Ausarbeitungen zum Seminar Computational Intelligence Methods

Wintersemester 2011/2012
Pascal Held (Hrsg.)

INF FAKULTÄT FÜR INFORMATIK

Inhaltsverzeichnis

Fuzzy Clustering without Apriori Knowledge

Christian Braune

Otto-von-Guericke-University of Magdeburg
Universitätsplatz 2, D-39106 Magdeburg, Germany
christian.braune@st.ovgu.de

Abstract. Clustering is one of the fundamental tasks in the data mining process. Be it the the first structural analysis of data, the selection of features suitable for prediction and classification or the detection of inherent structures within the data, clustering techniques are needed in many steps of the data mining process [1, 7]. Most of these techniques have in common that a certain *apriori* knowledge is required for the algorithms to perform well. Such knowledge may be the structure or density of the clusters to be found or something trivial like the number of clusters that there should be and with that, where the centers points might be. This overview will present some techniques that can be used to perform (unsupervised) clustering without such apriori knowledge and allow equally good results compared to common supervised techniques.

Keywords: prototypeless fuzzy clustering, optimal fuzzy clustering, k-means, fuzzy c-means, unsupervised clustering

1 Introduction and Motivation

Clustering is the task of detecting structures within a given data set. Data points (or entities) that are similar to each other are grouped together while data points that are dissimilar (or far from each other, in terms of distance measures) are grouped into different clusters. While upon visual inspection the human mind is quite able to distinguish between several groups ([14, 17]) such a task is either limited to at most three dimensional data sets in case of human, visual inspection or is still a challenging task for computers to do.

One of the best known clustering algorithms is the k-means algorithm [11]. With this algorithm a set of data points $X = \{x_1, \ldots, x_n\}, x_i \in \mathbb{R}^d$ can be partitioned into k groups of data points $\{X_1, \ldots, X_k\}$, such that

$$\forall i, j : X_i \cap X_j = \emptyset, 1 \leq i, j \leq k, i \neq j \tag{1}$$

and

$$\bigcup_{i=1}^{k} X_i = X \tag{2}$$

holds. To achieve this partitioning a set of cluster prototypes $C = \{c_1, \ldots c_k\}$ is initialized with random positions and iteratively updated. The updating process

recalculates the position of a prototype c_i to the mean of that points, to which c_i is the nearest of all prototypes, w.r.t. a given distance measure (e.g. euclidean distance). Resulting from this, to whole d-dimensional feature space will be partitioned into so-called Voronoi cells [5, 15] and all points within such a cell will be assigned to the cluster prototype lying at the balance point of that cell (see Figure 1, borders of Voronoi cells in grey).

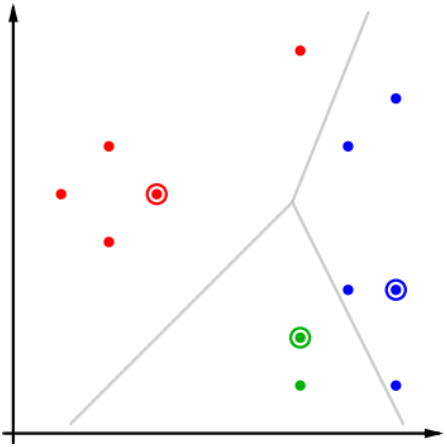

Fig. 1. Example of a k-means clustering with $k = 3$.[1]

The resulting paritioning of this algorithm heavily depends on the initial positions of cluster prototypes and is neither unique nor always optimal or intuitive. In addition to that, the k-means algorithm always finds a crisp partitioning of data points, i.e. each data point x_i is assigned to one and only one cluster. That is because each point can only lie in one of the aforementioned Voronoi cells. In a special case however, if the point is equidistant to at least two prototypes, i.e. the point is lying on the border line of at least two Voronoi cells, the point is assigned to one of the corresponding clusters randomly.

A better, more intuitive assignemnt of such points might be, that they do not belong to either cluster fully, but the degree of membership is shared between the involved clusters. Such a *fuzzy* assignment can then be applied to not only data points lying on the decision boundaries but to all data points. Prototypes are no longer updated by the points they are the nearest prototype to, but by all points. While points closer to the prototypes have more influence on the new position of this prototype, this influence vanishes with increasing distance. The data points themselves are not only assigned to one cluster alone, but - again w.r.t. their distance to the respective cluster center - are assigned membership

[1] Taken from lecture notes on Intelligent Data Analysis, R. Kruse, 2011, http://fuzzy.cs.uni-magdeburg.de/wiki/pmwiki.php?n=Lehre.IDA2011

degrees to each available cluster center. The resulting algorithm is known as fuzzy c-means [2] and returns not a crisp partitioning but a fuzzy partitioning matrix $U \in [0,1]^{k \times n}$ where each u_{ij} describes the membership of x_j to the cluster prototype c_i.

Still the initial membership degrees and cluster prototypes greatly influence the outcome of the clustering process and make it desirable to liberate oneself from such needs. The notion that a good clustering should minimize the distances of points within the same cluster and maximize the distances between points that are not in the same cluster can also be reformulated to an optimization problem. Such problem can then be solved with widely available toolkits for optimization.

For the fuzzy c-means algorithm the objective function would then be:

$$J(X, C, U) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^{\omega} \cdot d^2 \left(x_j, c_i \right) , \tag{3}$$

where ω is the so called *fuzzifier*, a real number greater than 1. With this parameter the crispiness of the resulting clusters can be controlled to a certain degree. Smaller values for ω result in crispier cluster. It should be noted, that if we restrict the values of each u_{ij} to be either 0 or 1, the clustering process would generate only crisp clusters and the result would be equal to a k-means clustering. This can be done by setting $\omega = 1$. A trivial - but useless - solution to this objective function would be $U = 0$, i.e. every $u_{ij} = 0$.

To prevent the optimization tools from finding such a solution, some constraints are employed such as

$$\forall 1 \leq j \leq n : \sum_{i=1}^{c} u_{ij} = 1 \tag{4}$$

and

$$\forall 1 \leq i \leq c : \sum_{j=1}^{n} u_{ij} > 0 \ . \tag{5}$$

Equations 4 and 5 ensure that each point is assigned to at least one cluster and that no cluster becomes empty. These constraints are usually incorporated in the objective function by means of Lagrange multipliers that incorporate the constraints directly into the target function (in Equation 6 the $lambda_k$s).

All aforementioned methods have in common, that at least some kind of apriori knowledge is required. The initial placement as well as the number of cluster prototypes have a significant influence on the clustering result. The choice of the distance measure used in computing the pair-wise distances can also lead to distinctively different cluster shapes, which are almost certainly different for different data sets and may be even different for clusters within the same data set. All these influence the outcome of the clustering process and the methods described above cannot cope well with greatly differing parameters of the clusters themselves, unless these are accounted for in the algorithms settings. With the notion of needed apriori knowledge one may call those common algorithms still

supervised despite the fact that there is no actual learning process. Yet the outcome heavily depends on the user's choice, which allows one think of the clustering process as if it was supervised.

This makes it highly desirable to have algorithms that can automatically choose which cluster prototypes are suitable, that do not need any prototypes for calculating the membership degrees or that do not even need to know the number of clusters to find beforehand. With this understanding, such clustering methods may be called *unsupervised*, although clustering itself is already known as unsupervised technique. Algorithms that cope with one or the other of these requirements will be presented in the following section.

2 Unsupervised Clustering Techniques

2.1 Prototypeless Fuzzy Clustering

In [3] Borgelt describes how the problem of having to place cluster prototypes in fuzzy clustering can be circumvented. Looking at the fuzzy c-means algorithm one can see that data points need their relative position to the cluster prototypes for recalculating their degrees of membership to these prototypes while the prototypes in return need the membership degrees of the data points to recalculate their positions. The approach here is to reformulate the objective function and the update rules in such a way, that the inherent goals of clustering are still reflected in the objective function, but only one update rule is needed. Borgelt here chose to keep the update rule for the membership degree instead of updating the prototype positions directly. This choice is derived naturally from the understandig, that the initial prototype placement can have a significant influence on the clustering outcome and thus should be avoided in an unsupervised algorithm. Nevertheless can the later approach be found in [10], were Hathaway and Bezdek reformulate the derived update rules to put the cluster prototypes to their new positions. The following part describes the derivation of an update rule for the membership degrees as it is decribed in [3].

In case of updating the membership degrees directly one first has to formulate the Lagrangian objective function:

$$J(X,U,D) = \sum_{i=1}^{c}\sum_{j=1}^{n}\sum_{k=1}^{j-1} u_{ij}^{\omega} u_{ik}^{\omega} d_{jk}^2 + \sum_{k=1}^{n} \lambda_k \left(1 - \sum_{i=1}^{c} u_{ik}\right) \tag{6}$$

Instead of using a distance measure this objective funtion solely depends on a dissimilarity matrix D, that can be either a distance matrix or any matrix containing dissimilarity values for data point pairs. This effectively eliminates the need for an embedding of data points in a metric space such that here also ordinal or even nominal features could be used if one somehow finds a way to measure the dissimilarity between such attributes. To find a minimum of this function a necessary condition is that all partial derivatives become zero, thus an equation system is derived by:

$$\frac{\partial J}{\partial u_{ab}} = \sum_{k=1}^{n} \omega u_{ab}^{\omega-1} u_{ak} d_{kb}^2 - \lambda_b = \omega u_{ab}^{\omega-1} \sum_{k=1}^{n} u_{ak} d_{kb}^2 - \lambda_b = 0 \tag{7}$$

Solving these partial derivates for $u_{ij}, \forall 1 \leq i \leq c, \forall 1 \leq j \leq n$ leads to

$$u_{ij} = \left(\frac{\lambda_j}{\omega \sum\limits_{k=1}^{n} u_{ik}^{\omega} d_{jk}^2} \right)^{\frac{1}{\omega-1}} \tag{8}$$

Taking into account that the membership degrees for each data point must sum up to one (each point is exhaustively distributed across the clusters), we can derive from

$$1 = \sum_{i=1}^{c} u_{ij} = \sum_{i=1}^{c} \left(\frac{\lambda_j}{\omega \sum\limits_{k=1}^{n} u_{ik}^{\omega} d_{jk}^2} \right)^{\frac{1}{\omega-1}} \tag{9}$$

that for each $\lambda_j, \forall 1 \leq j \leq n$ the update rule would be

$$\lambda_j = \left(\sum_{i=1}^{c} \left(\omega \sum_{k=1}^{n} u_{ik}^{\omega} d_{jk}^2 \right)^{\frac{1}{\omega-1}} \right)^{1-\omega} . \tag{10}$$

For directly updating the membership degrees we only need to insert the λ_j into the membership functions and have

$$u_{ij} = \left(\frac{\left(\sum\limits_{i=1}^{c} \left(\omega \sum\limits_{k=1}^{n} u_{ik}^{\omega} d_{jk}^2 \right)^{\frac{1}{\omega-1}} \right)^{1-\omega}}{\omega \sum\limits_{k=1}^{n} u_{ik}^{\omega} d_{jk}^2} \right)^{\frac{1}{\omega-1}} = \frac{\left(\sum\limits_{k=1}^{n} u_{ik}^{\omega} d_{jk}^2 \right)^{\frac{1}{1-\omega}}}{\sum\limits_{l=1}^{n} \left(\sum\limits_{k=1}^{n} u_{lk}^{\omega} d_{jk}^2 \right)^{\frac{1}{1-\omega}}} \tag{11}$$

The resulting system of non-linear equations is hardly solveable directly. Even in the case of $\omega = 2$ where the exponents of denominator and numerator become -1 this system is still hard to solve. In order to cope with this, Borgelt proposes an alternating optimization scheme (in [3]). In this case the derived update rule (Equation 11) is applied to each data point and cluster iteratively. Of the two possible ways to do so (online and batch) he argues that batch processing yields worse results is thus is to be avoided. Still this method requires some kind of initialization for the membership degress, where it does not seem to make a difference if all u_{ij} are initialied to $\frac{1}{c}$, where c is the number of desired clusters, or to random values and normalized to sum up to one.

In the end, when updating the degrees of membership, not all other data points need to be considered. Known as *constrained neighborhood clustering*,

only a subset of data points is used in the update process. These might be the k-nearest neighbors, the k-farthest neighbors or still all data points but weighted by the length of the shortest (w.r.t. euclidean distance) path along a k-nearest neighbor graph. Such a graph structure performs quite well in noisy data sets, as outliers will not be among the nearest neighbors of other, regular data points very often and thus have their influence automatically reduced.

2.2 Optimal Fuzzy Clustering

Another approach to allow for fuzzy clustering without apriori knowledge is described in [9]. Here the authors Gath and Geva do not aim to eliminate the need for cluster prototypes but the need to know the number of clusters in advance. They also handle the problem of different cluster shapes and densities in their approach of *unsupervised optimal clustering* by employing a different, statistical extension of the fuzzy c-means algortihm.

The general idea behind this approach is that the quality of a clustering can be measured somehow. If that is the case, the clustering for each number of clusters can be calculated by minimizing the objective function and then calculating the quality measure. The number of clusters that yields the best quality then should also lead to the optimal clustering.

For calculating each clustering itself the authors propose a slightly modified fuzzy c-means approach, the fuzzy maximum likelihood estimation clustering (FMLE).

Instead of using the euclidean distance in the objective function, they define a new metric

$$d^2(x_j, c_i) = (x_j - c_i)^T \cdot \Sigma \cdot (x_j - c_i) \tag{12}$$

where Σ can be any $d \times d$ positive definite matrix. If the identity matrix is chosen, the resulting clustering would be equal to that of a fuzzy c-means clustering with euclidean distance. But using the covariance matrix instead would give an explicit weighing of the features according to their statistical features.

The distance measure finally used is an exponential one, that uses a fuzzy covariance matrix F_i, i.e. the covariance matrix of the i^{th} cluster, which is calculated from only that subset of data that is assigned to cluster i. With this fuzzy covariance matrix an exponentially decaying distance measure can be formulated and used in the update rule. This narrows the search for the closest cluster prototype down to a small region in feature space. Because of that, this algorithm even more depends on the choice of proper cluster prototypes as it is more prone to local optima of the objective function.

To cope with the problem of getting stuck in a local optimum the authors propose an unsupervised method for finding good prototypes. The idea is to place the $(k+1)^{th}$ prototype in a region, where any the of k clusters has the lowest density, given a partitioning of the data into k clusters. For that a standard fuzzy c-means clustering with two cluster prototypes is performed. The first prototype is placed at the average position of all data points. Another cluster prototype

is then placed at an imaginary location that is equidistant to all points. With these a fuzzy partitioning is calculated. This process of adding prototypes and calculating a clustering is repeated until k cluster prototypes have been placed. These can then be used as starting seeds for the FMLE clustering.

The clustering process is repeated for $k = 1, \ldots, k_{max}$ and each time quality criteria are calculated. An optimal partitioning in the sense of the authors should give a clear separation of the resulting cluster, most data points should be located near the cluster center and the *volume* of the cluster should be minimal. As the determinant of a matrix can be interpreted as the volume described, the sum of all determinants of the fuzzy covariance matrices (each square rooted, as the volume is indeed spanned by the standard deviations) is a measure for the total (hyper-)volume of the clustering:

$$F_{HV} = \sum_{i=1}^{k} \left[\sqrt{det(F_i)} \right] \tag{13}$$

From this volume the average density can be computed by summing over the membership degrees of all points within that cluster. To fulfill the third requirement to a good clustering (most of the data points should lie near the cluster center) only those data points are considered when summing the membership degrees, which lie in a hypershpere of radius 1 (w.r.t. standard deviation) to the cluster center:

$$D_{PA} = \frac{1}{c} \sum_{i=1}^{c} \frac{S_i}{\sqrt{det(F_i)}} \tag{14}$$

where

$$S_i = \sum_{i=1}^{n} u_{ij}, \forall x_j \in \left\{ x_j : (x_j - c_i)^T F_i^{-1} (x_j - c_i) < 1 \right\} . \tag{15}$$

2.3 Other approaches

Due to space limitations other approaches to fuzzy clustering without prior knowledge will not be presented here in detail but yet they are worth mentioning.

In [12, 13] Pal et al. present a possibilistic approach that does not only use membership degrees but also *typicalities*, which are used to handle the presence of outliers in the prototype update process. An outlier may have a high degree of membership to its closest cluster but, given the cluster parameters, it might not be likely (hence: typical).

As mentioned above Hathaway and Bezdek derive a direct update rule for cluster prototypes without the explicit need for degrees of membership in [10]. Though the cluster prototypes can be updated directly, the computation of the membership degrees (which one is still more interested in than in the prototypes) has to be done afterwards.

In [4] Borgelt and Kruse elaborate on the procedure presented in Section 2.1. They alter the objective function again by inserting another (penalty) term for points that have high memberships in multiple clusters allows them to better

control the outcome of the clustering process. Doing so the clustering can be directed to crispier assignments in the core regions of the clusters while keeping fuzzy memberships in the border regions, allowing for the omittance of the neighborhood relations. The objective function with the lagrangian multipliers is extended to

$$J(X,U,D) = \sum_{i=1}^{c}\sum_{j=1}^{n}\sum_{k=1}^{j-1}(u_{ij}^{\omega}u_{ik}^{\omega} + \alpha(u_{ij}^{\omega} + u_{ik}^{\omega}))d_{jk}^{2} + \sum_{k=1}^{n}\lambda_k\left(1 - \sum_{i=1}^{c}u_{ik}\right) . \tag{16}$$

Setting α to small, negative values lead to almost crisp assignments of the data points to the clusters. For all data points clearly belonging to one class in the iris data set [8], $\alpha = -0.05$ will actually lead to assignments of degrees of membership to exactly one and only to fuzzy degrees in the overlapping regions.

3 Evaluation

Comparison of the presented approaches is difficult as both were tested on different data sets and had different goals. Furthermore only the implementation by Borgelt is publically available.

Fig. 2. Fuzzy c-means with $\omega = 2$ on the left, prototypeless fuzzy c-means with $\omega = 2$ on the right side. [3]

Nevertheless have both been using the well-known iris data set and have calculated the number of data points that were assigned to the wrong cluster (if the cluster result had been used as a classification criterion). Borgelt's approach (including the penalty term described in Section 2.3) here leads to 26 misclassifications. The same can be achieved with a standard fuzzy c-means algorithm and a fuzzifier $\omega = 2$. Without any additional information or constraints (i.e. $\alpha = 0$) the proposed algorithm will actually perform best, if only 2 clusters are required

on the iris data set. This is not surprising at all as the iris versicolor and iris virginica clusters are connected by significant overlap on at least 36 data points.

The later approach by Gath and Geva only *misclassifies* four out of 150 data points in the iris data set. The better recognition of the clusters can be attributed to the specific handling of cluster density and shape. Clearly this is one advantage of this proposal, as their experiments conclusively show. Differently shaped clusters are recognized as well as clusters with different densities. Even if clusters overlap their structure is revealed properly (as can be seen in the iris data set results).

4 Conclusions and Future Work

While the approach by Borgelt shares the same advantages as hierarchical agglomerative clustering (e.g. [6, 16]), it still suffers from the increased time complexity needed for the calculation. While normal k-means clustering can run in $O(knd)$ this algorithm needs $O(kn^2d)$ time complexity mainly due to the fact, that all pairwise distances need to be evaluated in the update process. The omitted need for initial cluster prototype placement (from which the second approach greatly suffers) reduced the need for apriori knowledge greatly. On the other hand the result of Gath and Geva's approach clearly leads to better results but has the significant drawback that it strongly depends on correct prototype placement. It also needs to calculate clustering for every possible value of k (up to a given maximum) such that its complexity is approximately $O(k^3nd)$. This is still in the range of computational feasible approaches, as k will usually be a rather small number compared to n. As for the size of d, the dimensionality of the data, this can be reduced by suitable techniques if the calculation of vector distances becomes a problem.

Although both approaches on their own show good results and achieve the goals they set themselves, it would be interesting to see, whether they could be combined or not. Borgelt's approach finds degrees of membership without initial placement of prototypes. From these the prototype locations could be computed (or one chooses Hathaways and Bezdek's reformulation approach altogether) and be used as initializations for Gath and Geva's approach. This would reduce the need for finding good cluster starting points, thus eliminating the dependency on the first part of their algorithm to find such starting points. In a second step the reformulation approach could also be applied to the clustering itself, reducing the apriori need to defining the parameter k_{max} that is the maximum number of clusters to be considered.

Borgelt's approach might even be used with a high value for k to find a lower boundary k_{min} for the number of clusters. As could be seen in the experiments with this approach, it performed best when only looking for two clusters because of the significant overlap. While not being able to distinguish between these clusters properly, it can be said, that *two* seems here to be a lower boundary on the number of clusters, which can be detected, as all data points in the $k=3$-case have almost equal degrees of membership to both the iris versicolor and the iris

virginica cluster. Thus, the lower boundary of clusters could be detected from a clustering with a large enough k. It would be interesting to see work on this in the future.

References

1. Berthold, M.R., Borgelt, C., Höppner, F., Klawonn, F.: Guide to intelligent data analysis. In: Gries, D., Schneider, F.B. (eds.) Guide to Intelligent Data Analysis, Texts in Computer Science, vol. 42, pp. 297–301. Springer London (2010)
2. Bezdek, J.C., Ehrlich, R., Full, W.: Fcm: The fuzzy c-means clustering algorithm. Computers & Geosciences 10(2-3), 191 – 203 (1984)
3. Borgelt, C.: Prototype-less fuzzy clustering. In: Proc. IEEE Int. Fuzzy Systems Conf. FUZZ-IEEE 2007. pp. 1–6 (2007)
4. Borgelt, C., Kruse, R.: An extended objective function for prototype-less fuzzy clustering. In: Proc. Annual Meeting of the North American Fuzzy Information Processing Society NAFIPS '07. pp. 146–151 (2007)
5. Chinrungrueng, C., Sequin, C.H.: Optimal adaptive k-means algorithm with dynamic adjustment of learning rate. Neural Networks, IEEE Transactions on 6(1), 157–169 (1995), http://dx.doi.org/10.1109/72.363440
6. Dong, Y., Zhuang, Y.: Fuzzy hierarchical clustering algorithm facing large databases. In: Proc. Fifth World Congress Intelligent Control and Automation WCICA 2004. vol. 5, pp. 4282–4286 (2004)
7. Ester, M., Sander, J.: Knowledge Discovery in Databases - Techniken und Anwendungen. Springer (2000)
8. Fisher, R.A.: The use of multiple measurements in taxonomic problems. Annals of Eugenics 7(7), 179–188 (1936)
9. Gath, I., Geva, A.B.: Unsupervised optimal fuzzy clustering. IEEE Transaction on Pattern Analysis and Machine Intelligence 11(7), 773–780 (1989)
10. Hathaway, R.J., Bezdek, J.C.: Optimization of clustering criteria by reformulation. IEEE Transactions on Fuzzy Systems 3(2), 241–245 (May 1995)
11. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Cam, L.M.L., Neyman, J. (eds.) Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability. vol. 1, pp. 281–297. University of California Press (1967)
12. Pal, N.R., Pal, K., Bezdek, J.C.: A mixed c-means clustering model. In: Proc. Sixth IEEE Int Fuzzy Systems Conf. vol. 1, pp. 11–21 (1997)
13. Pal, N.R., Pal, K., Keller, J.M., Bezdek, J.C.: A possibilistic fuzzy c-means clustering algorithm. IEEE Transactions on Fuzzy Systems 13(4), 517–530 (2005)
14. Rush, G.P.: Visual grouping in relation to age. Archives of Psychology (Columbia University) 94(No. 217) (1937)
15. Schreiber, T.: A voronoi diagram based adaptive k-means-type clustering algorithm for multidimensional weighted data. In: Bieri, H., Noltemeier, H. (eds.) Computational Geometry-Methods, Algorithms and Applications, Lecture Notes in Computer Science, vol. 553, pp. 265–275. Springer Berlin / Heidelberg (1991)
16. Wang, P.C., Leou, J.J.: New fuzzy hierarchical clustering algorithms. Information Science and Engineering 9(3), 461–483 (1993)
17. Xu, Y., Chun, M.M.: Visual grouping in human parietal cortex. Proceedings of the National Academy of Sciences 104(47), 18766–18771 (2007), http://www.pnas.org/content/104/47/18766.abstract

Challenges on Association Rule Mining On Data Streams in Contrast to Classical Association Rule Mining Algorithms

Anja Bachmann

Otto-von-Guericke-University of Magdeburg
Universitätsplatz 2, D-39106 Magdeburg, Germany
anja.bachmann@st.ovgu.de

Abstract. There are a lot of areas where association rule mining is applied. Unfortunately there are some challenges one has to face, e.g. a huge dataset or a great amount of frequent items. Especially when handling stream data there are some additional demands on association rule mining. They are among others caused by a time-critical response demand or a high arriving rate of new items. This paper lists some of those challenges like "Single Access to Data" or "Real-time Response". It also overviews several algorithms, e.g. *Apriori*, *Eclat*, *FP-Growth*, FUP and FUP_2, the algorithm of Giannella and Thomas' approach. It concludes the main facts and gives an outlook on possible future work.

Keywords: data mining, association rule mining, stream mining, challenges in association rule mining on stream data

1 Introduction

Nowadays association rule mining is applied more and more. It can be used to figure out dependencies between items and to look for linkages between them. Basis is frequent item set mining which has a vast amount of implementation forms stated in [3,10]. This paper concentrates on its adoption for association rule mining. The number of application areas is large and the cardinality of datasets such an algorithm has to handle can get huge. Therefore the algorithms have to be quite efficient without computational overhead.

With respect to association rules one not only has to take the cardinality of a dataset into account but also the dynamic of the data. On the one hand it is possible that the dataset is static and not evolving over time. On the other hand there are so-called data streams where the data is dynamic and changes over time. In both variants there are challenges one has to face. Some core problems, that usually appear, are that one has to handle huge datasets or that the results have to be presented fast. Therefore it is important to have efficient algorithms that assure fast performance. Especially in dynamic association rule mining long computations must be avoided. Otherwise it is possible that the data, which was

handled as current data during the performance, is already obsolete. Another occurring problem is that frequent items could become non-frequent over time or the other way round [13]. Those problems have to be considered and avoided as much as possible.

The rest of this paper is organised as follows. In Section 2 there will be an introduction into the topic of association rule mining. Afterwards there will be some considerations regarding the key challenges when applying association rule mining on data streams with the aims of effectiveness and efficiency. In Section 4 different established algorithms are introduced and considered. In the last section there will be a conclusion and an outlook to future work.

2 Basics and State of the Art

Association rule mining is a technique that was popularised by Agrawal et al. in 1993 as they proposed their algorithm AIS [1]. These authors also bring a first definition for item set mining and for association rule mining [1,2]: Let $I = \{i_1, i_2, ..., i_m\}$ be a set of items and D a set of transactions. Each transaction $T \in D$ is a set of items so that $T \subseteq I$. Assuming that $X \subseteq I$, then T contains X if $X \subseteq T$. An item set X has a support s in the transaction set D if $s\%$ of transactions in D contains X. Such an item set X is frequent if its support is higher than a user defined minimum support value $minsupp$.

An association rule was defined by Agrawal et al. in [1,2] as follows: It is an implication of the form $X \Rightarrow Y$ where $X \subset I$, $Y \subset I$ and the intersection of X and Y is the empty set. An often-used example for this is the myth of a person who buys diapers on Friday and additionally beer. An association rule for this is $diapers \Rightarrow beer$ which possibly states that a customer also buys beer when buying diapers [16]. Such a statement could help to rearrange a shop for improving its revenue. As [1,2] say, the rule $X \Rightarrow Y$ holds in the set of transactions D with a so-called confidence value c if $c\%$ of all transactions in D containing X also contain Y. A rule $X \Rightarrow Y$ has a support s if $s\%$ of all transactions in D contain the union of X and Y.

In [1,2] Agrawal et al. defined the problem of association rule mining by stating it as two subproblems: The first one is to find all item sets whose support is higher then the minimum support, i.e. all item sets that are so-called large or frequent. The second step is to generate all desired rules by using those frequent item sets. Also both Tan et al. and Yu and Chi state the problem out as those two steps [19,25]. There are several efficient techniques for solving the second step [25], e.g. the algorithm explained in [1]. Hence, most approaches concentrate on improving the first step [25], i.e. the challenge is to figure out efficient item set mining algorithms.

It is possible to subdivide such algorithms regarding certain criteria. One of them is their property of working on static or dynamic data. This means that an algorithm works on a dataset whose content either stays the same over time (static) or changes over times (dynamic). When handling dynamic data there

are new data items arriving at certain time points. The old data could either be kept or forgotten. Mining on such data is also called stream mining and can be defined as follows. A data stream is an ordered sequence of items that arrive in timely order" [13], but also "a sequence of unbounded, real time data items with a very high data rate" [14]. At different time points new data items arrive and can be proceeded.

Algorithms for stream mining can be assigned to one of three model types [13]: Landmark, Damped or Sliding Windows. By applying the Landmark model all frequent item sets over the entire history from one starting point, so-called landmark, until the end of this data stream are considered. In the Damped model there is an aging function that weights the arriving items and decrease the weight with age. By using the Sliding Window model frequent item sets in a sliding window can be found and maintained. Only the sliding window part of the stream is stored and processed. The selection of a model type depends on the application. In [13] Jiang and Gruenwald state that it is possible to switch from one type to another one by applying some adjustments.

3 Key Challenges

During the process of software engineering several phases are passed [4]. Especially the requirement analysis and the following design phase are important if aiming at creating an efficient and effective algorithm, because one has to know the general conditions.

There are some such challenges that one has to face when applying stream mining. In [25] Yu and Chi define three of them as "key challenges" shown in Table 1 in which *Challenge* represents the requirement that should be fulfilled to improve the efficiency of the algorithm and *Description* summarises the reasons for the requirement named in *Challenge*.

Table 1. Key challenges explicated by Yu and Chi [25]

Challenge	Description
Single Access to Data	It is very expensive to access data more than one time. Therefore it is necessary to just access the data once for detecting frequent item sets.
Handle Unbounded Data	The arriving data is unbounded. In contrast the storage and calculating capacity are limited. Hence it is inevitable to get used to work on limited resources.
Real-time Response	Stream mining algorithms are time-critical and require a short response time. Hence the speed of the association rule algorithm has to be higher than the incoming rate of the data.

In [13] Jiang and Gruenwald agree with these, but go further by stating two more challenges which are shown in Table 2.

Table 2. Key challenges explicated by Jiang and Gruenwald [13]

Challenge	Description
Data drifting	It is possible that the data distribution evolves over time. Stream mining algorithms have to be able to handle such events.
Incremental processing	Arriving data that evolves may also cause evolving analysis results. Therefore the mining process has to be an incremental one.

During the presentation of some association rule mining algorithms in Section 4, the focus will be on the challenges explicated by Yu and Chi [25]. They are the most important ones and due to space limitations only them will be considered in this paper. The challenges additionally named by Jiang and Gruenwald [13] could be examined in further researches.

4 Algorithms for Stream Mining

As said in Section 2, there are two main steps, but most approaches concentrate on improving the basic step, i.e. the item set mining part. Hence the focus is now on several item set mining algorithms. First of all there will be a presentation of some well-known static algorithms, namely *Apriori*, *Eclat* and *FP-Growth*. Afterwards there will be an introduction of some stream mining algorithms.

4.1 Item Set Mining Algorithms

As item set mining is the basis for association rule mining, efficient algorithms for item set mining have to be considered. Hence the next subsection will give an overview over several of such algorithms.

Apriori One of the best-known algorithms is *Apriori* that was introduced in [2] by Agrawal et al. The prefix tree, that corresponds to the lattice based on the dataset, is traversed by this algorithm by breadth-first-search. The core idea of *Apriori* is to prune all sub-trees whose root is infrequent. By doing this the candidate set can get very small and hence the algorithms works lots faster. There were several enhancements of this algorithm, e.g. *AprioriTID* [2], *AprioriHybrid* [2], *DIC* [5], or *Apriori*-like algorithms, e.g. the *Partition* algorithm [18], which will not be considered in this paper, but it is described in [12].

Eclat Another approach is called *Eclat* and was conceived by Zaki et al. in [27]. This algorithm uses depth-first-search for traversing the item set tree and accomplishes transaction list intersection. At this, one is only interested in those resulting transaction lists that exceed the minimum support. This means that each intersection should be stopped as soon as it is obvious that the resulting transaction list becomes infrequent [12].

FP-Growth *FP-Growth* is another well-known algorithm and like *Eclat* it also uses depth-first-search [11]. The characteristic of *FP-Growth* is its accomplishment of a preprocessing step for deriving a highly condensed representation of transactions, i.e. a FP-Tree [12]. In a second step the support values of all frequent item sets are derived [12].

Those traditional item set, but also association rule mining algorithms are applicable on static data and provide very good and fast results. Unfortunately they are not applicable for dynamic data. Therefore it was necessary to develop algorithms that are able to work on data streams [13].

4.2 Stream Mining Algorithms

Unfortunately it is not possible to use those *Apriori*-based algorithms for stream mining [13]. The problem is that they require multiple scans on the original data set and hence cause a lot of accesses and quite high CPU costs. Despite its efficiency in contrast to other algorithms, even *FP-Growth* is not applicable for data streams. Here the problem is that the FP-tree requires two scans of the data. As more and more applications use association rule mining for stream data, more research is conducted [13].

Like explained in Section 3 there are several key challenges and it is nearly infeasible to cope with all of them. Therefore it is advisable to make some distinctions. Stream mining algorithms can either be exact or approximate. If the dataset is pretty small then it is no problem to do an exact mining. In this case the result set consists of all item sets whose support is higher or equal to the minimum support [13]. But if the cardinality of the data is too large then an approximate mining is a good compromise between effectiveness and efficiency.

In the following subsections some stream mining algorithms will be presented.

FUP and FUP$_2$ In [7,8] Cheung et al. conceived two algorithms FUP and FUP_2. FUP is an algorithm for incrementally updating frequent item sets and quite similar to Apriori. FUP_2 is an extension of FUP that allows deleting old transactions. Due to this fact FUP_2 is also applicable for the sliding window model [25].

Thomas' approach Thomas et al. proposed an incremental algorithm [21] that is pretty similar to FUP_2, in addition that a negative boarder for item sets is maintained [25]. First the frequent item sets are mined and the counts of all frequent item sets are updated. Afterwards with a possible scan of the updated database, the negative boarder, the detected frequent item sets and the frequent item sets in the updated database are computed. This algorithm is pretty fast due to the fact that it mostly only needs one scan of the updated database. It is, like FUP_2, also applicable for the sliding-window data model.

Other exact algorithms Karp [15] and Yang [23] also present exact algorithms. Unfortunately both of them got significant drawbacks: Karps algorithm need two database scans for generating exact results, Yangs algorithm is only applicable for small datasets. That means that two of the key challenges from Section 3 fail, i.e. Single Access to Data and Real-time Response. In [9] Chi proposed an algorithm that only mines closed item sets over a Sliding Window model [13]. Mao presents an algorithm that maintains maximal frequent item sets over a Landmark model [13]. Unfortunately in both cases it is not clear how to get all the information to further generate association rules [13].

Giannella's approach In their work [10] Giannella et al. present an approximate stream mining algorithm. The characteristic is that it mines frequent item sets during arbitrary time intervals or rather tilted-time intervals. A so-called FP-stream is used to manage historic information and their frequencies over time. An aging function is used to weight the items in such a way as to give the newest data the highest weight. A benefit of that algorithm is that it provides different error levels for items at multiple time granularities [25] what is appropriate for applications where users are more interested in getting detailed information from the recent time period [13].

Other approximate approaches In [13] some other approximate stream mining algorithms are presented shortly. One of them is $FTP\text{-}DS$ conceived by Tang et al. [20] that mines frequent temporal patterns by using a Sliding Window model. Another one is *estDec* by Cheng et al. [6] that mines recent frequent item sets. For defining the frequency an aging function is used. Also as Giannellas approach this one mines on weighted transactions. Both $FTP\text{-}DS$ and *estDec* are thoroughly useful algorithms, but outperformed by other ones.

Further considerations It is also possible to differentiate between stream mining algorithms regarding the dimensionality of the underlying dataset (e.g. multidimensional data streams) or regarding methods to minimise the dataset (e.g. random sampling or partitioning). Information about it can be found in [17,24] (multidimensional data streams), [22,26] (random sampling) and [18] (partitioning).

Jiang and Gruenberg figured out that additional to a well-working algorithm also an efficient and compact data structure is needed [13].

5 Conclusions and Future Work

Association rule mining consists of two steps: First of all the frequent item sets must be retrieved, afterwards the association rules must be generated [1,2]. There are several statistical solutions for the second step. Hence the focus is on figuring out an efficient but also effective algorithm to detect frequent item sets [25], i.e. fasten an association rule algorithm means above all fasten the frequent item set mining algorithm. This holds for both static and dynamic association rule mining.

Unfortunately there are some challenges appearing when applying association rule mining on dynamic data, because it is not possible to use the classical algorithms for stream data. Therefore special association rule stream mining algorithms had to be conceived. Those algorithms differ not only in their exactness (exact vs. approximate), but also in the based model they work on (Landmark vs. Damped vs. Sliding Windows). There are some key challenges presented in Section 3 which every such stream mining algorithm should aim at. Unfortunately it is nearly impossible to cope with all of them. Therefore it is comprehensible that every algorithm solves these challenges differently.

Depending on the application and the cardinality of the data set one should choose a different model and a different exactness of an algorithm. Other choosing criteria are the accuracy of the results, but also the resource consumption and the computation time. The chosen algorithm should be a good compromise.

Apart from this, it seems like FUP_2 and Giannellas approach work pretty well.

Some future challenges could be to compare association rule mining algorithms that work on multidimensional data streams. Another task could be to create an approach that hybrids counting occurrences as frequency and transaction list intersection. It is also worth to be considered if it is possible to find algorithms for reducing the cardinality of the data set for computation with guaranty to take all items into account for frequent item set mining.

References

1. R. Agrawal, T. Imielinski and A. Swami. Mining Association Rules Between Sets of Items in Large Databases. *ACM SIGMOD Record*, Vol. 22, No. 2, p.207-216. June, 1993.
2. R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. *Proc. of the 20th Int'l Conf. on Very Large Data Bases (VLDB)*. Santiago, Chile. September 12-15, 1994.
3. M.R. Bertholt, C. Borgelt, F. Hppner and F. Klawonn. Guide to Intelligent Data Analysis. Springer. 2010.

4. B.W. Boehm. A Spiral Model of Software Development and Enhancement. *IEEE Computer*, Vol. 21, No. 5, p.61-72. May, 1988.
5. S. Brin, R. Motwani, J.D. Ullman and S. Tsur. Dynamic itemset counting and implication rules of market basket data. *ACM SIGMOD Record*, Vol. 26, No. 2, p.255-264. June, 1997.
6. J.H. Chang and W.S. Lee. Finding Recent Frequent Itemsets Adaptively over Online Data Streams. *Proc. of the 9th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD)*. Washington D.C., USA. August 24-27, 2003.
7. D.W. Cheung, J. Han, V. Ng and Y.C. Wong. Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique. *Proc. of the 12th Int.' Conf. on Data Engineering*. New Orleans, USA. February 26-March 1, 1996.
8. D.W. Cheung, S.D. Lee and B. Kao. A General Incremental Technique for Maintaining Discovered Association Rules *Proc. of the 5th Int'l Conf. on Database Systems for Advanced Applications (DASFAA)*. Melbourne, Australia. April 1-4, 1997.
9. Y. Chi, H. Wang, P.S. Yu and R.R. Muntz. Moment: Maintaining Closed Frequent Itemsets over a Stream Sliding Window. *Proc. of the 4th IEEE International Conference on Data Mining (ICDM)*. Brighton, UK. November 01-04, 2004.
10. C. Giannella, J. Han, J. Pei, X. Yan and P. S. Yu. Mining Frequent Patterns in Data Streams at Multiple Time Granularities. *Next generation data mining*, Vol. 212, p.191-212. 2003.
11. J. Han, J. Pei and Y. Yin. Mining Frequent Patterns Without Candidate Generation. *ACM SIGMOD Record*, Vol. 29, No. 2, p.1-12. June, 2000.
12. J. Hipp, U. Güntzer and G. Nakhaeizadeh. Algorithms for Association Rule Mining - A General Survey and Comparison. *ACM SIGKDD Explorations Newsletter*, Vol. 2, No. 1, p.58-64. July, 2000.
13. N. Jiang and L. Gruenwald. Research Issues in Data Stream Association Rule Mining. *ACM Sigmod Record*, Vol. 35, No. 1, p.14-19. March, 2006.
14. J. Joseph. Seminar Report On Data Stream Mining. Cochin. 2010.
15. R.M. Karp and S. Schenker. A Simple Algorithm for Finding Frequent Elements in Streams and Bags. *ACM Transactions on Database Systems*, Vol. 28, No. 1, p.51-55. March 2003.
16. B. Padmanabhan and A. Tuzhilin. Unexpectedness as a Measure of Interestingness in Knowledge Discovery. *Decision Support Systems*, Vol. 27, No. 3, p.303-318. December 1999.
17. H. Pinto, J. Han, J. Pei, K. Wang, Q. Chen and U. Dayal. Multi-Dimensional Sequential Pattern Mining. *Proc. of the 10th Int'l Conf. Information and Knowledge Management (CIKM)*. Atlanta, USA. November 5-10, 2001.
18. A. Savasere, E. Omiecinski and S. Navathe. An Efficient Algorithm For Mining Association Rules in Large Databases. *Proc. of the 21st Conf. on Very Large Data Bases (VLDB)*. Zrich, Switzerland. September 11-15, 1995.
19. P.N. Tan, M. Steinbach and V. Kumar. Introduction to Data Mining. Pearson Addison Wesley Boston. 2006.
20. W-G. Teng, M-S. Chen und P.S. Yu. A Regression-based Temporal Pattern Mining Scheme for Data Streams. *Proc. of the 29th Int'l Conf. on Very Large Data Bases (VLDB)*. Berlin, Germany. September 9-12, 2003.
21. S. Thomas, S. Bodagala, K. Alsabti and S. Ranka. An Efficient Algorithm for the Incremental Updation of Association Rules in Large Databases. *Proc. of the 3rd Int'l Conf. on Knowledge Discovery and Data Mining (KDD)*. Newport Beach, USA. August 14-17, 1997.

22. H. Toivonen. Sampling Large Databases for Association Rules. *Proc. of the 22nd Int.' Conf. on Very Large Data Bases (VLDB)*. Mumbay, India. September 3-6, 1996.
23. L. Yang and M. Sanver. Mining Short Association Rules with One Database Scan. *Proc. of the Int'l Conf. on Information and Knowledge Engineering*. Las Vegas, USA. June 21-24, 2004.
24. C-C. Yu and Y-L. Chen. Mining Sequential Patterns from Multidimensional Sequence Data. *IEEE Transactions on Knowledge and Data Engineering*. Vol. 17, No. 1, p.136-140. January, 2005.
25. P. S. Yu and Y. Chi. Association Rule Mining on Streams. *Encyclopedia of Database Systems*, p.136-139. Springer US. 2009.
26. M.J. Zaki, S. Parthasarathy, W. Li and M. Ogihara. Evaluation of Sampling for Data Mining of Association Rules. *Proc. of the 7th Int'l Workshop on Research Issues in Data Engineering*. Birmingham , UK. April 7-8, 1997.
27. M.J. Zaki, S. Parthasarathy, M. Ogihara and W. Li. New Algorithms For Fast Discovery of Association Rules. *Proc. of the 3rd Int'l Conf. on Knowledge Discovery and Data Mining (KDD)*. Newport Beach, USA. August 14-17, 1997.

Adaptive Classification

Roksolana Pleshkanovska
FIN / FGSE
Otto-von-Guericke Universität
Magdeburg, Deutschland
roksolana.pleshkanovska@st.ovgu.de

***Abstract*—„adaptive classification“ unterscheidet sich von klassischer Klassifizierung dadurch, dass der Klassifikator sich selbst anpasst. Zunächst wird auf ACT als ein Anwendungsbeispiel eingegangen. Im nächsten Schritt werden die „adaptive classification“ Prozeduren im Allgemeinen erläutert. Die Modell-basierte Klassifizierung, die „adaptive classification“ mit der Variationsmethode eines Kalman Filters, „adaptive prototype-based fuzzy classification“, „adaptive classification“ von zweidimensionalen Gel-elektrophoretischen Punktmustern mittels neuronaler Netze und mit Hilfe einer Cluster-Analyse, „adaptive classification“ für BCIs, „adaptive classification“ multispektraler Daten, „adaptive classification“ von EEG-Funktionen bei einem spärlichen Feedback werden ebenfalls näher erklärt. Letztendlich wird nur noch die grundlegende Klassifizierung an den Beispielen der Prototypmethoden (K-means Clustering und Gaussian Mixture) und der nächste-Nachbarn Methoden (k-nächste-Nachbarn Methoden) beschrieben.**

I. Anwendungsbeispiel: Was versteht man allgemein unter Adaptive Classification Technology (ACT)?

Nach einer erfolgten Analyse werden elektronische Dokumente durch eine Klassifizierung unterschiedlichen Dokumentenklassen zugeordnet (Abbildung 1). Die Einteilung geschieht auf Basis inhaltlicher Kriterien und nicht anhand des Layouts. Falls in einem Dokument spezielle Wörter, Sätze oder Folgen von Zeichen vorkommen, kann es auf diese Art und Weise einer speziellen Klasse zugewiesen werden. ACT ist in der Lage, diese Merkmale mit Hilfe einer Stichprobe selbstständig zu erkennen.

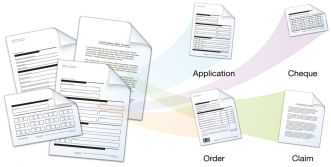

Abb. 1: siehe [1]

In der Praxis kann beispielsweise durch die Software erfasst werden, ob in einer Versicherung eine Schadensmeldung oder ein Mailing-Rückläufer als eingehendes Dokument vorliegt [2].

Im Allgemeinen existiert eine Ergänzung um Index- und Nutzdaten. Bei ACT handelt es sich um ein Konzept des Open Text Capture Document Reader (DOKuStar), der das vorliegende Dokument vom Inhalt her erschließen soll. Als besonders vorteilhaft erweisen sich dabei zum Beispiel folgende Tatsachen: Die Klassifizierung findet automatisch statt, das Verfahren ist vollständig selbstlernend, die Verwaltung ist ziemlich einfach, alle Daten befinden sich in einer Datenbank und außerdem kann man ATC in Kombination mit einigen anderen Klassifikationsmethoden nutzen.

Zunächst werden alle Beispieldokumente geladen, dann berechnet DOKuStar ACT einen geeigneten Klassifikator, mit dessen Hilfe zum Produktionszeitpunkt die einzelnen Dokumente klassifiziert werden sollen. Um einen solchen Klassifikator zu bestimmen, untersucht DOKuStar die Dokumente und stellt dabei fest, welche besonderen Merkmale auf allen Dokumenten einer Klasse aufzufinden sind und in welcher Hinsicht sich diese Merkmale von denen anderer Dokumentenklassen unterscheiden.

In der nachfolgenden Abbildung (Abbildung 2) repräsentieren die grünen Bereiche die korrekten Klassifikationsergebnisse, die gelben Bereiche entsprechen den zurückgewiesenen Dokumenten, da die Klassifikationsresultate zu unsicher gewesen sind, und die roten Bereiche sind den fehlerhaften Klassifikationen gleichzusetzen [3]:

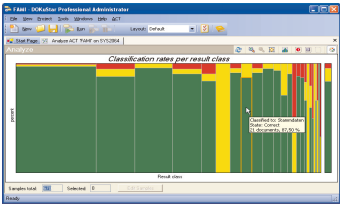

Abb. 2: siehe [2]

II. Unterschiedliche „adaptive classification“ Methoden

A. *Modell-basierte Klassifizierung – vorausschauendes „adaptive classification“- Model für Analysen und darauffolgende Mitteilungen – interne Bedrohungen als Problemstellung*

Das von einem Labor geleitete Forschungs- und Entwicklungsprojekt wird unter dem Schwerpunkt „Predictive Defense“ / „vorausschauende Verteidigung“ durchgeführt.

Abb. 3: siehe [3]

Eine derzeit bekannte Praxis bei der Bewältigung interner Cyber-Bedrohungen (Abbildung 3) besteht darin, das Netzwerk sowie die Einzelsysteme gleichermaßen zu überwachen, um dazu in der Lage zu sein, ermitteln zu können, wenn jemand nicht den festgelegten politischen Regeln folgt oder die für ihn genehmigte zulässige Höhe des Zugangs im eigenen Interesse auf eine Art und Weise missbraucht wird, die als schädlich für eine Organisation eingestuft werden würde.

Darin beinhaltet ist die Nutzung von Tools wie beispielsweise Firewall logs (Protokolle) oder IDS-Systemen (Abbildung 4) in Netzwerken oder auf Host-Systemen, die Aufzeichnungen von Aktivitäten erzeugen, mit dem Zweck, die später abzurufen. Dieses Handeln wird deswegen ausgeführt, weil man von einer von außen her verursachten Notwendigkeit ausgeht.

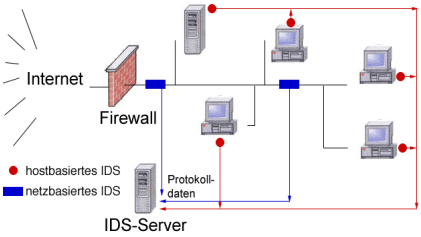

Abb. 4: siehe [4]

Eine große Herausforderung für die Forschung ist die Zeit zwischen dem eigentlichen „Vergehen“ und der Erkennung des „Vergehens“ zu reduzieren, sogar bis zu einem Punkt, wo die Erkennung von „Bedrohungsindikatoren“ dabei helfen kann, solche und ähnliche Angriffe vorherzusagen, bevor sie abgeschlossen werden.

Ein wesentliches Ergebnis beinhaltet eine Auswertung der vorherigen Forschung und Praxis im Hinblick auf ein „Insider-Bedrohungserkennungs-Tool“. Es wird also eine technische und wissenschaftliche Grundlage für diese Arbeit damit geschaffen.

Weiterhin wird ein Konzept für die angestrebten, vorausschauenden, adaptiven Aufgaben und Funktionen des abgebildeten Modells im untenstehenden Diagramm entwickelt, welches die Verarbeitung der ankommenden Sensordaten zeigt, um daraus Beobachtungen abzuleiten, die folgende Verarbeitung der Beobachtungen, um Indikatoren zu erschließen und die Analyse der Indikatoren, um den Umfang der Bedrohung zu messen. Die Studie beschreibt auch die „Bedrohungsindikatoren“, die bereits Vorstufen zu Taten anzeigen.

Der aktuelle Schwerpunkt beschäftigt sich damit, ausgewählte Klassifikationsalgorithmen zu implementieren und die Komponenten des vorrauschauenden Modells zu begründen [8].

Eine Model-basierte Klassifizierung (Abbildung 5) könnte man sich vorstellen, wie sie in der hier abgebildeten Grafik erläutert wird:

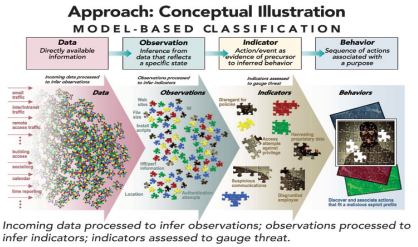

Abb. 5: siehe [5]

Zu Beginn müssen entsprechende Daten als direkt verfügbare Informationen zur Verfügung gestellt werden. Es könnte sich dabei beispielsweise um Email-Verkehr, Internet-Verkehr, Kalenderdaten und ähnliche Datenquellen handeln. Eingehende Daten werden dabei verarbeitet, um daraus entsprechende Beobachtungen abzuleiten.

Im nächsten Schritt werden viele Beobachtungen durchgeführt. Es findet eine Interferenz von Daten statt durch die jeweils ein gewisser Zustand reflektiert wird. Die entstandenen Beobachtungen werden außerdem verarbeitet, um sogenannte Indikatoren zu liefern.

Darauffolgend erkennt man Aktionen, die als Beweise dienen und eine Vorstufe für ein interferentes Verhalten darstellen. Die Indikatoren werden danach beurteilt und bewertet, ob sie die Bedrohungen messen können.

Zum Schluss geht es um das gesamte Verhalten. Eine Sequenz von Aktionen wird betrachtet, die mit einem Zweck assoziert wird.

B. „adaptive classification“-Prozeduren allgemein

Eine explizit berechenbare, notwendige und hinreichende Voraussetzung für die Existenz eines adaptiven Verfahrens zur Klassifizierung muss vorliegen.

Definitionsgemäß ist ein adaptives Verfahren dafür zuständig, eine Stichprobe aus einer alternativen Verteilung, die nur bis zu einem endlichen Störwertparameter bekannt ist, zu klassifizieren.

Es ist erforderlich, dass das gleiche asymptotische Verhalten (Abbildung 6 als ein Beispiel für asymptotisches Verhalten) der Fehlerwahrscheinlichkeiten für diese Familien vorliegt, genauso wie asymptotisch optimale Regeln für jede der Familien.

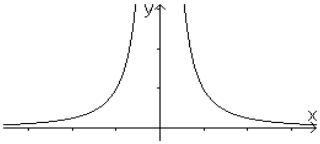

Abb. 6: siehe [6]

Es muss erforscht werden, unter welchen Bedingungen die gesamte Maximum-Likelihood-Prozedur (Abbildung 7) adaptiv ist und es muss eine adaptive Regel hergeleitet werden, wenn die Prozedur tatsächlich adaptiv sein sollte.

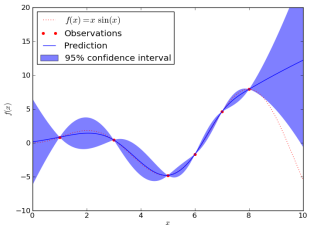

Abb. 7: siehe [7]

Man versucht, gewisse Übereinstimmungen der Prozeduren hervorzuheben. Es wird eine Untersuchung auf der Grundlage von Stichproben der Fehlerwahrscheinlichkeiten durchgeführt [7].

C. „adaptive classification“ mit der Variationsmethode eines Kalman Filters

In einem Paper von Peter Sykacek und Stephen Roberts wird ein probalistischer Ansatz, also ein Ansatz mit einer bestimmten Wahrscheinlichkeit, vorgeschlagen. Dieser bezieht sich auf die adaptiven Schlussfolgerungen einer zugrunde liegenden allgemeinen nichtlinearen Klassifizierung, die den rechnerischen Vorteil einer parametrischen Lösung mit der Flexibilität der sequenziellen Probenahme-Techniken kombiniert.

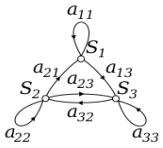

Abb. 8: siehe [8]

Man betrachtet die Parameter bei der Klassifizierung als latente beziehungsweise als nicht unbedingt sichtbare Zustände in einer Markov-Kette (Abbildung 8), die als ein spezieller stochastischer Vorgang als eine Basis dient. Es wird von einem sogenannten Zustandsraum E gesprochen, der abzählbar viele mögliche Werte enthält. Die „Markov-Eigenschaft“ charakterisiert einen Vorgang, bei dem die Wahrscheinlichkeit des Übergangs von einem Zustand in den nächstfolgenden Zustand von der „Vorgeschichte“ nicht abhängt [5].

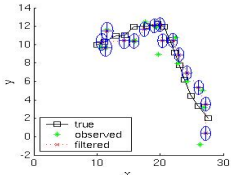

Abb. 9: siehe [9]

Es wird ein Algorithmus kreiert, der als eine Verallgemeinerung des Standard-Kalman Filters (Abbildung 9 als ein Beispiel für die Arbeit eines Kalman Filters) beurteilt werden dürfte. Der Kalman Filter, der ein Variationsproblem darstellt, beruht in diesem Fall auf zwei neuartigen unteren Schranken, die es ermöglichen, eine nicht-ausgeartete Verteilung über die Anpassungsrate des Kalman Filters zu verwenden. Eine Darstellung wird im Übrigen als nicht-ausgeartet bezeichnet, wenn es außer {0} keinen abschlossenen invarianten Unterraum gibt, so dass die Einschränkung darauf die Nulldarstellung ist.

Eine umfangreiche empirische Auswertung zeigte, dass die vorgeschlagene Methode fähig ist, wettbewerbsfähige Klassifikatoren zu erschließen. Sowohl in einer stationären als auch in einer nicht-stationären Umgebung wäre dies der Fall.

Obwohl der Schwerpunkt auf der Klassifizierung liegt, ist der Algorithmus leicht auf die anderen nicht-linearen Modelle ausgeweitet [6].

D. „adaptive classification“ von Zellbildern - „adaptive prototype-based fuzzy classification“

In einem Paper von Nicolas Cebron und Michael R. Berthold wird nach möglichen Lösungsansätzen gesucht, um die Probleme bei der Vorklassifizierung von unzähligen Daten erfolgreich zu bewältigen. Im Bereich der Bioinformatik in dem Hochgeschwindigkeitskameras eingesetzt werden, um die Zellbilder (Abbildung 10, als Beispiel das GFP-markierte Zytokeratin) zu analysieren, eröffnen sich viele neue Möglichkeiten. Bisher wurden spezielle aufwändig produzierte Skripte verwendet, um die Bilder zu klassifizieren. Die Klassifizierungsmethoden werden für je ein bestimmtes Problem entwickelt.

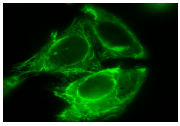

Abb. 10: siehe [10]

Das Ziel besteht darin, die Klassifizierung zu automatisieren. Die Idee geht davon aus, dass anhand von wenigen Beispielen in den Bilddaten, die man selbstständig selektiert hat und die vom Experten begutachtet und somit klassifiziert werden, ein Modell erstellt werden kann, welches anschließend fähig sein wird, den Rest der Bilddaten zu klassifizieren.

Bei diesem Projekt existieren zu Beginn des Prozesses noch keine klassifizierten Trainingsdaten, aus diesem Grund wird ein Modell benötigt, das die Verbindungen zwischen dem Konzept des unüberwachten und des überwachten Lernens herstellt.

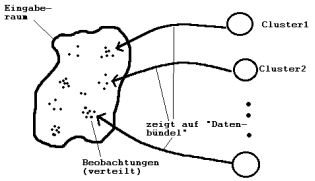

Abb. 11: siehe [11]

Unter einem unüberwachten Lernen („unsupervised learning“) (Abbildung 11 als eine mögliche Darstellungsweise) versteht man im Allgemeinen, dass es keinen externen Lehrer gibt, dieses Lernparadigma wird auch als „self-organized learning“ bereichnet. Das Netz versucht hierbei ohne eine äußere Beeinflussung die präsentierten Daten in Ähnlichkeitsklassen aufzuteilen [9].

Beim überwachten Lernen („supervised learning“) (Abbildung 12 als eine mögliche Darstellungsweise) wird durch einen externen Lehrer dem Netz zu jeder Eingabe die korrekte Ausgabe oder der Unterschied der korrekten zur tatsächlichen Ausgabe gegeben. Durch den vorliegenden Unterschied kann dann das Netz über eine Lernregel modifiziert werden. Die Trainingsdaten, die aus Paaren von Ein- und Ausgabedaten bestehen, müssen jedoch bei diesem Ansatz vorhanden sein. Es werden spezielle Schritte für alle Paare von Ein- und Ausgabemustern durchlaufen. Je ein Input-Wert hat somit je einen Output-Wert und bildet ein Datenpaar [10].

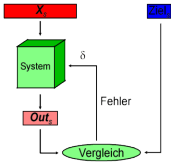

Abb. 12: siehe [12]

Ebenfalls zu beachten ist bei diesem Vorgang, dass ein Vergleich mit den Erwartungswerten geschieht und einen Fehler von der Größe δ liefert. Dieser entscheidet über die Notwendigkeit von weiteren Anpassungszyklen. Das Ziel besteht darin, die Ausgabewerte des neuronalen Netzes möglichst gut mit den Erwartungswerten Y übereinstimmend zu machen [11].

Zweckmässig wird in dem Paper ein neues aktives und adaptives Cluster- und Klassifikationsschema vorgestellt, welches mit einem anfänglichen Fuzzy-c-means Clustering (FCM-Algorithmus mit Noise-Erkennung) für das anfängliche Clustering (Abbildung 13) beginnt und dabei schrittweise die folgende Zielfunktion minimiert:

$$J_m = \sum_{i=1}^{|T|} \sum_{k=1}^{c} v_{i,k}^m d(\vec{w}_k, \vec{x}_i)^2 + \delta^2 \sum_{i=1}^{|T|} \left(1 - \sum_{k=1}^{c} v_{i,k}\right)^2$$

J_m wird unter folgender Randbedingung minimiert:

$$\forall i : 0 \leq \sum_{k=1}^{c-1} v_{i,k} \leq 1$$

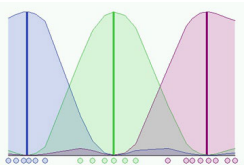

Abb. 13: siehe [13]

Auf den FCM-Algorithmus wird beispielsweise in dem Paper von James C. Bezdek, Robert Ehrlich und William Full detaillierter eingegangen. Dabei wird eine FORTRAN-IV Kodierung des Fuzzy-c-means (FCM) Programms übermittelt.

Das FCM Programm hat zum Beispiel eine wichtige Relevanz für eine große Auswahl an geostatistischen Datenanalyseproblemen. Durch dieses Programm werden Fuzzy-Teilungen und Prototypen für jegliche numerische Daten generiert.

Diese Fuzzy-Teilungen werden in der Hinsicht als sinnvoll erachtet, dass sie für die Bestätigung von bekannten Unterstrukturen in einem unerforschten Gebiet nützlich sind.

Das Clustering-Kriterium wurde benutzt, um Teilmengen in einer verallgemeinerten „Zielfunktion der kleinsten Quadrate“ zu bilden.

Die Merkmale dieses Programms beinhalten die Wahl zwischen drei Regeln (euklidisch, diagonal oder Mahalonobis). Außerdem gibt es einen einstellbaren Gewichtungsfaktor, der im Wesentlichen die Empfindlichkeit dem Rauschen gegenüber kontrolliert. Letztendlich werden auch verschiedene Mengen an Clustern akzeptiert und es existieren auch einige Outputs, die die Messungen zur Clustergültigkeit enthalten [13].

Auf diese Art und Weise werden große Datenmengen anfänglich unüberwacht gruppiert und die ersten Muster (Prototypen) für eine manuelle Klassifizierung werden selektiert. Diese Prototypen müssen im nächsten Schritt an das Modell angepasst werden. Das eingesetzte Verfahren trägt den Namen LVQ (Lernende Vektor Quantitierung). Dieser Algorithmus verbirgt sich hinter dem LVQ-Verfahren:

1: Wähle R initiale Prototypen für jede Klasse:
$m_1(k), m_2(k), \ldots, m_R(k), k = 1, 2, \ldots, K$.
2: **repeat**
3: Wähle zufällig ein Trainingsbeispiel $\vec{x}_i$. Sei $m_j(k)$ der nächste Prototyp zu $\vec{x}_i$. Sei g_i der Klassenname von $\vec{x}_i$ und g_j der Klassenname vom Prototyp.
4: **if** $g_i = g_j$ **then** // gleiche Klasse
5: Bewege den Prototyp in Richtung des Trainingsbeispiels:
$m_j(k) \leftarrow m_j(k) + \epsilon(\vec{x}_i - m_j(k))$, wobei ϵ die Lernrate ist.
6: **end if**
7: **if** $g_i \neq g_j$ **then** // verschiedene Klassen
8: Bewege den Prototyp in entgegengesetzer Richtung des Trainingsbeispiels:
$m_j(k) \leftarrow m_j(k) - w\epsilon(\vec{x}_i - m_j(k))$
9: **end if**
10: Verringere die Lernrate ϵ
11: **until** Maximale Anzahl Iterationen erreicht oder keine signifikante Änderung der Prototypen.

Es sollen nützliche und sinnvolle Beispiele ausgewählt werden, um den Aufwand gering zu halten.

Insgesamt wird in dem Paper ein Framework präsentiert, in dem die Analyse von Zellbildern erfolgen könnte [12].

Die Klassifizierung des Datenbestandes erfolgt dadurch, dass die Cluster-Prototypen entsprechend gekennzeichnet / markiert werden und indem an alle Datenpunkte die Markierung des nächsten Prototypen angebracht wird. Die Prototypen werden basierend auf den Markierungen der ausgewählten Beispiele an den Grenzen zwischen Clustern und den markierten Beispielen innerhalb der Cluster verschoben [25].

E. „adaptive classification“ von zweidimensionalen Gelelektrophoretischen Punktmustern mittels neuronaler Netze und mit Hilfe einer Cluster-Analyse

Aus dem Paper von Jiri Vohradsky geht hervor, dass die Deutung und die Interpretation der zweidimensionalen Gelelektrophoretischen Punktmuster beziehungsweise der Punktprofile durch statistische Programme und durch Programme für maschinelles Lernen erleichtert werden kann.

Unter einer Gelelektrophorese (Abbildung 14 als eine Möglichkeit wie man eine Gelelektrophorese durchführt) wird dabei ein Elektrophorese-Verfahren angenommen, bei dem ein Gel als Trägermedium benutzt wird. Das Verfahren dient zur Auftrennung von Gemischen [15].

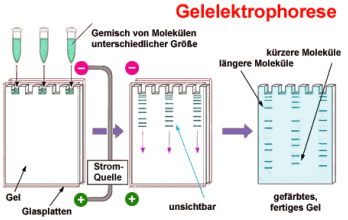

Abb. 14: siehe [14]

Zunächst muss eine Trennung der Moleküle stattfinden, erst dann wird das Gel mit dem entsprechenden Farbstoff (beispielsweise mit Ultramarinblau) angefärbt, um die in Banden getrennten unterschiedlichen großen Moleküle sichtbar zu machen [17].

Die Mischung aus zu trennenden Molekülen wandert unter dem Einfluss eines elektrischen Feldes durch ein Gel, das sich in einer ionischen Pufferlösung befindet [16].

Im weiteren Verlauf werden zwei unterschiedliche Ansätze zur Klassifizierung von Punktprofilen vorgestellt. Im Zusammenhang damit müssen auch die Cluster-Analysen durchgeführt werden und es werden auch die neuronalen Netze (Abbildung 15) genauer diskutiert.

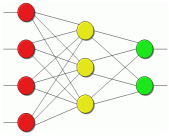

Abb. 15: siehe [15]

Schließlich werden bei dieser Vorgehensweise geeignete neuronale Netze für die beiden verschiedenen Modellmuster entworfen. Auch ein Algorithmus für die Ausbildung des Netzes zum Zweck der Klassifizierung wurde implementiert. Das Ergebnis der Untersuchungen verdeutlicht, dass die Leistungsfähigkeit der neuronalen Netze im Vergleich zu den Clustern und zu der Analyse der Hauptbestandteile höher ist.

Würden beide Ansätze miteinander kombiniert werden, sodass ein einziger Prozess entsteht, würde dies dazu führen, dass die Zuverlässigkeit und die Geschwindigkeit der Klassifizierung zunehmen könnten.

Auf künstlichem Weg erzeugte Beispielpunktemengen mit zusätzlichem Rauschen versehen, können bei dieser Methode für die Netzwerkausbildung verwendet werden.

Die Analyse wurde bei einer „Streptomyces coelicor“ (Abbildung 16) zweidimensionalen Gel-Datenbank angewendet [14].

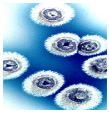

Abb. 16: siehe [16]

F. „adaptive classification“ für BCIs („brain computer interfaces“)

Das Paper von Pradeep Shenoy, Matthias Krauledat, Benjamin Blankertz, Rajesh P N Rao und Klaus-Robert Müller beschäftigt sich mit der „adaptive classification“ für BCIs.

Bereits in einem Paper von G. Schalk, D. J. McFarland und anderen, das in Biomedical Engineering erschien, wird ein „Allzweck-BCI-System“ - BCI2000- empfohlen.

Zwar haben schon viele Labore damit begonnen, BCI-Systeme zu entwickeln, die dabei helfen, Menschen mit schweren motorischen Störungen Kommunikations- und Steuerungsmöglichkeiten zu geben, jedoch hängen der weitere Fortschritt und auch die Umsetzung der praktischen Anwendungen beispielsweise von systematischen Auswertungen und von dem Vergleich der verschiedenen Gehirnsignale, von den weiteren Aufnahmemethoden, von den Verarbeitungsalgorithmen oder von den Ausgabeformaten ab. Dennoch muss einem bewusst bleiben, dass ein typisches BCI-System für eine bestimmte BCI-Methode geeignet ist. Das BCI2000 (Abbildung 17) wurde daher als eine spezielle Forschungs- und Entwicklungsplattform eingeführt.

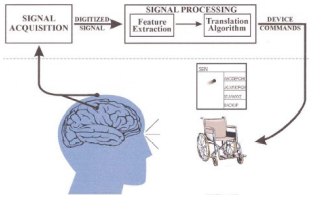

Abb. 17: siehe [17]

Im ersten Schritt werden die Gehirnsignale aufgenommen, in digitalisierter Form erfolgt die Weiterleitung. Als Nächstes muss eine Signalverarbeitung stattfinden, dabei geschieht eine Merkmalsextraktion und ein geeigneter Algorithmus muss ebenfalls gefunden werden, um die passenden Befehle an die Geräte weiterzuleiten.

Dieses kostenfreie System für Forschungs- und Lehrzwecke ist dazu in der Lage, Signalverarbeitungsmethoden, Ausgabevorrichtungen oder auch Protokolle zu integrieren. Es funktioniert gut in einer Online-Inbetriebnahme und es erfüllt die Echtzeit-Anforderungen [18].

Nicht-Stationaritäten werden bei den EEG-Signalen (Abbildung 18) als allgegenwärtig betrachtet. Diese Nicht-Stationäritäten sind besonders offensichtlich bei der Nutzung von EEG-basierten „Gehirn-Computer-Schnittstellen“ (BCIs). So erkennt man beispielsweise die Unterschiede zwischen der anfänglichen Kalibriermessung und der Online-Inbetriebnahme eines BCI. Ein anderer Fakt wäre, dass die Nicht-Stationaritäten ebenfalls durch die Veränderungen im Gehirn während des Experiments, zum Beispiel auf Grund von Müdigkeit, von neuen Aufgabenstellungen und ähnlichen Ursachen, verursacht werden können.

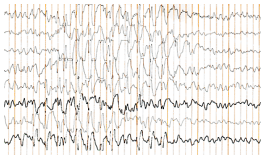

Abb. 18: siehe [18]

Im nachfolgenden Paper wird zum ersten Mal ein solcher systematischer Nachweis der statistischen Unterschiede der aufgezeichneten Daten während Offline- und Onlinesitzungen dokumentiert.

Darüber hinaus werden neue Verfahren zur Untersuchung und zur Visualisierung von Datenverteilungen vorgeschlagen. Diese Verfahren werden als sehr nützlich für die Analysen von Nicht-Stationaritäten betrachtet.

Laut der Studie können sich die Gehirnsignale wesentlich bei den Offline-Kalibierungssitzungen im Gegensatz zu den Online-Sitzungen verändern und dies auch innerhalb einer einzelnen Sitzung. Zusätzlich zu der allgemeinen Charakterisierung der Signale werden mehrere adaptive Klassifikationsschemata vorgeschlagen. Ihre Leistungsfähigkeit wird anhand der Daten, die man während der Online-Experimente aufgenommen hat, studiert.

Ein Ergebnis der Studie sagt aus, dass überraschenderweise einfache adaptive Methoden in der Kombination mit einem Offline-Funktion-Selektionsschema die BCI-Leistung stark erhöhen könnten.

Folgende „adaptive classification" Methoden werden untersucht:

- ORIG: Bei ORIG handelt es sich um einen unveränderten Klassifizierer, der anhand der Daten aus dem Offline-Szenario geschult wird. Er dient gewissermaßen als eine Basis.
- REBIAS: Es wird die Dauerleistung des unveränderten Klassifizierers genutzt und seine Ausgabedaten um einen Betrag verändert, der den Fehler auf den markierten Rückgabedaten minimieren sollte.
- RETRAIN: Man nutzt die Eigenschaften, die man bereits beim Offline-Szenario gewählt hat, jedoch wird der LDA-Klassifizierer (Linear Discriminant Analysis) neu umgeschult, um eine Hyperebene auszusuchen, die den Fehler auf den markierten Rückgabedaten minimieren sollte.

Eine Hyperebene ist die Verallgemeinerung einer normalen Ebene im 3D-Raum auf ein mathematisches Objekt im n-dimensionalen Raum. Eine dreidimensionale Hyperebene ist somit ein Teil des vierdimensionalen Raums [19].

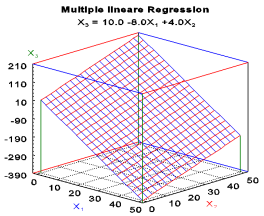

Abb. 19: siehe [19]

In dem oben abgebildeten Beispiel (Abbildung 19) wird eine 2D Hyperebene in einem 3D Variablenraum dargestellt [20].

- RESCP: Die Offline-Trainingsdaten werden komplett nicht berücksichtigt und es wird eine CSP-Merkmalsextraktion (CSP - Constraint Satisfaction Problem) and Klassifizierungstraining ausschließlich anhand der Rückgabedaten durchgeführt [21].

G. „adaptive classification" multispektraler Daten

In dem Paper von Hongzhi Zhao und Mita D. Desai geht es um eine neue „adaptive classification" Methode für multispektrale Fernerkundungsdaten, die auf lokalen Besonderheiten basiert.

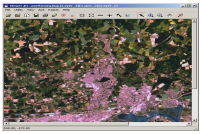

Abb. 20: siehe [20]

Im Allgemeinen wird für eine Multispektral-Klassifizierung ein Multispektraldatensatz (Abbildung 20) benötigt. Es könnte sich dabei um einen Datensatz (Satellitenbild und Ähnliches) mit mehreren Kanälen handeln, also neben Panchromatisch auch noch RGB und Infrarot verschiedener Spektralbereiche [22].

Typische Klassifizierungsmethoden, die auf globalen Merkmalen basieren, neigen dazu, im Laufe der Zeit nachzulassen, da alle Klassen oft in die gleiche Richtung projiziert werden. Bei den üblichen Methoden wird die Annahme gemacht, dass diese Trennbarkeit der Klassen für alle Richtungen konstant bleibt, was allerdings nicht immer so zutrifft.

Die neue Methode, die in dem Paper vorgestellt wird, überwindet diesen Nachteil, indem Merkmale ausgewählt werden, die die maximale Klassentrennbarkeit erlauben, dadurch, dass die Basis die lokalen Informationen über die Klassen und nicht die globalen Informationen bilden.

Es wird zuerst eine Projektionsmatrix für jede der Klassen gesucht, auf der Grundlage basierend, dass alle Trainingsbeispiele gut voneinander getrennt werden.

Die S-Matrix (Streumatrix) S_w innerhalb einer Klasse wird dabei definiert als:

$$S_w = \sum_{i=1}^{M} P(i) \sum_i$$

Die S-Matrix (Streumatrix) S_B zwischen den Klassen wird berechnet durch:

$$S_B = \sum_{i=1}^{M} P(i)(\mu_i - \mu)(\mu_i - \mu)'$$

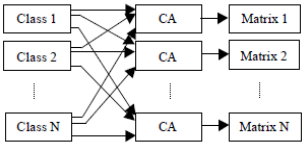

Abb. 21: siehe [21]

Das Bild (Abbildung 21) verdeutlicht, dass verschiedene Klassen ihre eigenen Projektionsmatrizen besitzen.

CA steht für eine kanonische Analyse / Korrelation, also für ein multivariantes Verfahren der Abhängigkeitsanalyse, um die Beziehung zwischen zwei Variablensätzen zu untersuchen [24].

Jeder Eingabevektor wird dann in einen anderen Raum durch jede Projektionsmatrix linear transformiert. In den transformierten Räumen kann eine Klassifizierung oder eine Art „Markierung" für die verschiedenen Klassen mit Hilfe der Maximum- Likelihood-Classification (MLC) stattfinden.

Dieser Vorgang (Abbildung 22) wird durch folgende Abbildung dargestellt:

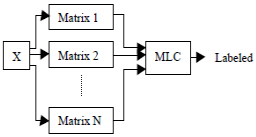

Abb. 22: siehe [22]

X stellt dabei einen Eingabevektor dar und wird klassifiziert beziehungsweise erhält eine Art „Markierung". Diese Prozedur wird auf alle Eingabevektoren angewendet.

Insgesamt besteht also die Vorgehensweise aus zwei Schritten: der Merkmalsextraktion, indem man Projektionsmatrizen für Klassen verwendet und MLC.

Um den Rechenaufwand gering zu halten, wird eine adaptive Dimensionsreduktion durchgeführt. Gute Ergebnisse wurden bei den Experimenten am Kennedy Space Center (KSC) erzielt [23].

H. „adaptive classification" von EEG-Funktionen bei einem spärlichen Feedback

Das Paper von D. R. Lowne, I. Rezek und S. J. Roberts beschäftigt sich mit der „adaptive classification", da es ein wichtiges Problem auf dem Gebiet der Online-EEG-Analyse darstellt. Einfache lineare Systeme werden als relativ wirkungslos eingestuft, da man die bekannte Nicht-Stationarität des EEG kennt. Darüber hinaus sind die Klassenbezeichnungen nur selten verfügbar. Es wird ein Algorithmus für die adaptive nichtlineare „zwei-Klassen-Unterscheidung" präsentiert, die die nicht-stationäre logistische Regression auf ein Umfeld, in dem Feedback nur spärlich vorhanden ist, erweitert:

Die logistische Regression ist auch unter dem Begriff des Logit-Modells bekannt. Es handelt sich um Regressionsanalysen zur Modellierung von diskreten abhängigen Variablen [26]. Die Abbildung 23 stellt ein Beispiel für eine logistische Funktion dar:

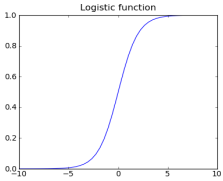

Abb. 23: siehe [23]

Zunächst werden die „Gaussian kernels" angewendet, um eine „Hilbert-Expansion" zu erschaffen, die sich als eine gute Methode erwiesen hat, um die Datentrennbarkeit zu gewährleisten und zu erhöhen. Der „Gaussian kernel" wird in 1-D, 2-D und N-D jeweils definiert als [27]:

$$G_{1D}(x,\sigma) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{x^2}{2\sigma^2}},\ G_{2D}(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}},\ G_{ND}(\vec{x},\sigma) = \frac{1}{(\sqrt{2\pi}\,\sigma)^N} e^{-\frac{\|\vec{x}\|^2}{2\sigma^2}}$$

Anschließend wird der Bayes'sche Ansatz angewendet, um einen zeitabhängigen Satz an Gewichten im Rahmen eines linearen dynamischen Systems zu entwickeln. Es wird ein geeignetes adaptives System in der Gegenwart von nicht-stationären Daten geschaffen. Man spricht von zwei entscheidenden Vorteilen des Bayes'schen Ansatzes:

Erstens werden feste Lernraten vermieden und stattdessen werden diese auf der Grundlage der Abweichung zwischen den vorhergesagten und den realen Unsicherheiten bei der Klassifizierung angepasst. Zweitens erlaubt die probalistische Natur des Systems einfache Behandlung der fehlenden Klassenbezeichnungen, welche lediglich auf der Basis der aktualisierten Klassenwahrscheinlichkeiten festgelegt werden, um fortfahren zu können, wie zuvor mit einer Art von Ziel.

Drei Testpersonen wurden darum gebeten, eine Streckung des Handgelenks (Abbildung 24) als eine Übung auf einen visuellen Hinweis hin durchzuführen.

Abb. 24: siehe [24]

Der Klassifizierer kannte diese Hinweise nicht und erhielt Feedback von den EMG-Daten (d.h. eine wahre Ausgabeklasse) in nur etwa 20% der Proben.

Die Elektromyografie (EMG) (Abbildung 25) ist eine Untersuchungsmethode der medizinischen Diagnostik / Neurophysiologie, welche die natürlicherweise auftretende elektrische Spannung in einem Muskel misst. Mit dieser Methode kann festgestellt werden, ob eine Erkrankung des Muskels vorliegt [29].

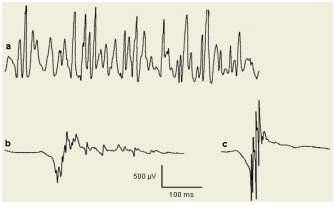

Abb. 25: siehe [25]

Die obere Abbildung zeigt unter a) ein normales Aktivitätsmuster (Musculus biceps brachii), b) und c) sollen schwer deformierte Aktionspotentiale bei einer Nervenläsion verbildlichen.

Es werden die spärlich-markierten EEG-Daten der Probanden analysiert und die vorhergesagten Markierungen mit denen von den EMG-Signalen verglichen.

X^2 – Hypothesentests zu den Verwechslungsmatrizen erlauben es zu der Schlussfolgerung zu gelangen, dass trotz eines Mangels an Zielvorgaben, der Algorithmus eine gute Klassifizierungsleistungsfähigkeit (ca. 81.9%) bei einer großen unmarkierten Datenmenge mit sich bringt.

Eine Verwechslungsmatrix $V_{i,j}$ gibt an, wie oft ein Muster der Klasse *i* einer falschen Klasse *j* zugeordnet wird. Die jeweilige Diagonale (Abbildung 26) enthält die richtigen Klassifikationen [30].

In der Abbildung 26 erkennt man beispielsweise eine Verwechslungsmatrix bei einer Klassifikation handgeschriebener Ziffern (1000 Trainingsmuster je Klasse).

	Classified as										Total
Digit	0	1	2	3	4	5	6	7	8	9	
0	964	2	7	7	4	1	8	1	11	0	1,000
1	0	961	1	1	3	1	7	2	23	1	1,000
2	9	6	897	17	19	2	10	14	26	0	1,000
3	4	5	6	952	1	9	3	3	12	5	1,000
4	1	7	2	1	942	0	8	2	9	33	1,000
5	5	2	4	31	2	927	13	1	10	5	1,000
6	8	10	1	1	6	2	970	0	7	0	1,000
7	4	11	2	8	2	0	0	917	5	51	1,000
8	5	13	2	16	8	16	2	5	924	9	1,000
9	3	4	0	7	9	2	0	11	15	949	1,000
Total	998	1021	992	1041	996	960	1011	956	1,042	1,053	10,000

Abb. 26: siehe [26]

Anfängliche Experimente zeigen, dass diese adaptive, nicht-lineare Klassifizierungsmethode besonders geeignet für nicht-stationäre Daten ist. Außerdem wird verdeutlicht, dass wenig Feedback nicht inkongruent mit „adaptive classification“ ist. Weitere Studien sollen eine Auswertung des Algorithmus in einer Online-Umgebung bei einer größeren Menge an Probanden enthalten. Zusätzlich sind weitere Untersuchungen über die Auswirkungen der Annahme der Klassenbeschriftungen größerer Sequenzen unbeschrifteter Daten oder einer größeren Klassenüberschneidung geplant [28].

III. Grundlagen: Klassifizierung und Objekterkennung – „klassische“ bekannte Klassifizierung

A. *verwendete Methoden*

Bis jetzt wurden die Daten üblicherweise an bereits vorhandene Modelle angepasst. Die Methoden, auf die nun im Folgenden eingegangen wird, zeichnen sich durch folgende Eigenschaften aus:

Man spricht von nicht komplizierten, modellfreien Methoden, die zum Zweck der Klassifizierung und der Objekterkennung angewendet werden.

Die Untergliederung in Klassen lässt sich jedoch dabei nicht immer nachvollziehen.

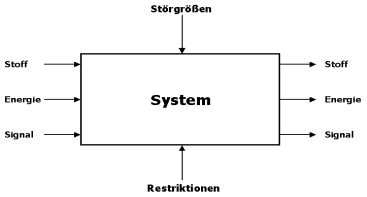

Abb. 27: siehe [27]

Als eine Black-Box-Anwendung (Abbildung 27 – die kybernetische Black-Box als Beispiel) ist diese sehr effektiv und liefert gute Ergebnisse. Unter einer Black-Box versteht man im Allgemeinen ein System, bei welchem nur das äußere Verhalten unter der Vernachlässigung des inneren Aufbaus eine Rolle spielt. Bei einer Black-Box handelt es sich um eine Einheit von der die Übertragungseigenschaften zwar bekannt sind, nicht aber deren innere Funktionsweise [31].

Die Black-Box dient als ein Werkzeug der Systemtheorie. Sowohl der Kern der Aufgabe, als auch alle Eingangs- und Ausgangsgrößen sowie die Rahmenbedingungen werden in der Black-Box definiert [32].

B. *Definitionen*

- Trainingsdaten: N Paare $(x_1,g_1),\ldots,(x_N, g_N)$
- x_i: Merkmal, für i ∈ {1,..., N}

- g_i: Bezeichnung für eine Klasse, für $g_i \in \{1,..., K\}$ und für $i \in \{1,..., N\}$
- Prototyp: ein Paar (x_k, g_k), dabei ist k in der Regel $\notin \{1,..., N\}$

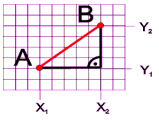

Abb. 28: siehe [28]

Wird ein Abstand in einem Merkmalsraum als ein Abstand zwischen standardisierten Merkmalen beurteilt, kann es der euklidische Abstand (Abbildung 28) sein, bei dem der Erwartungswert 0 und die Varianz 1 betragen. Die euklidische Distanz lässt sich dabei nach folgender Formel berechnen [33]:

$$|\vec{x}-\vec{y}|=\sqrt{\sum_{i=1}^{n}\left(xi-y_i\right)^2}$$

Die Summe aus der Anzahl der Merkmale x_i entspricht der Dimension des Merkmalsraums. Die Klassifizierung ist hierbei eine Strukturierung eines Merkmalsraums in Gebiete, denen aus Anwendungssicht eine bestimmte Bedeutung zukommt [34].

In der folgenden Abbildung (Abbildung 29) wird ein scharfer Klassifikator im 2D-Merkmalsraum eingesetzt:

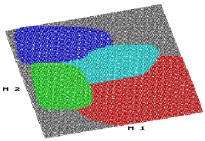

Abb. 29: siehe [29]

IV. Prototypmethoden

Prototypmethoden zeichnen sich durch die im Merkmalsraum befindlichen Trainingsdaten aus.

Aufgrund des Vorhandenseins an Prototypen ist es sehr einfach, auch neu ankommende Daten zügig einer Klassifizierung zu unterziehen.

Mit Hilfe der Prototypen findet eine Anordnung der Bayes'schen Entscheidungsgrenzen (Abbildung 30) statt. Diese werden als gestrichelte Linien in verschiedensten Beispielen gekennzeichnet. Bei der Theorie der Entscheidungsgrenzen („decision bound theory“) geht man von der Grundannahme aus, dass eine perzeptuelle (wahrnehmende) Identifikation eines bestimmten Stimulus ähnliche Prozesse wie eine Kategorisierung von Stimuli erfordert. Ein Beobachter repräsentiert dabei Stimuli in einem geometrischen Raum. Eine diskriminative (unterscheidende / trennende) Funktion spaltet den Raum in verschiedene Funktionen auf. Jede Teilregion ist mit einer Reaktion assoziert. Die Grenzen der Regionen sind die Entscheidungsgrenzen [35].

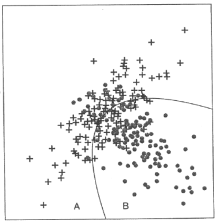

Abb. 30: siehe [30]

A. K-means Clustering

Beim K-means Clustering geht man grundlegend davon aus, dass eine Menge an Daten durch Häufungen klassifiziert wird.

Das bedeutet, dass man für die vorliegende Datenmenge eine feststehende Anzahl an Prototypen, bei denen es sich um sogenannte Häufungszentren handelt, festzulegen hat.

Das hierbei verfolgte Ziel besteht darin, iterativ den Abstand zwischen einem Merkmal und einem Häufungszentrum zu verkleinern und so minimal wie möglich zu halten.

K-means Clustering (Abbildung 31) arbeitet auf skalaren Daten. Die Anzahl der Cluster k wird jeweils vorgegeben. K-means Clustering erzeugt k Cluster, indem es k Zentrumspunkte findet.

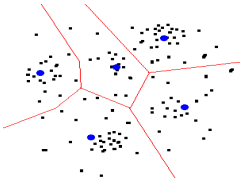

Abb. 31: siehe [31]

In der Abbildung 31 wird ein geclustertes Punktdiagramm dargestellt. Bei den schwarzen Punkten handelt es sich um Datenpunkte. Die roten Linien symbolisieren die Trennungen, die durch den K-means Algorithmus erzeugt werden. Die blauen Punkte sind dabei die Zentrumspunkte [36].

Der K-means Algorithmus läuft folgendermaßen ab:

```
proc K-MEANS(S, k)
    Z = {z_1, ..., z_k} := k zufällig gewählte Punkte aus S
    while Qualität wird besser
        C_i := {s ∈ S | i = argmin_{i=1,...,k} dist(s, z_i)} für i = 1, ..., k
        z_i := z(C_i) für i = 1, ..., k
        end
    return{C_1, ..., C_k}
```

z(C) ist in diesem Fall:

$$z(C) := \frac{1}{|C|} \sum_{\mathbf{c} \in C} \mathbf{c}$$

Die Zuordnung der Instanzen zu den einzelnen Klassen erfolgt durch die Bestimmung des ähnlichsten Zentrumspunktes.

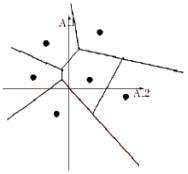

Abb. 32: siehe [32]

In der Abbildung 32 wird ein Voronoi-Diagramm für Nachbarschaftsbeziehungen aufgezeigt [37].

Man unterscheidet zwei entscheidende Fälle: Die unmarkierte Datenmenge und die markierte Datenmenge.

a) Fall 1: die unmarkierte Datenmenge

Dazugehörige Iterationsschritte beinhalten:

- Zu Beginn muss eine gewollte Anzahl an Startzentren per Zufall bestimmt werden.
- Als Nächstes wird durch eine Konstruktion eine Häufung durch eine Menge an Punkten, die sehr nah zum Zentrum hin liegen, geschaffen.

Unter dem Begriff der Punktmenge versteht man eine Menge an Punkten, die alle eine ganz bestimmte Eigenschaft haben [38].

In der Abbildung 33 sieht man eine erst mal ungeclusterte Punktmenge:

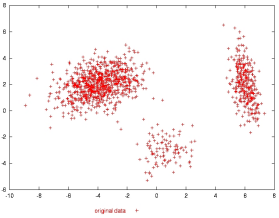

Abb. 33: siehe [33]

- Am Ende muss nur noch das neue Häufungszentrum berechnet werden.

Die letzten beiden Schritte werden bis zu einer Konvergenz hin wiederholt. Eine Divergenz würde nur vorliegen, wenn ein Punkt zur gleichen Zeit den gleichen Abstand zu zwei Zentren hätte.

b) Fall 2: die markierte Datenmenge

Dazugehörige Iterationsschritte beinhalten:

- Anfänglich muss K-means Clustering für jede der K Klassen angewendet werden. Pro Klasse existieren R Prototypen.
- Insgesamt stehen also K*R Prototypen zur Verfügung. Im Folgenden ordnet man jedem Prototypen ein Merkmal x_k und und eine Klassenbezeichnung g_k zu.
- Im Anschluss werden neu ankommende Daten der Klasse des nächsten Prototyps zugewiesen.

Bei der Abbildung 34 handelt es sich um ein simuliertes Beispiel mit drei Klassen $g_i \in$ {rot, grün, blau} und R=5 Prototypen pro Klasse. Die gestrichelte Linie ist in diesem Fall die Bayes'sche Entscheidungsgrenze.

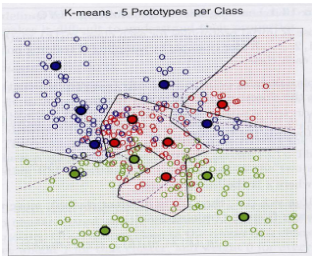

Abb. 34: siehe [34]

c) Beurteilung der Methode K-means Clustering

Bei der Methode K-means Clustering handelt es sich um eine relativ leicht verständliche Möglichkeit, um die Daten zu klassifizieren.

Die Bayes'schen Entscheidungsgrenzen sind jedoch nicht glatt. Deshalb sind auch falsche Klassifikationen, insbesondere an den Klassenrändern, nicht auszuschließen, was als nachteilig angesehen werden muss.

In Abhängigkeit davon, welche Startzentren und wie viele Startzentren man wählt, können die Ergebnisse variieren.

In einem Paper von J. A. Hartigan und M. A. Wong wird ein K-means Clustering Algorithmus beschrieben. Dieser Algorithmus beispielsweise erfordert als eine Eingabe eine Matrix von *M* Punkten in *N* Dimensionen und eine Matrix von *K* anfänglichen Cluster-Zentren in *N* Dimensionen. Die Anzahl der Punkte im Cluster *L* wir gekennzeichnet durch *NC(L)* und *D(I,L)* wäre dabei der euklidische Abstand zwischen einem Punkt *I* und dem Cluster *L*. Die allgemeine Prozedur besteht darin, nach einer „*K*-Trennung" mit einer lokal optimalen Quadratsumme innerhalb des Clusters zu suchen, indem man Punkte von einem Cluster in ein anderes bewegt [39].

Man unterscheidet generell zwischen hierarchischen und nicht-hierarchischen Clusterverfahren, nach denen vorgegangen werden könnte. Die ersteren fassen auf verschiedenen Stufen nahe beieinanderliegende Objekte zu neuen zusammen. Auf die nicht-hierarchischen Verfahren wird öfter Bezug genommen [40].

B. *Gaussian Mixture*

Bei Gaussian Mixture geht man von einer Annahme aus, dass die Möglichkeit besteht, jede Häufung durch eine parametrische Verteilung, wie beispielsweise durch eine Normalverteilung anzugeben. Die Normal- oder Gaußverteilung ist ein wichtiger Typ kontinuierlicher Wahrscheinlichkeitsverteilungen. Ihre Wahrscheinlichkeitskurve wird auch Gauß-Funktion genannt.

Der hier gültige Grenzwertsatz besagt, dass eine Summe von *n* unabhängigen, identisch verteilten Zufallsvariablen im Grenzwert $n \to \infty$ normalverteilt ist [41].

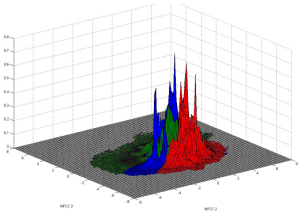

Abb. 35: siehe [35]

In der Abbildung 35 sieht man ein Gaussian Mixture Modell in diesem Fall für Klarinette (blau), Saxophon (grün) und Trompete (rot). Signale mit Merkmalen, die in ein koloriertes Gebiet fallen, werden als ein bestimmtes Instrument klassifiziert [42].

Gebildet wird eine Datenmenge durch eine Zusammensetzung / durch einen Mix einzelner Verteilungen. Die Dichten der Häufungen unterscheiden sich hierbei voneinander. Auch hier müssen die Häufungszentren herausgefunden werden.

Das Modell für Gaussian Mixture kann folgendermaßen beschrieben werden:

Zunächst muss angenommen werden, dass K Häufungen vorliegen.

Durch bestehende Normalverteilungen (Abbildung 36) mit den Parametern μ_k, Σ_k wird jede einzelne Häufung erzeugt.

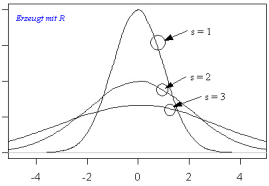

Abb. 36: siehe [36]

Die Abbildung 36 verdeutlicht den Zusammenhang zwischen der Normalverteilung und den verschiedenen Standardabweichungen *s*.

Daten werden als Vektoren x, $x \in R^N$ definiert.

Man hat *n* konkrete Daten von $x_1,\ldots, x_n$.

Die Dichte der Häufung *k* kann angegeben werden als: $f_k(x) = \phi(x; \mu_k, \Sigma_k)$.

Die “a priori” Wahrscheinlichkeit, also eine Wahrscheinlichkeit, die aufgrund von Vorwissen gewonnen wird, spielt insbesondere beim Bayes'schen Wahrscheinlichkeitsbegriff eine wichtige Rolle. Die “a priori” Wahrscheinlichkeit von k beträgt hier α_k, dabei gilt folgender Zusammenhang: $\sum_{k=1}^{K} \alpha_k = 1$. „A priori“ bedeutet auf Latein so viel wie: von dem, was vorher kommt. Im Zusammenhang mit dem Satz von Bayes werden die Wahrscheinlichkeiten $P(A_i)$ auch als „a priori“ Wahrscheinlichkeiten und $P(A_i|B)$ als „a posteriori“ Wahrscheinlichkeiten bezeichnet, weil $P(A_i)$ das Eintreten von A_i vor Kenntnis des Ereignisses B und das Eintreten dieses Ereignisses nach Kenntnis von B bewertet [43].

Weiterhin kann ebenfalls die Dichte der Gesamtmischung durch den folgenden Ausdruck berechnet werden: $f(x) = \sum_{k=1}^{K} \alpha_k \cdot f_k(x)$.

Dazugehörige Iterationsschritte des EM-Algorithmus beinhalten:

- Es beginnt alles mit einer Initialisierung.
- Darauffolgend geschieht ein Schätzschritt bei einer Iteration von p: Es muss jeder Klasse eine bestimmte Gewichtung zugeordnet werden. Das heißt, dass “a posteriori” Wahrscheinlichkeiten ausgerechnet werden. Unter “a posteriori” Wahrscheinlichkeiten versteht man im Allgemeinen Wahrscheinlichkeiten, die auf der Grundlage der Erfahrung ermittelt werden. Diese stellen einen Begriff aus der bayes'schen Statistik dar. Eine “a posteriori” Wahrscheinlichkeit gibt einen Wissensstand über einen unbekannten Umweltzustand θ nach der Beobachtung einer von θ abhängigen Zufallsgröße X an. Diese wird auch als eine statistische Wahrscheinlichkeit bezeichnet und es handelt sich um eine empirisch ermittelte Wahrscheinlichkeit. Die hier verwendete Formel lautet [44]:

$$p_{i,k} = \frac{\alpha_k^{(p)} \phi\left(x_i; \mu_k^{(p)}, \sum_k^{(p)}\right)}{\sum_{k=1}^{K} \alpha_k^{(p)} \phi\left(x_i; \mu_k^{(p)}, \sum_k^{(p)}\right)}, \quad i \in \{1 \ldots . n\}. k \in \{1 \ldots . K\}$$

- Zuletzt benötigt man nur noch einen sogenannten Maximierungsschritt. Bei diesem Schritt werden nun auch noch die “a priori” Wahrscheinlichkeiten sowie der Erwartungswert und die Kovarianzmatrix erneuert.

$$\alpha_k^{(p+1)} = \frac{\sum_{i=1}^{n} p_{i,k}}{n}$$

$$\mu_k^{(p+1)} = \frac{\sum_{i=1}^{n} p_{i,k} x_i}{\sum_{i=1}^{n} p_{i,k}}$$

$$\sum_{k=1}^{p+1} = \frac{\sum_{i=1}^{n} p_{i,k} \left(x_i - \mu_k^{(p+1)}\right)\left(x_i - \mu_k^{(p+1)}\right)^t}{\sum_{i=1}^{n} p_{i,k}}$$

Die letzten beiden Schritte müssen solange wiederholt werden, bis eine Konvergenz beim Gaussian Mixture (Abbildung 37 als ein Beispiel) vorliegt.

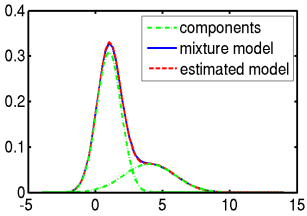

Abb. 37: siehe [37]

Folgende Vorteile werden dem EM-Algorithmus zugeschrieben:

- Der Algorithmus ist numerisch stabil, da er sich mit jeder Iteration dem Optimum nähert.
- Unter den meisten Voraussetzungen ist er global konvergent.
- Er ist einfach zu implementieren und braucht nicht viele Ressourcen.
- Außerdem liefert er Parameterschätzungen für Daten mit fehlenden Werten.

Die Nachteile des EM-Algorithmus wären:

- Dieser Algorithmus ist manchmal sehr langsam.
- Bei einigen Problemen sind die E- und / oder M-Steps schwierig zu bestimmen.
- Er liefert nicht automatisch eine Schätzung für die Kovarianzmatrix der Parameterschätzungen [45].

a) Beurteilung der Gaussian Mixture Methode:

Bei der Bewertung der Gaussian Mixture Methode fällt auf, dass diese Methode oft als eine “weiche” Methode präsentiert wird. Im Gegensatz dazu wird die vorher betrachtete K-means Clustering Methode als eine “harte” Methode beurteilt.

Prinzipiell gesehen kann durch die Gaussian Mixture Methode für jede Klasse die Merkmalsdichte ermittelt werden.

Außerdem berechnet man jedes Mal ganz glatte "a posteriori" Wahrscheinlichkeiten. Auf diese Art und Weise ist es realisierbar, für x eine Klassifizierung vorzunehmen.

In einem Paper von Zoran Zivkovic wird sich mit einem verbesserten „adaptive Gaussian Mixture"-Modell für eine Hintergrundsubtraktion beschäftigt. Eine Hintergrundsubtraktion (Abbildung 38 als ein Beispiel) ist eine alltägliche Bildverarbeitungsaufgabe.

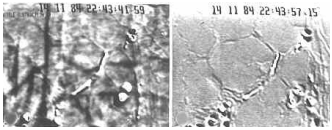

Abb. 38: siehe [38]

Es wird der übliche Pixel-Ebenen-Ansatz analysiert. Die Abbildung 39 verdeutlicht die Zusammenhänge zwischen Pixeln und Ebenen:

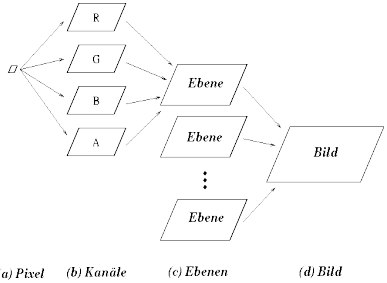

Abb. 39: siehe [39]

Eine Bildkomponentenhierarchie liegt allem zugrunde. Jede Ebene besteht aus einer Matrix von Pixeln und jedes Pixel ist aus einem R, G, B und Alphakanal [47].

Man versucht einen effizienten adaptiven Algorithmus zu entwickeln, indem man eine Gaussian Mixture Wahrscheinlichkeitsdichtefunktion einsetzt. Diese wird verwendet, um zu erkennen, welche Wertebereiche der Zufallsvariablen wahrscheinlicher sind als andere [48].

Rekursive Gleichungen werden benutzt, um die Parameter stetig zu aktualisieren und um auch gleichzeitig die passende Anzahl an Bestandteilen für jedes Pixel auszuwählen.

In der Praxis kann sich die Beleuchtung an einem Schauplatz nach und nach oder auch plötzlich verändern. Diesen Veränderungen muss sich angepasst werden, indem man den Trainingssatz durch Hinzufügen und Löschen von Beispielen aktualisiert.

Man wählt eine sinnvolle Zeitperiode *T* und bei Zeit *t* hätte man dann:

$$\mathcal{X}_T = \{x^{(t)}, ..., x^{(t-T)}\}$$

Für jedes neue Beispiel wird der Trainingsdatensatz aktualisiert und es wird neu geschätzt:

$$\hat{p}(\vec{x}|\mathcal{X}_T, BG)$$

Jedoch könnten unter den ehemaligen Proben einige Werte sein, die zu den Vordergrundobjekten gehören und diese Schätzung sollte dementsprechend gekennzeichnet werden als:

$$p(\vec{x}^{(t)}|\mathcal{X}_T, BG+FG)$$

Schlussendlich wird bei dieser Vorgehensweise GMM mit *M* Bestandteilen verwendet:

$$\hat{p}(\vec{x}|\mathcal{X}_T, BG+FG) = \sum_{m=1}^{M} \hat{\pi}_m \mathcal{N}(\vec{x}; \hat{\vec{\mu}}_m, \hat{\sigma}_m^2 I)$$

Insgesamt wird also ein verbessertes GMM Subtraktionsschema vorgestellt. Der neue Algorithmus kann automatisch die Anzahl der benötigten Bestandteile pro Pixel auswählen und wäre somit sehr gut an die zu beobachtende Szene angepasst. Doch auch die Verarbeitungszeit wird reduziert, was einen Vorteil darstellt [46].

b) Vergleich zwischen Gaussian Mixture und K-means Clustering

Wenn man die beiden Prototypmethoden Gaussian Mixture und K-means Clustering (Abbildung 40) vergleicht, stellt man fest, dass sich auf den ersten Blick zwar die Bayes'schen Entscheidungsgrenzen in vielerlei Hinsicht ähneln, doch bei der Gaussian Mixture Methode sind diese Entscheidungsgrenzen um Einiges glatter.

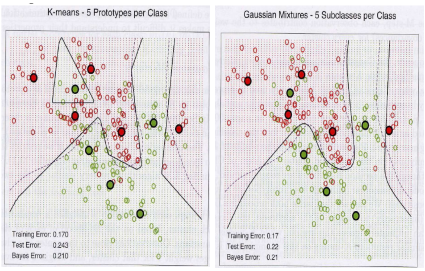

Abb. 40: siehe [40]

Auch können bei der Gaussian Mixture Methode bestimmte Regionen einfach vernachlässigt und nicht

berücksichtigt werden. Diese Regionen müssen dafür aber eine sehr geringe Merkmalsdichte aufweisen.

Da insgesamt eine Klasse in diesem Fall als eine Unterteilung in mehrere Unterklassen betrachtet wird, kann eine dieser Unterklassen durch die Wahrscheinlichkeiten einer anderen Klasse überdeckt werden.

Beim K-means Clustering hingegen würde man hier anders vorgehen und im Fall einer Häufung einfach einen Prototypen an der entsprechenden Stelle festlegen und eine Entscheidungsgrenze umher positionieren [49].

V. Nächste-Nachbarn Methoden

Die grundsätzliche Überlegung beschäftigt sich mit der Idee der parameterfreien Klassifizierung zur Abschätzung der Wahrscheinlichkeitsdichtefunktionen einer bestimmten Datenmenge jeweils in Abhängigkeit der nächsten Nachbarn. Die dafür benötigten Klassifizierer, die die Einteilung in Klassen vorzunehmen haben, basieren auf ihrer Erinnerung und brauchen auch kein anzupassendes Modell. Die nächste-Nachbarn-Regel lautet hierbei:

„Entscheide immer für die Klasse ω_i des nächstliegenden Trainingsdatenpunktes (Prototypen).“ [50]

Durch die nächste-Nachbar-Regel erzeugtes Voronoi-Mosaik (Abbildung 41):

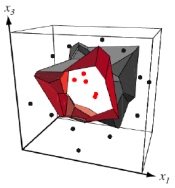

Abb. 41: siehe [41]

A. *k-nächste-Nachbarn Methode*

Die k-nächste-Nachbarn Methode (Abbildung 42 als ein Beispiel) erfordert eine gewisse Vorgehensweise, die eingehalten werden muss:

- Man gibt vorerst einen Punkt x_0 als einen Ausgangspunkt an, der noch nicht klassifiziert wurde.
- Im nächsten Schritt müssen k Trainingspunkte x_r mit $r = 1,..., k$ mit dem kleinsten euklidischen Abstand zu x_0 gefunden werden.
- Schlussendlich soll eine Klassifizierung des Punktes x_0 stattfinden. Diese Zuordnung erfolgt durch eine Mehrheitswahl der vorhandenen k Nachbarn. Damit ist gemeint, dass x_0 der Klasse zugewiesen wird, die die Mehrzahl der k Nachbarn beinhaltet [49].

Die k-nächste-Nachbarn-Regel lautet in diesem Fall:

„Entscheide immer für die Klasse der Mehrheit aller Prototypen innerhalb der k-nächste-Nachbarn-Zelle.“ [50]

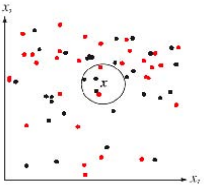

Abb. 42: siehe [42]

Der k-nächste-Nachbarn Methode entsprechend würde zum Beispiel der untere rote Punkt der k-nächste-Nachbarn-Zelle zu der Klasse der drei schwarzen Punkte drüber zugeordnet werden.

a) *Eigenschaften der k-nächste-Nachbarn Methode*

Durch folgende Eigenschaften zeichnet sich die k-nächste-Nachbarn Methode aus:

Die k-nächste-Nachbarn Methode wird sehr erfolgreich bei einer Vielzahl an Anwendungen verwendet. Dazu gehören beispielsweise: handgeschriebene Ziffern (Abbildung 43), Satellitenbilder, aber auch EKG-Bilder, u.v.m..

Abb. 43: siehe [43]

Besonders gute Ergebnisse können mit der k-nächste-Nachbarn Methode erzielt werden, wenn für jede Klasse sehr viele denkbare Prototypen erzeugt werden und wenn die Entscheidungsschranken relativ unregelmäßig festgelegt werden.

Es besteht ein direkter Zusammenhang zwischen der k-nächste-Nachbarn Methode und den Prototyp-Methoden. So ist bei der 1-nächster-Nachbar Methode, bei der eine Klassifizierung zustande kommt, ein Trainingspunkt nämlich gleichzeitig ein Prototyp [50].

Die k-nächste-Nachbarn Methode wird angewendet, wenn die Werte diskret oder reellwertige Vektoren sind. Eine relativ geringe Anzahl an Features muss vorliegen und eine große Menge an Trainingsdaten [51].

b) „Bias“ und „Varianz“

Die beiden Begriffe „Bias“ und „Varianz“ beschreiben mögliche Formen von Falschklassifizierungen. Im ersten Fall, “Bias” genannt, wird ein Trainingspunkt einer falschen Klasse zugewiesen. „Bias“ ist auch unter dem Begriff der Verzerrung bekannt. Es handelt sich um eine Differenz zwischen dem Erwartungswert einer Schätzfunktion und dem zu schätzendem Parameter [52].

Im zweiten Fall, “Varianz” genannt, wird ein Testpunkt falsch klassifiziert. Bei der „Varianz“ wird auch von einem Streuungsmaß gesprochen, welches die Verteilung der Werte um einen Mittelwert kennzeichnet [53].

Es existieren unterschiedliche Fälle, die beobachtet werden können:

- 1-nächster Nachbar: einerseits spricht man zwar von einer Überklassifikation, hingegen findet keine Falschklassifikation der Trainingsdaten statt. Das bedeutet, dass sich ein kleiner “Bias” ergibt, aber eine große Varianz dabei unvermeidbar bleibt.
- 15-nächste Nachbarn: sehr oft werden die Trainingsdaten falsch zugeordnet in Folge einer Klassifizierung. Aus diesem Grund sind ein großer “Bias” und eine kleine “Varianz” vorhanden.
- 7-nächste Nachbarn: wird als eine recht optimale Lösung angesehen, um den Testfehler so gering wie durchführbar zu halten [50].

c) asymptotische Eigenschaften

Von asymptotischen Eigenschaften wird ausgegangen, wenn man einen kleinen “Bias” bei der 1-nächste-Nachbarn Methode hat. Die Einteilung in Klassen ist sehr exakt auf die Trainingsdaten abgestimmt. Die große Varianz sagt aus, dass ein Testpunkt mit hoher Wahrscheinlichkeit falsch klassifiziert wird.

Aus asymptotischer Sicht betrachtet, kommt folgender Zusammenhang zum Vorschein, nämlich das Ergebnis, das schon Cover und Hart entdeckt haben [50]:

Fehlerrate

1-nächster-Nachbar ≤ 2 * Bayes'sche Fehlerrate

Klassifikator

d) Beurteilung der k-nächste-Nachbarn Methode

Folgende Definitionen müssen vorher vorgenommen werden: $p_k(x)$ bildet die Wahrscheinlichkeit ab, dass sich x in der Klasse k befindet. k^* ist hierbei die dominante Klasse der Nachbarn von x. Daraus ergibt sich: $p_k^*(x) \geq p_k(x)$, das gilt für alle $k = 1,..., K$. Dabei darf k nicht k^* entsprechen.

Asymptotisch gesehen gelten auch nachfolgende Zusammenhänge:

Bayes Fehler = $1 - p_k(x)$

1 – nächster-Nachbar-Fehler =

$$\sum_{k=1}^{K} p_k(x) \cdot (1 - p_k(x))$$

Ebenfalls als ein wichtiger Zusammenhang darf die nächste Zeile erwähnt werden:

$$1 - p_k(x) \leq \sum_{k=1}^{K} p_k(x) \cdot (1 - p_k(x)) \leq 2 \cdot (1 - p_k(x))$$

Die Bayes'schen Fehlerraten wurden in dem Paper „Nearest neighbor pattern classification“ von Cover und Hart im Jahr 1967 bewiesen [54].

e) Wie kann die 1-nächster-Nachbar Methode angewendet werden?

Grundsätzliches: Wenn man per Hand selber Ziffern aufschreibt (Abbildung 44), treten geringfügige Veränderungen auf. Das können zum Beispiel kleine Rotationen sein. Für menschliche Augen ergeben sich daraus keine Probleme. Allerdings darf man nicht vergessen, dass es große Unterschiede zwischen einem rotierten und einem nicht rotierten Bild gibt, denn die Graustufenwerte sind dann verschieden.

Abb. 44: siehe [44]

Wie schaut der Aufbau aus?

Man befindet sich in einem Merkmalsraum, der 256 Dimensionen in sich trägt. Mit dem Begriff der Dimension meint man ein Pixel (Abbildung 45) beziehungsweise ein Merkmal. Weiterhin ist ein Pixel in der Lage, Graustufenwerte aus einer Menge von {1,..., 1024} anzunehmen. Bei einem Punkt im Merkmalsraum geht es um einen 256-dimensionalen Vektor. Dieser Punkt repräsentiert eine Zahl (beispielsweise die Zahl 3).

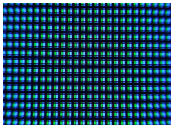

Abb. 45: siehe [45]

Falls der Fall vorliegt, dass sich zwei Bilder nur durch eine Rotation voneinander unterscheiden, gehören zwei Punkte zu einer Klasse [50].

Wie findet ein Kurvenvergleich statt?

Während einer Rotation erfolgt eine stetige Veränderung der Graustufenwerte der einzelnen Pixel. Eine glatte Kurve (Abbildung 46) kann im Merkmalsraum gezeichnet werden.

Graustufenbilder bestehen dabei aus 8 Bit an Informationen pro Pixel und verwenden 256 Grauschattierungen, um Farbabstufungen zu simulieren. Jeder Pixel eines Graustufenbildes hat einen Helligkeitswert zwischen Null (schwarz) und 255 (weiß) [55].

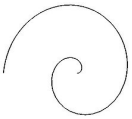

Abb. 46: siehe [46]

Jedenfalls befinden sich beim Beispiel einer 360° Rotation die Originalzahl und die rotierte Zahl auf der gleichen Kurve.

Sicherlich existieren auch hier Nachteile: Auf der einen Seite ist der Aufwand, den man beim Rechnen betreibt, sehr hoch. Andererseits muss zwischen den Ziffern "6" und "9" unterschieden werden, was eine Herausforderung darstellt [50].

Warum handelt es sich um eine invariante Metrik?

Der euklidische Abstand zwischen zwei Bildern wird durch eine Rotation beeinflusst und bleibt nicht unverändert. Der euklidische Abstand stammt aus der euklidischen Geometrie. Es ist eine Distanzfunktion über zwei Vektoren und berechnet als euklidische Norm des Differenzvektors zwischen beiden Vektoren. Dieser Abstand wird zwischen zwei Punkten in einer Ebene oder in einem Raum gemessen als eine Strecke durch die beide Punkte verbunden werden. Kommen Bewegungen zustande, bezeichnet man diesen Abstand als invariant oder auch als unverändert. Es liegen zwei Kongruenzabbildungen (Abbildung 47 – durch Translation / Parallelverschiebung entstanden in diesem Fall) vor [56]:

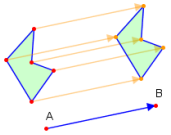

Abb. 47: siehe [47]

Das Problem besteht darin, dass man den minimalen Abstand zwischen zwei Kurven nur recht ungenau bestimmen kann, weil der euklidische Abstand im R^{256} sehr groß sein kann.

Die Rotationskurve bildet invariante Metrik ab. Unter einer Metrik versteht man eine mathematische Funktion, die je zwei Elementen eines Raumes einen positiven reellen Wert zuordnet, bei dem es sich um den Abstand der beiden Elemente handelt. Von der Bedeutung her bedeutet Metrik so viel wie Messung [57].

Die Metrik d ist invariant, wenn $d(x,y) = d(x+a, y+a)$ für alle x,y,a in einem Merkmalsraum zutrifft [50].

Welche Rolle spielen die Tangenten?

Da man die Ziffern selbst geschrieben hat, sind die Rotationen vergleichsweise gering. Man kann zum Beispiel kleine Rotationen stattfinden lassen, um zwei Bilder miteinander zu vergleichen. Von der Vorgehensweise her wird eine Tangente (Abbildung 48 als ein Beispiel) an die Kurve im Originalbild angelegt. Die invariante Kurve wird durch die Tangente approximiert. Durch die Rotation wird eine glatte Kurve abgebildet. Die Tangente wird dann einfach im Punkt x an diese Kurve angelegt.

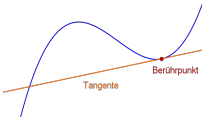

Abb. 48: siehe [48]

Die genaue Vorgehensweise setzt sich aus mehreren Schritten zusammen:

- Als Erstes müssen am Originalbild kleine Rotationen durchgeführt werden.
- Im weiteren Verlauf legt man eine Tangente an die Rotationskurve an.
- Nun muss die "ähnlichste" Tangente herausgefunden werden, also beispielsweise eine Tangente, die sich durch die gleiche Richtung oder durch den gleichen Winkel auszeichnet. Betrachtet werden dabei verschiedenste Tangenten aus der Trainingsmenge der Trainingspunkte, an die diese Tangenten angelegt werden. Das Bild, das die "ähnlichste" Tangente besitzt, wird der gleichen Klasse zugeordnet [50].

Wie hoch werden die Fehlerquoten bei diesem Verfahren geschätzt?

Die Fehlerraten bei diesem Verfahren, bei dem es um einen Tangentenabstand geht, sind als sehr niedrig einzustufen. Man kann diese Fehlerraten mit dem menschlichen Auge (Abbildung 49) gleichsetzen. Selbst bei einem neuronalen Netzwerk würden die auftretenden Fehlerraten höhere Werte erreichen. Beim euklidischen Abstand liegen die Fehlerraten hingegen sehr hoch [50].

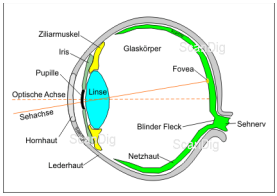

Abb. 49: siehe [49]

B. adaptive nächste-Nachbarn Methoden

Von den adaptiven nächste-Nachbarn Methoden ist die Rede, wenn sich ein Randpunkt in direkter Nachbarschaft befindet.

a) Welche Schwierigkeiten können auftreten?

Als sehr problematisch kann angesehen werden, falls der zu klassifizierende Punkt x_0 zwar in einer bestimmten Klasse (hier: in der grünen Klasse) bereits liegt, aber die Mehrzahl seiner Nachbarn (beispielsweise drei von fünf Nachbarn) einer anderen Klasse (hier: der roten Klasse) angehören (Abbildung 50). So würde x_0 zu der roten Klasse durch eine Mehrheitswahl unter den Nachbarn klassifiziert werden. Daraus lässt sich schlussfolgern, dass k-nächste-Nachbarn Methoden auch unpraktisch sein können. Das trifft vor allem zu, wenn man einen eingeschränkten Umfang an Trainingsdaten und gleichzeitig einen höherdimensionalen Merkmalsraum als eine Grundlage hat. Die Folge sind Falschklassifikationen [50].

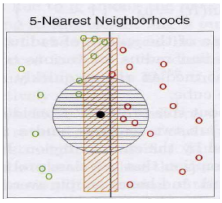

Abb. 50: siehe [50]

b) Welche Lösungen können herangezogen werden, um dieses Problem zu bewältigen?

Wenn man sich in höherdimensionalen Merkmalsräumen befindet, ist es entscheidend, die Punkte als Darstellungen von Zufallsvektoren (Abbildung 51) aufzufassen.

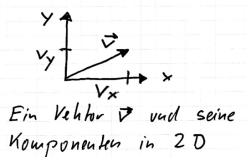

Abb. 51: siehe [51]

Durch die sogenannte Mahalanobis-Distanz wird der Abstand zwischen zwei Punkten $\underline{x}$ und $\underline{y}$ berechnet. Die Formel für die Mahalanobis-Distanz lautet:

$$d(\underline{x},\underline{y}) = \sqrt{(\underline{x}-\underline{y})^T S^{-1} (\underline{x}-\underline{y})}$$

Die Mahalanobis-Distanz ist ein Distanzmaß in einem mehrdimensionalen Vektorraum. Es handelt sich um ein multivariantes Verfahren. Im Fall von multivarianten Verteilungen stellt man sich die m Koordinaten eines Punktes als einen m-dimensionalen Spaltenvektor vor. Dieser Spaltenvektor entspricht dem Zufallsvektor $\underline{x}$ mit seiner Kovarianzmatrix S. Aus mathematischer Sicht ergibt sich die Mahalanobis-Matrix beim Logarithmieren der Dichte einer vorliegenden multivarianten Normalverteilung. Die Mahalonobis-Distanz berücksichtigt die Korrelation und die verschiedenen Skalierungen [58].

c) mögliche graphische Darstellungen

Im Allgemeinen wird die Mahalanobis-Distanz als skaleninvariant bezeichnet. Unter einer Skaleninvarianz beziehungsweise unter einer Skalenunabhängigkeit versteht man die Eigenschaft eines Zustandes, bei dem sich auch bei einer Skalierung die Eigenschaft oder Charakteristik weitestgehend nicht verändert. Man versteht darunter eine Symmetrie eines komplexen Systems. In der Abbildung 52 wird die statistische Skaleninvarianz von DLA (Diffusion-limited aggregation) dargestellt an einem Beispiel [59]:

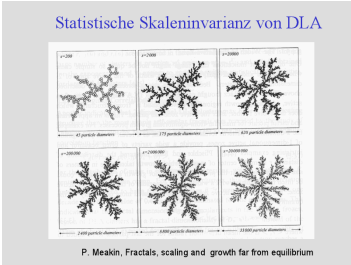

Abb. 52: siehe [52]

Außerdem wird die Mahalanobis-Distanz als translationsinvariant betrachtet. Unter einer Translation

verbirgt sich eine Parallelverschiebung. Es ist eine geometrische Abbildung, die jeden Punkt eines Raumes in derselben Richtung um dieselbe Strecke verschiebt [60].

Graphisch betrachtet bilden zwei Punkte, die die gleiche Mahalanobis-Distanz zu einem Zentrum besitzen, im zweidimensionalen Bereich eine Ellipse (Abbildung 53). Hingegen entsteht ein Kreis beim euklidischen Abstand der Punkte zum Mittelpunkt.

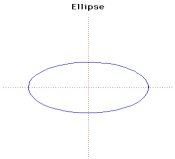

Abb. 53: siehe [53]

Ist die Kovarianzmatrix *S* die Einheitsmatrix, also trägt sie den Wert 1, so stimmen die Mahalanobis-Distanz und der euklidische Abstand überein.

d) praktische Anwendung

Der Grundgedanke besteht darin, einen Punkt zu einer bestimmten gegebenen Population mit Hilfe der Mahalanobis-Distanz zuzuweisen. Zu Beginn müssen sowohl die Erwartungswerte μ_1 und μ_2 als auch die Kovarianzmatrix *S* für die beiden Datensätze ermittelt werden. Darauffolgend sollte die Mahalanobis-Distanz eines Punktes zu den Mittelpunkten der Datensätze berechnet werden. Wo die Mahalanobis-Distanz zum Zentrum hin kleiner ist, wird der Punkt zu der Klasse klassifiziert [50].

e) ein Beispiel

In diesem Beispiel (Abbildung 54) werden sogenannte Zwei-Klassen-Daten erzeugt, also Daten die zu zwei verschiedenen Klassen gehören.

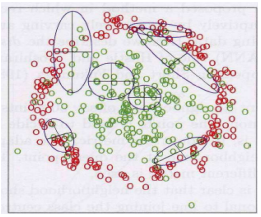

Abb. 54: siehe [54]

Die Daten, die sich in der Klasse 1 (rote Klasse) befinden, sind unabhängig standardnormalverteilt, allerdings mit einer wichtigen Nebenbedingung: Diese Daten der Klasse 1 werden nur auf einem Ring mit dem Radius $r \in (a,b)$, $a < b$ vom Zentrum entfernt, abgebildet. Die der Klasse 2 (grüne Klasse) zugehörigen Daten sind ebenfalls unabhängig standardnormalverteilt, aber diesmal ohne eine Nebenbedingung. Auf diesem Weg umrundet die Klasse 1 die Klasse 2 nahezu vollständig. Es können unterschiedliche Werte für die Anzahl an Trainingsdaten zur Untersuchung benötigt werden, beispielsweise 250 Trainingsdaten pro Klasse [1].

f) Bewertung der nächste-Nachbarn Methoden

Alles eingerechnet lässt sich zusammenfassend feststellen, dass nächste-Nachbarn Methoden als einfache Methoden eingegliedert werden. Man braucht kein Vorwissen über die Daten, mit denen gearbeitet wird. Sehr gute Ergebnisse können erzielt werden, wenn beispielsweise handgeschriebene Zahlen erkannt werden sollen. Dennoch bleibt als Nachteil der hohe Rechenaufwand für das Feststellen von Nachbarschaften bestehen [1].

Auch in der Praxis wird beispielsweise zur Vorhersage von Psychotherapieverläufen eines neuen Patienten mit differenziellen und auch mit adaptiven nächste-Nachbarn Methoden gearbeitet. Der Wunsch besteht darin, neue Subgruppen innerhalb von bereits bestehenden Subgruppen zu identifizieren. Die differenziellen nächste-Nachbarn Verfahren können mit „Expected Treatment Response"-Ansätzen verglichen werden. Durch adaptives Modellieren kann die Vorhersage noch weiter gesteigert werden [61].

VI. ABSCHLIESSENDE BEMERKUNGEN

Bei „adaptive classification" handelt es sich um partiell selbstlernende, dynamische Systeme, welche die bisherigen Klassifikationsverfahren erweitern und neue Möglichkeiten in der Technologie durch die Informatik aufgrund der unzähligen Anwendungsbereiche eröffnen. Im Verlauf der Arbeit wurde auf unterschiedliche praktische Beispiele basierend auf verschiedenen Klassifikationsmethoden eingegangen. Dabei wurde aus Motivationsgründen zu Beginn ein Anwendungsfall beschrieben, um das Interesse des Lesers auf das Themengebiet anzuheben. Da die Grundlagen auf dem Gebiet der künstlichen Intelligenz des Lesers vorausgesetzt werden, findet man die Grundlagen für das allgemeine Verständnis am Ende der Ausarbeitung, damit der Leser dazu aufgerufen wird, sich diese noch einmal ins Gedächtnis zu rufen.

Referenzen beziehungsweise das Literaturverzeichnis

[1] T. Hastie, R. Tibshirani, J. Friedman. “The elements of statistical learning.” Springer, 2001, Kap. 13

[2] http://www.captaris-dt.com/product/adaptive-classification-technology/de/

[3] http://www.captaris-dt.com/download/brochure-dokustar_act-de.pdf

[4] http://www.mathematik.uni-ulm.de/stochastik/lehre/ss07/seminar_sl/zierau.pdf

[5] http://wirtschaftslexikon.gabler.de/Definition/markov-prozess.html

[6] http://books.nips.cc/papers/files/nips15/AA32.pdf

[7] Journal of the American Statistical Association, June 1984, Volume 79, Number 386, Theory and Methods Section.

[8] http://i4.pnnl.gov/focusareas/class_model_desc.stm

[9] http://wwwmath.uni-muenster.de/SoftComputing/lehre/material/wwwnnscript/lern.html

[10] http://www.informatik.uni-ulm.de/ni/Lehre/WS02/LMCVCG/skript/Vorlesung2.pdf

[11] http://www2.ccc.uni-erlangen.de/projects/vsc/chemoinformatik/erlangen/datenanalyse/m_prozesse.html

[12] http://www.inf.uni-konstanz.de/gk/pubsys/publishedFiles/CeBe06c.pdf

[13] http://www.cse.hcmut.edu.vn/~chauvtn/data_mining/Reading/Chapter%205%20-%20Clustering/FCM%20-%20The%20Fuzzy%20c-Means%20Clustering%20Algorithm.pdf

[14] http://onlinelibrary.wiley.com/doi/10.1002/elps.1150181508/abstract, Article first published online: 14 APR 2005

[15] http://flexikon.doccheck.com/Gelelektrophorese

[16] http://www.uni-protokolle.de/Lexikon/Gelelektrophorese.html

[17] http://www.biokurs.de/skripten/13/bs13-11.htm

[18] http://www.bciresearch.org/paper.pdf

[19] http://www.formel-sammlung.de/ld-Hyperebene-855.html

[20] http://ifgivor.uni-muenster.de/vorlesungen/Geostatistik/kap/kap_10/k10_1.htm

[21] http://eprints.pascal-network.org/archive/00001346/01/SheKraBlaRaoMue06.pdf

[22] http://www.gis1.de/archives/266-Multispektral-Klassifizierung.html

[23] http://www.quest-innovations.com/images/stories/pdf/adaptive_classification_method.pdf

[24] http://www.wirtschaftslexikon24.net/d/kanonische-analyse/kanonische-analyse.htm

[25] http://www.sciencedirect.com/science/article/pii/S0165011408001516, Available online 27 March 2008

[26] http://www.lrz.de/~wlm/ilm_l11.htm

[27] http://www.stat.wisc.edu/~mchung/teaching/MIA/reading/diffusion.gaussian.kernel.pdf.pdf

[28] http://www.robots.ox.ac.uk/~parg/pubs/maia06.pdf

[29] http://flexikon.doccheck.com/Elektromyographie

[30] http://informatik.uibk.ac.at/teaching/ss2007/nn/slides/nnpraxis4.pdf

[31] http://www.itwissen.info/definition/lexikon/Black-Box-black-box.html

[32] http://www.hit-karlsruhe.de/EUG-MCL-Musterweb/blackbox.html

[33] https://docs.google.com/viewer?a=v&q=cache:q4va0S0zKi4J:kontext.fraunhofer.de/haenelt/kurs/folien/Haenelt_VektorAehnlichkeit.ppt+euklidischer+abstand+paper&hl=de&gl=de&pid=bl&srcid=ADGEESi3YLWgDi456wLGw4yEGMDpCakJq7NBN9ys1btoz6VpeDL2kEluLn4cJs-EPekRwGtw98YXA3jKCKAZiMnM2EbBpigqtPQHrmYnlWtF0L-D-G80OqoKo3Ba8Xun-Bq1PCDpD_rP&sig=AHIEtbTJDW5l0gGr9NePlG9j2H8jXJqog

[34] http://www.tu-chemnitz.de/etit/systh/forschung/projekte/alt/klassifikatorbox.php

[35] http://www.uni-hamburg.de/fachbereiche-einrichtungen/fb16/psych_1/Kategorisierung1.pdf

[36] http://mnemstudio.org/clustering-k-means-introduction.htm

[37] http://www.kietz.ch/DataMining/Vorlesung/folien/08-Deviation-clustering.pdf

[38] http://www.andiraez.ch/schule/DossierPunktmengenundDreiecke.pdf

[39] http://www.jstor.org/pss/2346830

[40] Zur Clusteranalyse, Joachim Büttner: http://onlinelibrary.wiley.com/doi/10.1002/bimj.19750170304/abstract?systemMessage=Wiley+Online+Library+will+be+unavailable+17+Dec+from+10-13+GMT+for+IT+maintenance.

[41] http://www.mathepedia.de/Normalverteilung.aspx

[42] http://cnx.org/content/m13205/latest/

[43] http://rosuda.org/lehre/WS03/Pausen/vortrag.pdf

[44] http://mars.wiwi.hu-berlin.de/mediawiki/wpstatde/index.php/A-posteriori-Wahrscheinlichkeit

[45] http://www.statistik.tu-dortmund.de/fileadmin/user_upload/Lehrstuehle/Genetik/BI09/Voigt_Vortrag.pdf

[46] http://staff.science.uva.nl/~zivkovic/Publications/zivkovic2004ICPR.pdf

[47] http://home.arcor.de/ulile/node28.html

[48] http://www.chemgapedia.de/vsengine/vlu/vsc/de/ch/13/vlu/daten/statistik/verteilungen.vlu/Page/vsc/de/ch/13/anc/daten/statistik/verteilungen/dichtefunktion.vscml.html

[49] http://www.mathematik.uni-ulm.de/stochastik/lehre/ss07/seminar_sl/zierau.pdf

[50] https://docs.google.com/viewer?a=v&q=cache:hGE0bEubsPUJ:www.ios.htwg-konstanz.de/joomla_mof/index.php%3Foption%3Dcom_docman%26task%3Ddoc_download%26gid%3D73%26%26Itemid%3D102+n%C3%A4chste+nachbarn&hl=de&gl=de&pid=bl&srcid=ADGEESgd0PfBKLxt_zCJsScdph1VDB5P87cdV1gDGmDUZw-hNmN74oIwGKEKkH7Cj7OUVql-fhncOunVLSwnrk3Xm--L9c1RgFfz8nPubBm8OpkFNslUL2ka-UNLp-jUSPC3bfD-9_cx&sig=AHIEtbTgrB6br74XmB22RfrD0n1WxigF7Q

[51] https://docs.google.com/viewer?a=v&q=cache:N-vksI3VWFQJ:www2.in.tu-clausthal.de/~hammer/lectures/seminar_ml/Prototypes.ppt+k-n%C3%A4chste+nachbarn&hl=de&gl=de&pid=bl&srcid=ADGEESj3apSc-LAsYx_Oja_o-AM7ZjBdCG-MAiVEOaMYgEqSvoSn67jyqmqZoDDCs0oBXgCHTo8-a9J-4XdG_bjOZMG5YthC5rPOrswjVsNh-EKtNKJ-kG6zt3vk6BYHZLK9J5WQrCkR&sig=AHIEtbRSnH6pT9xiJvyJ9lZIUuDdI1xk5g

[52] http://wirtschaftslexikon.gabler.de/Definition/bias.html

[53] http://de.statista.com/statistik/lexikon/definition/138/varianz/

[54] http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1053964

[55] http://home.vrweb.de/~rainerp/Theorie/Farbe/Spezialmodi.html

[56] https://docs.google.com/viewer?a=v&q=cache:fuSZ_e0tvOIJ:www2.in.tu-clausthal.de/~hammer/lectures/heursem/KNN.ppt+euklidischer+abstand+definition&hl=de&gl=de&pid=bl&srcid=ADGEEShvyIZNAwRZdWSoTNIC0YxLsqymSd5Da9LEX52Ux-DVQzIV_tLchjUtqzmrwDTm6_l7IDtc7Q6YJdisM16yUM09yQbAB8nsD-7ATTzdl5aMGbT5RpvSU9lIJZikF9f17532Zgi1&sig=AHIEtbQlVGQoIxZNmXsccLi80tmW9M4ehw

[57] http://www.itwissen.info/definition/lexikon/Metrik-metric.html

[58] http://www.statistics4u.info/fundstat_germ/cc_distance_meas.html

[59] https://docs.google.com/viewer?a=v&q=cache:r5wPBw8snycJ:www.thp.uni-koeln.de/krug/talks-Dateien/Selbst.ppt+skaleninvarianz&hl=de&gl=de&pid=bl&srcid=ADGEESjfxcVYpwR5JaOcCR5WxO9gNiF6-oKVnwEpz4sKzPZutkXSOzIS5ZLs6UDUmhr7qLKg2RXl2uVHxX84aegMwBiM0ufAhqgGWfLRYmOGFyXWQwkmQSvNT0aq0gQZQZzLegaJWpM_&sig=AHIEtbR1hxhd2RnVK4qQhwrhQyCrj8Rpcg

[60] http://www.mathematik-wissen.de/parallelverschiebung.html

[61] http://www.workshop-kongress2011.de/index.php?id=53&tx_tanmarkongress_pi1%5Bcontribution

%5D=138&tx_tanmarkongress_pi1%5Bevent%5D=23&tx_tanmarkongress_pi1%5Baction%5D=showPoster&tx_tanmarkongress_pi1%5Bcontroller%5D=contribution&cHash=db80bdf87942b46950c1886c401de0b6

Bilderverzeichnis

[1] http://www.opentext.de/3/extraction_cmyk_950x486.jpg

[2] http://www.captaris-dt.com/download/brochure-dokustar_act-de.pdf

[3] http://hackingarticles.com/wp-content/uploads/how-long-it-takes-to-hack-passwords.jpg

[4] http://www.tse.de/Bilder/IDS.gif

[5] http://i4.pnnl.gov/images/projects/pacman_lg.jpg

[6] http://www.mathematische-basteleien.de/hyperbel17.gif

[7] http://scikit-learn.sourceforge.net/dev/modules/gaussian_process.html

[8] http://sed-ferias09-20610043.wikispaces.com/file/view/175px-Simple_markov_chain.svg.png/85629151/175px-Simple_markov_chain.svg.png

[9] http://www.cs.ubc.ca/~murphyk/Software/Kalman/aima_filtered.jpg

[10] http://www.stauber-lab.de/Bilder/FLJ2.jpg

[11] http://www.clearclix.de/www/www.meermaedchen.de/Projekte/NeuroInf/1.gif

[12] http://www2.ccc.uni-erlangen.de/projects/vsc/chemoinformatik/erlangen/datenanalyse/bilder/ue_lernen.gif

[13] http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/images/image051.jpg

[14] http://www.biokurs.de/skripten/13/bs13-11.htm

[15] http://www.neuronalesnetz.de/nnbilder/large/neuronennetz_large.gif

[16] http://sciencev1.orf.at/static2.orf.at/science/storyimg/storypart_77649.jpg

[17] http://heservis.nl/locked-in.be/braintochair.jpg

[18] http://www.dr-werner-wolf.de/data/eeg/eeg-path.gif

[19] http://ifgivor.uni-muenster.de/vorlesungen/Geostatistik/kap/kap_10/k10_1.htm

[20] http://www.gis1.de/uploads/fernerkundung/iviewcapture_date_11_05_2010_time_08_55_11.jpg

[21]-[22] http://www.quest-innovations.com/images/stories/pdf/adaptive_classification_method.pdf

[23] http://www.mblondel.org/tlml/_images/8859d0e2e1.png

[24] http://www.gutachten-hand-arm.de/Bandriss_Handgelenk/Bilder_Bandriss/Handgelenk_Knochen_Web.jpg

[25] http://salerno.uni-muenster.de/data/bl/graphics/pics_big/elektrom000a_1.html

[26] http://informatik.uibk.ac.at/teaching/ss2007/nn/slides/nnpraxis4.pdf

[27] http://www.hit-karlsruhe.de/EUG-MCL-Musterweb/images/BlackBox.gif

[28] http://www.didmath.ewf.uni-erlangen.de/Verschie/Bertelsmann/Geometrie_HS/1_Grundbegriffe/Abstand_euklidisch.gif

[29] http://www.tu-chemnitz.de/etit/systh/forschung/projekte/alt/klassifikatorbox.php

[30] http://www.uni-hamburg.de/fachbereiche-einrichtungen/fb16/psych_1/Kategorisierung1.pdf

[31] http://mnemstudio.org/ai/cluster/images/k-means1.gif

[32] http://www.kietz.ch/DataMining/Vorlesung/folien/08-Deviation-clustering.pdf

[33] http://www.amspr.gfai.de/images/DataPlane.jpg

[34] http://www.mathematik.uni-ulm.de/stochastik/lehre/ss07/seminar_sl/zierau.pdf

[35] http://cnx.org/content/m13205/latest/Graphic2

[36] http://www.faes.de/Basis/Basis-Lexikon/Normalverteilung_Grafik.jpg

[37] http://www.maths.adelaide.edu.au/matthew.roughan/Code/gaussian_mixture_model.png

[38] http://www.biologie.uni-hamburg.de/b-online/fo03/21a.jpg

[39] http://home.arcor.de/ulile/img42.png

[40] http://www.mathematik.uni-ulm.de/stochastik/lehre/ss07/seminar_sl/zierau.pdf

[41]-[42] https://docs.google.com/viewer?a=v&q=cache:hGE0bEubsPUJ:www.ios.htwg-konstanz.de/joomla_mof/index.php%3Foption%3Dcom_docman%26task%3Ddoc_download%26gid%3D73%26%26Itemid%3D102+n%C3%A4chste+nachbarn&hl=de&gl=de&pid=bl&srcid=ADGEESgd0PfBKLxt_zCJsScdph1VDB5P87cdV1gDGmDUZw-hNmN74oIwGKEKkH7Cj7OUVql-fhncOunVLSwnrk3Xm--L9c1RgFfz8nPubBm8OpkFNslUL2ka-UNLp-jUSPC3bfD-9_cx&sig=AHIEtbTgrB6br74XmB22RfrD0n1WxigF7Q

[43] http://www.zaik.uni-koeln.de/AFS/Projects/Further/PatternRecognition/logo.gif

[44] http://www.mathematik.uni-ulm.de/stochastik/lehre/ss07/seminar_sl/zierau.pdf

[45] http://www.tech-talk.de/wp-content/uploads/2008/05/plasma-einzel-pixel.jpg

[46] http://upload.wikimedia.org/wikipedia/commons/thumb/d/d5/Smooth_curve.jpg/220px-Smooth_curve.jpg

[47] http://www.mathepedia.de/html/7_geometrie/f_elemgeo/a_planimetrie/2_grundbegriffe/d_kongruenzabbildung/transl.aspx?w=214&h=203

[48] http://www.serlo.eu/uploads/918.png

[49] http://www.filmscanner.info/BilderFM/Auge1.gif

[50] http://www.mathematik.uni-ulm.de/stochastik/lehre/ss07/seminar_sl/zierau.pdf

[51] http://www.scienceblogs.de/hier-wohnen-drachen/vektorZerlegung2d-thumb-540x362.jpg

[52] https://docs.google.com/viewer?a=v&q=cache:r5wPBw8snycJ:www.thp.uni-koeln.de/krug/talks-Dateien/Selbst.ppt+skaleninvarianz&hl=de&gl=de&pid=bl&srcid=ADGEESjfxcVYpwR5JaOcCR5WxO9gNiF6-oKVnwEpz4sKzPZutkXSOzIS5ZLs6UDUmhr7qLKg2RXl2uVHxX84aegMwBiM0ufAhqgGWfLRYmOGFyXWQwkmQSvNT0aq0gQZQZzLegaJWpM_&sig=AHIEtbRlhxhd2RnVK4qOhwrhQyCrj8Rpcg

[53] http://www.gap-system.org/~history/Curvepics/Ellipse/Ellipse1.gif

[54] http://www.mathematik.uni-ulm.de/stochastik/lehre/ss07/seminar_sl/zierau.pdf

Transaction-Sensitive Sliding Window Techniques in Stream Mining

Anja Bachmann

Otto-von-Guericke-University of Magdeburg
Universitätsplatz 2, D-39106 Magdeburg, Germany
anja.bachmann@st.ovgu.de

Abstract. Nowadays stream mining is an important process in both industry and research. There are different models for it, but for tracing recent trends it is most advisable to use the sliding window model. Here the challenge is to figure out the right window size; small enough to save memory, but big enough to detect all trends. In this paper, the transaction-sensitive sliding window approaches of Calders et al. and of Li and Lee are considered. The conclusion is that the $MFI\text{-}TransSW$ algorithm of Li and Lee works both effective and efficient due to its bit sequence representation.

Keywords: data mining, data streams, frequent item set mining, stream mining, sliding windows, different window sizes

1 Introduction

Most people create stream data unwittingly in their everyday life: They use their mobile phone to call friends, buy products on amazon.com or like statuses on Facebook. All these events create data items that are part of a stream of information [11]. This data stream can be mined in different ways and with different aims. The application areas vary from critical scientific and astronomical applications to important business and financial ones [7]. Therefore the importance of good stream mining increases more and more.

In stream mining there are several challenges one has to face, e.g. real-time response or single scan of the data. Such challenges are considered in different scientific works like [14,8] or "Challenges on Association Rule Mining On Data Streams in Contrast to Classical Association Rule Mining Algorithms" in this book. In addition it is problematical to detect frequent item sets in a data stream especially if the sliding window model is applied [11]. One important question is how to choose the window size. On the one hand it is advisable to set this quite small for saving memory. On the other hand doing this could cause a significant loss of information, because only most recent data will be considered. Especially seasonal trends would not be detected, e.g. the increased sale of ice cream on hot summer days [3]. Hence it is desirable to have a better possibility to adjust the window size suitable to the evolving dataset. This paper reviews some different approaches using sliding windows.

The rest of the paper is organised as follows. Section 2 gives some basic information and overviews some state of the art approaches. In section 3 the focus is on sliding window models, their properties and its adaptability. A listing and comparison of two different approaches for different window sizes will be presented. Afterwards they will be evaluated in section 4. At the end the topic will be concluded, followed by an outlook to future work.

2 Basics and State of the Art

Data mining is a process for retrieving information from data. There are several techniques that either predict or describe a datasets. Tan et al. state six main tasks for data mining [13]: Classification, Clustering, Association Rule Discovery, Sequential Pattern Discovery, Regression, Deviation Detection; where the first three ones are the most important ones. For different aims other algorithms are applied. A classification algorithm would try to figure out a class for an item by relying on some of its features and hence make some kind of prediction. In contrast a clustering algorithm would aim at finding items that seem to be similar to each other in some way and group them together. Association rule mining tries to find linkages and dependencies between items and aims at figuring out items that occur together. All of these tasks do not only exist in static, but also in dynamic data mining. In this paper the focus will be on the last one.

Li and Lee state that one of the most challenging problems in data mining is mining stream data [11]. Jiang and Gruenwald define a data stream as "an ordered sequence of items that arrive in timely order" [8]. Li and Lee characterise it as continuous and unbounded, appearing with high speed and with an over time often changing data distribution [11]. One big challenge in stream mining is frequent item set mining. An item set was defined by Agrawal et al. in [1,2] as follows: Let $I = \{i_1, i_2, ..., i_m\}$ be a set of items and D a set of transactions. Each transaction $T \in D$ is a set of items so that $T \subseteq I$. Assuming that $X \subseteq I$, then T contains X if $X \subseteq T$. An item set X owns a support s in the transaction set D if $s\%$ of transactions in D contains X. Such an item set X is frequent if its support is higher than a user defined minimum support value *minsupp*. A data stream can be defined in quite a similar way. In [3] Calders et al. defined it as a set of items, i.e. $S = \{i_1, i_2, ..., i_n\}$, where n is the lenght of the stream, i.e. $|S|$. The number of sets in S containing item i is defined as $count(i, S)$, which kind of represents the support value of i. The frequency of an item i in a stream S is denoted as $freq(i, S) := \frac{count(i,S)}{|S|}$.

There are several problems occuring when handling stream data. First of all there is an amount of data whose size is either known as very large or unbounded. Hence a lot of memory is needed and it is not possible to store all information. Every data item should only be scanned one time for detecting frequent items [14,8]. Furthermore frequent items could become non-frequent over time or the other way round [8]. Stream mining algorithms have to be aware of those problems.

It is possible to sub-divide stream mining algorithms into different model types. In [3] they are called landmark, time-fading and sliding window model. In the landmark model the time period is fixed and hence the entire history from one starting point, so-called landmark, until the end of this data stream is considered for item set mining [8]. The time-fading model is also known as Damped model [8]. Here, new arrived data points got an higher weight due to an aging function. In this paper the focus will be on the third model type: sliding windows. They are composed of recently generated data, i.e. the data within the window, which are considered for item set mining [11]. Another distinction criterion in stream mining is the exactness of the results. They can either be exact or approximated. Hence, there are algorithms for every combination of model and exactness type.

3 Different Approaches

Several scientific papers consider sliding window algorithms. They can be distinguished between time-sensitive and transaction-sensitive sliding window ones.

When applying a time-sensitive sliding window, the basis processing unit is a certain time unit, e.g. a minute oder a day [11]. Some known algorithms using time-sensitive sliding windows are the approaches of Lee and Ting [10] or Lin et al. [12].

Another shape of sliding window is the transaction-sensitive one. In this case the basis processing unit is an expired transaction [11]. There are different established algorithms for transaction-sensitive sliding windows, e.g. the approaches of Chang and Lee [5] or Chi et al. [6], but also the approach of Calders et. al [3] and the $MFI\text{-}TransSW$ algorithm by Li and Lee [11].

In [9] Lee et al. also present an algorithm that uses windows consisting of a sequence of partitions.

However in this paper only the transaction-sensitive algorithms of Calders et. al [3] and Li and Lee [11] will be considered further. They present new features and ideas for transaction-sensitive sliding windows that appear to be interesting.

3.1 Calders' Approach

In their works [3,4] Calders et al. propose an exact algorithm using transaction-sensitive items. They defined the term data stream and its content as follows [3]. The item set containing item a and b is denoted as ab. The stream S with item set $\{a, b\}$ arriving first, $\{a\}$ arriving second and $\{a, b, c\}$ arriving third, is represented by $S =< ab\ a\ abc >$.

Calders et al. also define a so-called max-frequency $mfreq(i, S)$ as the maximum frequency of i over all windows starting at the end of the stream, i.e. $mfreq := \max_{k=1\ldots|S|}(freq(i, last(k, S)))$ where $last(k, S)$ is a sub-stream of S containing the last k itemsets of S [3]. The largest window, in which this maximum frequency of item i is found, is denoted as the maximum window, i.e. $maxwin(i, S)$. Since [3], the presented value max-frequency is used as a new

frequency measure. It describes the maximum frequency of an item i over all evolving windows from a defined start point until the end of the stream S [3]. Unfortunately there is an occuring problem: If the last arriving item coincides with the item i then $maxfreq(i, S)$ would become 1. Therefore a minimum window length should be chosen to only consider windows with a length higher than this.

Another characteristic of this approach is the summary, i.e. $summary(S)$, which is created to keep in mind the starting points of those windows that may contain the maximum frequency. Those points are called *boarders* [3]. They target at being able to everytime create a current summary immediately. Moreover the algorithm provides a function $Get_m freq$ to receive the maximum frequency and a procedure $Update$ to handle incoming item sets and then updating the $summary$. Two challenges arise from that: First of all $Update$ has to be very efficient and second $summary$ has to be indepentend from the number of items in the stream or at least only grow slowly with increasing number. In section 4 there will be an evaluation about the mastering of those challenges.

3.2 MFI-TransSW

Li and Lee present a transaction-sensitive algorithm that uses a bit-sequence representation of items for saving memory and increasing speed [11].

They define a transaction data stream as a continuous sequence of transactions [11], as defined in section 2 . They also define a transaction-sensitive window $TransSW$ as a window that moves forward for every transaction and that contains a number of transactions, called w[11]. For all items X in the current sliding window $TransWS$ a bit sequence $Bit(X)$ of lenght w is created. A *bit-sequence transform* is procedured that sets the ith bit to 1 if the ith transaction contains the item X.

There are three phases: "Window initialization, window sliding and frequent itemsets generation" [11]. First of all there are as much windows collected in $TransSW$ as necessary to reach a length of w. Afterwards if a new transaction comes in then the oldest window is removed from $TransSW$ by a bitwise left shift. Subsequently *Item-Prune* is applied, i.e. dropping all items that no longer occur in $TransSW$. The third phase is only proceeded if the "up-to-date set of frequent itemsets is requested" [11]. In this case, a set CI_k of candidate item sets from the pre-known set of frequent item sets is generated by a method called $CIGA$. $CIGA$ is standing for "Canditate Itemset Generation using the Apriori property". The *Apriori* property was conceived by Agrawal et. al in [2] and state that the subsets of a frequent item set also have to be frequent or the other way round that all supersets of an infrequent itemset have to be infrequent, too. Then for all candidates a bitwise AND operation is proceeded over all transactions to get their frequency.

One special ability of $MFI\text{-}TransSW$ is that it can be extended to also work on time-sensitive sliding windows. This extension is called $MFI\text{-}TimeSW$ and is considered in [11], but it will not be part of this paper. Hence, only $MFI\text{-}TransSW$ will be considered further and evaluated in the next section.

4 Evaluation

Two approaches were presented in section 3, the one by Calders et al. and the one of Li and Lee.

Calders' approach works fairly well in the tested scenario without relying on a fixed window length or a time-decaying factor [3]. The experiments in [3] show that the generation and storage of the *summary* takes less memory. Unfortunately they only run experiments with data streams containing two distinct items a and b. It is assumable that the storage consumption would be higher if the number of different items would become almost as large as the cardinality of S. Referring to the two challenges named in subsection 3.1 one can say that the procedure *Update* seems to work quite fast and therefore master the challenge. Unfortunateley it is not possible to say that the summary is that independent from the number of items in the stream. In the case of a data stream only containing two different items, the summary is quite independent, but it is not possible to state a level of mastering for other datasets. The biggest problem regarding this is that Calders et. al themselves say that their algorithm is only "feasible if and only if the summary remains small"[3]. As long as this cannot be guaranteed the algorithm cannot yet be treated as a good-working one.

The algorithm $MFI\text{-}TransSW$ of Li and Lee [11] was compared to three established algorithms, namely SWF [9], Moment [6] and SWFI-stream [5]. The evaluation showed that the memory usage grows proportional to the window size for all four algorithms. However above these algorithms $MFI\text{-}TransSW$ took least memory. Another experiment showed that memory consumption especially grows during the window sliding phase, but not for $MFI\text{-}TransSW$. This algorithms also outperforms the other ones regarding memory consumption in the frequent item set mining phase. Regarding both the response time and the sliding time, $MFI\text{-}TransSW$ works also fairly good and outperforms the other algorithms, due to its bit sequence representation. Concluding, $MFI\text{-}TransSW$ can be seen as a time-efficient method for mining frequent item sets from data streams within a transaction-sensitive sliding window [11].

5 Conclusions and Future Work

Both algorithms show that they are able to present accurate results in quite a short time. Unfortunately the algorithm of Calders et al. leaves some questions unanswered and is suggestive of needing to be improved. In contrast the algorithm $MFI\text{-}TransSW$ shows good results both in memory usage and processing time and is able to outperform established algorithms. Due to its bit sequence representation, operations can be processed quite fast with a very low memory consumption. Moreover there is an extension of this algorithm to enable it to work on time-sensitive sliding windows. Hence the final statement of this paper is that $MFI\text{-}TransSW$ is a very recommandable algorithm.

Some future challenges could be to test Calders' approach on other datasets to see if its good performance can still be guaranteed. Furthermore Calders et al. could consider the mentioned minimum window size further and run experiments regarding it. An option for $MFI\text{-}TransSW$ is to extend this approach so that it is not only applyable for item set mining, but also for association rule mining.

References

1. R. Agrawal, T. Imielinski and A. Swami. Mining Association Rules Between Sets of Items in Large Databases. *ACM SIGMOD Record*, Vol. 22, No. 2, p.207-216. June, 1993.
2. R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. *Proc. of the 20th Int'l Conf. on Very Large Data Bases (VLDB)*. Santiago, Chile. September 12-15, 1994.
3. T. Calders, N. Dexters and B. Goethals Mining Frequent Items in a Stream Using Flexible Windows. *Proc. of the ECML/PKDD- 2006 Int'l Workshop on Knowledge Discovery from Data Streams (IWKDDS)*. Berlin. September 18, 2006.
4. T. Calders, N. Dexters and B. Goethals A New Support Measure For Items in Streams. , 2007.
5. J.H. Chang and W.S. Lee. A Sliding Window Method for Finding Recently Frequent Itemsets over Online Data Streams. *Journal of Information Science and Engineering*, Vol. 20, p.753-762. 2004.
6. Y. Chi, H. Wang, P.S. Yu and R.R. Muntz. Catch the Moment: Maintaining Closed Frequent Itemsets over a Data Stream Sliding Window. *Knowledge and Information Systems*, Vol. 10, No. 3, p.265-294. 2006.
7. M.M. Gaber, A. Zaslavsky and S. Krishnaswamy. Mining Data Streams: A Review *ACM SIGMOD Record*, Vol. 34, No. 2, p.18-26. June, 2005.
8. N. Jiang and L. Gruenwald. Research Issues in Data Stream Association Rule Mining. *ACM Sigmod Record*, Vol. 35, No. 1, p.14-19. March, 2006.
9. C.H. Lee, C.R. Lin and M.S. Chen. Sliding Window Filtering: An Efficient Method for Incremental Mining on a Time-Variant Database. *Information systems*, Vol. 20, No. 3, p.227-244. 2005.
10. L.K. Lee, and H.F. Ting. A simpler and more efficient deterministic scheme for finding frequent items over sliding windows. *Proc. of the 25th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. Chicago, USA. June 2628, 2006.
11. H-F. Li and S-Y. Lee. Mining Frequent Itemsets over Data Streams Using Efficient Window Sliding Techniques. *Expert Systems with Applications*, Vol. 36, No.2, p.1466-1477. March, 2009.
12. C-H. Lin, D-Y. Chiu, Y-H. Wu and A.L.P. Chen. Mining Frequent Itemsets from Data Streams With a Time-Sensitive Sliding Window. *Proc. of the 5th SIAM International Conference on Data Mining*. Newport Beach, USA. April 21-23, 2005.
13. P.N. Tan, M. Steinbach and V. Kumar. Introduction to Data Mining. Pearson Addison Wesley Boston. 2006.
14. P. S. Yu and Y. Chi. Association Rule Mining on Streams. *Encyclopedia of Database Systems*, p.136-139. Springer US. 2009.

Linear Dimension Reduction Techniques

Christian Braune

Otto-von-Guericke-University of Magdeburg
Universitätsplatz 2, D-39106 Magdeburg, Germany
christian.braune@st.ovgu.de

Abstract. With sizes of datasets to be analyzed rising and rising the need for selection of meaningful attributes to describe the data rises likewise. Dimensionality reduction techniques can help to cope with problems like storing unnecessary data or the curse of dimensionality [2]. Different linear dimension reduction techniques (like e.g. PCA or Optimal Seperation) are introduced in this paper and their advantages and disadvanteges are briefly discussed. With the tools to reduce dimensionality given one has to keep in mind that neither of them can perform best for all data situations (see [18, 19]). A procedure, called Meta Learning, for choosing the best method for the data at hand is described and the evaluation of its performance is discussed.

Keywords: dimension reduction, principal component analysis, meta learning, sammon mapping, canonical correlation analysis, linear discriminant analysis, feature selection, fisher criterion, multidimensional scaling, self organizing maps, no free lunch theorem

1 Introduction and Motivation

With sizes of datasets to be analyzed rising and rising the need for selection of meaningful attributes to describe the data rises likewise. Out of the 211 data sets listed at the UCI database for machine learning[1] more than 125 data sets have more than ten attributes and more than 25 have more than 100 attributes. These are only sample training sets used for comparing algorithmic results. Real World data may easily have a lot more attributes such that even multidimensional visualization techniques such as parallel coordinates [6] become hard to interpret. When trying to cluster the data given or to build classifiers from it, one also faces the problem that not every attribute may be informative for the given problem. Such an attribute might be the credit card number of a client, when looking for clients that might be interested in a new product.

Another problem with high-dimensional data is the so-called curse of dimensionality [2]. For any distance measure the difference between maximum and minimum distance become inevitably undistinguishable (see Eq. 1).

$$\lim_{d\to\infty} \frac{d_{max} - d_{min}}{d_{min}} = 0 \tag{1}$$

[1] http://archive.ics.uci.edu/ml/datasets.html

This and the extended need for storaging the possibly meaningless information make it necessary to reduce the number of features used for classification tasks.

A naïve approach for doing so is to test all subsets of the features whether they are suitable to represent the data or not. This leads to an exponentially increasing computational cost, as possibly all elements of the power set of the set of features have to be tested. In other approaches (such as frequent itemset mining, [1]) enumeration of exponentially growing sets can be sped up by pruning techniques. Unfortunately this is not possible with feature selection as a feature added to a not correlated set of features might reveal the hidden correlation while the same feature itself might not bear enough information to be used as a feature set alone. Hence, pruning approaches here will not be successful and more advanced techniques than simple enumeration have to be employed.

Fig. 1. Iris data set, (a) all four attributes used, (b) only first two principal components used

An example for the benefits of feature selection or dimensionality reduction is shown in Figures 1. The scatter plot matrix of the four original attributes (Figure 1) compared to the scatter plot of the first two principal components shows that the inherent structures in the data can still be perceived, although only half of the original dimensions are available. The method used here is a principal component analysis (PCA) which will be explained in more detail in Section 3.1. In contrast to simple feature selection, PCA is used to recombine attributes to form a set of new attributes. This new set is then able to keep a certain amount of variance of the data. Along with the reduced need to store information, while still preserving certain features of the data, make dimension reduction techniques highly suitable for data analysis.

2 Related Work

Beside the linear dimension reduction techniques that will be explained in greater detail in Section 3, there are also non-linear mappings that are used to reduce the number of dimensions of a dataset. Other than explaining most of the variance of a given data set, these algorithms work directly on the distance matrices that can be obtained from pair-wise comparison of the data points. For each data point $x_j \in \mathbb{R}^d$ these techniques try to find points $x'_j \in \mathbb{R}^{d'}, d' < d$ such that

$\forall i, j : d(x_i, x_j) \approx d(x'_i, x'_j)$. This distance preserving feature was first introduced by Sammon [16] by directly reducing a stress function that is derived from the squared pair-wise differences of the corresponding distance matrices. Primarly he intended this mapping to be used for presenting high-dimensional data in two- or three-dimensional spaces, but the algorithm works well for any target dimension. Reduction to only one dimension has been shown (see [3, 4]) to work as a sorting criterion as well.

Another well-known technique for non-linear dimensionality reduction is closely related to neural networks. Teuvo Kohonen proposed the usage of self-organizing maps [10] to approximate spaces of target concepts. This can also be used to represent high dimensional spaces with low dimensional maps.

The probably most used technique today is multidimensional scaling (MDS, [17]) which was originally used in psychology, but has been applied to several other fields nowadays. Here, the setup is similar to that of Sammon's mapping but either uses a different stress function (classical MDS) or reduces the minimization of stress to an eigenvalue/eigenvector problem.

All the aforementioned methods have in common that they do not create new features as linear combinations of old ones, but they rather try to maintain the inter-point distances to represent the structure within the data as closely as possible. Algorithms that recombine existing features into new ones will be presented in Section 3.

With all the choices of techniques and methods to reduce the dimensionaltiy of data there are still situations where one or the other procedure performs poorly and produces unexpected results. The fact that there cannot be a single, best method is known from other domains of machine learning and data analysis as *no free lunch theorem* [18, 19]. Wolpert stated with this theorem (initially for learning algorithms) that under the least restrictive assumptions (the most unbiased situation) for roughly every concept a learner performs well at, there is another concept it will perform poor at. Thus, on average all algorithms that minimize a target function behave the same when averaging over all possible target functions.

How to choose that algorithm which performs best for reducing dimensionality for the data at hand will be covered in Section 4.

3 Linear Dimension Reduction Techniques

In the following section some popular techniques for dimensionality reduction will be described. Some of them have been in use for more than a hundred years while others were just recently proposed. All of them have in common that they choose or recombine existing features into new features as linear combinations of the existing features. For d existing features and k features to be created new features are calculated by

$$x'_j = \sum_{i=1}^{d} \alpha_{ij} \cdot x_i, \forall 1 \leq j \leq k \ . \tag{2}$$

The main differences between the algorithms lie in the coefficients that are chosen for each of the existing features to be used to calculate each of the new features.

3.1 Principal Component Analysis (PCA)

First presented by Pearson in [14] the principal component analysis (PCA) uses the eigenvalues $\lambda_1, \ldots, \lambda_d$ of the covariance matrix of the data at hand to find the dimensions of largest variance. For that the eigenvalues of the covariance matrix are sorted in descending order. The corresponding eigenvectors form an orthogonal matrix Γ which can be used to re-align the data points along the axis formed by the principal components. The directions of maximal variance are then aligned to the axis of the new coordinate system, hence this is also called principal axis transformation. If only the first k principal components (i.e. eigenvalues and their corresponding eigenvectors) are used to create Γ and the rest is set to zero, the resulting data transformation will also only use k dimensions. Thus the other $d-k$ dimension are omitted and the PCA can be used to keep maximum variance for a fixed number of target dimensions. On the other hand one could also use the smallest $k' < d$ such that

$$\frac{1}{\sum_{i=1}^{d} \lambda_i} \cdot \sum_{i=1}^{k'} \lambda_i \geq \rho \tag{3}$$

as number of dimensions that keep at least ρ of the variance.

In [20] the sparse PCA (SPCA) is described that chooses feature weights to be either zero or one, making interpretation of the resulting features easier, e.g. in the context of gene expression data.

3.2 Canonical Correlation Analysis (CCA)

In canonical correlation analysis one originally aims to find a way to maximize the correlation coefficient between two sets of random variables, e.g. different classes of labelled data. For this the random vectors $X = (x_1, \ldots, x_m)$ and $Y = (y_1, \ldots, y_n)$ are each multiplied with weighing vectors a and b respectively. The objective function is to maximize the correlation between $a^T X$ and $b^T Y$:

$$J(a, b) = \frac{a^T \Sigma_{XY} b}{\sqrt{a^T \Sigma_{XX} a}\sqrt{b^T \Sigma_{YY} b}} , \tag{4}$$

where Σ_{IJ} denotes the (possibly non-square) coveriance matrix of the vectors I and J.

The solution of this optimization problem is then reduced to an eigenvector problem, where a and b are the eigenvectors belonging to a' or b' respectively.

$$a' \in \operatorname{spec}\left(\Sigma_{XX}^{-1} \Sigma_{XY} \Sigma_{YY}^{-1} \Sigma_{YX}\right) \text{ and } b' \in \operatorname{spec}\left(\Sigma_{YY}^{-1} \Sigma_{YX} \Sigma_{XX}^{-1} \Sigma_{XY}\right) \tag{5}$$

The entries in the vectors themselves give the correlation between the features from X and Y.

As highly correlated features represent redundancy in the data, only one of the highly correlated features needs to be used for representation. Thus, dispensable features may be omitted and the dimensionality of the data can be reduced.

3.3 Linear Discriminat Analysis (LDA)

Closely related to the PCA, but taking the class distribution of labelled data into account, is the linear discriminant analysis (LDA, first described by Fisher [8]). Originally, Fisher's discriminant is used to find a decision boundary between two classes of objects. To use the LDA as classifier one only needs to calculate the squared mahalonobis distance ([5], Eq. 6) between the point to be classified and the two class centers. The class center that lies nearer w.r.t. that distance is the class the new data point gets assigned to.

Using the LDA as dimensionality reduction technique requires some small changes to this procedure. For k classes one finds a linear discriminant to separate one and the other $k-1$ classes. This is repeated until a set of $k-1$ linear discriminants has been found. These are pairwise uncorrelated and span a $k-1$-dimensional space that contains most of the classes variability.

$$d(x,y) = (x-y)^T \Sigma^{-1}(x-y), \Sigma \text{ is covariance matrix.} \tag{6}$$

3.4 Fisher's Criterion

A different approach to find a solution of the LDA is, again, to reduce this problem to an eigenvector problem of covariance matrices. As the number of classes within the training data is known, one can calculate the covariance matrix for only those points that belong to a class as Σ_i for each $1 \leq i \leq k$. With this and the a priori probability π_i of each class the within-class (Σ_W) and between-class (Σ_B) variance can be calculated as

$$\Sigma_W = \sum_{i=1}^{k} \pi_i \Sigma_i \text{ and } \Sigma_B = \sum_{i=1}^{k} \pi_i (\mu_i - \mu_0)(\mu_i - \mu_0)^T \ , \tag{7}$$

where μ_i is the mean of class i and μ_0 is the overall mean.

The criterion itself, that has to be maximized, is then

$$J_F(\gamma_j) = \frac{\gamma_j^T \Sigma_B \gamma_j}{\gamma_j^T \Sigma_W \gamma_j} \ , \tag{8}$$

where $\gamma_j \in \mathbb{R}^d, 1 \leq j \leq r$ are the so-called discriminant vectors which are closely related to the decision boundaries mentioned in Section 3.3. The optimization problem is subject to the constraints

$$\gamma_i^T \Sigma_W \gamma_j = 0, i \neq j, \forall i,j : 1 \leq i,j \leq r \ . \tag{9}$$

In [9] it was shown that this problem can be solved by eigenvalues and -vectors.

3.5 Optimal Seperation (OS)

Unfortunately Fisher's criterion does not take into account how points are classified according to its results. That may lead to the misclassification rate that rises when reducing the dimensionality of the data. In [11] Luebke and Weihs therefore propose a different approach that aims at giving an upper bound of errors the criterion is prone to. Parameters for for dimensionality reduction are then chosen such that this upper bound is thoroughly reduced. While still not able to calculate the true error beforehand, reducing the upper error bound gives a good and important estimate for optimizing the dimensionality reduction. Starting from the squared mahalonobis distance (see Eq. 6) the distance given a projection $\Gamma = (\gamma_1, \ldots, \gamma_r) \in \mathbb{R}^{d \times r}$ can be defined as

$$d'(x, y|\Gamma) = ((x-y)^T\Gamma)(\Gamma^T\Sigma_W\Gamma)^{-1}(\Gamma^t(x-y)) \ , \tag{10}$$

which is the squared mahalonobis distance in the projection space (where Σ_W is the same as in Section 3.4).

The optimal separation criterion is then derived from that distance measure as

$$J_{OS}(\Gamma) = \frac{k-1}{k} \sum_{i=1}^{k} \Phi\left(-\frac{1}{2} \cdot \min_{j=1,\ldots,k; i \neq j} d'(\mu_i, \mu_j|\Gamma)\right) \ , \tag{11}$$

where Φ is the cumulative distribution function of the Gaussian distribution. This criterion directly represents the upper error bound (under the assumption that it is normally distributed) such that minimizing $J_{OS}(\Gamma)$ is equal to finding optimal vectors $\gamma_1, \ldots, \gamma_r$ for the dimensionality reduction.

3.6 Minimum Error Criterion (MEC)

In [15] Roehl and Weihs give a criterion that does not aim at finding a good projection Γ explicitly. The minimum error criterion (MEC) rather minimizes the error which an LDA classifier is prone to. For this let $\alpha(x)$ be a classifier that assigns a class i to the vector $x \in \mathbb{R}$. Thus, given a projection matrix Γ, $\alpha(x^T\Gamma)$ is the class that is assigned to x after dimension reduction and $P(\alpha(x^T\Gamma) \neq i | x \in C_i)$ is the probability of assigning x to the wrong class. Here C_i is the set of all points belonging to class i. The probability of the prediction errors can be estimated by bootstrapping [7] allowing the minimization of the MEC:

$$J_{MEC}(\Gamma) = \sum_{i=1}^{k} \pi_k P(\alpha(x^T\Gamma) \neq i | x \in C_i) \ . \tag{12}$$

Still one has to assume, that the error rate is normally distributed for the bootstrapping process to work properly.

3.7 Discussion

All the aforementioned methods have in common that they are in general not optimal. The PCA only performs an alignment of the data along its principal axis but does not take available class information into account. One might argue though that this is a benefit of the PCA as it does not need any class information. This is different for the rest of the methods described. They all rely on the presence of correctly classified training data to generate a projection matrix.

Although the Fisher Criterion can be optimized analytically it may not always yield the optimal result. LDA and CCA are more often used as classifier rather than as dimension reduction techniques. The two further techniques described in this paper suffer from the need of bootstrapping or similar techniques to estimate the error probabilities. Neither of them can be optimized analytically and the computational effort for their approximation may be quite high.

4 Choosing the Best Technique: Meta-Learning

What is still left as an open question: Which method should be used? It has already been discussed at the end of Section 1 that there is no single best method. The *no free lunch theorem* forbids such a method. But we still might be able to decide from the characteristics of the data itself which algorithm might perform best in this special case.

In [12] Luebke and Weihs propose a 4-step algorithm, called Meta-Learning, to classify high-dimensional data with prior dimensionality reduction:

1. Identify data space
2. Calculate selection statistic
3. Apply selection statistic
4. Apply selected method

4.1 3-Class Case

The authors look at the most simple case under which the Meta-Learning algorithm can be applied. Since the Fisher criterion (see Section 3.4) is optimal in a two class case, but suboptimal in a $k \geq 3$ case, the most simple case to consider is a three class scenario.

First of all the relevant data space has to be identified for this whole data space to be analyzed. Following the argumentation of Luebke and Weihs, one can project any k-class problem into a $k-1$-dimensional space (see also [13]). Furthermore one may assume (without loss of generality) that the within-covariance matrix is the identity matrix such that only one case has to be considered:
$d = 2, r = 1, \Sigma_W = I_2$, where I_2 is the 2×2 identity matrix.

To consider all possibly relevant data situations on has to vary the three central moments (mean (v_1), variance (v_2) and skewness(v_3)) of the data itself. As the mean is in some way a scaling parameter, Luebke and Weihs only consider the second and third moment on their analysis and conclude that for certain

combinations of skewness and variance different criterion perform better for prediction tasks.

By simulating data with the given parameters they train an LDA classifier with 100 training examples and calculate the error on 1000 test observations. The classification is either done with Fisher's criterion (see 3.4) or the optimal separation criterion (see 3.5). Repeating the process 50 times allows them to give intervals of v_2 and v_3 where either Fisher's criterion or the OS criterion performs significantly better ($\alpha = 0.1$, see Figure 2). From the results they generate two rules which help decide when to use the OS criterion:

- if $v_2 < 0.2$ use OSC,
- if $v_3 < -0.5$ use OSC.

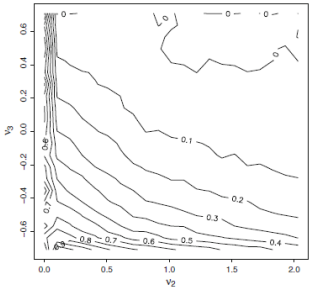

Fig. 2. Performance gain of OSC against Fisher's criterion.

4.2 4-Class Case

For generalization to four classes some additional assumptions have to be made. First of all the apriori probabilities for each class within the data have to be equal as well as all within-covariance matrices which also should be regular. This leads to the classes within the data only varying in their means' locations.

To distinguish between four classes, $2^{4-1} = 8$ variables are needed. So Luebke and Weihs chose to use the first four central moments (kurtosis = v_4) in addition to the condition number of Σ_W, the number of observations per class, the skewness of the data distribution per class, and the kurtosis of the data distribution per class.

The rule set is not that simple as in the 3-class case, but can be visualized as a decision tree (see Figure 3). The decision tree is based on the german business cycle and shows (in grey) which method should be used for dimension reduction during which years.

Based upon an adaptive procedure that uses different methods for different time periods the prediction error rate could be reduced to 0.088 whereas the optimal separation criterion alone yields an error rate of 0.093. All other criteria tested yield even higher error rates.

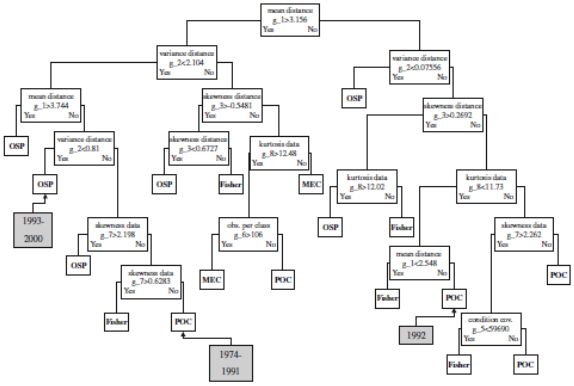

Fig. 3. Decision Tree derived from the german business cycle data set. Leaf nodes contain the criterion that should be used for dimension reduction, if the data has the characteristic given by the non-leaf nodes.

5 Conclusions and Future Work

Dimensionality reduction techniques have been in use for more than a hundred years now and still they are of great interest. With advancing computational power more sophisticated measures and methods become feasible and less restrictive assumptions are possible. Still the *no free lunch theorem* states that there can not be a single best method and thus there will always be need to find methods that improve results in certain data situations. With the techniques presented in this paper we already have some tools at hand that cover different aspect of what good data separation is in classification tasks. Yet it is still hard to choose which method should be applied when. In this case Meta-Learning can help and can even lead to adaptive procedures that improve the classification results even further.

Yet, there this method can still be improved upon in the future. All the methods described here lack of some kind of generalization capabilities. They assume linear separability of the data or normal distribution of equal parameters within the data - both assumptions that are certainly not often met in real life data sets. While kernel methods might help with the linear separability of data, this induces additional dimensions instead of reducing the dimensionality. Selection and combination of suitable features becomes even more important and a combination of these techniques has to be investigated in to see, if there are any benefits to be gained.

Also the sensitivity to outliers is still a concern for most of the proposed methods. Some filtering techniques or lower weighing of outliers may help to improve classification quality and should be investigated in the future.

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proc 20th Int Conf Very Large Data Bases VLDB. pp. 487–499 (1994)
2. Beyer, K., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is nearest neighbor meaningful? In: Beeri, C., Buneman, P. (eds.) Database Theory ICDT99, Lecture Notes in Computer Science, vol. 1540, pp. 217–235. Springer Berlin / Heidelberg (1999)
3. Borgelt, C., Braune, C., Kötter, T., Grün, S.: New algorithms for finding approximate frequent item sets. Soft Computing - A Fusion of Foundations, Methodologies and Applications pp. 1–15 (2011)
4. Braune, C., Borgelt, C., Grün, S.: Finding ensembles of neurons in spike trains by non-linear mapping and statistical testing. In: Gama, J., Bradley, E., Hollmén, J. (eds.) Advances in Intelligent Data Analysis X, Lecture Notes in Computer Science, vol. 7014, pp. 55–66. Springer Berlin / Heidelberg (2011)
5. De Maesschalck, R., Jouan-Rimbaud, D., Massart, D.: The mahalanobis distance. Chemometrics and Intelligent Laboratory Systems 50(1), 1–18 (2000)
6. d'Ocagne, M.: Coordonnées parallèles et axiales: Méthode de transformation géométrique et procédé nouveau de calcul graphique déduits de la considération des coordonnées parallèles. (1885)
7. Efron, B.: Bootstrap methods: another look at the jackknife. The annals of Statistics 7(1), 1–26 (1979)
8. Fisher, R.A.: The use of multiple measurements in taxonomic problems. Annals of Eugenics 7(7), 179–188 (1936)
9. Jin, Z., Yang, J., Tang, Z., Hu, Z.: A theorem on the uncorrelated optimal discriminant vectors. Pattern Recognition 34(10), 2041–2047 (2001)
10. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological cybernetics 43(1), 59–69 (1982)
11. Luebke, K., Weihs, C.: Improving feature extraction by replacing the fisher criterion by an upper error bound. Pattern recognition 38(11), 2220–2223 (2005)
12. Luebke, K., Weihs, C.: Linear dimension reduction in classification: adaptive procedure for optimum results. Advances in Data Analysis and Classification pp. 1–13 (2011)
13. McCulloch, R.E.: Some remarks on allocatory and separatory linear discrimination. Journal of Statistical Planning and Inference 14(2-3), 323–330 (1986)
14. Pearson, K.: Principal components analysis. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 6(2), 559 (1901)
15. Roehl, M., Weihs, C.: Direct minimization of error rates in multivariate classification. Computational Statistics 17, 29–46 (2002)
16. Sammon, J.: A nonlinear mapping for data structure analysis. Computers, IEEE Transactions on 100(5), 401–409 (1969)
17. Torgerson, W.: Multidimensional scaling: I. theory and method. Psychometrika 17(4), 401–419 (1952)
18. Wolpert, D.H.: The lack of a priori distinctions between learning algorithms. Neural Computation Vol. 8(No. 7), 1341–1390 (October 1996)
19. Wolpert, D.H.: The supervised learning no-free-lunch theorems. In: In Proc. 6th Online World Conference on Soft Computing in Industrial Applications. pp. 25–42 (2001)
20. Zou, H., Hastie, T., Tibshirani, R.: Sparse principal component analysis. Jcgs 2006 15(2), 262–286 (2006)

Training von neuronalen Netzen anhand von nicht eindeutig benannten Mustern mittels des Nächster-Nachbar-Algorithmus

Severin Orth

Fakultät für Informatik
Otto-von-Guericke-Universität
39106 Magdeburg
severin.orth@st.ovgu.de

Zusammenfassung Auf Basis einer Veröffentlichung von E. Hüllermeier and J. Beringer wird das Problem des Trainings mit nicht eindeutig benannten Mustern besprochen [1]. Die Ermittlung des adäquaten Bezeichner erfolgt dabei unter Zuhilfenahme des Nächster-Nachbar-Algorithmus. Es erfolgt eine Implementierung in Python.

1 Einführung

Ein Problem der Informatik ist die Zuordnung von Objekten, welche zum Beispiel von einer Kamera aufgenommen, in bestimmte Klassen.

Klassifizierung wird maschinenunterstützt häufig mit Neuronalen Netzen oder ihren Verwandten gelöst.

Trainieren von intelligenten Systemen zur Klassifizierung erfolgt normalerweise mit eindeutig benannten Mustern.

Single-Labeled Klassifizierung heißt, dass zu jedem Trainingsbeispiel eine eindeutige Bezeichnung vorhanden ist und dieses eindeutig in eine Klasse einzuteilen ist.

Können die Beispiele in mehrere Klassen eingeteilt werden, spricht man von der Multi-Labeled Klassifizierung. Viele Bücher werden beispielsweise mit mehreren Genre kategorisiert [2].

Bei nicht eindeutig benannten Mustern (Ambiguous-Labelled Klassifizierung) gehört zwar jedes Exemplar eindeutig zu einer Klasse, zum Trainieren tragen aber einige Beispiele mehrere Bezeichner[3]. Programmiertechnisch zu lösen ist dann die Aufgabe, welches der Bezeichner der korrekte ist. Der Trainingsimplementierung wird hierfür ein Algorithmus vorgeschaltet, welcher aus dem Set der Bezeichner den zum Training adäquaten entscheidet.

1.1 Klassifizierung

Im Gegensatz zum Clustering, bei welchem man versucht anhand einer Menge von Daten Zusammenhänge und Strukturen zu erkennen, sind bei der Klassifizierung für die Objekte bereits mögliche Klassen bekannt. Es erfolgt eine eindeutige Zuweisung dieser Klassen zu den Objekten [4].

2 Kognitive Systeme

Maschinenunterstützte Klassifizierung erfolgt in der Informatik meist mittels kognitiver Systeme. Kognitive Systeme ermöglichen Informationsverarbeitung anhand ihrer Umgebung [5]. Wir betrachten im Folgenden technische kognitive Systeme. Einem solchen System wird eine Menge an Umgebungscharakteristiken übergeben anhand dessen es handelt.

Bei der Klassifizierung soll das System für jedes Datum entscheiden, welche Klassen zugeordnet werden. Dies sollte eindeutig geschehen, d.h. die Entscheidung soll nur anhand der übergebenden Musterchararakteristiken (Beispielsweise Farbe, Anzahl Ecken, Dichte) geschehen.

2.1 Künstliche neuronale Netze

Kognitive Systeme nehmen sich das menschliche Gehirn als Vorbild. Abstrahiert man das Gehirn in eine algorithmische Form lässt es sich als Menge von vernetzten Neuronen beschreiben. Jedes Neuron besteht aus gewichteten Eingängen, einer Aktivierungsfunktion mit Schwellenwert und einem Ausgang. Überschreitet die Summe der gewichteten Eingänge den Schwellenwert feuert das Neuron, ansonsten nicht. Siehe Abbildung 1.

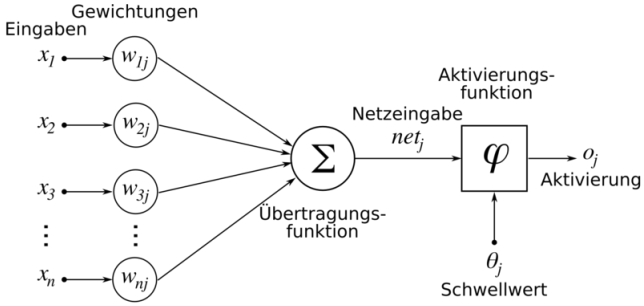

Abbildung 1. Ein Neuron mit seinen gewichteten Eingängen der Aktivierungsfunktion mit Schwellenwert. [6]

Ein Neuronales Netz ist ein gerichteter Graph aus Neuronen und Kanten, welche die Neuronen verbinden. Wir betrachten hier eine Implementierung in der das Netz aus drei Schichten besteht und vollständig vernetzt ist: Einer Eingabeschicht, welche nur die externe Eingabe des Netzes entgegennehmen. Eine versteckte Schicht, die alle Eingabeneuronen als Eingang haben und eine Ausgabeschicht, die mit allen Neuronen der versteckten Schicht vernetzt sind und deren Ausgabe die Netzausgabe beschreibt [7].

Ein künstliches neuronales Netz zur Klassifizierung nimmt als Netzeingabe die vorhandenen Mustercharakteristiken entgegen und ermittelt die Klassen des Datums.

2.2 Training

Zur Ermittlung aller Schwellenwerte und Gewichte des Neuronalen Netzes muss dieses trainiert werden. Das Verfahren der Backpropagation wird meist dafür genutzt [8]. Hierbei werden alle Schwellenwerte und Gewichte zufällig initialisiert. Danach erfolgt die Berechnung eines Lernbeispiels von Eingabe bis Ausgabe. Für das Lernbeispiel ist das gewünschte Ergebnis bekannt und es erfolgt ein Vergleich der Berechnung mit dieser. Eine daraufhin erfolgende Modifizierung der Gewichte rückläufig von Ausgabeschicht nach Eingabeschicht führt zu einer Verbesserung des Netzes. Nach endlicher Wiederholung dieses Vorgangs für eine Menge von Lernbeispielen ist das Neuronale Netz dann zur Klassifizierung der Problemstellung trainiert.

3 Nicht eindeutig benannte Klassifizierung

Lässt der Algorithmus es zu, kann ein Muster mehreren Klassen angehören.

Bei der Klassifizierung mit nicht eindeutig benannten Trainingsdaten gehört jedes Objekt nur einer Klasse an, Teile der Lernbeispiele sind aber, sowohl mit der richtigen Klasse, als auch mit noch weitere Klassen markiert.

Für diese mehrfach markierten Muster muss entschieden werden, welche der Klassen die Richtige sei, um diese dann als Lernbeispiel für das Trainieren benutzen zu können [1] Siehe Abbildung 2.

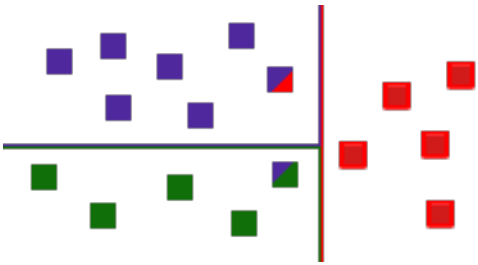

Abbildung 2. Nicht eindeutig benannte Klassifizierung: Zwei nicht eindeutig bezeichnete Punkte, die sich aber eindeutig einer Klasse zuordnen lassen

3.1 k-Nearest-Neighbor-Algorithmus

Bereits erwähnte Problemstellung ist demnach die Erkennung der richtigen Klasse wenn ein Trainingsbeispiel mit mehreren Klassen markiert ist. Eine Möglichkeit hierfür bietet der k-Nearest-Neighbor-Algorithmus. Dieser geht davon aus, dass alle Lernbeispiele untereinander über einen Abstand verfügen. Verfügt ein Lernbeispiel nun über mehreren Bezeichner, wählen wir als mutmaßlich richtigen denjenigen aus, der am häufigsten unter den k nächsten Nachbarn vorhanden ist [9]. Siehe Abbildung 3.

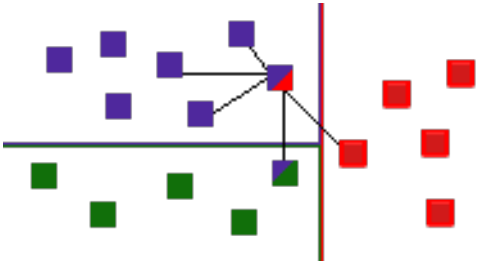

Abbildung 3. Für k = 5, würde der lila/rote Punkt im Nearest-Neighbor-Algorithmus als lila identifiziert werden, da der häufigste Bezeichner unter den nächsten fünf Nachbarn lila ist.

4 Implementierung

Die Implementierung erfolgt in Python [Anhang 1]. Es wird eine Mengen von Trainingsdaten generiert. Ein Neuronales Netz mit vorgeschaltetem 5-Nächster-Nachbar-Algorithmus und eins ohne vorgeschalteten Algorithmus werden gleich trainiert und ausgewertet. Aufgetragen wird eine Gegenüberstellung der Erkennungsraten der beiden Netze.

4.1 Setup

Der Algorithmus wird zur Veranschaulichung durch Beutel mit Getreide implementiert, die auf einem Reisbrett fallen gelassen werden. Jedes Feld kann maximal mit einem Korn belegt werden. Dabei wird der Beutel über einem gewissen Punkt fallen gelassen und die Körner werden zufällig um diesen Punkt verteilt. Jeder Beutel enthält zu einem gewissen Anteil Körner einer fiktiven Getreidesorte. Unterscheidungsmerkmal der Körner sei Länge, Dicke und Breite und unsere Implementierung soll danach wieder erkennen aus welchem Beutel ein Korn stammt. Die Körner sind dabei mit der Getreidesorte des Beutels aus dem sie fallen benannt. Ein Teil der Körner wird zufällig zusätzlich mit anderen Getreidesorten bezeichnet. Über die xy-Koordinaten der Körner auf dem fiktiven Reisbrett werden beim Trainieren für den k-Nearest-Neighbor-Algorithmus die nächsten Nachbarn bestimmt. Im Anhang kann eine Beispielverteilung für 48 Körner gefunden werden [Anhang 2].

4.2 Testdaten

Der Testlauf erfolgt mit 1000 bis 2000 Körnern je Beutel.

Die drei fiktiven Getreidesorten Nais, Veizen und Leis werden auf einem 300 x 300 großen Reißbrett verteilt.

4.3 Neuronales Netz

Die Implementierung erfolgt mittels eines vorwärts gerichtetem und vernetztem Neuronalen Netzes, dessen Eingangsschicht aus Identitätsneuronen besteht, welche die Identität anstatt einer Schwellenwertfunktion benutzen. Die Gewichte und Schwellenwerte werden mit Zufallsgrößen zwischen 1 und 100 initialisiert.

5 Ergebnisse

Folgend aufgelistet sind die Ergebnisse für fünf Testläufe, jeweils ohne Betrachtung der nicht eindeutig benannten Trainingsbeispiele und mit. Als k für den Nearest-Neighbor-Algorithmus wurde fünf gewählt. Siehe Tabelle 1.

Tabelle 1. Auflistung der Ergebnisse aus fünf Testläufen

Testlauf	Körner	nur eineindeutige	mit uneindeutigen
1	3843	66,0%	70,5%
2	4191	61,7%	64,1%
3	4991	71,3%	73,5%
4	4406	41,5%	42,8%
5	4702	52,3%	52,6%

6 Fazit

Wir sehen, dass mit Betrachtung von nicht eindeutig benannten Testmustern eine Klassifizierung, sowie das Lernen eines Neuronalen Netzes zur Klassifizierung möglich ist. Es zeigt sich eine Tendenz einer Verbesserung durch die Einbeziehung der nicht eindeutig benannten Trainingsbeispielen. In der Veröffentlichung sind weitere Möglichkeiten aufgezeigt aus dem Set der Bezeichner den Korrekten zu ermitteln. Diese sollten bessere Ergebnisse erzielen.

Anhang

[1] Implementierung in Python: .
http://severin.orth.privatepaste.com/3bf11411b4 .
[2] Körnerverteilung für 48 Körner auf einem 20x20 Reisbrett:.
http://severin.orth.privatepaste.com/acf53aa259 .

Literatur

[1] E. Hüllermeier and J. Beringer. Learning vom Ambiguosly Labeled Examples.
[2] Grigorios Tsoumakas and Ioannis Katakis Multi-Label Classification: An Overview In International Journal of Data Warehousing & Mining, 3(3), 1-13, July-September 2007.
[3] R. Jin and Z. Ghahramani. Learning with multiple Labels. In 16th Annual Conference on Neural Information Processing Systems, Vancouver, Canada 2002.
[4] D. Michie , D. J. Spiegelhalter , C.C. Taylor . Machine Learning, Neural and Statistical Classification.
[5] G. Görz, C.-R. Rollinger, J. Schneeberger. Handbuch der künstlichen Intelligenz. Oldenbourg Wissenschaftsverlag GmbH, München 2005.
[6] Chrislb. Schema eines künstlichen Neurons. Wikimedia Commons, abgerufen am 04.01.2011.
http://commons.wikimedia.org/wiki/File:ArtificialNeuronModel_deutsch.png
[7] C. Borgelt, F. Klawonn, R. Kruse, D. Nauck. Neuro-Fuzzy-Systeme. Friedr. Vieweg und Sohn Verlagsgesellschaft, Wiesbaden 2003.
[8] R. Hecht-Nielsen. Theory of the backpropagation neural network. In Neural Networks, 1989. IJCNN., International Joint Conference on.
[9] D.V. Dasarathy, editor. Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques. IEEE Computer Society Press, Los Alamitos, California, 1991. 1994.

Interactive Machine Learning for Classification

Julia Hempel
julia.hempel@st.ovgu.de

Faculty of Computer Science
Otto-von-Guericke University Magdeburg

Abstract. To evaluate large unstructured datasets, the data need to be classified. Machine learning methods are used widely to classify data automatically. However, classical machine learning systems require much expert knowledge and cannot adapt to their environment after training. Interactive machine learning suggests a more flexible approach by enabling the user to change the reasoning of the machine-learning system interactively. This causes many challenges both in algorithm design and human-computer interaction. In this paper we will give a brief introduction to interactive machine learning and present recent research results concerning appropriate algorithms and user interfaces.

Keywords: Machine learning, Interaction, Classification, HCI

1 Introduction

Nowadays terabytes of data accrue every day, e.g. pictures of the earth made by satellites, video recordings of security cameras or the daily mails. All this unstructured data need to be evaluated. To evaluate data, it is useful to find patterns in the data and classify it. Because of the huge amount of data it is not longer possible to categorize it manually. Instead it needs to be classified automatically. To assign one object to a category, the category needs to be defined. In many cases the definition of a category is not trivial, e.g. if a system shall recognize concepts like abnormality, importance or danger [Amershi et al. 2010]. One common approach to build such concepts is using machine learning. The definition of general categories is learned here from positive and negative examples of the category [Mitchell 1997]. The resulting classifier is used to categorize data automatically. For example, the NASA uses machine-learning methods to automatically classify objects in the Sky Survey.
In classical machine learning (CML) the system is trained first and after that it can be used interactively. In case of errors or changes in the overall system, the CML system has to be retrained. To train a CML algorithm much knowledge of the underlying algorithm is necessary. The features that shall be trained need to be selected and parameters of the algorithm need to be defined. Unfortunately most users do not have such a deep understanding. Moreover the construction of example data requires domain knowledge, which is difficult to obtain. To avoid

those problems and to make machine learning more flexible, another approach has been developed.
Interactive machine learning (IML) enables the user to change the reasoning of a machine-learning system interactively. The system provides feedback to the user, the user makes corrections and the system adapts to the users feedback [Fails and Olsen 2003]. In this way the users understanding and trust in the machine learning system can be improved and the accuracy of the system, too. However, there are several challenges in IML. The machine-learning algorithms need to be fast to enable real-time usage and involve the users feedback. Moreover appropriate techniques in human-computer interaction are necessary to enable a rich collaboration between humans and machine-learning systems.
In this paper we will provide an overview of recent research results in the field of interactive machine learning. For this purpose, we will first discuss the differences between CML and IML. After that we will introduce the challenges in IML algorithms and human-computer interaction.

2 Differences between CML and IML

Machine learning provides a possibility to build classifiers automatically. In contrast to interactive machine learning (IML) models, classical machine learning (CML) models are not interactive and generally slow to train [Fails and Olsen 2003]. In figure 1 the CML model is summarised. To build a classifier, first of all the features need to be selected. After that the training with prepared data takes place. When the training finished, the classifier is build. The CML model is optimized for efficient classification. So the classifier usually runs quickly and enables real-time usage. This optimization of run time goes with the expenses of long training periods [Amershi et al. 2010].

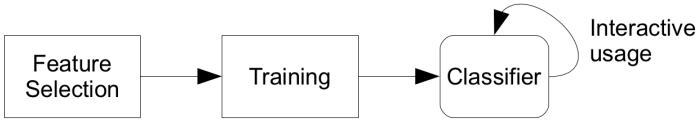

Fig. 1. Classical machine learning model [Fails and Olsen 2003]

The user of CML can affect the building of the classifier only by choosing parameters of the algorithm and selecting the features. However, the correct selection of the parameters and features requires a deep understanding of the underlying algorithm [Ware et al. 2001]. Unfortunately most users have only a limited technical understanding and even experts select often by trial and error. At the same time CML algorithms are very sensitive to feature selection

[Fails and Olsen 2003]. So the quality of the classifier suffers greatly from a false decision. Also the preparation of the training data is very important for the quality of the classifier, because the distribution of the training examples should fit the distribution of the perspective application [Mitchell 1997]. If there occur situations during the operation of the classifier that were not trained at all, the performance of the classifier might be weak. As a result the construction of the training data need to be done by a domain expert.
With the prepared data the training is done offline [Amershi et al. 2010] and often requires extensive training periods [Amershi et al. 2011]. After training is finished there is no possibility to change the classifier. As a result CML must be retrained if there are any errors or changes in the system [Amershi et al. 2011].
In contrast to CML, IML is much more flexible. The classifier changes dynamically, because it constantly learns from human guidance [Amershi et al. 2011]. For that IML model uses a train-feedback-correct cycle that is summarized in figure 2. The IML algorithm classifies the new data and gives rapid feedback to the user. Now the user can correct errors in the classification immediately. This correction does not require a deep understanding of the underlying algorithm. Also inexperienced users are able to build a classifier that way. Moreover the domain knowledge of the user can be introduced directly in the building of the algorithm.

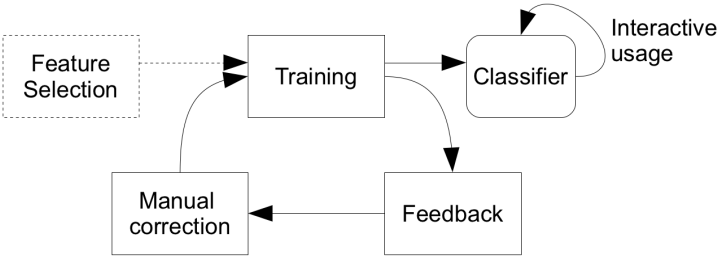

Fig. 2. Interactive machine learning model [Fails and Olsen 2003]

In CML the quality of an algorithm is measured by its inductive power. That is, how well the algorithm will perform on new data based on the extrapolations made on the training data [Fails and Olsen 2003]. A high inductive power requires a careful design of the learning algorithm and long training times. Because in IML the user is able to correct errors very quickly, a strong induction is not relevant. Here the speed of the algorithm is focused. To allow an interactive use of the algorithm the training part of the loop needs to take less than 5 seconds. So the interactive cycle can be iterated very quickly and more frequently [Fails and Olsen 2003]. As well as in CML overfitting can occur in IML, but it also can be corrected easily by the user [Ware et al. 2001]. So complex algorithms to prevent overfitting are not necessary. This increases the speed of the learning

algorithm.
In the next chapter we will discuss requirements and possible implementations of such machine-learning algorithms.

3 IML algorithms

In general most CML algorithms can be adapted to IML. Though there are some requirements in IML that restrict the selection. As explained before an IML algorithm has to learn very fast, because training is done interactively. As a result some types of learning algorithms are not appropriate for IML, e.g. neural networks. Even though artificial neural networks are appropriate for problems with many features and can classify instances very fast, the training period is too long for interactive usage [Mitchell 1997].
Moreover the selection of the machine-learning algorithm depends on the application field. Below we will introduce two examples of IML applications and their algorithmic implementation.

Image processing with CRAYONS

[Fails and Olsen 2003] built a tool to create image classifiers. The tool is addressed to user interface designers, who want to integrate recognition or perception in their user interface. The goal is to enable the designers to create image classifiers by simply selecting areas in a picture. For example a classifier that recognizes skin can be built by painting lines on areas of skin in a picture. No understanding of the underlying algorithms should be necessary. As a result the machine-learning algorithm has to make the feature selection automatically.
In summary the requirements for the machine-learning algorithm are:

- Learn/train very quickly
- Accommodate many features
- Perform feature selection
- Allow plenty of training examples

[Fails and Olsen 2003] selected decision trees as machine-learning algorithm. Decision trees have the advantage that the algorithm itself realises the process of feature selection by selecting the feature to test at each node in the tree [Mitchell 1997]. In CRAYONS the feature with the lowest impurity (entropy) in a collection of training examples is selected. The impurity is a measure of effectiveness of a feature in classifying the training data [Mitchell 1997]. If a set of examples belongs to the same class, the impurity is zero. The feature with the lowest impurity is chosen and the set is divided. After that the algorithm is applied recursively to the divided subset. So only features that provide discrimination are used.
[Fails and Olsen 2003] found, that the implementation based on decision trees achieves fast training-times and effective feature selection. As a result decision trees are appropriate for usage in IML systems.

Network alarm triage with CueT

[Amershi et al. 2011] developed a tool that automatically generates recommendations for network alarm triages and visualise them. The goal of the tool is to assist network operators to find and fix problems. For that it groups and prioritizes a continuous stream of alarm signals. The importance and correlation between alarm signals is learned interactively from the decisions of the operator. To implement the learning mechanism [Amershi et al. 2011] uses a nearest neighbour classification. The incoming alarms are classified by similarity, which is measured using a distance function. The distance is calculated based on predefined distance metrics e.g. device type and event name. For an incoming alarm the average distance between this alarm and each alarm in a category is calculated using Mahalanobis distance. The importance and correlation among individual features is taken into account by the covariance matrix. This matrix is learned from the operator actions. For that an online metric learning algorithm is applied [Jain et al. 2008]. Learning the matrix from the operator actions has the advantage that no expert tuning is necessary, because this expert knowledge is difficult to obtain and does not adapt immediately if the network changes.
The related user study by [Amershi et al. 2011] shows, that CueT enables network operators to triage alarms significantly faster and improves their accuracy.

In a nutshell, there are many different approaches to implement IML algorithms. The requirements of the algorithms depend strongly on the specific applications field. Typical requirements to IML algorithms are:

- Short training periods
- Enabling many training examples
- Performing feature selection
- Involving user feedback

However, involving user feedback effectively is not only a requirement in algorithm design, but also a challenge in human-computer interaction. Below we will discuss recent findings in the field of interaction between users and machine-learning systems.

4 IML user interfaces

In the chapter above we have shown that it is possible to change the reasoning of a machine-learning system interactively. This offers great opportunities in the collaboration of human and computer. The users understanding and trust in the machine-learning system can be improved and the accuracy of the system as well [Stumpf et al. 2009]. However, it is a challenge to develop effective strategies for end-users interacting with machine learning, because a shared understanding of the concept needs to be reached [Amershi et al. 2010]. Although many applications today use machine-learning algorithms, only little research is done in the field of effective interaction between users and those algorithms. There already exist applications that implement interactive machine learning,

but they mostly enable only simple right/wrong judgements to the user [Stumpf et al. 2009]. A common example are interactive email spam filters, which adapt to the end-users preferences by learning from the No Spam/Spam classification of the user. However to enable a rich collaboration between machine-learning systems and users more feedback need to be given from both users and systems. [Stumpf et al. 2009] found that interactions with a machine-learning system can be divided in three components. These components conform to the IML model summarized in figure 2.

- System's ability to explain it's reasoning to the user (Feedback)
- Users reasoning corrections (Correction)
- System's ability to make use of the user feedback (Training)

Below we will discuss different research approaches and results in each component.

System's ability to explain it's reasoning to the user

To enable the user to give rich feedback to the system, the user needs to understand its reasoning. In a user study made by [Stumpf et al. 2009] three different explanation paradigms were tested:

- rule-based explanation
- keyword-based explanation
- similarity-based explanation

[Stumpf et al. 2009] found that the rule-based explanation paradigm was the most understandable, second best was the keyword-based one. The similarity-based explanation caused serious problems to the user. However no paradigm was a real winner, because the preferences of the participants in the study were not unit. Because of that [Stumpf et al. 2009] suggests the support of multiple paradigms of explanation.
Another approach to explain the systems reasoning to the user was developed by [Ware et al. 2001]. He built a tool to construct a classification model visually. By drawing a polygon in the data visualisation, the user builds a classifier. Each polygon represents a node in a decision tree. This decision tree can be examined by the user, too. [Ware et al. 2001] found that humans are able to build good classifiers that way, if the data can be illustrated in two dimensions and the dataset is not too large. The advantage of this approach is that the inductive process is illustrated in every step. So the user gets educated about the meaning of the result.

Users reasoning corrections

User corrections can be made though critic and adjustments. [Stumpf et al. 2009] found in her study, that users give rich feedback in many different ways e.g. by reweighting features or combining features. In general the participants were more

accurate than the machine but they also did mistakes. As a result user feedback can also introduce errors into the reasoning.
Interaction techniques to adjust the classifier were discussed by [Amershi et al. 2010]. He developed a tool to train concepts for re-ranking web image search results. The user trains the classifier by labelling example images. Instead of giving direct feedback to the system, the user is supposed to experiment with different labellings and compare the results. Finally the user has to decide upon a label that guides the system to learn the desired concept. To enable the user to experiment with the machine learning system, [Amershi et al. 2011] implemented two approaches and compared them in a user study.

1. Undo/redo mechanism
2. History visualization showing previous models

The evaluation shows, that the history visualization was not helpful. More time was spend without improving the overall model quality. In contrast, the revision mechanism did very well. The participants used it often and achieved better final models in the same amount of time. Moreover the undo/redo function was important to users when the quality of the model appears to drop. So including a revision mechanism can improve the performance of users training a machine learning system.

System's ability to make use of the user feedback

The third component to enable rich human-computer collaboration in IML is the systems ability to make use of the given user feedback. The simplest technique to introduce feedback is to allow the users to generate new training data as implemented in [Fails and Olsen 2003] and [Amershi et al. 2011]. However, there are more complex techniques, which enable more sophisticated user feedback and as a result deeper changes in the machine-learning algorithm. One example for such a technique is the co-training algorithm introduced in [Stumpf et al. 2009].

In a nutshell, the user interface of an IML system should fulfil the following requirements:

- Feedback to the user should be provided using multiple explanation paradigms, for example a combination of a visualisation and rule-based explanations.
- User should be allowed to offer feedback in different ways and undo/redo her actions
- The System should apply the users feedback in a meaningful way.

5 Conclusion

Interactive machine learning provides great opportunities by enabling users to affect machine-learning algorithms. In a train-feedback-correct cycle the system offers a classification, the user corrects this classification and the system uses

this feedback to improve the classifier [Fails and Olsen 2003]. This interaction between user and machine-learning system makes a rich collaboration possible. On one hand, the users understanding and trust in the machine-learning system improves. On the other hand, the accuracy of the system increases, because the domain knowledge of the user is involved directly and errors can be corrected rapidly.
To enable interactive training of a machine-learning system, the algorithm must be fast to train and integrate user feedback well. Moreover the user interface must be appropriate. The system has to provide understandable feedback to the user, e.g. by multiple explanation paradigms like a visualisation combined with rule-based explanation [Stumpf et al. 2009]. Also the user should be allowed to give feedback in different ways and undo/redo her actions. In the last step the algorithm has to integrate the feedback of the user in a meaningful way.
All in all interactive machine learning is a very innovative and promising field at the intersection between computational intelligence and human-computer interaction. Much more research needs to be done in this field to enable humans to work hand-in-hand with a machine-learning system.

Bibliography

[Amershi et al. 2010] Amershi, S. ; Fogarty, J. ; Kapoor, A. ; Tan, D.: Examining multiple potential models in end-user interactive concept learning. In: *Proceedings of the 28th international conference on Human factors in computing systems* ACM (Veranst.), 2010, S. 1357–1360

[Amershi et al. 2011] Amershi, S. ; Lee, B. ; Kapoor, A. ; Mahajan, R. ; Christian, B.: Human-Guided Machine Learning for Fast and Accurate Network Alarm Triage. In: *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011

[Fails and Olsen 2003] Fails, Jerry A. ; Olsen, Dan R.: Interactive machine learning. In: *Proceedings of the 8th international conference on Intelligent user interfaces*. New York, NY, USA : ACM, 2003 (IUI '03), S. 39–45. – URL http://doi.acm.org/10.1145/604045.604056. – ISBN 1-58113-586-6

[Jain et al. 2008] Jain, P. ; Kulis, B. ; Dhillon, I.S. ; Grauman, K.: Online metric learning and fast similarity search. In: *Advances in Neural Information Processing Systems* 22 (2008)

[Mitchell 1997] Mitchell, T.M.: *Machine learning*. Mcgraw-Hill Higher Education, 1997. – 368 S

[Stumpf et al. 2009] Stumpf, S. ; Rajaram, V. ; Li, L. ; Wong, W.K. ; Burnett, M. ; Dietterich, T. ; Sullivan, E. ; Herlocker, J.: Interacting meaningfully with machine learning systems: Three experiments. In: *International Journal of Human-Computer Studies* 67 (2009), Nr. 8, S. 639–662

[Ware et al. 2001] Ware, M. ; Frank, E. ; Holmes, G. ; Hall, M. ; Witten, I.H.: Interactive machine learning: letting users build classifiers. In: *International Journal of Human-Computer Studies* 55 (2001), Nr. 3, S. 281–292

Explorative Datenanalyse mithilfe hierarchischer Fuzzy-Regelsystemen

Anett Hoppe

Otto-von-Guericke-University of Magdeburg
Universitätsplatz 2, D-39106 Magdeburg, Germany
anett.hoppe@st.ovgu.de

Abstract. Regelsysteme haben in der Analyse großer Datenmenge bisher kein besonderes Interesse erregt. Sie erweisen sich in vielen Fällen als zu einfach, um tatsächlich sinnvolle Informationen hervorzubringen, oder als zu anfällig gegen verrauschte oder fehlerhafte Daten empfunden. Die vorliegende Arbeit stellt eine Methode vor, die statt eines einfachen Regelsystems eine Hierarchie von Regelbasen über einen Datensatz aufbaut. Auf diese Art und Weise kann ein grober Überblick über die Daten in Regelform vermittelt, gleichzeitig jedoch über Nutzerinteraktion ausgewählte Interessengebiete genauer, bis hin zum einzelnen Datenpunkt betrachtet werden.

Keywords: fuzzy, rule systems, hierarchical, visualization, exploration

1 Einleitung

Während Fuzzy-Regel-Systeme insbesondere in mechatronischen Problemstellungen bereits gute Dienste leisten , sind sie in der Anwendung zur Analyse größerer Datenmengen bisher weitgehend unbeachtet geblieben. Einen Grund dafür bilden einige Schwachstellen der Regelsysteme, die mit häufigen ungünstigen Eigenschaften realweltlicher Daten korrelieren. Große, reale Datenbasen sind oftmals durchsetzt von unvollständigen, verrauschten oder fehlerhaften Datensätzen. Zudem kann die Anzahl der einen Datenpunkt beschreibenden Attribute sehr hoch sein, die Daten also sehr hochdimensional sein. Auf dieser Basis erstellte Regelsysteme, die die Daten vollständig, korrekt und widerspruchsfrei zu beschreiben, wären für den Nutzer aufgrund ihrer Komplexität nur schwer zu interpretieren. Eine hohe Anzahl berücksichtigter Features führt gegebenenfalls zu sehr langen und daher unübersichtlichen Einzel-Regeln. Die Notwendigkeit, durch Rauschen oder Fehler entstandene Outlier-Artefakte ebenfalls korrekt zu beschreiben führte unter Umständen zu zahlreichen zusätzlichen Regeln, die die Regelbasis komplexer und schwerer verständlich machen. Das resultierende Regelmodell wäre also womöglich vollkommen korrekt, aber schlußendlich nutzlos, da es vom menschlichen Nutzer nicht mehr interpretierbar ist.
Ein möglicher Lösungsansatz ist die Einführung eines zusätzlichen Arbeitsschrittes der Datenaufbereitung, in dem auffällig abweichende oder unvollständige

Datensätze aus der Datenbasis entfernt werden und hoch korrelierte Attribute zusammengefasst werden. Dies löst zwar zunächst das Problem, führt aber unter Umständen zu ungewolltem Informationsverlust. Denn schließlich kann nicht sicher gestellt werden, daß eine entfernte Abweichung tatsächlich immer fehlerhaft ist – es könnte sich im Gegenteil um ein sehr spezielles, interessantes Phänomen in den Daten handeln.
Der in der vorliegenden Arbeit dargestellte Ansatz setzt darauf, an dieser Stelle den Nutzer wieder in den Wissensextraktionsprozess einzugliedern und sein Expertenwissen, Kontextbewusstsein oder auch Intuition einzubringen. Es wird die Möglichkeit gegeben, die erzeugte Regelbasis explorativ zu erfassen. Hierzu wird ein hierarchisches System von Regeln aufgebaut. In der obersten Schicht werden nur wenige Regeln geboten, die den kompletten Datensatz sehr grob beschreiben – sozusagen eine Übersichtsdarstellung. In dieser Darstellung kann der Nutzer interaktiv einen für ihn interessanten Teilbereich auswählen, um in die nächste Hierarchieebene zu gelangen. In dieser wird der zuvor gewählte Teilbereich etwas genauer, wiederum in Regelform dargestellt. Die Komplexität wird durch die Aufteilung der Daten erträglich gehalten. Die Möglichkeit des iterativen Absteigens in den Hierarchieebenen befähigt den Nutzer trotzdem eine Betrachtung feingranularer Phänomene.

2 Aufbau hierarchischer Regelbasen

Nachdem zahlreiche Algorithmen zum automatischen Erzeugen von Regelbasen bereits entwickelt wurden , liegt die eigentliche Schwierigkeit in der Erstellung der Hierarchie. Der folgende Abschnitt stellt zunächst kurz das hier genutzte, die Regelbasis erzeugende Verfahren vor und widmet sich dann ausführlicher der Art und Weise, wie daraus eine Hierarchie von Regeln gewonnen werden kann.

2.1 Zugrundeliegender Clusteringalgorithmus

Eine genaue Beschreibung des Verfahrens findet sich in [1]. Die Trainingsbeispiele werden dem Algorithmus zum Aufbau der Regelbasis fortlaufend pr¨sentiert. Wenn notwendig, wird dem Modell eine neue Regel hinzugefügt, um das neue Trainingsbeispiel abzudecken oder die Abdeckung einer bereits vorhandenen Regel angepasst. Dabei sind folgende drei Schritte für jedes Beispiel durchzuführen:

Abdeckung: Wird das neu eingeführte Trainingsbeispiel von einer bereits existierenden Fuzzy-Regel überdeckt, wird lediglich deren Kernregion bis zur Abdeckung des neuen Beispiels erweitert und die Gewichtung der entsprechenden Regel erhöht.

Hinzufügen: Ist das neue Muster noch nicht von einer Regel abgedeckt, muß eine neue Regel in das System eingefügt werden. Diese neue Regel wird zunächst mit unbeschränkten Wertebereichen angelegt, ihre Gültigkeit erstreckt sich also zunächst über den kompletten Attributraum.

Einschränken: Sind Muster unkorrekt von einer Fuzzyregel der konkurrierenden Klasse überdeckt, wird die Supportregion der entsprechenden Regel soweit eingeschränkt, dass der Konflikt entfernt wird. Dabei ist der Volumenverlust des von der Regel abgedeckten Werteraums möglichst gering zu halten.

Gegebenenfalls müssen die kompletten Trainingsdaten mehrfach durchlaufen werden, damit der Algorithmus terminiert. [4]

2.2 Erzeugen der Hierarchie

Der dargestellte Fuzzy-Lerner sucht die Wertebereiche der erstellten Regeln möglichst weit zu fassen, und damit die Länge der erstellten Regeln nach Möglichkeit zu beschränken. Er zeigt jedoch die bekannten Schwächen hinsichtlich verrauschter oder fehlerhafter Datensätze. Die durch sie hervorgerufenen Artefakte führen zu zusätzlichen und die Regelbasis unnötig verkomplizierenden Regeln.
Vom System sind diese überflüssigen Regeln jedoch nicht zuverlässig von korrekten zu unterscheiden. Da im hierarchischen System die Relevanzbewertung jedoch dem Nutzer überlassen bleibt, steht lediglich die Aufgabe, Regeln zu identifizieren, die für die nächste, gröbere Hierarchiestufe nicht dienlich sind. Hierbei werden korrekte oder durch Rauschen/Fehler erzeugte Regeln nicht unterschieden, diese Entscheidung kann also allein auf Grundlage der Daten getroffen werden.

Für die Entscheidung, ob eine Regel im Schritt zur nächstabstrakteren Hierarchieebene gefiltert werden muß, müssen effiziente Bewertungskriterien eingeführt werden.
Die einfachste ist dabei der Anteil durch die Regel abgedeckter Datenpunkte an der Gesamtdatenbasis:

$$Relevanz(R) = \frac{von R abgedeckte Datenpunkte}{Gesamtzahl Datenpunkte}$$

Eine weitere Möglichkeit besteht darin, den Informationsverlust zu messen, der entsteht, wenn die Regel R von der Regelbasis entfernt wird:

$$Relevanz(R) = I(alle Regeln) - I(Regel ohne R)$$

dabei ist die Funktion I eine den Informationsgehalt einer Regelbasis bemessende Größe. Hierfür können zum Beispiel der Gini-Index oder die Entropy herangezogen werden. .

Zum Erstellen der eigentlichen Regelhierarchie wird nun zunächst ein vollständiges Modell über alle Daten gelernt. In der resultierenden, riesigen Regelbasis werden mithilfe des gewählten Relevanzmaßes die Regel identifiziert, die in Relation zum Restmodells als Outlier angesehen werden können. Von diesen Outlier-Regeln überdeckte Datenpunkte werden aus der Datenbasis entfernt und mit den verbleibenden die Regelbasis für die nächsthöhere Hierarchieebene

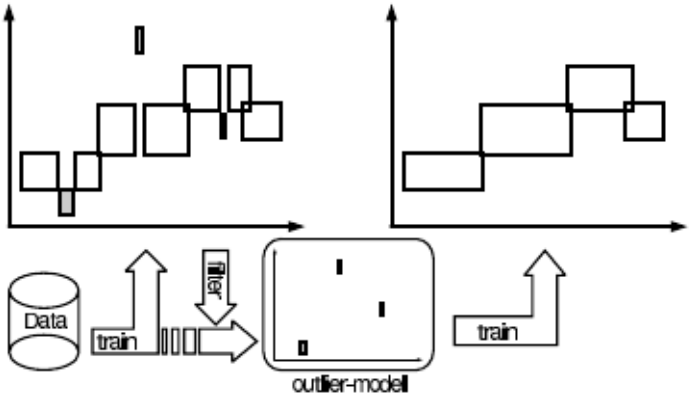

Fig. 1. Zwei-stufiges Outlierfiltering [4]

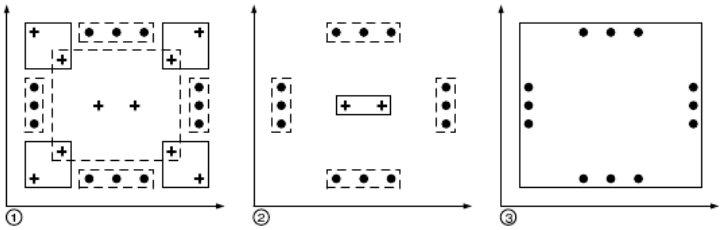

Fig. 2. Beispiel: Regelhierarchie, basierend auf regelbasiertem Filtern [4]

trainiert (siehe Bild 1.
Mithilfe eines mehrstufigen Prozesses kann so die komplette Modelhierarchie erhalten werden. Die Filterprozedur wird rekursiv angewandt, bis alle noch enthaltenen Regeln eine Relevanzschwelle bedingen und das Verfahren terminiert. Bild 2 zeigt eine beispielhaft erstellte Hierarchie für einen einfachen Datensatz mit zwei Klassen (Kreuz und Kreis). Die Rechtecke verdeutlichen dabei nur die Kernregionen der Cluster. Es wird eine dreistufige Hierarchie aufgebaut, mit einer Outlier-Schwelle von 2 und eine Filterschwelle von 1. Das heißt, Regeln, die zwei oder weniger Datenpunkte beschreiben, werden dem Outliermodell hinzugefügt; nur Datenmuster, die in eine Kernregion eines Outlierclusters fallen, werden tatsächlich aus dem Datenmodell entfernt.
Bild 2(1) zeigt das ursprüngliche Regelmodell, mit neun Regeln, die im ersten Arbeitsschritt ermittelt wurden. Die vier Regeln in den Ecken bedecken jeweils nur zwei Datenpunkte, werden also dem Outliermodell hinzugefügt und die entsprechenden Datenpunkte aus dem Datensatz entfernt. Bild 3(2) zeigt das resultierende Modell der nächsten Ebene, es enthält nur noch fünf Regeln.

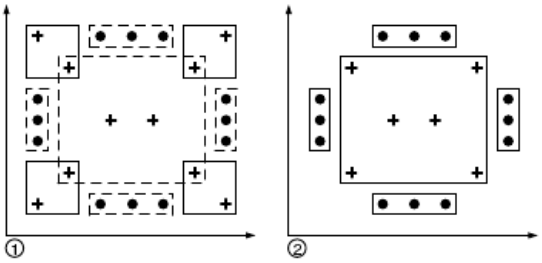

Fig. 3. Beispiel: Hierarchie basierend auf model-basiertem Filtering [4]

Im letzten Schritt wird das finale Regelmodell entworfen, nur eine Regel und überdeckend alle übrig gebliebenen Instanzen.
Es ist zu sehen, dass potenziell in einem einzigen Arbeitsschritt zahlreiche Instanzen aus dem Datensatz entfernt werden, die Hierarchie möglicherweise sehr flach ausfällt. Im ursprünglichen Ansatz wird nur überprüft, ob eine Instanz von einer Outlierregel überdeckt wird, also in diesem Moment nur der Informationsgehalt dieser einen Regel in Betracht gezogen. Um eine aussagekräftigere Hierarchie zu erhalten, wird dieser Test erweitert, um einen modell-weiten Blick zu bewahren. In der neuen Herangehensweise wird eine von einer Outlierregel überdeckte Instanz nur dann aus dem Datensatz entfernt, wenn sie nicht gleichzeitig auch von einer der regulären Regeln beeinflusst wird. Bild 3 zeigt das Ergebnis dieser Änderung auf dem bereits in Bild 2 genutzten Datensatz.

2.3 Klassifikation mithilfe der Hierarchy

Es ist bereits bekannt, wie nicht-hierarchische Regelsysteme zur Klassifikation verwendet werden können, um fuzzy-Zugehörigkeitswerte für neue Instanzen zu bestimmen (genauere Beschreibung in [4]). In der hierarchischen Form müssen nun die Ergebnisse jeder einzelnen Ebene miteinander kombiniert werden, um die Zugehörigkeitswerte der neuen Instanz zu den Klassen endgültig zu ermitteln (Bild 4.
Es können an dieser Stelle zwei Ansätze verfolgt werden:

Summe: Es wird die Fuzzy-Summe über die Zugehörigkeitsgrade einer jeden Ebene gebildet und dieser als Endergebnis ausgegeben.

Bottom-Up-Strategie: Die Ausgabe wird anhand der Hierarchieschicht bestimmt, die das neue Muster zuerst überdeckt.

2.4 Mögliche Visualisierungen

Parallele Koordinaten Parallele Koordination [7,8] erlauben die Darstellung multidimensionaler Daten in zwei Dimensionen. Die multiplen Dimensionen des

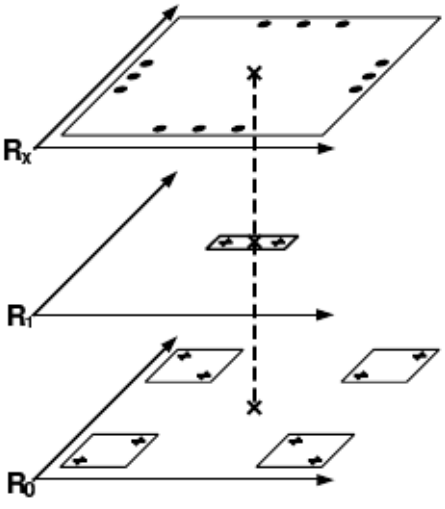

Fig. 4. Einfügen einer neuen Instanz und Klassifikation [4]

Problems werden dabei auf mehrere vertikale Achsen projiziert, sodass ohne Informationsverlust zweidimensionale Muster entstehen. Eine genaue Beschreibung, wie die Darstellung von fuzzy-Regeln in parallelen Koordinaten geschehen kann, findet sich in [2]. Zur kurzen Veranschaulichung soll wiederum der Iris-Datensatz [3] herangezogen werden. In Bild 5 findet sich eine flache Darstellung der erzeugten Regelbasis, ohne Hierarchie. In Bild 6 erfolgt die Darstellung des gleichen Datensatzes allerdings in Form einer hierarchischen Regelbasis. Die oberste Schicht enthält lediglich drei Regeln, die generelle Trends in den Daten verdeutlichen. In den niedrigeren Leveln wird die Darstellung feingranularer, Einzelphänomene werden sichtbar.

Multi-Dimensionale Skalierung Zur Darstellung hochdimensionaler Datenräume können multi-dimensionale Skalierungstechniken eingesetzt werden, um diese auf niedrig-dimensionalere, leichter darstellbare Räume abzubilden. Um dabei Informationsverlust zu vermeiden, suchen MDS-Methoden, die paarweisen Distanzverhältnisse zwischen den Datenpunkten bei der Abbildung zu erhalten. Es wird hierbei eine adäquate Fehlerfunktion minimiert.
Nach der Transformierung in niedrige Dimensionen können traditionelle Methoden zur Darstellung verwendet werden. Eine genauere Beschreibung des Vorgehens findet sich in [6]. Das Ergebnis des multi-dimensionalen Skalierens für den Iris-Datensatz findet sich in Bild 7.

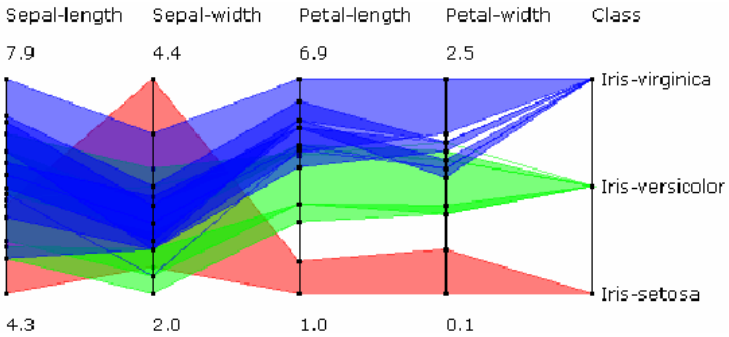

Fig. 5. Flaches Regelsystem für den Iris-Datensatz [5]

References

1. M.R. Berthold. Mixed fuzzy rule formation. *International journal of approximate reasoning*, 32(2-3):67–84, 2003.
2. M.R. Berthold and L.O. Hall. Visualizing fuzzy points in parallel coordinates. *Fuzzy Systems, IEEE Transactions on*, 11(3):369–374, 2003.
3. R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(7):179–188, 1936.
4. T. Gabriel and M. Berthold. Constructing hierarchical rule systems. *Advances in Intelligent Data Analysis V*, pages 76–87, 2003.
5. T. Gabriel, A. Pintilie, and M. Berthold. Exploring hierarchical rule systems in parallel coordinates. *Advances in Intelligent Data Analysis VI*, pages 741–741, 2005.
6. T. Gabriel, K. Thiel, and M. Berthold. *Multi-dimensional scaling applied to hierarchical rule systems*. Bibliothek der Universität Konstanz, 2009.
7. A. Inselberg. Multidimensional lines. *Parallel Coordinates*, pages 63–113, 2009.
8. A. Inselberg and B. Dimsdale. Multidimensional lines ii: proximity and applications. *SIAM Journal on Applied Mathematics*, pages 578–596, 1994.

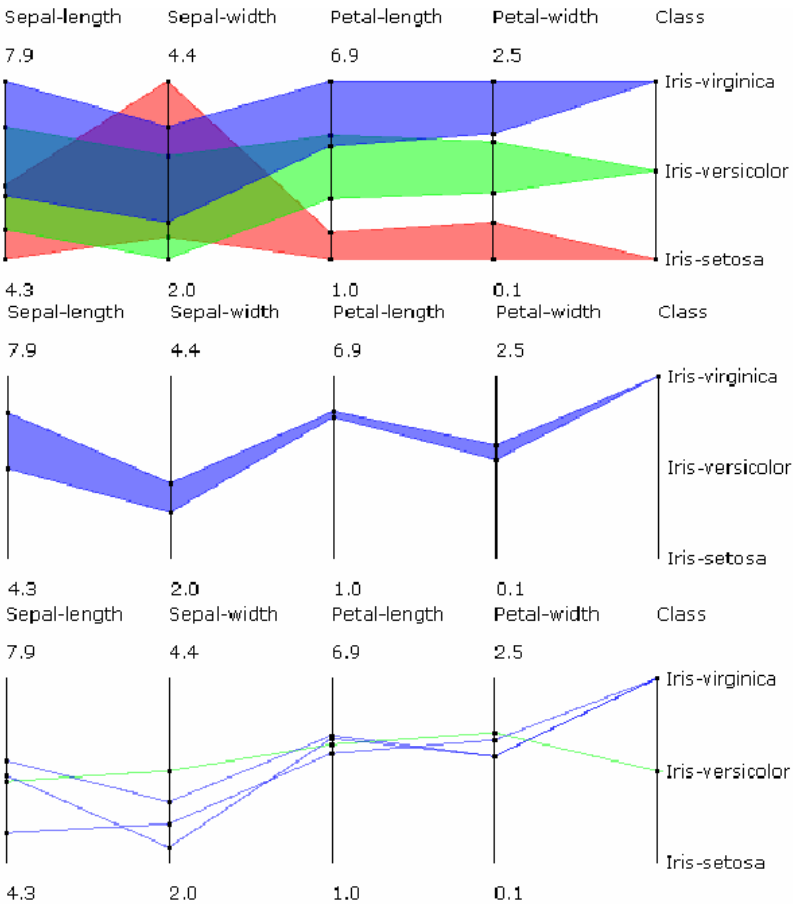

Fig. 6. Hierarchisches Regelsytem des Iris-Datensatzes in parallelen Koordinaten [5]

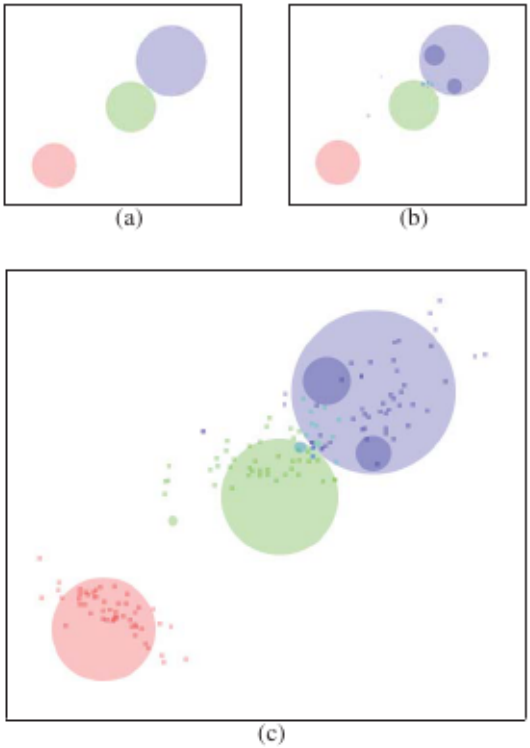

Fig. 7. Zwei-Schichten-Hierarchie, trainiert auf dem Iris-Datensatz: (a) oberste Ebene, drei Regeln, (b) zusätzliche elf Regeln aus der zweiten Ebene, (c) alle Datenpunkte

Interaktive Exploration von Fuzzy-Clustern unter Nutzung von Neighborgrams

Anett Hoppe

Otto-von-Guericke-University of Magdeburg
Universitätsplatz 2, D-39106 Magdeburg, Germany
anett.hoppe@st.ovgu.de

Abstract. Es wird eine interaktive Methode zur Erzeugung von Fuzzy-Clustern ein kleinen bis mittelgroßen (Teil-)Datensätzen vorgestellt. Basis bildet dabei eine neuartige Datenstruktur, das sogenannte Neighborgram, eine eindimensionale Darstellung der Nachbarschaft eines Datenpunktes. Die Einfachheit dieser Struktur bietet übersichtliche Möglichkeit der Visualisierung und kann in Kombination mit geeigneten Interaktionstechniken eine Grundlage zur visuellen Exploration des Clusterergebnisses bieten.

Keywords: fuzzy clustering, neighborgrams, visual exploration

1 Einführung

Dank moderner Technik und der Weiterentwicklung von Speichermedien ist es heutzutage möglich, immense Datenmengen automatisiert aufzunehmen und für die spätere Verarbeitung vorzuhalten. Dies geschieht bereits bei sehr einfachen Vorgängen des täglichen Lebens – der Einkauf im Supermarkt liefert Daten, die einer besseren Sortierung der Waren im Markt dienen können; automatische Fahrkartenscanner in Großstädten ermöglichen eine bessere Planung und Vorhersage von hohen Passagieraufkommen in öffentlichen Verkehrsmitteln.
Je mehr Daten dabei für die Auswertung zur Verfügung stehen, desto allgemeiner ist die aus ihnen extrahierbare Aussage. So kann man aus den Einkaufsdaten *eines* Kunden Aussagen über das Einkaufsverhalten genau dieses einen Kunden treffen – und anhand der erkannten Muster Sortierung oder Preis der Waren optimieren. Werden allerdings die Daten zahlreicher Kunden zugrunde gelegt, kann günstigenfalls ein Optimum geschaffen werden, das möglichst viele Kunden des Marktes zufriedenstellt – und damit deren Wiederkommen sichert.
Es resultiert daraus der Wunsch in jedem Fall ausreichend Daten zu erfassen, der im Rahmen der letzten Jahre zu einer regelrechten Sammelwut geführt hat. Das Ergebnis sind enorme Datenmengen, die allein mit Mitteln des menschlichen Geistes nicht mehr zu verarbeiten sind. So schätzten Forscher der Universität von Berkeley die Menge jährlich neu erzeugter Daten auf etwa 1 Exabyte, was einer Million Terabyte entspricht [3]. Aufgrund des Alters dieser Schätzung kann

davon ausgegangen werden, daß die tatsächliche Menge heute neu belegten Speichers noch um einiges höher liegt. Nachdem die Aufnahme der Angaben weitgehend automatisch erfolgt, stellt sich heutigen Forschern ein neues Problem: Wie soll auf dieser unüberschaubaren Masse an Einzeldaten analysiert und die ihnen gegebenenfalls inneliegende Information extrahiert werden? Aus dieser Fragestellung entwickelte sich die wissenschaftliche Teildisziplin des Data Mining, die Methoden und Werkzeuge der Mathematik nutzt, um interessante Muster in Daten zu entdecken [2]. Außerdem das Maschinelle Lernen, daß sich mit der Vorhersage zukünftiger Entwicklungen aus den in bereits gesehenen Datensätzen Erfahrungen beschäftigt.
Die seit den in den 1950er Jahren aufkommenden ersten Clusteranalyseverfahren immer weiter verbesserten Algorithmen sind inzwischen fähig effizient mit großen Datenmengen, verschiedenen Datentypen und diversen zu entdeckenden Mustern umzugehen. Probleme entstehen jedoch immer noch, wenn die zugrundeliegenden Muster der Datenentstehung sehr komplex sind bzw. ihre Entdeckung ein hohes Maß an Expertenwissen voraussetzt. Neben der Modellierung dieses Spezialistenwissens direkt in das System zur Wissensentdeckung, entwickelt sich zudem eine Methodik, die dem Datenanalysten das Eingreifen in den Analyseprozeß ermöglicht. Hierbei werden grundlegende Verfahren der Datenanalyse mit aufschlußreicher Visualisierung der Analyse und ihrer Ergebnisse kombiniert und dem Nutzer Möglichkeit zur Interaktion gegeben. Es entsteht hierbei eine Symbiose aus Data-Mining-Methoden und den dem menschlichen Nutzer eigenen Fähigkeiten wie Hintergrundwissen, Kontextinformation und Intuition. So übersteigt die menschliche Kapazität, in realen, verrauschten Daten fehlerhafte Punkte zu erkennen und auszusortieren oder Ansammlungen unterschiedlicher Dichte bzw. Form zu erkennen oftmals noch die heutiger Algorithmen. Gleichzeitig kann die algorithmische Unterstützung die Auswertung der enormen, vom menschlichen Geist nicht mehr faßbaren Datenmengen ermöglichen.
Die vorliegende Arbeit stellt einen Ansatz aus dem Bereich des visuellen Data Minings vor. Dessen Ziel ist es, in bis zu mittelgroßen Datenbanken ein Fuzzy Clustering durchzuführen – also ein Clustering bei dem jeder Datenpunkt zu einem bestimmten Anteil einem Cluster angehört, nicht notwendigerweise nur einem einzigen. Die Grundlage des Clusterings und der Präsentation an den Nutzer bilden sogenannte “Neighborgrams”, eine eindimensionale Darstellung der nächsten Nachbarn eines Punkts. Diese wird zunächst genutzt, um die algorithmischen Clusterzugehörigkeiten zu ermitteln. Beim späteren Finetuning der Ergebnisse dient als übersichtliche Darstellung der Cluster für den Nutzer – und bietet die entsprechenden Interaktionsmöglichkeiten zur Verschiebung der Clustergrenzen bzw. Clusterzugehörigkeiten.

2 Verwandte Arbeiten

Überschneidungen finden sich im ersten Teil der Arbeit hauptsächlich mit dem Feld der Intelligenten Datenanalyse bzw. des Data Minings. Die Zielstellung ist die Entdeckung von Mustern und Zusammenhängen in bisher unbekannten

Daten. Die zugrundeliegende Datenstruktur der Neighborgrams als eindimensionale Struktur ist neu, ähnliche Betrachtungen von Nachbarschaftsbeziehungen fanden speziell in der Bioinformatik bereits Beachtung (vgl. [4]). Der genutzte Clusteringalgorithmus zeigt Ähnlichkeit zur Mountain-Methode [5], jedoch in leicht abgewandelter Form.
Im zweiten Teil des Artikels zeigen sich starke Parallelen zu den Aufgabenstellungen der explorativen Datenanalyse, es werden einfache Techniken der Visualisierung adaptiert, um eine übersichtliche Darstellung der Neighborgrams zu schaffen.

3 Clustering mithilfe von Nachbarschaftsbeziehungen

Der im Folgenden dargestellte Algorithmus nutzt zur Bestimmung der in den Daten vermuteten Cluster die Nachbarschaftsbeziehungen zwischen den betrachteten Datenpunkten und bereits berechneten Clustern. Es ist ersichtlich, daß die Berechnung der Nachbarschaft eines jeden Punktes/Clusters mit hohem Aufwand verbunden ist. Das Verfahren ist von daher nur für bis zu mittelgroße Datenmengen oder die Modellierung bestimmter Untermengen einer Datenbasis anwendbar. Letzteres trifft man beispielsweise in der pharmazeutischen Forschung an, wenn beim Test einer Wirkstoffgruppe in Bezug auf bestimmte Krankheitserreger zwar Unmengen von Daten aufgenommen werden, schlußendlich aber nur die Eigenschaften derer Wirkstoffe nher untersucht werden sollen, die sich als tatsächlich wirksam erwiesen.

3.1 Neighborgrams

Als Grundlage des Clusterings und der späteren Visualisierung wird eine neuartige Datenstruktur genutzt, die sogenannten Neighborgrams. Es handelt sich dabei um eine eindimensionale Darstellung der k nächsten Nachbarpunkte eines Datenpunktes. Für jeden für die Analyse interessanten Datenpunkt p_i wird also eine Liste der k ihm am nächsten liegenden Datenpunkte/Cluster hinterlegt, verbunden mit der jeweiligen Distanz zwischen p_i als Zentrum und dem jeweiligen Nachbarpunkt p_j mit $0 \leq j \leq k$.Dabei ist jeder Datenpunkt p_i mit der Distanz 0 zu sich selbst der erstgenannte in seinem eigenen Neighborgram. Treten zwei Datenpunkte mit gleicher Distanz zu p_i werden diese in zufälliger Reihenfolge dem Neighborgram hinzugefügt. Ein Beispiel für die Darstellung findet sich in Bild 1.

3.2 Clustering

Grundsätzlich wird jeder als interessant identifizierte und mit einem Neighborgram verbundene Datenpunkt p_i als potenzielles Clusterzentrum angenommen. Die Aufgabe des Clusteralgorithmus' besteht nun darin, aus der Menge der Neighborgrams in jedem Iterationsschritt die besten Clusterzentren zu identifizieren und ihnen gegebenenfalls andere Datenpunkte als "Clusterinhalt" zuzuordnen. Die zu absolvierenden Arbeitsschritte ergeben sich also wie folgt:

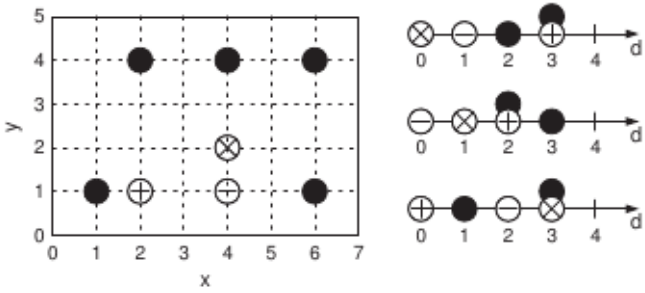

Fig. 1. Beispiel: Attributraum mit drei Neighborgrams

1. Ordne jedes Neighborgram einem Clusterkandidaten zu (oder erzeuge einen neuen für das entsprechende Neighborgram).
2. Erzeuge ein Ranking aller Clusterkandidaten und erzeuge den besten tatsächlichen Cluster.
3. Entferne alle im neu erzeugten Cluster enthaltenen Datenpunkte aus der Datenbasis.
4. Wieder hole ab Schritt 1 bis die Abbruchkriterien erfüllt sind.

Um diese Schritte tatsächlich durchzuführen müssen allerdings noch einige Maße definiert werden: die Definition eines Clusterkandidaten und eine Möglichkeit diese zu evaluieren und zu ranken; wie das Entfernen überdeckter Datenpunkte/-muster zu erfolgen hat; und unter welchen Bedingungen die Bearbeitung als beendet angesehen werden kann.

3.3 Eigenschaften der Neighbourgrams

Auch wenn die Struktur der Neighborgrams vergleichsweise einfach ist – eine einfache geordnete Liste der R Punkte, die dem aktuell betrachteten Zentrum am nähsten liegen, lassen sich bereits hieraus Informationen über die enthaltenen Punkte ableiten.

Coverage Γ: In der R-Nachbarschaft eines Datenpunktes befinden sich nicht zwangsläufig nur andere Datenpunkte der gleichen Klassenzuordnung. Die Coverage, oder "Abdeckung", gibt an, wie viele der Beispiele in der Nachbarschaft durch einen Cluster positiv erklärt würden, also der gleichen Klasse wie das Nachbarschafts- bzw. Clusterzentrum haben. Dieses Maß kann im Rahmen *einer* Nachbarschaft schrittweise betrachtet werden – die Abdeckung unter Betrachtung des ersten Punktes der Nachbarschaftsliste, unter Betrachtung der ersten beiden etc., bis die maximale Anzahl R von Punkten

im Neighborgram erreicht ist.

– **Beispiel:** Für das Muster $\oplus$ im Beispiel aus Bild 1 ergibt sich schrittweise für die Abdeckung: $\Gamma_{\oplus}(1) = 1$; $\Gamma_{\oplus}(2) = 1$; $\Gamma_{\oplus}(3) = 2$; $\Gamma_{\oplus}(4) = 3$; $\Gamma_{\oplus}(5) = 3$;

Purity Π: Die Purity steht in direktem Verhältnis zur vorher betrachteten Coverage, nur bezieht sie sich auf den *Anteil* der positiv erklärten Beispiele an der Gesamtmenge betrachteter Datenpunkte.

– **Beispiel:** Für das Muster $\oplus$ im Beispiel aus Bild 1 ergibt sich schrittweise für die Purity: $\Pi_{\oplus}(1) = 1$; $\Pi_{\oplus}(2) = \frac{1}{2}$; $\Pi_{\oplus}(3) = \frac{2}{3}$; $\Pi_{\oplus}(4) = \frac{3}{4}$; $\Pi_{\oplus}(5) = \frac{3}{5}$;

Optimale Tiefe Ω: Jedes bei der Berechnung mit einbezogene Negativbeispiel lässt den Wert der Purity absinken. Die optimale Tiefe gibt die Entfernung vom Zentrumspunkt an, die bei der Berechnung betrachtet werden kann, ohne einen bestimmten Schwellwert für die Purity zu unterschreiten.

Minimale Größe: beschreibt die minimale Größe, die ein Cluster haben muß. Die Nutzung der Purity als Qualitätsmaß hat den Nachteil, daß anfällig für verrauschte Daten ist. Eine fehlerhaft negative Instanz, angetroffen in der Nähe des Zentrums eines Neighborgrams kann dazu führen, daß die Optimale Tiefe unvernünftig klein ausfällt. Um dies zu vermeiden, wird der zusätzliche Parameter eingeführt, der uns eine Mindestgröße eines potenziellen Clusters festlegen läßt.

3.4 Clusterkandidaten und Algorithmus

Mit diesen zusätzlichen Maßen über die Eigenschaften der Neighborgrams kann nun genauer festgelegt werden, wie Clusterkandidaten zu erzeugen und zu bewerten sind. Hierzu wird zunächst ein vom Nutzer definierter Mindestwert für die Purity benötigt, der die Berechnung von Coverage und Optimaler Tiefe für jedes Neighborgram ermöglicht. Als vielversprechendster Clusterkandidat gilt jener, der die höchste Purity erreicht. Alle von ihm überdeckten Datenpunkte (also alle in seiner R-Nachbarschaft) werden diesem Cluster zugeordnet und aus dem Datensatz entfernt. Mit den übrigen Datenpunkten wird der Clusterauswahlprozess erneut durchgeführt.
Dieses Vorgehen ist sehr strikt – ein Datenpunkt wird genau einem Cluster zugeordnet und dann nicht weiter betrachtet. Als Schritt von diesem strikten Clustering zum Fuzzy-Clustering wird die sogenannte *Partial Coverage*, also "teilweise Abdeckung" eingeführt. Mithilfe einer *fuzzy membership function* (dt. etwa "Zugehörigkeitsfunktion) kann ein Grad der Zugehörigkeit eines Datenpunktes zu einem bestimmten Clusters modelliert werden. Die Basis zur Bestimmung der Zugehörigkeitsfunktion zu einem Cluster dient wiederum die Purity – so werden als Eckpunkte zum Beispiel der maximale Radius mit maximaler Purity (d.h.

Purity = 1), der maximale Radius mit "guter" Purity (d.h. Purity ¿ Schwellwert) oder der minimale Radius "schlechter" Purity (d.h. Purity ¡ Schwellwert) herangezogen.
Eine genauere Betrachtung verschiedener Zugehörigkeitsfunktionen und der mit ihnen erzielten Ergebnisse auf Testdatensätzen findet sich in [1].

4 Visualisierung und Interaktion

4.1 Neighborgram-Visualisierung

Die vereinfachte Darstellung der Nachbarschaftsbeziehungen im eindimensionalen Neighborgram hat einen entscheidenden Vorteil: die Möglichkeit sehr einfacher Visualisierung. Ein Beispiel dafür findet sich in Bild 2. Es handelt sich um zwei beispielhafte Neighborgrams aus dem Iris-Datensatz, horizontal abgetragen die Distanz zum Nachbarschaftszentrum, darauf abgetragen die Datenpunkte in der Umgebung. Die Farbe codiert hierbei die bekannte Klassenzuordnung die im Hintergrund grau hinterlegte Fläche die Zugehörigkeitsfunktion des Clusters für eine bestimmte minimale Purity p_min. Für die Darstellung der Datenpunkte hat die vertikale Achse keine symbolische Bedeutung – sie werden lediglich übereinander dargestellt, um Überdeckungen zu vermeiden und damit die Übersichtlichkeit zu erhalten. Für die Zugehörigkeitsfunktion wird auf der Vertikalen jedoch der Grad der Zugehörigkeit codiert, in ihrem Maximalwert 1, ab einem bestimmten Wert absteigend bis zur 0.

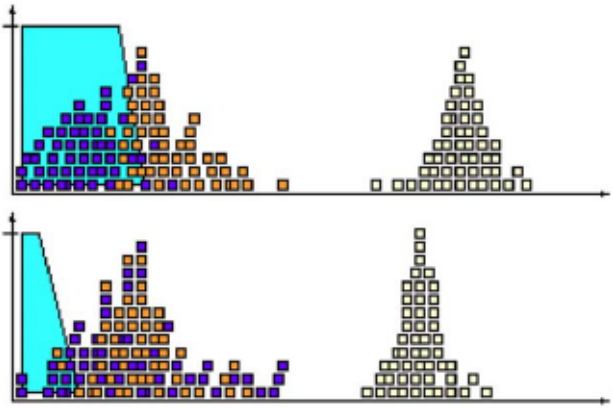

Fig. 2. Zwei Beispiel-Neighborgrams aus dem Iris-Datensatz

4.2 Interaktion

Möglichkeit für die Nutzerinteraktion ergibt sich speziell im Ranking-Schritt des Algorithmus. Dem Nutzer wird die sortierte Liste potenzieller Clusterkandidaten präsentiert, dieser kann nach Begutachtung Clusterkandidaten als unwichtig markieren oder gegebenenfalls in ihrer Bedeutung höher werten. Weiterhin denkbar wäre eine direkte Beeinflussung der Zugehörigkeitsfunktion.

5 Zusammenfassung

Die vorliegende Arbeit stellt eine Methode vor, die ein Clustering basierend auf Neighborgrams vornimmt. Der zugrundeliegende Algorithmus nutzt dabei zwei Eingabeparameter, die minimale Purity, die die Grenzen eines Clusters festlegt und ein Abbruchkriterium.
Laut [1] können im Clustering zufriedenstellende Ergebnisse erzielt werden, die sich mit der fuzzy-Version des Algorithmus noch weiter verbessern.
Ein wichtiges Novum der Methode stellt jedoch die dazugehörige Visualisierungstechnik dar, die, bedingt durch die einfache Struktur der zugrundeliegenden Neighborgrams, dem Nutzer die Möglichkeit gibt, den Clusteringprozess einfach nachvollziehbar zu begleiten und auch zu beeinflussen. Es wird ihm somit ermöglicht, Expertenwissen in den Clusteringprozess einfließen zu lassen. In [1] wird ein Beispiel aus dem Bioinformatik-Bereich aufgezeigt, das die Notwendigkeit solcher explorativer Syteme betont.
Leider wurden bisher keine Nutzerstudien angeboten, die die tatsächliche Nutzbarkeit des Systems auch durch Laien untersucht.

References

1. M.R. Berthold, B. Wiswedel, and D.E. Patterson. Interactive exploration of fuzzy clusters using neighborgrams. *Fuzzy sets and systems*, 149(1):21–37, 2005.
2. Christopher Clifton. Encyclopedia britannica: Definition of data mining, 2010.
3. D.A. Keim. Information visualization and visual data mining. *Visualization and Computer Graphics, IEEE Transactions on*, 8(1):1–8, 2002.
4. D.E. Patterson, R.D. Cramer, A.M. Ferguson, R.D. Clark, and L.E. Weinberger. Neighborhood behavior: a useful concept for validation of molecular diversity descriptors. *Journal of medicinal chemistry*, 39(16):3049–3059, 1996.
5. R.R. Yager and D.P. Filev. Approximate clustering via the mountain method. *Systems, Man and Cybernetics, IEEE Transactions on*, 24(8):1279–1284, 1994.

Fuzzy Clustering across Parallel Universes

Karen Otte

Otto-von-Guericke-University of Magdeburg
Universitätsplatz 2, D-39106 Magdeburg, Germany
karen.otte@st.ovgu.de

Abstract. Clustering is a data mining technique, where new information are generated by identifying new groups in data sets. This learning method is a so called 'unsupervised learning' because there is no information about the classification of the data objects. Some data, e.g. 3D models, have more than one representation or are described in more than one description space. These data sets need a different approach for clustering, because not all spaces are sufficient for representing clusters. The difference between existing approaches and the new one developed by Wiswedel, Höppner and Berthold [5] is, that it constructs a global model over all existing descriptor spaces based on the individual local models. The approach and results are presented and discussed.

Keywords: Fuzzy Clustering, Possibilistic Clustering, multi representation-spac 3D-Data clustering

1 Introduction

The aim of cluster analysis is to partition a given set of data or objects into clusters[2]. These clusters represent the relation between single objects. Objects are described by e.g. experts and sometimes there are different approaches to describe them. For example 3-dimensional models could be described by their volume, their shape or in an image-based way. These objects have more than one representation and so also different description spaces. According to Wiswedel et al. [5] most learning techniques are restricted to learning in exactly one descriptor space. Learning in the presence of several object representations is in practice often solved by using one of the following three schemes:

1. Reduce the analysis to any descriptor space separately. This is a rather strong limitation, because it may ignore information that is encoded in the remaining object representations.
2. Construct a joint descriptor space by e.g. concatenating features of the individual descriptor spaces. Such a combination is often impossible due to different feature domains. It may also introduce artifacts as it blurs the information that is present in different object representations.
3. Perform a separate analysis of each descriptor space and fuse the individual models afterwards. This ad hoc solution delays the task of merging results

to a post processing step. It has the major drawback of not paying attention to possible structural overlaps among descriptor spaces since the individual model construction is carried out independently. These overlaps, however are often of special interest and can be matched in order to reach consensus.

All of these strategies have limitations because they either ignore or obscure the multiple facets of the objects (given by the descriptor spaces) or do not respect overlaps. This often makes them inappropriate for practical problems. The new learning scheme encompasses a simultaneous analysis of all given descriptor spaces, hereafter referred as *universes*. The main objective of learning in parallel universes is therefore the construction of a global model based on local models that outperforms schemes and provides new insights with regard to overlapping as well as universe-specific patterns. The learning task can also used for supervised learning e.g. for building classifiers [5]. Learning in parallel universes is rather difficult, because single descriptor spaces are not useful for learning. By analyzing the relations between clusters in the different description spaces it is possible to gain new information. For the clustering of objects with multiple-representations like 3D-models, image data or molecular data there is no applicable solution. This work is a summary of the approaches and papers by Wiswedel, Höppner and Berthold. Following their extensions of the fuzzy c-means and possibilistic clustering for parallel universes were presented.

2 Fuzzy C-Means-Clustering and Possibilistic Clustering

The presented Methods based on fuzzy c-mean clustering (FCM) [1] and possibilistic clustering (PCM) [3], which are modified for data sets with more than one description space. Both algorithms are extentions of the c-mean-algorithm, so they iteratively minimize the object function. This function uses the weighted intracluster distances between instances/objects. So the lower the summed distance, the better the clustering. In the following the value P stands for the overall Numbers of Objects in one Universe. The object representation/instance i, $1 \leq i \leq P$, for an object is x_i. K is the number of all clusters in one Universe. These number must be set for each universe before using the algorithm. And the distance function between a instance and an prototype is shown by $d(w_k, x_i)$, where w_k is the k-th prototype. Normally the Euclidean distance is used.

2.1 Fuzzy C-Means-Clustering

The Fuzzy c-means-clustering (FCM) was invented by Bezdek in 1981 [1]. In the FCM algorithm the clusters are presented by their centers, which are called prototypes. The objective function (1) represents the accumulated sum of distances between the object representations (instances) x_i and the cluster centers w_k. It is weighted by the degree of membership $v_{i,k}$ to which an object x_i belongs to cluster k. The parameter $m \in (1, \infty)$ is the so called 'fuzzification parameter'.

The object function and the update functions for the values $v_{i,k}$ and w_k are:

$$\min_{v_{i,k},w_k} \sum_{i=1}^{P}\sum_{k=1}^{K} v_{i,k}^m \; d\;(w_k, x_i) \qquad s.t. \quad \forall i : \sum_{k=1}^{K} v_{i,k} = 1 \tag{1}$$

$$v_{i,k} = \left(\sum_{\overline{k}=1}^{K} \left(\frac{d_u\;(w_k, x_i)}{d\;(w_{\overline{k}}, x_i)} \right)^{\frac{1}{m-1}} \right)^{-1} \tag{2}$$

$$w_k = \frac{\sum_{i=1}^{P} v_{i,k}^m x_i}{\sum_{i=1}^{P} v_{i,k}^m} \; . \tag{3}$$

According to Wiswedel et al. [5] one serious drawback of FCM is that outliers and noise can heavily influence the clustering result.

2.2 Possibilistic Clustering

As response to the problem with noisy data, Krishnapam and Keller [3] developed the possibilistic clustering (PCM) which does not have such a side constraint. The main difference to the FCM is that instead of distributing an object among the clusters it uses a penalty term η_k to avoid the trivial solution in which all clusters are empty. For each membership $v_{i,k}$ of an object i to a cluster k it uses a non-membership value $1 - v_{i,k}$. This will take an accordingly large value if an object has a large distance to the cluster prototype. This non-membership value is weighted by the penalty term η_k. This penalty term is determined using certain heuristics. Krishnapuram and Keller [3] suggest using the average intra-cluster distance of cluster k, e.g. calculating the average distance of instances with a high membership to the cluster prototype. The PCM objective function is:

$$\min_{v_{i,k},w_k} \sum_{i=1}^{P}\sum_{k=1}^{K} v_{i,k}^m \; d\;(w_k, x_i) + \sum_{i=1}^{P} \eta_k \sum_{k=1}^{K} (1 - v_{i,k})^m. \tag{4}$$

3 Clustering in Parallel Universes

Fuzzy clustering in parallel universes is an unsupervised learning concept published first in 2007 by Wiswedel and Berthold [4]. In contrast to existing ones, their approach constructs a global model that is based on only partially applicable, local models in each descriptor space.
FCM and PCM are not directly applicable for problems described in parallel universes. In the previous work of Wiswedel and Berthold [4], they suggested modifying the objective function and including additional terms to represent the universes.
Although this extension enabled a simultaneous analysis of different universes, it had a few drawbacks [5]:

– The user has to specify the number of clusters for each universe separately.
– Clusters in different universes are independent of each other. They are not interconnected and therefore it is also not possible to identify universe overlapping information, i.e. objects that group well in multiple universes.
– The results are difficult to interpret.
– The Clustering are affected by noise and outliers due to its partitioning property. Although the influence of noisy objects in a universe is reduced when these objects clusters well in the remaining universes, they could still prove problematically if they do not contribute to any of the clusters.

These disadvantages limited the general usage of this approach, because often there is no prior knowledge on how many clusters to expect in a given universe and also the missing connection between the universes is often inappropriate. These concerns was eliminated by using a different interpretation of the used membership values and a different formulation of the target function.
The first approach of Wieswedel et al. was, to add an membership values to the objective function, which represent the universes, so the FCM and PCM could be used for parallel universes [4]. These membership values $0 <= z_{i,u} <= 1$ denote the object memberships to universes and are also optimized during the training process. The higher this value is, the better the corresponding object representation is covered by on of the clusters in a universe. Wieswedel, Höppner and Berthold give two different interpretations of the membership values [5], a fuzzy and a possibilistic way.

3.1 Fuzzy Clustering across Parallel Universes

For this fuzzy clustering, there is a fixed number of global clusters, whereby their significance to the universes is learned throughout the training. The objective function or the new clustering has the following form:

$$\min_{v_{i,k},z_{k,u},w_{k,u}} \sum_{u=1}^{U}\sum_{k=1}^{K} z_{k,u}^{m\prime} \sum_{i=1}^{P} v_{i,k}^{m}\ d_u\ (w_{k,u}, x_{i,u}) \tag{5}$$

$$s.t. \quad \forall i : \sum_{k=1}^{K} v_{i,k} = 1 \quad \forall i : \sum_{k=1}^{K} z_{k,u} = 1\ . \tag{6}$$

The function represents a classical fuzzy c-means within each universe, whereby this sum of weighted distances is multiplied by the membership value $z_{k,u}$, i.e. the degree to which a cluster is represented in a universe. This scaling is analogous to the weighting of the distances in a classical c-means approach: $z_{k,u}$ (or $v_{i,k}$) will be large if the weighted sum of distances $\sum_{i=1}^{P} v_{i,k}^{m} d(x_{i,u}, w_{k,u})$ is small. The membership values are learned to be large, i.e. $z_{k,u} - - > 1$ if a cluster k forms primarily in universe u. If these values are similar between different universes, it indicates that this cluster forms in different universes equally well. The output of this approach is a set of cluster prototypes along with their respective membership values and the objects with their membership values to

the clusters. The update function for the membership values and the cluster prototypes are:

$$v_{i,k} = \left(\sum_{\overline{k}=1}^{K} \left(\frac{\sum_{u=1}^{U} z_{k,u}^{m\prime} \; d_u \; (w_{k,u}, x_{i,u})}{\sum_{u=1}^{U} z_{\overline{k},u}^{m\prime} \; d_u \; (w_{\overline{k},u}, x_{i,u})} \right)^{\frac{1}{m-1}} \right)^{-1} \tag{7}$$

$$z_{k,u} = \left(\sum_{\overline{u}=1}^{U} \left(\frac{\sum_{i=1}^{P} v_{k,u}^{m} \; d_u \; (w_{k,u}, x_{i,u})}{\sum_{i=1}^{P} v_{k,u}^{m} \; d_u \; (w_{k,\overline{u}}, x_{i,\overline{u}})} \right)^{\frac{1}{m-1}} \right)^{-1} \tag{8}$$

$$w_{k,u} = \frac{\sum_{i=1}^{P} v_{i,k}^{m} x_{i,u}}{\sum_{i=1}^{P} v_{i,k}^{m}} \; . \tag{9}$$

3.2 Possibilistic Clustering across Parallel Universes

The new objective function uses the objective function of standard fuzzy c-means within the individual universes, whereby they are linked using a possibilistic membership value to represent a clusters typicality for a universe.

$$\min_{v_{i,k}, z_{k,u}, w_{k,u}} \sum_{u=1}^{U} \sum_{k=1}^{K} z_{k,u}^{m\prime} \sum_{i=1}^{P} \; v_{i,k}^{m} d_u \; + \sum_{u=1}^{U} \eta_u \sum_{k=1}^{K} (1 - z_{k,u})^{m\prime} \sum_{i=1}^{P} v_{i,k}^{m} \tag{10}$$

$$s.t. \forall i : \sum_{k=1}^{K} v_{i,k} = 1$$

The second term of the objective function corresponds to the possibilistic interpretation as in standard PCM. The role of the multiplier $\sum_{i=1}^{P} v_{i,k}^{m}$ is important here as it penalizes clusters that do not belong to any universe. Ignoring this multiplier would otherwise attract all objects into these no-universe clusters. Similar to the classical PCM the second term is scaled by a universe specific parameter η_u. The value of this parameter can be heuristically determined usage, i.e. the average intra-cluster distance in a universe. That is

$$\eta_u = \kappa \frac{\sum_{i=1}^{P} \sum_{k=1}^{K} z_{k,u}^{m\prime} v_{i,k}^{m} \; d_u(x_{i,u}, w_{l,u})}{\sum_{i=1}^{P} \sum_{k=1}^{K} z_{k,u}^{m\prime} v_{i,k}^{m}} \; , \tag{11}$$

whereby κ is a user parameter to scale this term. According to Wiswedel et al. [5] this is usually set to $\kappa = 2$.

4 Results

The presented algorithms were tested with two types of data, artificial data and 3D object data. The artificial data have 3 universes with two-dimensional feature spaces each. That allows a simple visualization and generation of clusters.

Each universe has two Gaussian distributed clusters with a total of 1,400 objects. Figure XY shows the clusters without noise and with 2/3 of the data are noise. This data generation process does not create any overlapping clusters across universes, so its not so applicable. Wiswedel et al. used this data to compare there last approaches against further approaches. The conclusion was, that it clustered well for the assumptions that there are 3 universes with 3 clusters each. There was no other tests with different cluster sizes or comparisons with other clustering technics (expect a standard FCM on the joint feature space which was not applicable at all).

The second data set (3D object data by 3D Benchmark 2008 [**?**]) was used to analyze the influence of the fuzzifiers m and m'. The data set has the following 4 representations [**?**]:

DBF a 366 dimensional image-based descriptor,
SIL a 510 dimensional image-based descriptor, similar to DBF but simpler,
SSD a 432 dimensional shape-based descriptor which uses the curvature properties of an object surface and
VOX a 343 dimensional volume-based descriptor.

Wiswedel et al used the Euclidean norm to calculate the distances between objects representations and prototype. The domain of the distance function were normalized to have comparable distance ranges in all universes. Figure 1 shows a

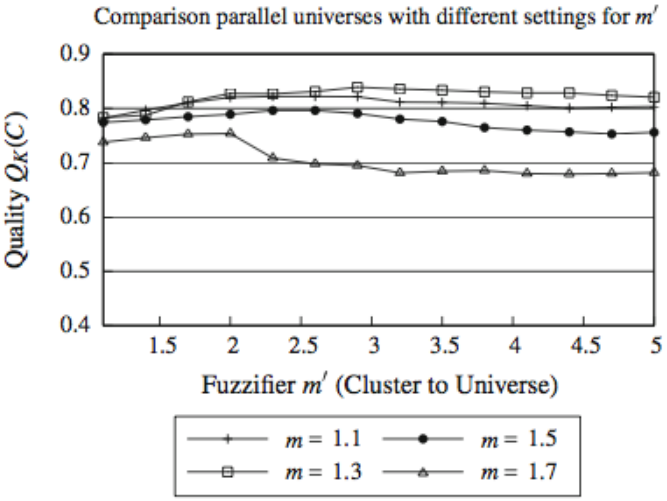

Fig. 1. Quality values for parallel universe approach with increasing fuzziness

quality factor $Q_K(C)$ with respect to the fuzzifiers m and m'. The quality value

$Q_K(C)$ uses an entropy based comparison of the computed clustering K with the referencing clustering C. The higher the value Q_K, the better the clustering. Figure 1 shows also that a smaller value of m tend to give better clustering results, relatively independent on how m' is chosen. The value m' seems to have a far less impact than m. According to Wiswedel larger the value for m' produce more overlap between universes, whereas small values allow more crisp assignments.

5 Conclusion and Further Work

Wiswedel et al. developed different extensions of a family of fuzzy clustering algorithms, which are able to cluster data sets across parallel universes. In the classical setup these algorithms minimize an objective function representing a weighted sum between object representations and their (partially) associated cluster prototypes. The new membership values can be interpreted (after learning) as, e.g. the typicality of a cluster belonging to a certain universe. According to Wiswedel the algorithms are better suited for real-world problems. This was demonstrated with one data sets that contains descriptions of 3D objects given in different universes. They mentioned that the clustering of the 3D data set produced a very intuitive clustering, that could not be detected with other methods. Wiswedel, Höppner and Berthold compared their results only with previous versions of the algorithms and not against other approaches. This is preferable for their further work, because this is an important way to verify their results. At this moment, there is no real-world usage of these clustering algorithms and there was no further work after the paper "Learning in parallel Universes" in 2007 [5].

References

1. J. C. Bezdek, R. Ehrlich, and W. Full. The fuzzy c-means clustering algorithm. *Computers and Geosciences*, 1984.
2. F. Höppner, F. Klawonn, R. Kruse, and T. Runkler. *Fuzzy Clustering Analysis.* John Wiley and Sons, 1999.
3. R. Krishnapura and J. M. Keller. A possibilistic approach to clustering. *IEEE Transaction on Fuzzy Systems*, pages 98–110, 1993.
4. B. Wiswedel and M. R. Berthold. Fuzzy clustering in parallel universes. *International Journal of Approximate Reasoning*, 45, 2007.
5. B. Wiswedel, F. Höppner, and M. R. Berthold. Learning in parallel universes. *Data mining knowladge discovery*, 21:130–152, 2010.

Einleitung in das spektrale Clustering

Jan Zelmer

Otto-von-Guericke-University of Magdeburg
Universitätsplatz 2, D-39106 Magdeburg, Germany,
jan.zelmer@st.ovgu.de

Zusammenfassung. In dieser Arbeit wird das Grundprinzip von Spectral Clustering erklärt, so dass man in der Lage ist weiterführende Literatur zu dem Thema zu verstehen, einfache Modelle zu implementieren und im besten Falle das Erweiterungsprinzip auf andere Modelle des Machine Learnings anzuwenden.

Schlüsselwörter: Spectral, Clustering, K-Means

1 Einleitung

In der Zukunft wird es immer und mehr Daten geben, also auch mehr Informationen, die daraus gewonnen werden können. Aber vor allem wird es sehr viel mehr Datenmüll geben, und dementsprechend steht der Bedarf nach guten Methoden beides von einander zu trennen. Und da sich Kontexte laufend ändern und es immer noch Optimierungspotential wird die Forschung in diesem Bereich wohl nie endgültig abgeschlossen sein. Deswegen möchte ich in dieser Arbeit ein weiteren Schritt im Clustering vorstellen - das spektrale Clustering. Das spektrale Clustering ist eine Methode die an der Datenvorverarbeitung ansetzt um gute Ergebnisse zu produzieren. Es werden also die Eingabedaten verändert um dem letztendlichen Modell, welches ein bekanntes Modell sein kann (wie z.B. K-Means), optimierte Daten gegeben, damit dieses Modell bessere Ergebnisse liefert. Da dieser Ansatz nur Berechnungen auf den Eingabedaten vornimmt und diese verändert, kann man das eigentliche Clustering von gut bekannten, erforschten und implementierten Modellen nutzen.

2 Theoretische Grundlagen

2.1 Distanzmaße

Distanzmaße sind ein sehr wichtiges Grundelement in Machine Learning, da die Distanzen Aussagen darüber treffen, wie nah sich zwei Datenpunkte sind und somit auch wie ähnlich sie sind. Die Ähnlichkeit ist meistens als Inverse des Distanz definiert (z.B. Ä = 1 - D). Es gibt viele verschiedene Arten von Distanzmaßen und ich werde hier nur eine kleine Auswahl von meistgenutzten vorstellen. Bei der Auswahl des Distanzmaßes ist die Struktur des Datensatzes entscheidend.

Gegeben sind 2 Datenpunkte x und y mit $x, y \in \mathbb{R}^n$

Euklid:
Der Euklidische Abstand ist wohl der am meisten Bekannte. Er nutzt die Summe der quadratischen Abständen zwischen den einzelnen Komponente der beiden Datenpunkte.(Der „Satz des Pythagoras“beruht auf dieser Berechnung, obwohl geschichtlich gesehen es andersherum gewachsen ist)

$$d(x, y) = \sqrt[2]{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Kosinus:
Der Kosinus Abstand wird oft in Textprocessing genutzt, da hier die Länge der Vektoren, beziehungsweise der Größe der Dimensionen keinen Einfluss auf den Abstand hat. Wie man der Formel entnehmen kann, findet eine Normalisierung statt, die dies ermöglicht.

$$d(x, y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$$

Mahalanobis:
Bei diesem Maß findet eine Verzerrung des Raumes statt. Diese Distanz findet häufig in der Statistik Anwendung um zufällig verteilte Punkte zu Gruppen zuordnen zu können. Da die Verteilung der verschiedenen Gruppen mit angegeben werden kann, ist es recht einfach so zu unterscheiden, zu welcher Gruppe nun der Punkt am Besten passt.

$d(x, y) = \sqrt[2]{(x-y)^t S^{-}1(x-y)}$, wobei S die Kovarianzmatrix ist

2.2 K-Means und Abwandlungen

Dieses Modell ist eines der ersten und einfachsten Modelle und wird hier beispielhaft erklärt damit der Leser anhand dieser Arbeit Spectral Clustering implementieren kann. Das Modell besteht im Wesentlichen aus 3 einfachen Komponenten: dem initialen Raten der Clusterzentren, das Zuordnen der Punkte zu den Clusterzentren und schließlich das Neuberechnen der Clusterzentren. Komponente 2 und 3 werden dabei solange ausgeführt bis ein gewisses Abbruchkriterium erfüllt ist.

Gegeben ist eine Menge von Punkten $X = \{x_1, ..., x_j\}, x_i \in \mathbb{R}^n$ und ein Distanzmaß d, welches im entsprechendem Vektorraum $\mathbb{R}^n$ definiert ist.

Erster Schritt des Algorithmus ist das Festlegen der Startpunkte. Hierzu gibt es auch verschiedene Möglichkeiten, auf die ich aber nicht vollständig eingehen kann. Die naivste Möglichkeit ist diese zufällig auszuwählen, die k so ausgewählten Punkte sind die initialen Clusterzentren $c_1, ..., c_k$. Das k bestimmt also wie viele Gruppen es gibt und wird vom Anwender von vornherein festgelegt. Nachdem nun unsere Clusterzentren initialisiert worden sind, ist der nächste Schritt jedem Datenpunkt ein Clusterzentrum zuzuordnen. Dafür brauchen wir das Distanzmaß (z.B. Euklid), es wird also für jeden Datenpunkt die Entfernung zu jedem Clusterzentrum berechnet. Anschließend wird dem Punkt ein Cluster

C zugeordnet und zwar zu dem Cluster wo die Distanz zwischen Clusterzentrum und Datenpunkt am geringsten ist. Ein Cluster C bezeichnet dabei alle Punkte x die dem Cluster zugeordnet sind, also zum dem jeweiligen Clusterzentrum die kleinste Distanz haben.

$$argmin_{i \in k}(d(x_j, c_i))$$

Die Zugehörigkeit wird so für Datenpunkte berechnet und wir haben somit unsere erste Gruppierung. Diese Gruppierung ist aber in der Regel nicht optimal, so dass wir noch weiter iterieren müssen. Der letzte Schritt der Iteration ist es die Clusterzentren neu auszurechnen. Dabei wird der Mittelwert eines Clusters ermittelt, und dieser Mittelwert ist dann das neue Clusterzentrum. Normalerweise wird das arithmetische Mittel genommen, man kann aber auch hier wieder andere Methoden nutzen.

$$c_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i$$

Es gibt nun mehrere Abbruchkriterien: entweder man lässt dies eine bestimmte Anzahl an Iterationen durchlaufen oder was typischer ist, man lässt den Algorithmus so lange laufen, bis sich die Clusterzentrum kaum bis gar nicht mehr verändern. Was gleich bedeutend ist, mit die Zugehörigkeiten der Datenpunkte ändert sich nicht mehr.

Die Schritte zusammengefasst:

- 1. Wahl der initialen Clusterzentren $c_1, ..., c_k$
- 2. Bestimmen der Zugehörigkeiten der Datenpunkte d zu den Clusterzentren $c_1, ..., c_k, d_i \in c_n$
- 3. Neu Errechnen der Clusterzentren $c_1, ..., c_k$
- 4. Wiederhole Schritt 2 und 3 bis das Abbruchkriterium erfüllt ist

Generell ist dieses Modell sehr flexibel und man kann nach belieben verschiedene Aspekte austauschen:

1. Startpunkte der Clusterzentren
2. Distanzmaß
3. Mittelwert

Dieses Modell ist leicht zu implementieren, liefert recht schnell, teils gute Ergebnisse, aber es gibt Szenarien wo dieses Modell die zugrunde liegende Gruppierung nicht vernünftig erfassen kann. Am Rande erwähnt sei noch, dass man Clusterings als Minimierungsproblem auffasen kann die man mit Hilfe von Laplace Operatoren formulieren und lösen kann. Die Formulierung ist nicht trivial und auch nicht Bestandteil dieser Arbeit, deswegen wird hier nicht ausführlich darauf eingegangen.

2.3 Ähnlichkeitsgraphen

Ähnlichkeitsgraphen sind Graphen mit einer speziellen Semantik. Sie geben an wie ähnlich zwei Knoten zueinander sind. Dabei hängt es von der Art des Ähnlichkeitsmaßes ab, wie die Kanten aussehen. Ein Ähnlichkeitsgraph ist definiert als $G = (V, E)$, wobei G ungerichtet ist. Die Menge der Knoten $V = \{v_1, ..., v_n\}$ entspricht unseren Datenpunkten und die Kanten ($E = \{e_1, ..., e_m\}$, wobei $0 <= m <= n^2, e(v_1, v_2) = sim(v_1, v_2)$) die Ähnlichkeiten zwischen den Punkten. Bei dem Ähnlichkeitsmaß muss beachtet werden, dass der kleinste Wert im Definitionsbereich 0 und der Größte 1 nicht überschreiten darf und 0 bedeutet absolut keine Ähnlichkeit (oder das genaue Gegenteil) während 1 genau gleich (Identität) bedeutet.

Wenn man die Ähnlichkeiten ausgerechnet hat, gibt es verschiedene Möglichke die Kanten zu setzen.

ϵ-Neighborhood: Hier werden alle Kanten eingetragen, deren Werte einen gewissen Grenzwert ϵ überschreiten. Da nur Werte die den Grenzwert überschreite eingetragen werden, ähneln sich die Kantenwerte relativ stark, so dass man auch gut auf sie verzichten kann. Der Graph ist also nicht gewichtet und ungerichtet.

k-Nearest-Neighborhood: Ähnlich der ϵ-Neighborhood werden hier nur die Kanten eingetragen, die eine bestimmte Voraussetzung erfüllen. Sie müssen innerhalb der ersten „k“Nachbarn sein. Das bedeutet dass so jeder Punkt mindestens k Kanten besitzt. Man kann diese Forderung verstärken, indem man sagt, dass der andere Punkt zu dem eine Kante gesetzt werden soll, auch von sich aus die Bedingung für das Setzen einer Kante erfüllt. Dies nennt man dann „mutual-k-nearest-neighbor “
und die Punkte haben dann maximal k Kanten. Die andere Möglichkeit den Graph ungerichtet zu machen ist einfach die Richtung zu ignorieren und nicht zu übernehmen.

Fully-connected-Graph: Wie der Name schon sagt, werden alle Kanten eingetragen und erhalten als Gewichte die Ähnlichkeit zwischen den Knoten.

3 Spectral Clustering

3.1 Allgemein

Wie bei allen Clusteringalgorithmen ist eine Partitionierung der Eingabedaten das Ziel. Dabei bedient sich dieses Modell zweier Kernkomponenten. Zum ersten die Umformung der Eingaben und des Eingaberaums in eine andere Form und des anschließenden Clusterings der umgeformten Eingabedaten. Das anschließende Clusteringmodell kann dabei frei gewählt werden.

Spectal Clustering erinnert dabei an die Spektalanalyse, wo es dabei geht eine Energiequelle anhand ihrer abgegebenen Strahlung zu untersuchen und dabei ein Energiespektrum aufzustellen, was charakteristisch für die Quelle ist. (siehe Fraunhofer-Linien [4])

Um die Grundidee von Spectral Clustering besser zu verdeutlichen, kann man SVMs heranziehen. Dort wird zuerst der Featureraum transformiert und anschließend wird versucht eine Hypertrennebene zwischen die einzelnen Mengen zu konstruieren. Bei Spectral Clustering wird auch der Featureraum transformiert, in diesem Fall wird er nicht aufgebläht sondern auf die Dimensionen reduziert, wo die wichtigen Features vermutet werden. Man könnte es auch mit PCA vergleichen, wo auch eine Featureraumreduktion stattfindet. Nur in diesem Fall werden die Datenpunkte nicht in den neuen Raum umgerechnet, sondern die berechneten Eigenvektoren sind die neuen Datenpunkte. Dem passioniertem Leser wird [5] ans Herz gelegt, den Anderen erkläre ich kurz die Idee hinter PCA. Gegeben ist eine Datenmatrix, aus der man die Eigenwerte und Vektoren errechnet, anschließend wird der größte Eigenvektor (Eigenwerte und Vektoren bilden ein Paar, und somit ist der größte Eigenwert und der dazu gehörige Vektor gemeint) die neue Hauptachse. Die Daten werden in den neuen Raum umgerechnet und es wird wieder eine neue Achse ermittelt, bis man genügend Achsen gefunden hat. Somit werden n-dimensionale Räume in den Dimensionen reduziert.

Der wissenschaftliche Neugewinn ist die Umformung der Daten. Welches Modell zur Grundlage des anschließenden Clustering benutzt wird, ist dem Anwender überlassen. Zum Zwecke der Erklärung nutzen wir hier den K-Means.

Gegeben ist der Ähnlichkeitsgraph G mit den Kanten E, diese Kanten werden in eine Matrix W überführt, in dem die einzelnen Abstände abgetragen sind. Die Matrix enthält auf der Hauptdiagonalen somit nur Einsen und die Werte sind auch an der Hauptdiagonalen gespiegelt. Wir definieren die Matrix D als Diagonalmatrix wobei die Werte der Diagonalen sich wie folgt ergeben:

$$d_{ii} = \sum_{j}^{n} w_{ij} \text{ , auch Grad eines Knotens genannt}$$

Nun können wir die nicht normalisierte Laplace Matrix L definieren.

$$L = D - W$$

Mit diesen Definitionen sind wir in der Lage nun das Verfahren zu beschreiben.
Eingabe: Ähnlichkeitsmatrix $S \in \mathbb{R}^{nxn}$ und die Anzahl der zu erstellenden Cluster k.

- Konstruiere aus der Matrix ein Ähnlichkeitsgraph, mit der Adjazenzmatrix W
- Errechne die nicht normalisierte Laplace Matrix L
- Berechne aus L die ersten k Eigenvektoren $u_1, ..., u_k$
- Die Eigenvektoren u spaltenweise aneinandergereiht bilden die Matrix U

- Die neuen Datenpunkten y sind die Reihen der Matrix U (Erste Reihe ist der erste Datenpunkte, usw.)
- Clustere die neuen Datenpunkte in Cluster $C_1, ..., C_k$ (K-Means wird empfohlen, andere Modelle sind aber auch nutzbar)

Ausgabe: Cluster $A_1, ..., A_k$ wobei $A_i = \{j | y_j \in C_i\}$

3.2 Verfahren nach Shi und Malik

Der Unterschied zum allgemeinen Verfahren ist die Extraktion der Eigenvektoren. Statt der einfachen Berechnung der Eigenvektoren findet eine generalisierte Form davon statt, ansonsten bleibt der Algorithmus gleich. Eingabe: Ähnlichkeitsmatrix $S \in \mathbb{R}^{nxn}$ und die Anzahl der zu erstellenden Cluster k.

- Konstruiere aus der Matrix ein Ähnlichkeitsgraph, mit der Adjazenz Matrix W
- Errechne die nicht normalisierte Laplace Matrix L
- **Berechne aus dem generalisiertem Eigenproblem** $Lu = \lambda Du$ **die ersten, generalisierten** k **Eigenvektoren** $u_1, ..., u_k$
- Die Eigenvektoren u spaltenweise aneinandergereiht bilden die Matrix U
- Die neuen Datenpunkten y sind die Reihen der Matrix U (Erste Reihe ist der erste Datenpunkte, usw.)
- Clustere die neuen Datenpunkte in Cluster $C_1, ..., C_k$ (K-Means wird empfohlen, andere Modelle sind aber auch nutzbar)

Ausgabe: Cluster $A_1, ..., A_k$ wobei $A_i = \{j | y_j \in C_i\}$

3.3 Verfahren nach Ng et al.

In dieser Variante des Verfahrens findet auch eine veränderte Berechnung der Eigenvektoren statt und auch noch eine nachträgliche Normalisierung der neu errechneten Datenpunkte.

Folgendes wird für diese Variante noch definiert:

$$L_sym = D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$$

Eingabe: Ähnlichkeitsmatrix $S \in \mathbb{R}^{nxn}$ und die Anzahl der zu erstellenden Cluster k.

- Konstruiere aus der Matrix ein Ähnlichkeitsgraph, mit der Adjazenz Matrix W
- Errechne die nicht normalisierte Laplace Matrix L
- **Berechne aus** L_{sym} **die ersten** k **Eigenvektoren** $u_1, ..., u_k$
- Die Eigenvektoren u spaltenweise aneinandergereiht bilden die Matrix U
- **Erstelle eine Matrix** $T \in \mathbb{R}^{nxn}$ **aus** U **durch Normalisierung der Reihen zu** 1
- Die neuen Datenpunkten y sind die Reihen der Matrix U (Erste Reihe ist der erste Datenpunkte, usw.)
- Clustere die neuen Datenpunkte in Cluster $C_1, ..., C_k$ (K-Means wird empfohlen, andere Modelle sind aber auch nutzbar)

Ausgabe: Cluster $A_1, ..., A_k$ wobei $A_i = \{j | y_j \in C_i\}$

4 Anwendungsbeispiel

In [2] nutzen sie Spectral Clustering um die Knoten eines large scale Network zu clustern. Dabei nutzen sie eine angepasstere und weiter entwickelte Form dieses Clusterings um eine sogenannte Q Function zu maximieren. Dabei handelt es sich um objektive Funktion die die Anzahl von Clusterzentren eines Graphen schätzt.

In [3] wird Spectral Clustering zum ersten mal für 3-D Mesh Segmentation genutzt. Dabei wird zuerst eine Ähnlickeitsmatrix erstellt, welche die Likelyhood codiert, dass zwei Gesichter der gleichen Gruppe angehören. Anschließend werden die Eigenvektoren erstellt oder die Laplace ähnliche Matrix. Um anschließend darauf zu clustern. Im anderen Teil entwickeln sie einen Algorithmus, darauf basierend, der die 3-D Meshes entlang der konkaven Regionen segmentiert.

5 Auswertung

Spektrales Clustering hat 2 Vorteile gegenüber „normalen "Clustering Methoden. 1. Es ist robust gegenüber der Form den die Verteilung hat. Während K-Means zum Beispiel nur Kreise darstellen kann, kann Spektrales Clustering auch andere Formen erkennen, wie sich überlappende Spiralen. 2. Der Berechnungsaufwand ist linear, sobald ein geeigneter Ähnlichkeitsgraph gefunden wurde. Es besteht auch nicht die Gefahr in einem lokalen Minima hängen zu bleiben oder die Notwendigkeit den Algorithmus mehrmals zu mit unterschiedlichen Parametern zu starten. Nichts desto Trotz hängt die Qualität vom gewählten Graphen ab, und auch kleine Unterschiede im Graphen können zu abweichenden Ergebnissen führen.

Generell verspricht diese Verfahren gute Ergebnisse in schneller Zeit, wenn man ein Gefühl dafür hat, wie man den Graphen anhand der Eingabedaten, wählen soll. Den optimalen Graphen zu finden ist aber weiterhin eine nicht triviale Aufgabe. Dieses Prinzip lässt sich auch auf andere Problemstellungen übertragen, da das hier vorgestellte Prinzip ja hauptsächlich die Eingabedaten transformiert.

6 Quellen

1. Luxburg, U. (2007). A tutorial on spectral clustering. Statistics and Computing, 17(4), 395-416. doi:10.1007/s11222-007-9033-z

2. Smyth, P., White, S. (2005). A Spectral Clustering Approach To Finding Communities in Graphs*. Proceedings of the fifth SIAM international conference on data mining (Vol. 119, p. 274). Society for Industrial Mathematics.

3. Liu, R., Zhang, H. (n.d.). Segmentation of 3D meshes through spectral clustering. 12th Pacific Conference on Computer Graphics and Applications, 2004.

PG 2004. Proceedings., 298-305. Ieee. doi:10.1109/PCCGA.2004.1348360

4. Joseph Fraunhofer: Bestimmung des Brechungs- und des Farbenzerstreungs-Vermögens verschiedener Glasarten, in Bezug auf die Vervollkommnung achromatischer Fernröhre. In: Annalen der Physik. 56, Nr. 7, 1817, S. 264–313

5. Karl Pearson: On lines and planes of closest fit to a system of points in space. In: The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science. Series 6, 2, S. 559–572 (1901)

Using Fuzzy Decision Trees for Ranking and Regression Problems

Sebastian Mai

Otto-von-Guericke-Universität
39106 Magdeburg
sebastian.mai@st.ovgu.de

Abstract. Fuzzy decision trees are interpretable and comprehensible tools. They thereby provide a compromise between decision trees and more complex methods. A general overview about fuzzy decision trees is provided. Furthermore soft decision trees are presented as one example of a specific fuzzy decision tree.

Keywords: fuzzy decision tree, machine learning, classification, ranking, regression, soft decision trees, roc-metrics

1 Introduction

1.1 This is a fuzzy decision tree

In a nutshell a fuzzy decision tree is a decision tree making use of fuzzy logic. When using decision trees the data is usually modelled as a set of objects that are each represented as a set of attribute values [3] .

A simple algorithm using a decision tree to classify an object could be described as follows: The object is inserted at the root of the tree. There an attribute-test is performed. The result of this test determines which child node the object shall be passed to. This happens at all internal nodes until a leaf of the tree is reached. The label of the leave then indicates which class the object belongs to. [3]

So how do we change the tree to make use of fuzzy-logic?
Fuzzy logic makes it possible to model uncertainty. This is achieved by replacing a boolean variable that can either be true or false. The new value can take any value in [0,1] that could be interpreted as the probability that a predicate is true. The original behaviour is hereby not changed at all.
A solution is to add such a value $p \in [0, 1]$ to each node indicating whether the attribute test on the parent nodes were positive $p = 1$ or negative $p = 0$.

The new parameter p can be interpreted as the degree of membership a given object has towards a class (or a set of classes) and thus allows to represent the results of the attribute-tests on a much finer scale. Very important is that now not only one single path can be taken into consideration. Multiple paths are followed at once. The degree of membership to a class is determined by the sum of the values p of the leaves assigned to the class.

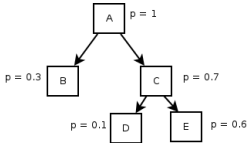

Fig. 1. a fuzzy decision tree with degree of membership

2 Advantages of fuzzy decision trees

2.1 Problems that can be solved by fuzzy decision trees

There are three classes of problems that can be solved using fuzzy decision trees.

classification: Given is a set of objects belonging to a set of classes. The question now is: To which class does the object belong?

> "[However] it is still unclear whether fuzzy decision trees can systematically and significantly outperform non-fuzzy trees in terms of classification accuracy." [1]

regression: When it comes to produce a numerical output fuzzy decision trees can clearly outperform non-fuzzy trees in terms of accuracy.
Let's assume we want to approximate the first half-period of the sinus-function:

$$f(x) = \sin x, x \in [0, \pi]$$

A simple solution would be to pick some sampling points i.e.

$$\{0, \frac{1}{3}\pi, \frac{1}{2}\pi, \frac{2}{3}\pi, \pi\}$$

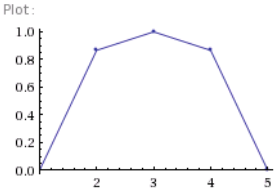

Fig. 2. piecewise linear output

And afterwards split up the input into intervals around those points.
By introducing fuzzy values only at the last internal node you would be able

to add a fuzzy value to the two sampling points surrounding the point you are processing indicating where the point is located in between the two sampling points. With this two values and the values of the function it is possible to estimate a point in between the two sampling points. This would result in a piecewise linear function as in Fig. 2 - a remarkable improvement to the normal decision tree.

ranking: Given the same data as in classification-problems the aim is now to provide an order of classes that fits a specific object most to least.
An example: There are three classes describing the weather outside: sun, rain and snow. Let's assume it's a hot, sunny day. The correctly ranked classes would be sun, rain, snow. If it is freezing cold outside and the sun shines the correct order would be sun, snow, rain. In both cases the correct answer to the classification-problem would be the same, whereas the order of the classes change.
This simple example shows that a tree that is useful for classification not automatically is a good ranker and vice versa. A particular advantage in this field is, that a fuzzy decision tree can be both a good ranker and a good classifier. [1]
Another advantage of fuzzy decision trees is, that the degree of membership p for each class can be used to order the classes and it is not necessary to come up with additional features in the tree.

2.2 General advantages of fuzzy decision trees

A fuzzy decision tree might handle numerical input with more accuracy than a normal decision tree.
Furthermore a fuzzy decision tree might be smaller than an equal non-fuzzy tree. For example if you would want to analyse colors you would be able to split orange into red and yellow in one step, whereas you would need either more levels or more child-nodes in a non-fuzzy tree.
Another interesting possibility is to rate the objects in the training set. For example measurements could be available from two different sources A and B. The measurements from source A are very exact whereas the measurements from source B aren't as reliable. Nevertheless both sources shall be used to train a fuzzy decision tree. By lowering the initial degree of membership p at the root node of the tree for objects from source B it can easily be accomplished to enhance the influence of the more accurate measurements from source A on the resulting tree.

A fuzzy decision tree is well readable. This characteristic makes it very interesting for educating. Another use-case are systems that shall be validated by hand after the tree is trained.
Furthermore fuzzy decision trees are easier to debug than more complex tool. The main reason for this is that there are many ways of tree traversal and tools to visualize trees are available because of the important role the play in computer science in general.

3 Measuring the quality of fuzzy decision trees

It is not easy to rate the performance of a decision tree. There are of course some obvious metrics like CPU-time, average error or the complexity of the algorithm. One metric that especially applies to fuzzy decision trees is called area under curve (AUC). The AUC is the area under the receiver operating characteristic. Assume a certain threshold is used to decide whether an instance belongs to a class or not. Tuning the threshold up would increase the false-negative-rate. Tuning it down on the other hand would result in a higher false-positive-rate.

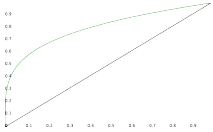

Fig. 3. roc-graph for a typical and a random classifier

Therefore we are able to compute the true-positive-rate as function of the false-positive-rate (turning down the threshold). When classifying completely at random the resulting function would be $f(x) = x$. The optimal curve would be $f(x) = 1$ meaning, that all instances are classified correctly if the threshold is set to the correct value. Usually the roc-curve is somewhere in between. The goal now is to maximize the area under the curve to reach the optimal ratio between false-positive and false-negative classifications.
To rate the accuracy of regression trees mostly mean squared error and related statistical measurements such as variance are used. [5]

4 Algorithms to generate fuzzy decision trees

I have explained now, how a fuzzy decision tree works and which problems can be solved by fuzzy decision trees. When generating a decision tree there are generally two questions to be answered. Which leave shall be split into more leaves? What is the attribute-test to perform in the new split? At first general approaches and afterwards two specific algorithms are proposed.

4.1 General methods

adding fuzzy behaviour after building the tree There are many great algorithms to generate decision trees most important C4.5 and CART. [1] The fuzzy behaviour is then added interpreting the resulting tree another way than originally intended. Therefore additional data for each node can be collected while performing the algorithm.

using a fuzzy technique from start to end Another method is to use fuzzy values all the time. Finding the function for splitting a node is much more difficult than in non-fuzzy trees. The reason for that is that there are usually more free parameters than in normal trees. Furthermore you cannot use the same method to measure the information-gain of a new node.

4.2 Soft decision trees

Soft decision trees are fuzzy decision trees producing numerical output. A node in the tree is characterised by two numerical values α and β. α represents the location of a predicate like $x < 5$ which would lead to $\alpha = 5$.
The value of β is used to create an interval around α. In this interval the split is softened. That means the value p of the left child p_{left} would be 0 if $x \geq \alpha + \beta$ and p if $x \leq \alpha - \beta$. In the interval the value would be on the line connecting both the interval-boarders. An additional requirement is: $p_{left} + p_{right} = p$

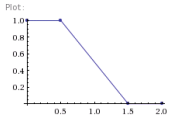

Fig. 4. the function used for splitting $\alpha = 1, \beta = 0.5$

If the p-Values in all leaves are calculated the result is calculated for example by computing $\sum p_{leaf} \cdot l$ where l is the label (value) of the leaf. According to the use-case other functions might be chosen here.

Building of a soft decision tree consists of three steps:

growing For every node $\alpha\beta$ and the labels of the child nodes are chosen to minimise the squared error function. [2] A leave is reached when either the level of the tree reached a certain limit or the error function decreased to a certain value.

pruning Pruning is used to remove irrelevant parts of the tree thus enhancing readability and performance of the tree. First the nodes are ordered from least to most relevant. Node after node is collapsed and the mean absolute error (MAE) of the sub-tree is computed. Afterwards the best tree is picked. The best tree in this case is the one with the MAE next to $\min(MAE) + \sigma_{MAE}$[2]. For growing and pruning the training set is split into two disjoint subsets the growing set GS and the pruning set PS. The pruning set is used to compute the MAE and σ_{MAE}.

refitting / backfitting Refitting and backfitting both are techniques to optimize the parameters in the tree once the layout of the tree is fixed. Refitting is the less consuming of both algorithms and relies on optimising only the leave-parameters via matrix inversion. Backfitting is the alternative that uses more computation-time thus leading to better results. It uses back-propagation to optimise all parameter values using the objects from pruning set and growing set.

Soft decision trees have some advantages over crisp decision trees. The main advantage is that using soft splitting strongly reduces the variance of the tree. This is caused by the much better approximates at the split locations. The overlapping classes lead to less differing values at those points.
However the improvements have to be paid with much greater resource consumption.

5 Conclusion

Fuzzy decision trees can overcome some disadvantages of normal decision trees. This can be achieved staying readable and interpretable. However it is not to be expected that fuzzy decision trees outperform non-fuzzy ones at their best use-cases. Good use-cases for fuzzy decision trees are those with naturally overlapping classes or regression problems with low variance. [2] suggests that the resource consumption of soft decision trees could be improved by further research.

References

1. Eyke Hüllermeier, Stijn Vanderlooy: Why Fuzzy Decision Trees are Good Rankers, 2011
2. Cristina Olaru , Louis Wehenkel: A complete fuzzy decision tree technique, Fuzzy Sets and Systems 138 (2003) 221 – 254
3. J.R. Quinlan: Induction of decision trees, Machine Learning 1, 1986 Kluwer Academic Publishers, Boston
4. Tom Fawcett: ROC Graphs: Notes and Practical Considerations for Data Mining Researchers, HPL-2003-4, 2003
5. D. Sculley: Combined Regression and Ranking, 2010

Learning Association Rules using Evolutianary Algorithms

Kai Dannies
kai.dannies@st.ovgu.de

Otto-von-Guericke University Magdeburg

Abstract. Association Rules are widely used espacially in data mining such as market-basket analysis. Some applications are product placement and peronalized advertisements. One of the challenges in finding good association rules is the rapidly growing number of possible rules with increasing number of attributes. This paper shall present an overview about a specific approach handling this problem: Using evolutionary algorithms to find a good set of association rules.

1 Introduction

Evolutionary algorithms are commonly used to find good solutions in problems with a huge number of possibilities, e.g. finding the best move of a chess game. Basically they are imitating the biological evolution with an evolving basis population. The "weakest" individuals, where strongness is defined by a fitness function, will die out (survival of the fittest). When generating association rules with a great number of possible parameters, e.g. product placement in a supermarket, the number of possible association rules can raise to an enormous number.

It seems logical to use evolutionary algorithms finding good association rules in high-dimensional search spaces. This paper will cover this topic. Therefore, in chapter 2, I will explain the basics of evolutionary algorithms and association rules. Chapter 3 will present some challenges of this topic. Chapter 4 will give an overview about recent research covering this topic. In Chapters 5 i will outline one chosen algorithms. In Chapter 6 I will conclude this paper by a short comparison und a summary of this topic.

2 Basics

2.1 Evolutionary Algorithms

As mentioned above, evolutionary algorithms are imitating biological evolution. So the general steps of an EA are easy to understand:

1. Initialize Population (e.g. generate random potential solutions)
2. Evaluate individuals in population by a defined fitness function
3. Repeat until Stopping Criterion is met:

(a) Select individuals from current generation
(b) Recombine them to obtain new individuals (offsprings)
(c) Evaluate new individuals
(d) Replace some or all individuals of the current population by offsprings
4. Return the best individuals so far

These algorithm uses the idea that strong individuals combined or slightly changed will generate good and, hopefully, better individuals, so the "Survival of the fittest" strategy will result in a good approach to the best solution by using limited space and time. Of course there are several (problem specific) questions to answer:

- How to represent a single individual?
- How to combine several individuals to a new solution?
- Which stopping criterion should i choose?
- Which individuals shall proceed to the next generation?
- How to measure the quality of an individual?

Some of this questions have standard answers which can be used to many different problems. The stopping criterion could be a fixed number of iterations (generations) or a too small improvement of the quality of the best solution.

For the recombination genetic algorithms, a subset of the evolutionary algorithms, propose to use operations which have an equivalent in nature: Crossover and Mutation. Normally the representation of an individual is a list, where each position in the list represents one attribute and the value at this position defines the shape of the attribute. The mutation changes randomly one or more attribute values. Crossover choose two individuals and takes for each attribute randomly one of the values. Of course there are several different answers to each of the given questions. After explaining Association rules there will be more specific replies.

There are several advantages using evolutionary algorithms for a wide range of problems. At first we don't make any assumptions about the problem. Secondly by using more than one starting candidate we try different ways to the solution to raise the chance to get a good solution. By recombination of different good solutions a too early convergence is avoided. Additionaly the used space and time can be very small by only storing the best individuals and choosing an appropriate stopping criterion.

2.2 Association Rules

Association Rules are mainly used to find interesting relations between variables in large data sets. The original definition, given by Agrawal et al.[1] is the following:

$I = \{i_1, i_2, ..., i_n\}$ is a set of n binary attributes, also called items. $D = \{t_1, t_2, ..., t_m\}$ is called the given database. Each transaction t_i contains a subset of the items in I. An association rule(AR) is defined by $A \Rightarrow B$ where $A, B \subset I$ and $A \cap B = \emptyset$. A is called antecedent and B consequent. For example one could find following association rule:

buy diaper, friday afternoon ⇒ buy beer

Which means that someone buys diaper at friday afternoon he will also buy beer. This unintuitive rule has actually an interesting explanation: Young fathers are buying in for the weekend and besides the needed diaper they buy beer for themselves.

The general algorithm to find all association rules is the following:

1. minimum support is applied to find all frequent item sets in database
2. frequent itemset with mimimum confidence are used to form association rules

Where support $supp(X)$ is defined as the probability $P(X)$ to find the specific item X set in a randomly chosen transaction. The confidence of a rule $X \Rightarrow Y$ is defined by $\frac{supp(X \wedge Y)}{supp(X)}$.

The second step is pretty easy, so let's have a glance at the first one. For n binary variables there are $2^n - 1$ possibilities to combine them, so the number of variables grows exponentially. We can exploit some properties of those sets:

– if an itemset with k+1 members is frequent, all of its subsets is frequent, too
– if an itemset with k members is not freqent, neither of its supersets is frequent

But even with this properties the number of frequent itemsets can be too big to be computed efficiently. This is where Evolutionary algorithms can help.

3 Using EA finding AR's

As promised in chapter two i will give some more precise answers to the given questions:

How to represent a single individual?

A single individual should be a possible solution of the problem. Here we are looking for a set of association rules so we could conclude one individual is a set of rules. On the other hand we could choose a complete generation as the solution and therefore each individual could be one rule. The first possibility is known as Pittsburgh approach [9] but the second one was also described in literature, e.g. in iterative rule learning [11] or genetic cooperative-competetive learning [7]. One example for the second possibility is shown in chapter five.

How to combine several individuals to a new solution?

In Chapter two I already described two standard operators; Crossover and Mutation. This works just fine for binary or discrete values of our attributes. We have to adapt these operators slightly when dealing with continous values. Our rules will change from

$$i_1 \wedge i_2 \wedge ... \wedge i_n \Rightarrow i_m \wedge i_m \wedge ... \wedge i_{m+k}$$

to

$$(l_1 \leq i_1 \leq u_1) \wedge ... \wedge (l_n \leq i_n \leq u_n) \Rightarrow$$
$$(l_m \leq i_m \leq u_m) \wedge ... \wedge (l_{m+k} \leq i_{m+k} \leq u_{m+k})$$

Where l and u are the lower and upper borders of the values for this rule. Here we don't have to switch the boolean value but to combine the borders during the crossover, respectively modifying them during mutation.

Which stopping criterion should I choose?

In literature mostly the standard stopping criteria are applied: Either you reach a maximum number of iterations or your recent improvement was so small that we don't expect a significant improvement in further generations.

Which individuals shall proceed to the next generation?

Before asking which you should know how many individuals shall survive. This number is of course strongly dependent on the first question. If you have chosen that one candidate is a complete rule set you don't have to keep much individuals to get good results. But if you have chosen one chromosom to be just one rule you have, dependent of your number of variables, to keep a lot of rules to get a feasible rule base.

After determining the number of candidates to keep you usually take just the n best individuals to build the pool for the next generation.

How to measure quality of an individual?

In supervised learning we can always look at a training set. One widely used possibility is to divide the training set in two distinct sets: One which is used to derive the rule base and another one to test it. For testing it there are different criteria: In the last section i already explained the support as a measure for the overall impact of an attribute set and the confidence as the measurement of dependence between two attribute sets. There are a lot of additional proposes in literature: (to explain)

- comprehensibility
- diversity
- J-Measure
- perplexity
- coverage
- ...

To get a single number this quality measurements are used e.g. in a linear combination.

4 Recent Research

There are different types of association rules to be learned. [6] differentiates mainly between three different types:

- boolean/categorical association rules
- quantitative/numerical associaton rules
- fuzzy association rules

The first category of this topic is well explored, e.g. by Dehuri et al.[5], Wakabi-Waiswa and Baryamureeba[12], Shenoy et al.[8] or Yan et al.[15]. One logical extension to purely boolean variables is the use of continous variables. Therefore at least the representation of the solution candidates have to be extended from boolean values to some continous interval $lowerBorder < item < upperBorder$. Algorithms were developed by e.g. Aumann and Lindell[3] or Webb[14]. However, whatever algorithm you choose, you will often find the rules overemphasized at the borders. Imagine you find a rule which say e.g. if you buy at least 1kg meat from a duck you will buy a flask wine, too. Of course it's not much of a difference for a human if you buy 990 grams or 1kg, but for the system the rule would have a sharp border.

This shall be overcome by the third class of algorithm: The fuzzy association rules. With help of them we can model unsharp decision boundaries. You may learn two different things using evolutionary fuzzy systems[4][10]: Learning/tuning membership functions of the fuzzy variables and possibly in addition the minimum fuzzy support. I will explain a procedure developing membership functions in chapter five.

5 Genetic Algorithm Optimization of Membership Functions for Mining Fuzzy Association Rules

This algorithm was published in 2000 by [13]. It presents a method for computing parameters of fuzzy sets. As application example they use the algorithm for intrusion detection of a computer network. The general algorithm is:

1. let an expert generate a rule set for the given problem
2. mutate the rule set until a certain number of solution candidates is generated (first generation)
3. evolute the population by using crossover and mutation until some stopping criterion is fullfilled

Representation of individuals

For their algorithm the authors expect that each variable has three fuzzy sets, e.g. "low", "mid" and "high". Each of the fuzzy sets is represented by two fuzzy parameters. This six parameters build one gene of the chromosome. For each fuzzy variable one of these genes is used. In the given example four fuzzy variables are analysed:

– number of SYN, FIN and RST flags in TCP-Headers of last two seconds
– number of destination ports in last two seconds

Overall there are 24 parameters to tune.

Evaluation of fitness

At first, a measurement for similarity of rules, respectively rule sets is needed. Therefore definitions given by [2] is used. For the similarity of two rules $R_1 : X \Rightarrow Y, c, s$ and $R_2 : X' \Rightarrow Y', c', s'$ with c and s as confidence and support:

$$sim(R_1, R_2) = \begin{cases} max\left(0, 1 - max\left(\frac{|c-c'|}{c}, \frac{|s-s'|}{s}\right)\right) & , if\ X = X'\ and\ Y = Y' \\ 0 & , else \end{cases}$$

Furthermore the similiraty of two rule sets is given by:

$$sim(S_1, S_2) = \frac{1}{|S_1||S_2|} \cdot \sum_{R_1 \in S_1} \sum_{R_2 \in S_2} sim(R_1, R_2)$$

As every supervised learning model, genetic algorithms has to be trained on preclassified data. Therefore the authors used 3 different classified sets: One set in a normal situation without any attacks, further divided into a training and a control set. Additionaly there are sets representing two different types of attack, namely IP-spoofing and port scanning. [13] defined five fitness functions, where n is the normal training set, r the control or reference set and a1 and a2 the two abnormal sets.

$$F_1 = \frac{S_{rn}^2}{S_{ra1}S_{ra2}}$$
$$F_2 = \frac{S_{rn}}{S_{ra1}} + \frac{S_{rn}}{S_{ra2}}$$
$$F_3 = 2S_{rn} - S_{ra1} - S_{ra2}$$
$$F_4 = \frac{S_{rn}}{S_{ra1}}$$
$$F_5 = S_{rn} - S_{ra1}$$

Results

The algorithm produces parameters for the needed fuzzy sets. The results are, compared to the rules given by human experts, much better, as you can see in the following table:

	experts	F1	F2	F3	F4	F5
Normal	0.74	0.81	0.81	0.84	0.85	0.84
Abnormal1	0.31	0.05	0.07	0.05	0.04	0.04
Abnormal2	0.32	0	0	0.03	0.06	0.04

Remarkably even for the fitness functions only trained on one type of attack (F_4, F_5), the other was recognized, too. So using evolutionary algorithms to find good fuzzy sets is much better than the human expert, even in this limited space.

6 Conclusion

I presented an alternative theory of generating association rules. Therefore evolutionary algorithms were used. The topic is covered in literature for nearly twenty years now and there are a lot of publications towards this. Jesus et al. [6] gives a broad overview.

Three different types of association rules were explained: Boolean ones, continous ones and fuzzy association rules. For the last type I presented an algorithm which found the parameters of the fuzzy sets. It proves that it generates very good rules for this special application.

This is one of the problems of this topic, the reason for the dozens of publications: There is no general algorithm. Each problem has its one specialities which requires an own solution. This could be topic of further research: Building a framework which can deliver problem specific algorithms.

References

1. Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. *SIGMOD '93 Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, 1993.
2. Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. *Proceedings of the 20th international conference on very large databases*, 1994.
3. Yonatan Aumann and Yehuda Lindell. A statistical theory for quantitative association rules. *Fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, 1999.
4. Oscar Cordn, Francisco Herrera, Frank Hoffmann, and Luis Magdalena. *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases*. World Scientific, 2001.
5. S. Dehuri, A.K. Jagadev, A. Ghosh, and R. Mall. Multi-objective genetic algorithm for association rule mining using a homogeneous dedicated cluster of workstations. *American Journal of Applied Sciences*, 2006.
6. Mara J. del Jesus, Jos A. Gmez, Pedro Gonzlez, and Jos M. Puerta. On the discovery of association rules by means of evolutionary algorithms. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2011.
7. David Perry Greene and Stephen F. Smith. Competition-based induction of decision models from examples. *Machine Learning*, 1993.
8. P. Deepa Shenoy, K. G. Srinivasa, K. R. Venugopal, and L. M. Patnaik. Evolutionary approach for mining association rules on dynamic databases. *Proceedings of the 7th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, 2003.
9. Stephen Frederick Smith. *A learning system based on genetic adaptive algorithms*. PhD thesis, University of Pittsburgh, 1980.
10. F. Herrera und M. Lozano. Genetic fuzzy systems: taxonomy, current research trends and prospects. *Evolutionary Intelligence*, 2008.
11. Gilles Venturini. Sia: A supervised inductive algorithm with genetic search for learning attributes based concepts. *Lecture Notes in Computer Science*, 1993.

12. Peter P. Wakabi-Waiswa and Venansius Baryamureeba. Extraction of interesting association rules using genetic algorithms. *International Journal of Computing and ICT Research*, 2008.
13. Wengdong Wang and Susan M. Bridges. Genetic algorithm optimization of membership functions for mining fuzzy association rules. *International Joint Conference on Information Systems, Fuzzy Theory and Technology Conference*, 2000.
14. Geoffrey I. Webb. Discovering associations with numeric variables. *Seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, 2001.
15. Xiaowei Yan, Chengqi Zhang, and Shichao Zhang. Genetic algorithm-based strategy for identifying association rules without specifying actual minimum support. *Expert Systems with Applications*, 2009.