FUZZY CLUSTER ANALYSIS FOR IDENTIFICATION OF GENE REGULATING REGIONS

Lars PICKERT[a], Frank KLAWONN[b], Edgar WINGENDER[c]

[a] Institut für Betriebssysteme und Rechnerverbund, Technische Universität Braunschweig, Germany

[b] FB Elektrotechnik und Informatik, FH Ostfriesland, Constantiaplatz 4, D-26723 Emden, Germany

[c] Gesellschaft für Biotechnologische Forschung mbH, Braunschweig, Germany

Abstract. The main approach of this work is the implementation of a cluster analysis program for identification of regulatory regions in genomes. These regions are important parts of the genetic pool in higher developed organisms. They are composed of several basic elements, so called transcription factor sites, which can be identified by special analysis tools more or less vaguely. The program we have developed is able to search for two-dimensional clusters in the results of such analysis tools to give hints on gene regulatory regions. For this purpose two fuzzy clustering algorithms have been implemented: The fuzzy c-means (FCM) and the Gath and Geva fuzzy clustering algorithm (GG) with two conventional cluster validity methods and one which has been developed especially for this application. All results of the cluster analysis program can be visualized and documented automatically.

Keywords. fuzzy clustering, fuzzy cluster analysis, genetics, gene regulation

1 Introduction

The genome of an organism contains the complete information to organize its development and function. The coordinate realization of this information, i.e. the precise tuning of the expression of its genes, is of extreme importance. One of the most important steps on which gene expression is regulated is transcription, the copying of DNA into RNA sequences. It is controlled by the regulatory regions on the genomic DNA termed promoters and enhancers, both of them being composed of short sequence elements of 5-25 base pairs (**bp**) length (consider a bp as one character of the DNA sequence code). They interact with a functional class of proteins called transcription factors (**TF**). Occasionally, several individual TF binding sites (**TFS**) together constitute a "composite element" [6] which exerts a function which may be qualitatively different from those of the constituents. The computer-aided identification of enhancers or promoters as gene regulating regions is a difficult task because they cannot be identified by only one search criterion. This problem is due to the abundance of satisfying search criteria and the lack of knowledge about rules of composition, location, coding grammar etc. So far it is impossible to make direct predictions on possible gene regulating regions. So we have to look for another strategy to search for potential enhancers and promoters in eukaryotic DNA sequences. The underlying ideas for this approach base on the following facts:

1. Most (known) eukaryotic enhancers and promoters contain several TFS in an area normally of some hundred base pairs. Figure 1 shows the known enhancer of the simian virus 40 with the contained TFS.
2. For their biological function, TFS depend on a defined sequence context including other TFS.
3. There exist at least three programs for the sequence based identification of potential TFS in DNA. These programs are PatternSearch [9], MatInspector [7] and ConsInspector [2]. These programs produce lists of potential TFS with two attributes: DNA position and score. The score should coincide with the binding affinity of the considered transcription factor.

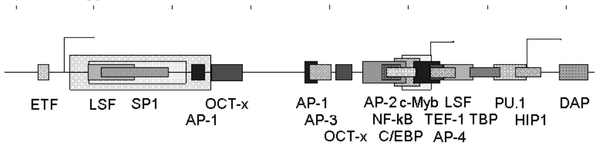

Figure 1: Enhancer of Simian Virus 40 genome as a cluster of several TFS of many different TF. Total sequence length: 5243 bp

From (1.) and (2.) we can conclude that many enhancers and promoters build clusters of TFS on DNA. Point (3.) allows us to expect that we are able to identify many or most of the TFS in a given DNA sequence (at least most of the TFS with high binding affinity). So the idea of a procedure to give hints on possible enhancers and promoters is to search for clusters of potential TFS in the output of available TFS analysis programs. Expected high scored TFS as true positives this means to to look for clusters which contain significant high scored TFS.

2 Strategies for TFS cluster - analysis

According to the available data describing each potential TFS we could imagine two different ways to look for TFS clusters:

2.1 One-dimensional clustering

The simplest idea of a TFS cluster analysis is to cluster only on the DNA position of TFS and so get a one-dimensional cluster space. But depending on the TFS analysis program used, a lot of false positives (error type two) have been produced along with false negatives (error type one). This fact makes it obvious that the data to be clustered contain a lot of noise data. In many cases the scores of TFS could help to filter many false positives. But in the case of one-dimensional clustering this information is not appropriately used and thus cluster analysis would not be able to give helpful hints on possible enhancers or promoters in most cases. Figure 2 demonstrates this problem using an example of one-dimensional TFS data.

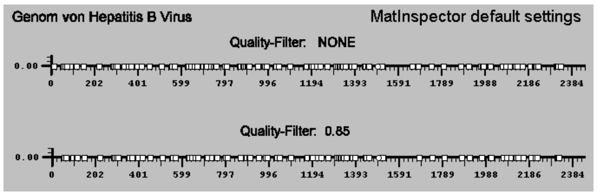

Figure 2: One-dimensional TFS data of the hepatitis B virus (**HBV**) genome. Each white rectangle a potential TFS.

2.2 Two-dimensional cluster analysis and DNA projection

Adding a TFS quality based on the score of the potential TFS as a second dimension in cluster space improves the results drastically as shown in figure 3. Now we are able to differentiate the cluster quality by simple projection on the second dimension (quality) and can choose all clusters with a significant degree of quality for DNA projection. Selected clusters can be projected onto the first axis of cluster space, the DNA position. Such resulting intervals are named potential transcription regulating regions (= **TRR**). Several strategies for selecting 'high quality clusters' can be imagined, e.g. usage of all clusters which contain at least one data point with a quality greater than a given threshold or usage of all clusters which contain only data points that exceed a given quality threshold or even manual cluster selection. The current work uses the first strategy, since it has been experimentally proven that high-scoring TFS may act as an anchor to include low-scoring TFS into a TRR.

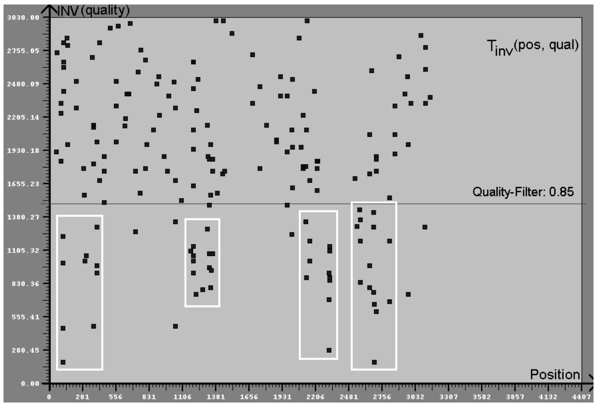

Figure 3: Two-dimensional TFS data of HBV genome.

2.3 Why fuzzy clustering ?

Predictions on potential gene regulating regions can be done only vaguely because of vague data (scored TFS) and vague semantics of potential TFS (false positives, data points between neighboured clusters, etc.). It is obvious that fuzzy clustering can solve these problems in a better way than 'hard clustering' algorithms. So the user can choose one of the best three membership degrees of each data point to a cluster for the defuzzification procedure of clustering assignments. Figure 4 demonstrates the meaning of applied fuzzy clustering for DNA-projection.

2.4 Creating data points from TFS

Before we can start reading TFS data from TFS analysis output and transform it into cluster space we have to ensure that the TFS scores will satisfy the following two conditions:

- true positives should have a similar high quality score,
- true negatives should have a similar low quality score,

to be able to compare the quality of potential TFS of different transcription factors independent of the

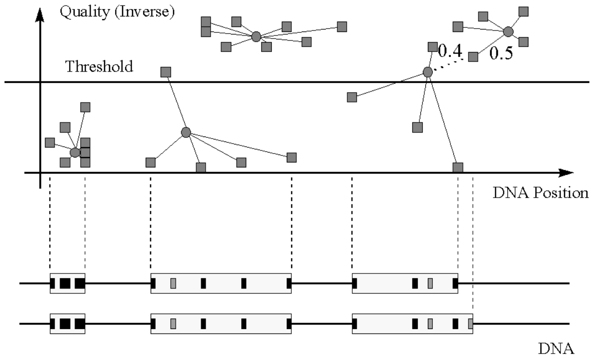

Figure 4: Displaying two-dimensional TFS clusters with two DNA projections depending on used defuzzification: the upper one uses best and the lower one second best level of membership degrees.

used search pattern or search matrix. For this purpose a quality function has been developed which is based on TFS scores and if necessary on search pattern characteristics for each recognized TFS analysis program (PatternSearch, Mat- and ConsInspector). This has been done by empirical work. These quality scores were used as the quality attributes of TFS data for cluster analysis.

After reading the TFS data from a selected TFS analysis output file this data is stored in a relational database. For a new cluster analysis run all or a user-defined subset of the TFS data can be prefiltered by quality threshold filters to get rid of obviously low scored data. A second optional filter, the so called *shrink filter* eliminates *false TFS hot spots*. These hot spots are multiple TFS hits of one or more (functionally similar) factors at a certain DNA position which do not occur as multiple TFS at this position in reality. Using a shrink filter recognized hot spots of selected factors will be shrunk into one data point for clustering to avoid overweighting a DNA position. On the other hand, a weighted clustering process could be performed using multiple search patterns for special factors and skipping the shrink filter procedure.

As a final step of creating data points from TFS data a transformation from world-coordinates DNA-position and quality score into cluster space with 'compatible axis scalings' must be defined. Obviously, different scalings of coordinate axes against each other could influence the result of data partitioning drastically. In this case, the data space could be limited to a finite area because DNA positions are limited by sequence length. The range of possible TFS quality can be limited by simple thresholds for minimum and maximum quality since true positives could be estimated against a minimum and false positives against a maximum quality score. So coordinate transformation could be reduced to a simple interval transformation with three user-definable parameters: minimum and maximum quality threshold (Y_{min} and Y_{max}) and maximum quality value in cluster space (V_{max}). Formula (1) shows the transformation T_{inv} from TFS data to cluster data points. Note that for improving the visualization of the DNA projection the quality scores will be transformed inversely into cluster space so that the higher scores are nearer to zero.

$\mathbf{T}_{\text{inv}}$:

$$[0, X_{\text{max}}] \times [Y_{\text{min}}, Y_{\text{max}}] \mapsto [0, U_{\text{max}}] \times [0, V_{\text{max}}]$$

$$(x, y) \mapsto \left(x, \frac{V_{\text{max}}}{(Y_{\text{max}} - Y_{\text{min}})} \cdot (Y_{\text{max}} - y) \right) \quad (1)$$

X_{max}	:	Sequence length in bp
Y_{max}	:	Est. upper quality for false positives
Y_{min}	:	Est. lower quality for true positives
V_{max}	:	Max. range of quality in cluster space
U_{max}	:	Sequence length in cluster space

$$U_{\text{max}} := X_{\text{max}} \quad \text{and} \quad Y_{\text{min}} < Y_{\text{max}}.$$

In most cases best results have been achieved with setting the maximum quality range V_{max} in cluster space two to three times higher than the expected maximum width of TRR regions (300-500 bp). This maximum TRR width will play a special role for the automatic determination of the optimal number of clusters as we will see later (cluster quality constraint (=**CQC**)).

3 Fuzzy clustering algorithms

In order to understand in which way fuzzy clustering can help us in the search for gene regulating regions, we take a brief look at the fundamental ideas of objective function based fuzzy clustering in general. (For an overview see for example [5].)

3.1 Objective function based fuzzy clustering

In objective function based fuzzy clustering a suitable fuzzy partition for a given data set $X = \{x_1, \ldots, x_n\} \subseteq \mathbb{R}^p$ has to be found. That means that for each datum x_k and each cluster i a membership degree u_{ik} between 0 and 1 has to be determined. u_{ik} indicates the degree to which datum x_k is assigned to cluster i. For probabilistic (fuzzy) clustering the constraint

$$\sum_{i=1}^{c} u_{ik} = 1 \qquad \text{for all } k \in \{1, \ldots n\}$$

is required, i.e., the membership degree 1 of each datum can be distributed over different clusters.

A cluster is represented by a typical member (usually the center of the cluster). Thus, objective function based fuzzy clustering aims at minimizing the objective function

$$J(X,U,v) \;=\; \sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^m d^2(v_i,x_k) \qquad (2)$$

under the above mentioned constraint by a suitable choice of the cluster centers v_i and the matrix U of membership degrees u_{ik}. $d(v_i, x_k)$ is the distance between prototype v_i and datum x_k. The parameter $m > 1$ is called fuzziness index. For $m \to 1$ the clusters tend to be crisp, i.e. either $u_{ik} \to 1$ or $u_{ik} \to 0$, for $m \to \infty$ we have $u_{ik} \to 1/c$. Usually $m = 2$ is chosen.

For the moment, we assume the number of clusters c to be fixed.

Differentiating (2) leads to the necessary conditions

$$u_{ik} \;=\; \frac{1}{\sum_{j=1}^{c}\left(\frac{d^2(v_i,x_k)}{d^2(v_j,x_k)}\right)^{\frac{1}{m-1}}} \qquad (3)$$

and

$$v_i \;=\; \frac{\sum_{k=1}^{n}(u_{ik})^m x_k}{\sum_{k=1}^{n}(u_{ik})^m} \qquad (4)$$

for a minimum. These equations are therefore applied alternatingly for the computation of the clustering result until there are (almost) no changes in the matrix U.

The most simple fuzzy clustering algorithm is the fuzzy c–means (FCM) (see f.e. [1]) where the distance d is simply the Euclidean distance. It searches for spherical clusters of approximately the same size.

Gustafson and Kessel [4] and later on Gath and Geva [3] extended the FCM in order to adapt to different cluster shapes and sizes. Their main idea is to introduce a symmetric, positive definite matrix for each cluster, carrying out a transformation on the difference vector $(v_i - x_k)$ whose length determines the distance of a datum to a cluster center.

3.2 Determining the number of clusters

The usually unknown number of clusters can be determined on the basis of a suitable cluster validity measure (see for instance [1, 3, 5, 8, 11]) that evaluates a given fuzzy partition. The fuzzy clustering is carried out with varying numbers of clusters and as the final result the fuzzy partition is chosen which obtained the best evaluation value. However, it turned that for our problem the common validity measures are not adequate. Therefore, we designed a specific strategy.

The maximal size of gene regulating regions is bounded and thus clusters larger than this size do not make sense. Therefore, we defined the function twd (too wide cluster):

1. Let $\max_{ext}$ be the maximal admissible length of a cluster (in the direction of the x-axis for the DNA positions).
2. We fix a quality value T in order to neglect clusters that contain only low scorers.
3. We defuzzify the fuzzy partition by assigning each datum to the cluster for which it yields the highest membership degree.
4. For these crisp clusters that contain at least one datum with a quality better than T (i.e. a value lower than T), we compute their extension in x-direction.
5. The function twd yields the value 0 if the extension in x-direction of all clusters considered in 4. is less than $\max_{ext}$, otherwise twd is 1.

The number of clusters is determined by increasing it stepwise starting from 1 until twd reaches the value zero. When we have to increase the number of clusters, we do not add new prototypes randomly, but within those clusters that cause twd to be 1. The new prototype is placed on the position of the datum which is assigned to the corresponding cluster but has the lowest membership degree or near the datum in the cluster with the largest Euclidean distance to the prototype. In this way, clusters that are too large are split.

4 Results

The verification process of the TFS cluster analysis results was done on sequences with known gene regulatory regions. Such information was obtained from TRANSFAC [10], a relational database available via WWW that contains detailed information about experimentally verified TFS. The following two subsections report two examples of performed TFS cluster analysis: one completely successful (genome of simian virus 40) and one that demonstrates some problems related to the TFS data (genome of hepatitis B virus).

4.1 Cluster Analysis of whole simian virus 40 (SV40) genome

Figure 1 shows the known enhancer of this DNA sequence. After a TFS analysis using MatInspector with all available search matrices (vertebrates) from TRANSFAC, we obtained the result shown in figure 5. To improve speed and accuracy of TFS cluster analysis, all data points in the area where obviously no clusters could be found were filtered out with a

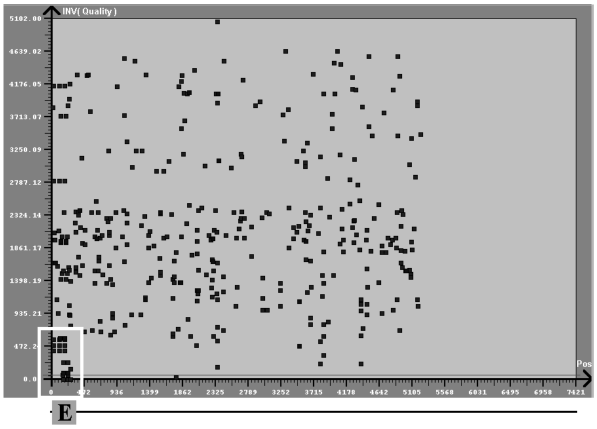

Figure 5: Two dimensional TFS data of the SV40 genome. A strong cluster of high-quality TFS is marked by a white rectangle. The position of the experimentally verified enhancer is shown by grey 'E' box.

quality score filter (in this example quality threshold = 0.95, choosen from visualization). Figure 6 shows the result of a Gath and Geva run and following DNA projection that leads to the displayed TRRs. Different matrices for most known tran-

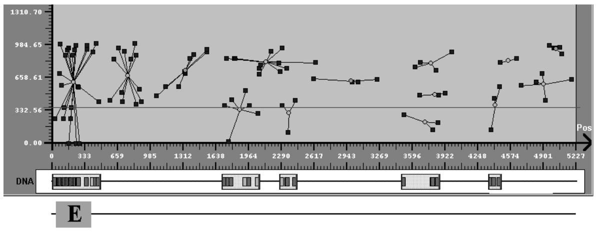

Figure 6: Gath and Geva clustering of prefiltered (quality > 0.90) SV40 data with DNA projection. The position of the experimentally verified enhancer is shown by grey 'E' box.

scription factors have been used. Resulting false TFS hot spots were compensated by a special shrink filter. The leftmost TRR with the high density of contained potential TFS represents nearly exactly the range of the authentic SV40 enhancer. By looking at the best TFS hit of each TF in this automatically documented TRR, we get a good description of the real SV40 enhancer:

```
TRR_2  FROM: 27 TO: 465
   AP-1        AT:  27 (-)    QUAL: 0.9750  MATR: V$AP1_Q
   Sp1         AT:  72 (+)    QUAL: 0.9300  MATR: V$SP1_Q6
   Sp1         AT:  93 (+)    QUAL: 0.9420  MATR: V$SP1_Q6
   AP-1        AT:  112 (+)   QUAL: 0.9750  MATR: V$AP1_Q2
   Oct-1       AT:  117 (+)   QUAL: 0.9640  MATR: V$OCT1_Q6
   NF-kappaB   AT:  163 (+)   QUAL: 1.0000  MATR: V$NFKB_C
   AP-1        AT:  184 (+)   QUAL: 0.9750  MATR: V$AP1_Q2
   Oct-1       AT:  189 (+)   QUAL: 0.9640  MATR: V$OCT1_Q6
   NF-kappaB   AT:  235 (+)   QUAL: 1.0000  MATR: V$NFKB_C
   C/EBPbeta   AT:  238 (-)   QUAL: 0.9022  MATR: V$CEBPB_01
   AP-1        AT:  254 (+)   QUAL: 0.9430  MATR: V$AP1FJ_Q2
   AP-4        AT:  268 (+)   QUAL: 1.0000  MATR: V$AP4_Q6
   c-Myb       AT:  465 (+)   QUAL: 0.9580  MATR: V$CMYB_01
END OF TRR_2
```

The meaning of the other TRRs is to be proofed by experimental verification. Only the one enhancer is actually known in the SV40 genome.

4.2 Cluster analysis of whole hepatitis B virus (HBV) genome

The identification of the two HBV enhancers with TFS cluster analysis is more difficult than in the first example. First, the composition complexity of both enhancers is very low (figure 7) and second, the main factor C/EBP can not be identified very well by currently available search matrices (true positives often get a poor score and thereby become false negatives). As shown in figure 3 the TFS data

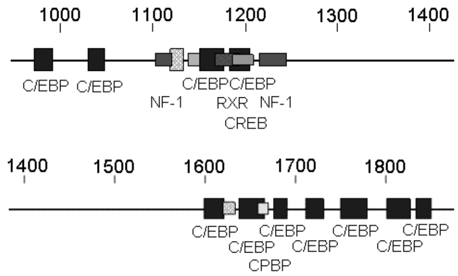

Figure 7: Two Enhancers of HBV genome as clusters with low composition complexity (over 75% C/EBP). Total sequence length: 3182 bp

of the HBV genome contains no such strong 'high-quality' clusters like the SV40 genome. Although a search for the known TFS fails in most cases for C/EBP, the Gath and Geva algorithm was able to detect the first enhancer, as shown in figure 8. This is an example of composing effects from noisy and incomplete data in TFS cluster analysis.

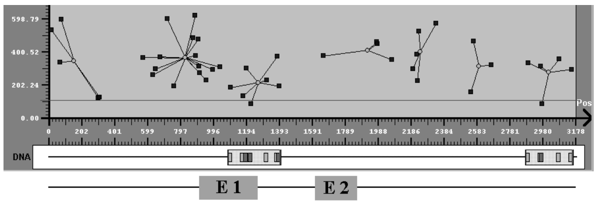

Figure 8: Gath and Geva clustering of prefiltered (quality > 0.90) HBV data with DNA projection. Although the first enhancer (grey 'E1' box) was detected appropiately, the second one (grey 'E2' box) could not be found since abundance of correct C/EBP TFS data. The quality of TRR for the first enhancer can be improved by adding two further cluster prototypes.

5 Discussion

We have shown that clusters of high scored potential TFS can be found in many DNA sequences and often such clusters coincide with experimentally verified enhancers or promoters. In some cases this procedure fails for the following reasons:

1. Failure of TFS analysis. For some transcription factors they produce too much false positives even with restrictive search parameters.

2. In the case of homeogenous clusters, the cluster analysis depends on the quality of the respective search pattern. This risque is largely diminished in the case of clusters of high complexity.

3. Not all TFS exhibit a strong factor/ DNA binding affinity.

Problem (1.) could be reduced by selecting good and reliable search patterns (verified by many tests) or improved TFS analysis programs (e.g. based on physical DNA structure). Point (3.) could be handled by searching for composite elements in (and near) the TRRs. Further investigations will improve the knowlegde about good search patterns, good parameters, like coordinate transformation etc. If more knowledge about gene regulatory regions and improved TFS analysis tools accumulates, the TRR data from cluster analysis could be used as evidence knowledge for a knowledge based system. Although more information than only sequence based TFS analyis data is neccessary for an exact identification of enhancers and promoters (e.g. physical structure analysis), this new method could give important hints on potential gene regulating regions.

References

[1] J.C. Bezdek, Pattern recognition with fuzzy objective function algorithms, (Plenum Press, New York, 1981).

[2] K. Frech, G. Herrmann, T. Werner. Computer-assisted prediction, classification and delimitation of protein binding sites in nucleic acids. Nucleic Acids Res. 21 (1993), 1655-1664.

[3] I. Gath, A.B. Geva, Unsupervised optimal fuzzy clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (1989), 773–781.

[4] D.E. Gustafson, W.C. Kessel, Fuzzy clustering with a fuzzy covariance matrix, Proc. IEEE CDC, San Diego (1979), 761–766.

[5] F. Höppner, F. Klawonn, R. Kruse, Fuzzy–Clusteranalyse: Verfahren f"ur die Bilderkennung, Klassifikation und Datenanalyse (in German). Vieweg, Braunschweig (1997).

[6] O.V. Kel, A.G. Romaschenko, A.E. Kel, E. Wingender. A compilation of composite regulatory elements affecting gene transcription in vertebrates. Nucleic Acids Res. 23 (1995), 4097-4103.

[7] K. Quandt, K. Frech, H. Karas, E. Wingender, T. Werner. MatInd and MatInspector: new fast and versatile tools for detection of consensus matches in nucleotide sequence data. Nucleic Acids Res. 23 (1995), 4878-4884.

[8] M. Sugeno, T. Yasukawa, A Fuzzy–Logic–Based Approach to qualitative modeling, IEEE Transactions on Fuzzy Systems 1. 1993. 7–31.

[9] E. Wingender, H. Karas, R. Knüppel. TRANSFAC database as a bridge between sequence data libraries and biological function. Biocomputing, proceedings of the 1997 Pacific Symposium (1997), 477-485.

[10] E. Wingender, A.E. Kel, O.V. Kel, H. Karas, T. Heinemeyer, P. Dietze, R. Knüppel, A.G. Romaschenko, N.A. Kolchanov. TRANSFAC, TRRD, and COMPEL: towards a federated database system on transcriptional regulation. Nucleic Acids Res. 25 (1997), 265-268.

[11] X.L. Xie, G. Beni, A validity measure for fuzzy clustering, IEEE Transactions on Pattern Analysis 13 (1991), 841–847.