A Combination of Genetic Programming and Cluster Analysis for Single Class

Cuong To[1], and Jiri Vohradsky[1]

[1] Laboratory of Bioinformatics, Institute of Microbiology, ASCR, Videnska 1083, 142 20
Prague, Czech Republic
cuongto@biomed.cas.cz

Abstract. In this paper, a novel algorithm for single class based on genetic programming (GP) is introduced. Single class problem is converted into symbolic regression, and then genetic programming is used to search a polynomial function which approximates symbolic regression problem. Four real databases (one transcriptomics, one proteomics, and two breast cancers) were used to test the algorithm and a comparison with six well-known algorithms was done. The results prove that the algorithm is a rather good one.

1 Introduction

In supervised pattern classification, there are three kinds of problems: single class, multi-class classification, and binary classification. Single class [1] is novelty detection. Multi-class classification [2-8] is usually converted into multiple binary classifications. Maybe, binary classification is the most popular form of supervised pattern classification. In binary classification [9-14], the training set consists of two sub sets called positive set and negative set. The positive set contains patterns which are labeled in the same class. The negative set includes the patterns which are not included in the positive set.

Although patterns of the negative set help the classifier recognizes patterns which are different from the positive set. But there exist some problems of the negative set that can affect searching result as:

+ Number of patterns in the negative set: If the number of patterns in the negative set is high, resulting in a low number of misclassified patterns, the number of true patterns will be low. On the contrary, if the number of patterns in the negative set is low, resulting in a high number of misclassified patterns, the number of true patterns will be high. There is no general rule to determine the best number of patterns in the negative set for every database. It is still a trial and error task.

+ Patterns of the negative set: two negative sets having the same number of but different patterns may give very different results (one result is ideal, the other is not). Therefore, pattern selection for the negative set is critical and is largely a function of trial and error.

Supervised pattern classification methods which do not contain the negative set in the training set remove the disadvantages of the negative set. These methods are called single class classification.

In this paper, we propose an algorithm for single class based on genetic programming [15]. Firstly, single class problem is represented as the symbolic regression. Then genetic programming searches a polynomial function describing the symbolic regression problem.

2 Method

Let the training set be a set of patterns, $TS = \{\mathbf{x}_i \in \mathrm{R}^n, i = 1..m\}$, with m as the number of patterns in the training set.

The main idea of the algorithm is to find a function which represents the mean similarity between one pattern and other patterns of the training set. In other words, we have a problem of symbolic regression. In the symbolic regression, we need a set of pairs ($\mathbf{x}_i$, y_i). The values, $\mathbf{x}_i$, are known in the training set while the values, y_i, need to be determined. In the algorithm, we use Euclidean distance to describe y_i. The following describes y_i:

Average Euclidean distance between two patterns $\mathbf{x}_r = \left(x_{r1}, x_{r2}, ..., x_{rn}\right)^{\mathrm{T}}$ and $\mathbf{x}_s = \left(x_{s1}, x_{s2}, ..., x_{sn}\right)^{\mathrm{T}}$ is:

$$Dis(\mathbf{x}_r, \mathbf{x}_s) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_{ri} - x_{si}\right)^2} \qquad \textbf{(1)}$$

Because x_{ri} and $x_{si} \in [0, 1]$ so $Dis(\mathbf{x}_r, \mathbf{x}_s) \in [0, 1]$. $Dis(\mathbf{x}_r, \mathbf{x}_s) = 0$ means $\mathbf{x}_r \equiv \mathbf{x}_s$. We change 0 to 1 as:

$$Sim(\mathbf{x}_r, \mathbf{x}_s) = 1 - Dis(\mathbf{x}_r, \mathbf{x}_s) \qquad \textbf{(2)}$$

Equation (2) is used to measure similarity of two patterns. The difference between two patterns is calculated:

$$Dif(\mathbf{x}_r, \mathbf{x}_s) = 1 - Sim(\mathbf{x}_r, \mathbf{x}_s) = Dis(\mathbf{x}_r, \mathbf{x}_s) \qquad \textbf{(3)}$$

The mean of similarity between pattern $\mathbf{x}_r$ and other patterns in the training set, TS, is computed as:

$$M(\mathbf{x}_r) = y_r = \frac{\sum_{i}^{m} Sim(\mathbf{x}_r, \mathbf{x}_i)}{m} \qquad \textbf{(4)}$$

$M(\mathbf{x}_r)$ is used to describe y_r in symbolic regression problem. In other words, we use genetic programming to search function $f(\mathbf{x})$ that satisfies the following criterion

$$f(\mathbf{x}_i) \approx M(\mathbf{x}_i),\ \forall i = 1..m \tag{5}$$

2.1 The Fitness Function

In reality, we do not know function type of the training set. Furthermore, it is possible for different training sets to have different functions. So we use polynomial function because polynomial function can approximate any function in principle. The fitness of a tree in population is computed as:

$$Fitness = \sum_{i=1}^{m} \left| f^{k}(\mathbf{x}_i) - M(\mathbf{x}_i) \right| + 2 \times \sum_{i=1}^{m} \sum_{j=i+1}^{m} \left\| f^{k}(\mathbf{x}_i) - f^{k}(\mathbf{x}_j) \right| - Dif(\mathbf{x}_i, \mathbf{x}_j) \right| \tag{6}$$

where:

- f^{k} : polynomial function of k-th tree in population.
- $M(\mathbf{x}_i)$: mean of similarity between pattern $\mathbf{x}_i$ and other patterns in training set computed by Eq. (4).
- $Dif(\mathbf{x}_i, \mathbf{x}_j)$: difference between two patterns $\mathbf{x}_i$ and $\mathbf{x}_j$ computed by Eq. (3).

2.2 Prediction

Let $f(\mathbf{x})$ be the best polynomial function that GP creates. $Min\_TS = \min \{f(\mathbf{x}_i),\ \forall \mathbf{x}_i \in$ training set$\}$. $Max\_TS = \max \{f(\mathbf{x}_i),\ \forall \mathbf{x}_i \in$ training set$\}$. If a tested pattern, **a**, has $f(\mathbf{a}) \in [Min\_TS, Max\_TS]$, then tested pattern, **a**, belongs to the training set.

2.3 Pattern Splitting

There are some reasons why a pattern is split. First, the higher dimension pattern is the lower probability of obtaining good results. Second, pattern splitting can decrease the number of misclassified patterns. Third, each sub pattern can be assigned to one computer and runs independently to allow for parallel computing.

An original pattern is split into q non-overlapping sub patterns such that the sum of dimension of all sub-patterns is equal to the number of dimensions of the original pattern. Therefore, the original pattern search problem becomes q sub pattern search problems. Applying fitness function, Eq. (6), to each sub pattern, we obtain separate result of each sub-pattern. Results of all sub patterns are then combined according to Eq. (7) to breed result of the original pattern.

$$rs = \bigcap_{i=1}^{q} rs_i \tag{7}$$

with

- rs_i : result set of i-th sub pattern.

- *rs* : result set of original pattern.

2.4 Control Parameters of GP

Artificial data were used to determine the value of parameters of genetic programming and created by the following scheme:

Step 1: Create a random set of 2000 *21*-dimensional patterns called set R.
Step 2: Create a random template A.
Step 3: Create a random set of 100 patterns by adding 30% Gaussian noise to pattern template A, having set C.
Step 4: Mix set C with set R, having set B (2100 patterns).
Step 5: Apply algorithm to search C in B.

The performance of the algorithm is measured using two indicators [5], namely sensitivity (*Se*) and specificity (*Sp*):

$$Se = \frac{TP}{|\mathrm{C}|} \tag{8}$$

$$Sp = \frac{|\mathrm{R}| - FP}{|\mathrm{R}|} \tag{9}$$

With

- *TP (true positive)*: the classifier predicts that the pattern is in the training set and the pattern is in the training set.
- *FP (false positive)*: the classifier predicts that the pattern is in the training set but the pattern is not in the training set.
- |C| is number of similar patterns.
- |R| is number of other patterns in database.

Population size

The following results were tested with a probability of crossover of 0.9 and a number of 500 generations. Results are summarized in Table 1. Based on the results of Table 1, we selected 1000 trees for a population.

Table 1. *Se* and *Sp* of different population sizes

Population size	*Se*	*Sp*
40	0.975	0.8135
100	0.9875	0.87
300	1.0	0.8675
500	1.0	0.8715
1000	1.0	0.8915
1500	1.0	0.8755

2000	1.0	0.847
2500	1.0	0.8375
3000	0.9875	0.928

Number of generations

The following results were tested with a probability of crossover of 0.9 and a population size of 1000. Results are summarized in Table 2. We chose 500 for the number of generations.

Table 2. *Se* and *Sp* of different number of generations

Number of generation	*Se*	*Sp*
10	1.0	0.736
20	1.0	0.805
100	1.0	0.813
200	1.0	0.84
300	1.0	0.863
400	1.0	0.8665
500	1.0	0.8915
700	0.9875	0.9025
1000	0.9875	0.914
2000	0.95	0.93

Probability of crossover

The following results were tested with a population size of 1000 and maximum number of generations of 500. Results are summarized in Table 3. We chose 0.9 for the probability of crossover.

Table 3. *Se* and *Sp* of different probability of crossovers

Probability of crossover	*Se*	*Sp*
0.5	0.9875	0.85
0.6	0.9875	0.9
0.7	0.9875	0.9185
0.8	1.0	0.8665
0.9	1.0	0.8915

Control parameters of GP

Table 4 lists all control parameters which are used in the algorithm.

Table 4. Control parameters of genetic programming

Population size:	1000
Maximum generation:	500
Probability of crossover:	0.90

Probability of reproduction:	0.10
Maximum depth for tree created during run:	10
Maximum depth for initial random tree:	7
Terminal set:	$\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$
Function set:	+, -, ×, pow2, pow3, …, pow10

with powX is power of X

How many sub patterns

Pattern splitting method (section 2.3) was tested and the results are summarized in Table 5. The results show that the number of sub patterns from 3 to 6 is optimal depending on the power of computing and which *Sp* we would like to obtain.

Table 5. *Se* and *Sp* of different number of sub patterns

Number of sub patterns	*Se*	*Sp*
1	1.0	0.717
2	1.0	0.747
3	1.0	0.8915
4	1.0	0.9475
5	1.0	0.9805
6	1.0	0.9915
7	0.9875	0.998
10	0.975	1.0

2.5 A Combination of Cluster Analysis and GP

In order to improve the performance of the algorithm, we used cluster analysis. The idea of this technique is that the more complex the training set the higher the number of rules necessary to cover it [16]. Therefore, K-means cluster method [17] is used to partition the training set into multiple sub training sets. Each sub training set is then independently computed by genetic programming. The scheme of this combination is in Figure 1.

The result of training set is calculated as:

$$\text{rs\_ts} = \bigcup_{i=1}^{k} \text{rs\_grp}_i \qquad \textbf{(10)}$$

with

- rs_grp_i : the result set of *i*-th group.
- rs_ts : the result set of training set.

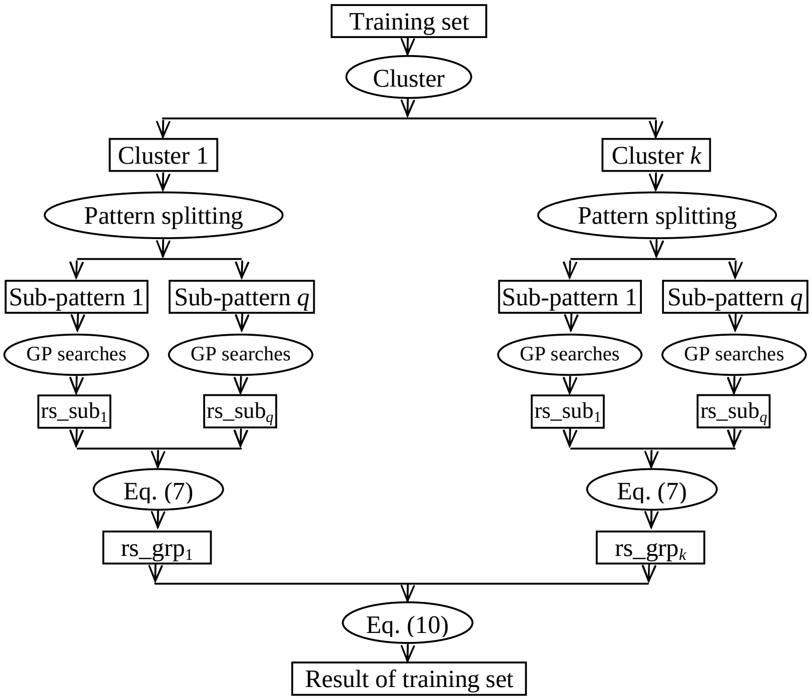

Fig. 1. A combination of cluster analysis and genetic programming

3 Experiments

Four real databases (one transcriptomics, one proteomics, and two breast cancers) were used to test the algorithm and compare with six other algorithms.

3.1 Transcriptomics Database of Response of Fibroblasts to Serum

This database [18] has 517 genes monitored in 19 different time points using DNA chips to represent the response of fibroblasts to serum. Therefore, the database has 517 patterns whose dimension is 19. The database was firstly analyzed using clustering methods. We selected the initial training set for the algorithm by random selection from the clusters identified by the previous cluster analysis. For each selected cluster an initial training set containing a set of patterns randomly selected from each cluster, was created. Then, the algorithm was applied for identification with other members of the selected cluster. The results are listed in Table 6.

Table 6. *Sp* and *Se* for transcriptomics database of 7 algorithms (null value means algorithm does not work). Binary SVM [1] is support vector machine for binary classification. Single SVM [1] is support vector machine for single class. LogitBoost [19, 20]. LR is logistic regression [20]. LDA is linear discriminant analysis [20]. LS is linear regression and least square [20]

Clus-ter	Cluster + GP		Binary SVM		Single SVM		LogitBoost		LR		LDA		LS	
	Se	*Sp*	*Se*	*Sp*	*Se*	*Sp*	*Se*	*Sp*	*Se*	*Sp*	*Se*	*Sp*	*Se*	*Sp*
1	1	0.9399	0.9	0.8302	1	0.8406	0.6	0.6708	0.5	0.7288	0.65	0.7495	0.65	0.7495
2	1	0.9515	0.9545	0.8861	1	0.8502	1	0.7194	1	0.7511	1	0.7405	1	0.7405
3	1	0.986	1	0.7594	0.8571	0.9583	-	-	0.7143	0.5249	1	0.6978	0.1429	0.5249
4	1	0.9919	1	0.7825	1	0.878	1	0.439	0.8	0.376	1	0.4106	1	0.4106

3.2 Caulobacter Proteomics Database

The database [21, 22] contains 145 patterns whose dimension size is 5. The database was firstly analyzed using clustering method. The average pattern of each cluster was then calculated. These average patterns were used in the initial training set for the algorithm to find other patterns of clusters. The results are summarized in Table 7.

Table 7. *Sp* and *Se* for proteomics database of 7 algorithms

Clus-ter	Cluster + GP		Binary SVM		Single SVM		LogitBoost		LR		LDA		LS	
	Se	*Sp*	*Se*	*Sp*	*Se*	*Sp*	*Se*	*Sp*	*Se*	*Sp*	*Se*	*Sp*	*Se*	*Sp*
1	1	0.9669	1	0.719	0.5417	0.9587	1	0.7438	1	0.7025	1	0.6529	1	0.6364
2	1	0.9826	1	0.8609	0.8	0.8087	1	0.9304	1	0.887	1	0.8783	1	0.8435
3	1	0.9722	0.973	0.9167	0.9459	0.8704	1	0.7407	1	0.75	1	0.713	1	0.713
4	1	0.9926	1	0.637	0.8	0.8889	1	0.7481	1	0.7481	0.9	0.7407	0.7	0.7556
5	1	1	1	0.9478	0.7667	0.8435	1	0.6348	1	0.6348	1	0.6261	1	0.6261
6	1	1	0.963	1	0.8889	0.989	1	1	1	1	1	0.967	1	0.967
7	1	0.9714	0.975	0.9238	0.8	0.8571	0.975	0.8476	0.975	0.8381	1	0.9238	1	0.9238

3.3 Two Breast Cancer Databases

The Wisconsin Breast Cancer Database [23] has 699 patterns with nine dimension of each pattern. There are two classes in this database, namely benign and malignant. The number of patterns in benign class and malignant class are 458 and 241, respectively.

The second database called Wisconsin Diagnostic Breast Cancer [24] contains 569 instances each of which belongs to benign class or malignant class (357 benign, 212 malignant). Each instance is described by 30 real-valued attributes. Attributes are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

For both of these breast cancer databases, the rate of the training set and test set is 50%-50%. Also patterns in the training set and test set were randomly selected. The results are shown in Table 8 and 9.

Table 8. *Sp* and *Se* Wisconsin breast cancer database

Class	Cluster + GP		Binary SVM		Single SVM		LogitBoost		LR		LDA		LS	
	Se	Sp	Se	Sp	Se	Sp	Se	Sp	Se	Sp	Se	Sp	Se	Sp
Benign	0.9955	0.7908	0.9595	0.7531	1	0.3473	1	0.3808	0.991	0.3891	0.973	0.795	1	0.3598
Malignant	1	0.9189	0.9832	0.9392	0.9832	0.9077	1	0.8581	1	0.8581	0.9748	0.8694	0.9916	0.8018

Table 9. *Sp* and *Se* Wisconsin diagnostic breast cancer database

Class	Cluster + GP		Binary SVM		Single SVM		LogitBoost		LR		LDA		LS	
	Se	Sp	Se	Sp	Se	Sp	Se	Sp	Se	Sp	Se	Sp	Se	Sp
Benign	0.9607	0.816	0.9831	0.717	0.5112	0.0142	1	0.5943	1	0.5189	1	0.5613	1	0.4387
Malignant	0.9057	0.7647	0.8774	0.9664	0.9528	0.7115	0.9434	0.6751	0.9434	0.7031	0.9811	0.5574	1	0.0896

4 Discussions

Based on the results of the transcriptomics database (Table 6), we see that the Algorithm finds all similar patterns in the four clusters (*Se* = 1) with very high *Sp*. The six other algorithms only find all of the similar patterns in some clusters. The lowest *Sp* of the Algorithm falls into cluster 1, this case has the lowest value of mean of correlation (0.63 ≈ 50% noise level).

Using the Algorithm, the 7 clusters of the proteomic database were compared with the results of pattern extraction. The clusters were chosen to cover different sizes of the target group and different within group correlation ranging from 0.63 (≈ 50% noise level) to 0.88 (≈ 30% noise level). Results are summarized in Table 7 and show that the desired selectivity was always satisfied, and all profiles in the cluster were correctly classified. In comparisons with the six popular pattern classification methods, the Algorithm is the best one.

In our observations, if two training sets have the same noise level but different dimension (cluster 7 of Table 7 and cluster 1 of Table 6; cluster 1 of Table 7 and cluster 4 of Table 6) then traditional methods (e.g LogitBoost, LR, LDA, and LS) will give better result in lower dimension cases. Therefore, traditional methods are suitable for low dimensions. This is the reason why feature selection methods are used to reduce dimensions when these methods are applied to high dimensional patterns.

In the Wisconsin Breast Cancer Database, one of the classes has a mean of correlation of about 0.30 (>> 50% noise level) and the minimum of correlation of two patterns of about -0.80. That means two different patterns (measured by simple correlation) are in the same class.

On the contrary, in the Wisconsin Diagnostic Breast Cancer, the minimum value of correlation between two patterns in the database is 0.91 (about 10% noise level). In other words, two similar patterns (measured by simple correlation) belong to different classes.

The results (Table 8 & 9) show that the Algorithm can identify almost all patterns in the same class with rather high precision and is always better than the six other algorithms. The Algorithm is really outstanding in such difficult databases.

5 Conclusions

Single class is a new trend in supervised pattern classification. It overcomes the disadvantages of the negative set in binary classification. In this work, we present an algorithm solving single class by using genetic programming. Based on the results of the four databases compared against six well-known algorithms, we see that the Algorithm can find most similar patterns with rather high precision. This study not only proves the Algorithm is better than the six other algorithms but also proves that genetic programming is a very powerful method for the symbolic regression problems.

We used three techniques to improve the performance of the Algorithm. The first one uses kernel principal component analysis to project patterns onto feature space. This simplifies the problem (data not shown). The second technique uses cluster method to partition the training set into groups. Each group will have a private classifier. The last technique uses parallel evolutionary computation (data not shown). Among these three techniques, the second technique provides the best result but at the expense of greater computation time.

Acknowledgment

We would like to thank Josef Panek and Ivan Janda for their helpful advice. This work was supported by grants of GACR No. 310/03/0293, 310/04/0804 and EC Integrated Project ActinoGEN, LSHM-CT-2004-005224.

References

1. Scholkopf B., Smola J. A.: Learning with kernels. MIT Press (2002)
2. Valentini G.: Gene expression data analysis of human lymphoma using support vector machines and output coding ensembles. Artificial Intelligent in Medicine 26: 281-304 (2002)
3. Teredesai, A. M. and Govindaraju V.: Issues in Evolving GP based Classifiers for a Pattern Recognition Task. Proceedings of the 2004 IEEE Congress on Evolutionary Computation 1: 509- 515 (2004)
4. Zhang, M. and Smart W.: Multiclass Object Classification Using Genetic Programming. 6th European Workshop on Evolutionary Computation in Image Analysis and Signal Processing, EVOIASP2004: 367-376 (2004)
5. Bojarczuk C. C., Lopes S. H., Freitas A. A.: Data mining with constrained-syntax genetic programming: applications to medical data sets, Proceedings Intelligent Data Analysis in Medicine and Pharmacology (IDAMAP-2001)
6. Kishore, J. K., Patnaik L. M., et al.: Application of genetic programming for multicategory pattern classification. IEEE Transaction on Evolutionary Computation 4(3): 242-258 (2000)
7. Loveard, T. and Ciesielski V.: Representing Classification Problems in Genetic Programming. Proceedings of the Congress on Evolutionary Computation: 1070-1077 (2001)
8. Folino, G., Pizzuti C., et al.: A cellular genetic programming approach to classification. Proceedings of the Genetic and Evolutionary Computation Conference 2: 1015-1020 (1999)

9. Innes, A., V. Ciesielski, et al.: Reducing False Alarms using Genetic Programming in Object Detection. In Proceedings of the 2004 International Conference on Artificial Intelligence (IC-AI'04): 569-574 (2004)
10. Song, A. and V. Ciesielski: Texture Analysis by Genetic Programming. Proceedings of the 2004 IEEE Congress on Evolutionary Computation: 2092-2099 (2004)
11. Ciesielski, V., A. Innes, et al.: Genetic programming for landmark detection in cephalometric radiology images. International Journal of Knowledge-Based Intelligent Engineering Systems 7(3): 164-171 (2003)
12. Ross, B. J., A. G. Gualtieri, et al.: Hyperspectral Image Analysis Using Genetic Programming. GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference: 1196-1203 (2002)
13. Agnelli, D., A. Bollini, et al.: Image classification: an evolutionary approach. Pattern Recognition Letters 23: 303-309 (2002)
14. Eggermont, J., A. E. Eiben, et al.: A comparison of genetic programming variants for data classification. Advances in Intelligent Data Analysis, Third International Symposium, IDA-99: 281-290 (1999)
15. Koza, J. R.: Genetic programming: on the programming of computers by means of natural selection. London, MIT Press (1992)
16. Freitas, A. A.: Data mining and knowledge discovery with evolutionary algorithms. Berlin, Springer Verlag (2002)
17. Jain A.K., Murty M.N., Flynn P.J.: Data clustering: a review. ACM Computing Surveys, 31(3): 264-323 (1999)
18. Iyer VR, Eisen MB, Ross DT, Schuler G, Moore T, Lee JC, Trent JM, Staudt LM, Hudson J, Jr., Boguski MS et al: The transcriptional program in the response of human fibroblasts to serum. Science, 283(5398):83-87 (1999)
19. Dettling M. and Buhlmann P.: Boosting for tumor classification with gene expression data. Bioinformatics 19: 1061-1069 (2003)
20. Hastie T., Tibshirani R., Firedman J.: The elements of statistical learning – data mining, inference, and prediction. Springer (2003)
21. Vohradsky J, Janda I, Grunenfelder B, et al.: Proteome of Caulobacter crescentus cell cycle publicly accessible on SWICZ server. Proteomics, 3(10):1874-1882 (2003)
22. Grunenfelder B., Rummel G., Vohradsky J., et al.: Proteomic analysis of the bacterial cell cycle. Proc Natl Acad Sci USA, 98(8):4681-4686 (2001)
23. Mangasarian O. L., Wolberg W. H.: Cancer diagnosis via linear programming. SIAM News 23(5): 1-18 (1990)
24. Street W.N., Wolberg W.H., Mangasarian O.L.: Nuclear feature extraction for breast tumor diagnosis. IS&T/SPIE 1993 International Symposium on Electronic Imaging: Science and Technology 1905: 861-870 (1993)