Chapter 23
Handwritten Digit Recognition Using Adaptive Classifier Construction Techniques

Tuan Trung Nguyen

Institute of Mathematics, Warsaw University, ul. Banacha 2, 02-097 Warsaw, Poland
nttrung@mimuw.edu.pl

Summary. Optical character recognition (OCR) is a classic example of a decision making problem where class identities of image objects are to be determined. This concerns essentially finding a decision function that returns the correct classification of input objects. This chapter proposes a method of constructing such functions by using an adaptive learning framework based on a multilevel classifier synthesis schema. The schema's structure and the way classifiers on a higher level are synthesized from those on lower levels are subject to an adaptive iterative process that allows learning from input training data. Detailed algorithms and classifiers based on similarity and dissimilarity measures are presented. Also, results of computer experiments using the techniques described on a large handwritten digit database are included as an illustration of the application of the proposed methods.

1 Introduction

Optical character recognition (OCR) is an important field in pattern recognition (PR), where a class assignment function for input character images is to be determined. Historically, pattern recognition algorithms can be grouped within two major approaches: statistical (or decision-theoretical), which assumes an underlying and quantifiable statistical basis for generating a set of characteristic measurements from the input data that can be used to assign objects to one of n classes, and syntactic (or structural), which favors the interrelationships or interconnections of features that yield important structural descriptions of the objects concerned. Both approaches are widely used in pattern recognition in general, but in the particular field of optical character recognition, the structural approach, especially methods based on trees and attributed graphs, is gaining popularity [7].

Typically, a structural OCR system attempts to develop a descriptive language that can be used to reflect the structural characteristics of the input image objects. Once such a language has been established, it is used to describe the characteristic features of the target recognition classes so that new images can be assigned to one of them when checked against those features. Most existing systems employ some kind of hierarchical descriptions of complex patterns built from *primitives*, elemental blocks that can be extracted directly from input data. However, the vast majority of such

systems assume a priori knowledge or a model of *primitives* and/or description hierarchy, leaving little room for adaptability to the input environment and, therefore, hindering the system's efficiency.

Based on the assumption that the construction of a recognition system itself needs to reflect the underlying nature of the input data, we propose a new framework in which the extraction of *primitives*, the development of the descriptive language, and the hierarchy of description patterns are all subject to an iterative adaptive process that learns from the input data. The framework is essentially based on the granular computing model, in which representational primitives equipped with similar measures play the part of information granules, whereas the pattern hierarchy implements the idea of the granular infrastructure comprising interdependencies among information blocks. (For a more comprehensive description of granular computing, see [6].) This allows for great flexibility of the system in response to the input data and as a consequence, a gradual improvement of the system's suitability to the underlying object domain. Moreover, it is noteworthy that the whole process is conducted automatically.

Later, we show that the same framework can also be used effectively to generate class dissimilarity functions that can be combined with similarity measures in the final recognition phase of the system; this makes our approach distinct from the majority of existing systems, which usually employ only class similarity when classifying new, unseen images.

Finally, we present the results of experiments on the large NIST 3 handwritten digit database, which confirm the effectiveness of the proposed methods.

2 Structural OCR Basics

Both statistical and structural approaches proved equally effective in PR in general, but the graphical nature of the input data in OCR intuitively favors employing structural methods. A major structural approach is the *relational graph* method, where the image objects from the training data set $U = \{u_1, u_2, \ldots, u_n\}$ are first converted to graphs $\{g_1, g_2, \ldots, g_n\}$, where specific image features are encoded in *nodes* and relations between them are represented by *edges*. Then, for each target class $\{CL_1, CL_2, \ldots, CL_c\}$, a set (library) of protopypical graphs $\{G_1, G_2, \ldots, G_c\}$ is developed, most often by means of some similarity measures.

These prototypes, also called *skeleton* graphs, are considered to contain characteristic traits for each target class and, in a way, represent images of that class. Now, given a new image object u_N, a representation graph g_N is extracted and *compared* with the skeleton graphs from each set G_i. The final class assignment can be denoted as

$$CL(u_N) = \Gamma[D_1(u_N, G_1), D_2(u_N, G_2), \ldots, D_c(u_N, G_c)],$$

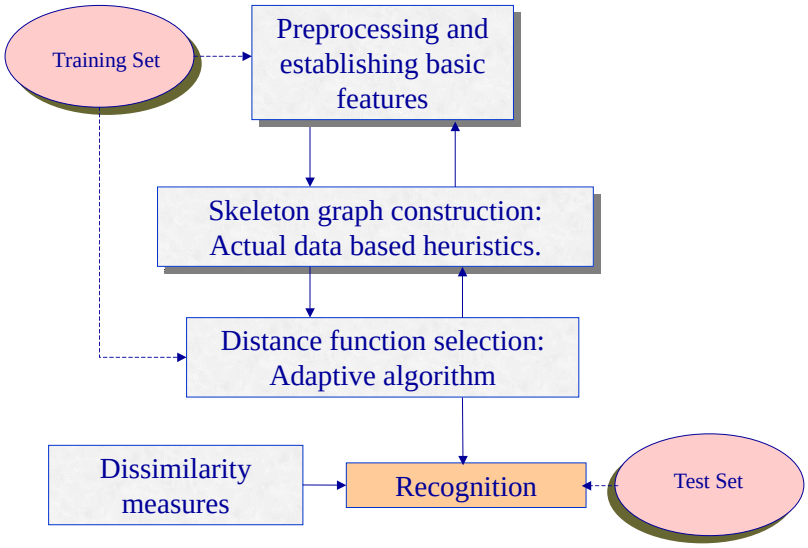

Fig. 1. Adaptive recognition system overview

where D_i are class similarity measures or distance functions and Γ reflects the chosen classification strategy [7].

It is obvious that the successful recognition depends on the choice of

- The graph model for the image data.
- Similarity measures used to build skeleton graphs.
- The distance functions and classification strategy in the final recognition phase.

Most existing OCR systems assume a priori domain knowledge about the input data and as a consequence, employ some or all these components as established in advance, with little or no feedback from the actual input training data. In this chapter, we shall show that all three components can be dynamically constructed by an adaptive process based extensively on the actual input image domain.

3 Relational Graph Model for Handwritten Digits

The development of OCR in general and handwritten digit recognition in particular over the years yielded many highly effective description models for analyzing digit images. For the research in this chapter we have chosen the enhanced loci coding scheme because of its simplicity, sophistication, low computational cost and its suitability to our higher level system components. The enhanced loci algorithm, though simple, has proved very successful in digit recognition [4].

3.1 Feature Extraction: The Enhanced Loci Coding Scheme

The enhanced loci coding scheme consists essentially of three major steps:

- *Feature extraction*: During this phase, each white pixel in the image is characterized by the black pixel regions that surround it in each of the four scanning directions: north, east, south and west. There are 16 possible code values for white pixels. Black pixels are labeled either as part of a horizontal, vertical, slanting edge or as completely enclosed by other black pixels. As a result, we have a description of basic local shapes of the digit.
- *Feature unification*: In this phase, the digit image is globally scanned to correct the pixels that had been wrongly labeled as result of "noise" in the picture.
- *Feature concentration*: In this phase, codes are combined with codes from surrounding regions to reflect both local and nonlocal region topologies. Black pixel codes are combined with the nearest white region in each scanning direction. White pixel codes are combined with those white regions outside of the nearest black strokes in the scanning direction.

Once the loci coding is done, the digit image is segmented into regions consisting of pixels with the same code value. These regions then serve as *primitives* to build the graph representation of the image. Suppose that an image I has been segmented into

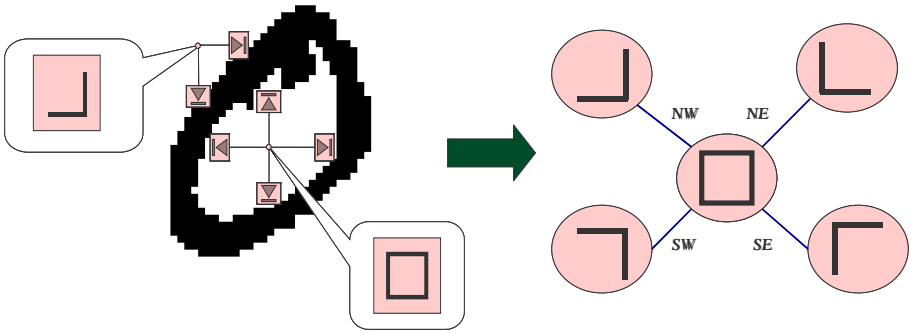

Fig. 2. Graph model based on loci encoding.

regions $R_1, R_2, \ldots, R_k$, each characterized by coordinates of the pixels contained and their shared loci code. The graph representation of the image is an attributed labeled graph denoted as

$$G_I = \{N, P, E\},$$

where

- $N = \{n_1, n_2, \ldots, n_k\}$ is a set of *nodes* representing $R_1, R_2, \ldots, R_k$.

- P is a set of *properties* of the nodes, containing, among other things, the loci code, the number of pixels, and the gravity center of the corresponding regions. We assume that the set of possible properties will remain the same for all digits from the input domain.
- E is a set of directed labeled edges between pairs of nodes, describing the relative direction between corresponding regions. These directions (labels) may be N, E, S, W, NE, SE, NW and SW. We also assume that the set of possible labels remains the same for all digits from the input domain.

It is easy to observe that such a graph G_I will contain local information about black strokes in nodes and nonlocal features about the shapes of the digit as edges (see also [5]).

3.2 Base Skeleton Graph Construction

Definition 1. A *base segment* $S_b = \{N_b, P, E_b\}$ is any two-node segment of any digit representation graph, i.e., $|N_b| = 2$ and $|E_b| = 1$. We shall say that a base segment S_b *matches* a graph G_I, or $match(S_b, G_I)$, if S_b is isomorphic to a subgraph of G_I.

Similar base segments can then be merged:

Definition 2. Given a set of base segments with common node and edge sets $S_1 = \{N, P_1, E\}$, $S_2 = \{N, P_2, E\}$, $\ldots$, $S_k = \{N, P_k, E\}$, a *base skeleton segment* is defined as

$$S_{bs} = \{N, P_{bs}, E\},$$

where P_{bs} is a combined set of properties

$$P_{bs} = \{(L_1, f_1), (L_2, f_2), \ldots, (L_k, f_k)\},$$

with loci code $L_i \in P_i$, and *the frequency of occurrence* of L_i,

$$f_i = \frac{|\{G_m : match(S_i, G_m)\}|}{|\{G_m : \exists 1 \leq j \leq k : match(S_j, G_m)\}|}.$$

We shall say that a base skeleton segment,

$$S_{bs} = \{N, \{(L_1, f_1), (L_2, f_2), \ldots, (L_k, f_k)\}, E\}$$

, *matches* a graph G_I, or $match(S_{bs}, G_I)$, if $\exists_{1 \leq j \leq k} : match(\{N, L_k, E\}, G_I)$.

Having constructed base skeleton segments, we can build *base skeleton graphs*.

Definition 3. A *base skeleton graph (BSG)* is any set of base skeleton segments.

Though this concept allows constructing of a broad range of skeleton graphs, a particular type, linked directly to the loci coding scheme, can be distinguished as highly useful:

Definition 4. A *loci base skeleton graph (LBSG)* is a BSG in which there exists one node, called a *center node*, which is connected to every other node. Furthermore, an *extended loci base skeleton graph (ELBSG)* is a graph in which several LBSGs are connected.

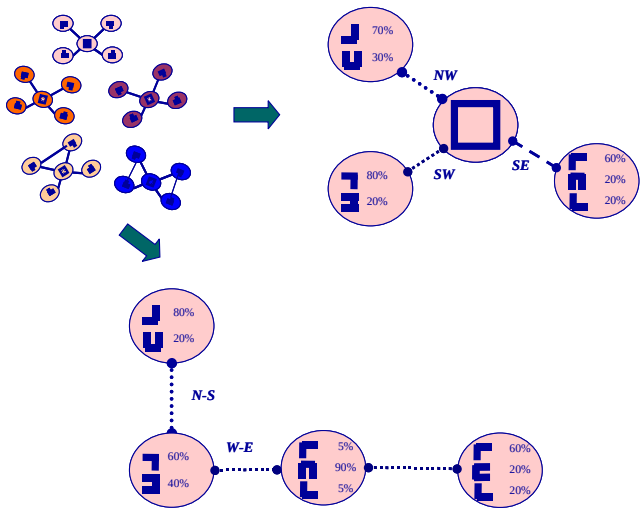

Fig. 3. Skeleton graph consolidation

LBSGs correspond to concentrated loci codes, where a region (represented by the center node) is characterized not only by its own topology, but also by topologies of all regions immediately surrounding it. ELBSGs, in turn, describe features that are more global, rather than direct neighborhoods.

One can look at BSGs as “soft” or “blurred” prototypical graphs that may be used to represent a class of digits. By fine-tuning various parameters of the model, e.g., the set of properties or connection labels, or by imposing various cutout thresholds, we can dynamically control the primitives extraction process.

3.3 Graph Similarity Measures

To build prototype libraries for each digit class from the input domain, we need to be able to measure the similarity between representation and skeleton graphs.

Rather than concentrate on a specific, predefined measure, we aim to develop an entire framework for the flexible construction of families of similarity measures.

Given a BSG S and a digit representation graph G, similarity $\tau(S,G)$ is established as follows:

Definition 5. Suppose that for each node $n \in S$, $P(n) = \{(L_1, f_1), (L_2, f_2), \ldots, (L_{k_n}, f_{k_n})\}$ is the set of $(code, frequency)$ pairs at n. Then

if $match(S,G)$ **then for each** $n \in S$,

$$\tau_n(S,G) = \sum_{i=1}^{k_n} f_i \tau_C[L_i, L^n(G)],$$

where $L^n(G)$ is the loci code found at the node matching n in G, and τ_C is a code-defined similarity function that returns the similarity between two given loci codes:

$$\tau(S,G) = \sum_{n \in S} w_n \tau_n(S,G),$$

where w_n are connection-defined weight coefficients,

else

$$\tau(S,G) = 0.$$

This definition provides a tolerant matching scheme between representation graphs and skeleton graphs, which allows us to concentrate on the specific aspects of the graph description concepts at each given stage of the learning process. By fine-tuning code-defined and connection-defined weight coefficients, we can achieve significant flexibility in information granules construction.

Now, let $S = \{S_1, S_2, \ldots, S_k\}$ be an ELBSG where S_i are loci BSGs. The similarity $\tau(S,G)$ can be defined as

$$\tau(S,G) = \mathbb{F}[\tau_1(S_1,G), \tau_2(S_2,G), \ldots, \tau_k(S_k,G)],$$

where τ_i are single LBSG-defined similarity measures and $\mathbb{F}$ is a synthesis operator. The choice of $\mathbb{F}$ is greatly influenced by the actual structure of the granules' hierarchy, which in our case is the interconnections between local patterns in skeleton graphs. Both explicit a priori and data-driven learned synthesis types of operators can be employed.

Definition 6. A distance function between two graphs G_1, G_2 *with regard to* a skeleton graph S is defined as

$$d_S(G_1,G_2) = |\tau_S(G_1) - \tau_S(G_1)|.$$

Suppose that for a digit class k, a set (library) of prototypical graphs PG_k has been established. Then, we can consider different distance functions *with regard to* that class using various synthesis operators, e.g.,

$$d_k(G_1, G_2) = \max_{S \in PG_k} d_S(G_1, G_2),$$

$$d_k(G_1, G_2) = \sum_{S \in PG_k} w_S d_S(G_1, G_2),$$

where w_S are weight coefficients.

4 Adaptive Construction of Distance Functions

Based on the relational graph, similarity measure, and distance function models defined in previous sections, we can construct an iterative process that searches for an optimal classification model as follows:

Algorithm

Step 1 Initial skeleton graph set for each digit class k:

1. **select** a start node: $S = n_0$.
2. Among its neighboring node–candidates, **pick** a node n_i that
   - has class k as dominant among the supporting population, **or**
   - best splits class k from the others, namely, has the maximal number of pairs (G_1, G_2) of which only one belongs to class k and only one matching $S \cup \{n_i\}$.
3. $S := S \cup \{n_i\}$.
4. **repeat** 2–3 until *quality criteria* are met.
5. **output** S.

Also, for each digit class, Loci BSGs are constructed based on

- Frequency and histogram analysis.
- Adjustment of connection-defined weights using a **greedy clustering scheme** with the recognition rate as quality criteria.
- Manual selection of a number of core initial EBSGs using some domain knowledge.

The purpose of this step is to establish an initial BSG set that bears as much characteristic information as possible specific to each class. Manual selection allows for flexible integration of an expert's domain knowledge in the feature construction process.

Step 2 Distance function evaluation

With the established sets of skeleton graphs for each digit class, develop graph similarity measures and distance functions, as described in Sect. 3.3.

Based on developed distance functions, perform *k*-NN clustering on the input training data collection to obtain class separation.
Evaluate the recognition rate based on developed clusters.

Step 3 Adjustment of parameters

Using a **greedy strategy** with regard to the recognition rate, make adjustments to:

- Single code similarity function.
- Code-based and connection-based similarity weights.
- BSG-based distance function weight.

Reconstruct the skeleton graph set as needed.

Repeat steps 2–3 until quality criteria are met.

These two steps allow us to establish the hierarchy of information granules through distance functions. The clustering-based learning on both infrastructure building and granule construction levels ensures that the final structure reflects the nature of the input domain.

End algorithm

It can be observed that this is an adaptive iterative process with a two-layered *k*-NN clustering scheme, aimed at optimizing three components crucial to the recognition process:

- Primitives extraction process, implemented by loci coding scheme, code-defined and connection-defined similarity measures.
- Similarity measures model, represented by base skeleton graphs.
- Class distance functions (discriminants), synthesized from extended skeleton graphs.

5 Dissimilarity Measures

So far, similarity measures are used to construct libraries of prototypes so that future input data may be checked against them. Thus, the recognition process relies on how a new object resembles those that had been learned. However, sometimes

it could really help if we knew whether an object *u does not* belong to a class *c*. At times in the recognition process, it might be crucial to know that, for example, a digit is neither a '8' nor a '9,' though we still do not know for sure if it is a '1' or a '7.' Furthermore, there may be situations when, based on similarity analysis, we know that a digit may be a '0' or a '1,' and at the same time, by using dissimilarity measures, we can state that it cannot be a '0.' Hence, dissimilarity measures are very useful for constructing classification discriminants.

In our relational graph model, dissimilarity measures can be defined as follows:

Antigraphs

Definition 7. Let $G = \{V, p, E, l\}$ where V is a set of nodes, E is a set of connections, p is an node attribute assignment function, and l and is a connection label assignment function. Further, let P be a set of all possible node properties and L be a set of all possible connection labels. Then, graphs

$$G_V = \{V, p_V, E, l\} \text{ where } p_V(n) = P \setminus \{p(n)\} \, \forall n \in V,$$

and

$$G_E = \{V, p, E, l_E\} \text{ where } l_E(e) = L \setminus \{l(e)\} \, \forall e \in E,$$

are called *antigraphs* of G.

Then, for instance, the dissimilarity to a digit representation graph may be defined as similarity to its antigraph.

Based on the same framework described in Sect. 4, we can construct antigraph skeleton sets for each digit target class or several target classes and use them as discriminants to improve the classification quality. In this way, similarity and dissimilarity measures can be effectively combined, using a common framework, to augment the final recognition rate.

6 Results of Experiments

Extensive testing has been conducted to verify the developed methods. We chose the U.S. National Institute of Standards and Technology (NIST) Handwritten Segmented Character Special Database 3, a major reference base within the handwritten character recognition community, as the main data collection. The base contains 223,125 128×128 normalized binary images with isolated handwritten digits from 2100 different people. (For details, see [3])

As a reference experiment's data collection, we chose a random portion of the whole base that contained

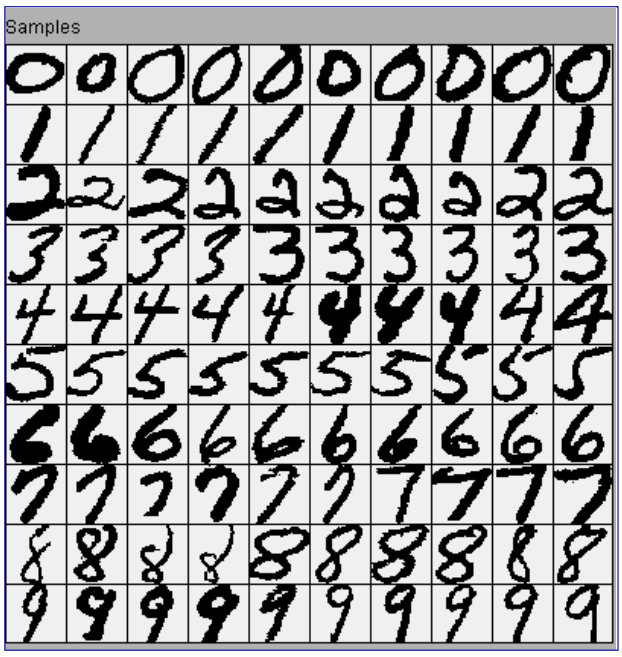

Fig. 4. Samples from the NIST SD 3 collection

- 44,000 digits as a training table, of which 4000 have been separated for tests during the learning process.
- 4000 digits for the final test table

Most skeleton graphs are simple in size (5 to 19 nodes). Complexity varies from class to class; the maximum connection depth does not exceed 5. The LBSG-type graphs are the main engine for the main recognition phase, whereas general BSG-type and antigraphs were particularly useful in fine-tuning postprocessing.

The system has been implemented as a Win32 application under Windows NT 4.0 using Microsoft Visual C++ 6.0. The learning phase took, on average, 4 hours 15 minutes on a Intel Pentium III 733MHz computer with 128MB RAM. The final recognition speed was 0.045 s per digit.

It is interesting to observe that there were no rejections, which showed that the chosen graph model and loci coding were highly appropriate for the digits concerned.

The results obtained qualify our system close to the leading recognition packages tested at NIST, of which the average zero-rejection error rates were 1.70% (see [3]).

Table 1. Adaptive distance function recognition results.

Class	No. of skeleton graphs	No. of digits	Misclassified	Reject
0	8	439	1.6%	0%
1	7	328	1.5%	0%
2	12	417	6.7%	0%
3	11	375	6.1%	0%
4	14	421	5.5%	0%
5	9	389	9.8%	0%
6	11	397	5.3%	0%
7	8	366	3.3%	0%
8	13	432	4.9%	0%
9	9	436	6.9%	0%
	Total	4000	**5.2** %	0 %

Table 2. Recognition results with dissimilarity improvement.

Class	No. of skeleton graphs	No. of digits	Misclassified	Reject
0	8	439	0.91%	0%
1	7	328	1.52%	0%
2	12	417	2.16%	0%
3	11	375	4.53%	0%
4	14	421	3.80%	0%
5	9	389	2.57%	0%
6	11	397	2.02%	0%
7	8	366	1.37%	0%
8	13	432	2.78%	0%
9	9	436	1.83%	0%
	Total	4000	**2.35**%	0%

7 Conclusion

We presented a uniform framework for automatically constructing classifiers based on an adaptive scheme. A model for synthesizing similarity measures from input data primitives through higher level features has been proposed. The method allows flexible learning from the input training data during the construction phase and proved effective. The same framework can be used to develop dissimilarity measures that are highly useful in improving the classification quality. Experiments conducted on a large handwritten digit database showed that the method can be applied to practical problems with encouraging results. The framework can easily

be adapted to recognizing other structured objects such as handprinted characters, fingerprints, iris images, and human faces.

References

1. M. R. Anderberg. *Cluster Analysis for Applications*. Academic Press, New York, 1973.
2. J. Bazan, H. S. Nguyen, T. T. Nguyen, J. Stepaniuk, A. Skowron. Application of modal logics and rough sets for classifying objects. In M. De Glas, Z. Pawlak, editors, *Proceedings of the Second World Conference on the Fundamentals of Artificial Intelligence (WOCFAI'95)*, 15–26, Ankor, Paris, 1995.
3. J. Geist, R. A. Wilkinson, S. Janet, P. J. Grother, B. Hammond, N. W. Larsen, R. M. Klear, C. J. C. Burges, R. Creecy, J. J. Hull, T. P. Vogl, C. L. Wilson. The second census optical character recognition systems conference. NIST Technical Report NISTIR 5452, 1–261, 1994.
4. K. Komori, T. Kawatani, K. Ishii, Y. Iida. A feature concentrated method for character recognition. In B. Gilchrist, editor, *IFIP Proceedings*, North Holland, Amsterdam, 29–34, 1977.
5. H. S. Nguyen, T. T. Nguyen. An approach to the handwriting digit recognition problem based on modal logic. Master's Thesis, Institute of Mathematics, Warsaw University, 1993.
6. L. Polkowski, A. Skowron. Towards adaptive calculus of granules. In L.A. Zadeh, J. Kacprzyk, editors, *Computing with Words in Information/Intelligent Systems*, 201–227, Physica, Heidelberg, 1999.
7. R. J. Schalkoff. *Pattern Recognition: Statistical, Structural and Neural Approaches*. Wiley, New York, 1992.
8. Y. Kodratoff, R. Michalski. *Machine Learning: An Artificial Intelligence Approach*, Vol. 3. Morgan Kaufmann, San Francisco, 1990.