LEARNING TO CLASSIFY IMAGES BY MEANS OF ITERATED FUNCTION SYSTEMS

Matteo BALDONI, Cristina BAROGLIO, Davide CAVAGNINO and Lavinia EGIDI
Dipartimento di Informatica — Università degli Studi di Torino
Corso Svizzera, 185 — I-10149 Torino (Italy)
E-mail: {baldoni,baroglio,davide}@di.unito.it, lavinia@al.unipmn.it

The full automation of object visual recognition is a hard and computationally expensive task, mainly for two reasons: on one hand, it is difficult to extract the necessary distinctive information from the raw data, while, on the other, the obtained representations must have characteristics that make them apt to training adaptive classifiers to perform the recognition task of interest. In this paper we present a new method for representing 2-D images, based on the extraction of a set of *fractal features*, which exploits the approximation of an image with an *Iterated Function System*, a technique that is already at the basis of many successful image compression tools. In particular, we show that such features have a high discriminatory power, can be easily extracted in an automatic way from the raw data and can effectively be used to train adaptive classifiers to discriminate between different kinds of objects.

Keywords: Iterated Function Systems, automatic image recognition, automatic feature extraction.

1 Introduction

An automatic classifier capable of recognizing imperfectly drawn bidimensional images can be used for a variety of applications, ranging from easier-to-use CAD systems to indexing systems and retrieval engines for large repositories of images and documents [1]. In order to have such a capability, the system should be able to abstract from inessential details in the image by focussing its attention on *distinctive features*. The recognition process can, therefore, be seen as a sequence of two separate phases. A first one of *feature extraction*, in which distinctive traits are somehow singled out, followed by a classification process, in which the image is recognized as belonging to a certain family. We consider here classes of objects that share the same shape structure.

The main problem when dealing with image recognition is that normally the data corresponding to images are too large to be handled. For instance, a simple 6×4 inch image with 256 gray levels and a resolution of 100 dpi (dots per inch) requires more than 230 Kbytes. But consider as an example the case of recognizing a hand-written digit: all that is required is the ability to tell which out of ten different classes the image belongs to. A partial (and hence more concise) representation of the image should be sufficient.

In this paper, we show that Iterated Function Systems (IFSs) can be used to produce encodings that help solving similar decision problems. IFSs are already exploited in the Image Analysis field but only for image compression [2,3,4]. Briefly, algorithms for fractal compression are based on the idea that if two portions of the image look the same, they do not need to be repeated twice in the coding and it is sufficient to record their similarity. This observation is carried to the extremes by

dissecting and deforming portions of the image to obtain more and more cases of likeness. The similarities themselves and the operations carried out on the picture are recorded by means of a collection of contractive transformations of the two dimensional space, i.e. by means of an IFS (see Section 2 for the definition). In particular, since the target is compression, IFSs are built so as to deflate the image at hand as much as possible, keeping all the information necessary for later precise reconstruction.

However, recognizing an image does not mean being able to reproduce it exactly, therefore all we need is an approximate representation of it. We propose to extract characteristic features using a methodology based on IFS theory, but also exploiting the fact that we need much less data than in the case of compression, since we only must capture *distinctive* self-similarities.

The novelty of our work is that IFSs were never used before for image classification purposes. We chose fractal methods because of the nice properties of IFSs that allow the extracted features to be used by *automatic learning systems*. This is very important because it offers an alternative to the current approaches to image classification, that are based on the beforehand creation of prototype models of the relevant classes. (See Section 2 for a more extensive discussion of these issues.)

The algorithm that we propose here derives from the experience of a prototype system [5], written in Constraint Logic Programming, that could only deal with linear shapes. Our ideas can be traced back to a preliminary work [6] in which experiments on the use of IFSs for image recognition were carried out, although the features were extracted manually and only from artificial images with fractal structure. On the contrary, we test the system on more natural images, namely digits and trees. We chose the first set of images because we could profit from a large database of handwritten digits and compare our results with experiments described in the literature; besides, digits offer a challenge to a feature extraction system based on fractal techniques, because the latter seem to be better suited for working on shapes that have a planar extension than on purely linear figures. On the other hand we believe that the method we present may be paricularly interesting to deal with classification of very irregular shapes, on which traditional methods do not perform well. For this reason we tested it also on trees, as an example of class of images with an instrinsic complex structure. The tests are carried out using neural nets (see Section 4).

In this paper we propose our new application of techniques based on the mathematics of fractals, mainly with the hope of raising enough interest in the community to *have some feed-back from experts* in the area. We feel that our research might evolve in interesting directions and yield useful applications, but we are limited by our little experience with fractal encoding methods.

In the next section we briefly review IFS theoretical backgrounds [7,8] and fix the notation. We also highlight the properties of IFSs that make them suitable for the use in image classification and explain the novelty of our approach. In Section 3 we present our algorithm for feature extraction. In Section 4 we give a brief introduction to neural nets and summarize the results of our experiments, discussing their meaning and implications. We conclude with a review of the related issues that we think should be addressed first, mostly focussing on those problems that are directly connected to the fractal based techniques we use.

2 IFS and image recognition

The basic idea underlying *fractals* is that a fractal is, at any scale, *self-similar*. In the plane, fractal images can be generated by iterating a geometrical transformation that maps the given image to smaller copies of itself. The fractal is the limit figure obtained when the number of iterations goes to infinity. In principle, this mapping can be any transformation, in practice, *affine* transformations, which preserve the object's shape, are preferred. The only necessary condition imposed on the transformations is that they be *contractive*, which intuitively means that they must move points closer to each other.

An *affine transformation* in a bidimensional space has the general form:

$$M(\begin{bmatrix} x \\ y \end{bmatrix}) = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}. \quad (1)$$

Notice that the parameters e and f encode the *translation* component of the transformation.

An *Iterated Function System* (IFS) is a set $\{M_1, \ldots, M_r\}$ of contractive transformations (*contractions*) on compact sets that define a set-valued mapping S, such that, for any compact set E, $S(E) = \bigcup_{k=1}^{r} M_k(E)$. S is contractive in the Hausdorff metric. The property of IFSs that is more relevant to this work is that for each IFS defined on some subset E of $\mathbf{R}^n$ there *always exists* a *unique* non-empty compact set $F \subseteq E$ such that $F = \bigcup_{k=1}^{r} M_k(F)$ (*Fixed Point Theorem* [8]). F is known as the *invariant set* of the IFS.

Moreover, as a consequence of the *Collage Theorem* [7,8], given any compact subset E of $\mathbf{R}^n$ and an arbitrary precision of approximation, it is always possible to find an IFS, whose invariant set approximates E as finely as desired.

In practice images are represented in a discrete way as *matrices of pixels*. Thus, in principle each image can be represented by the transformations of the IFS. Since each transformation is described by 6 parameters (see Eq. (1)), the *number of features* necessary to represent a two-tone image is equal to $n = r \times 6$, where r is the cardinality of the IFS. This method allows to encode images with no matter how complex shape, e.g. a natural scene, a tree, or a cloudy sky [9]. For this reason the Collage Theorem and more often its variant for Partitioned IFS (PIFS) are at the basis of many image compression systems. However, in both cases designing an algorithm for encoding arbitrary shapes so that they can be reconstructed with arbitrary accuracy is still an open research problem.

One of the main advantages of IFS-based encodings is that, unlike other methods, they produce very concise representations of extremely complex images, the number of features not depending on the image size [10,11]. A few transformations are usually enough for capturing a meaningful approximation of the shape.

Another characteristic of IFSs is that they are *robust*. In fact, even though the covering produced (the invariant set) is not perfect, we know that the reproduction of the image is *stable*, i.e. small changes in the transformations produce small changes in the invariant set, which means that varying the coefficients in a continuous way, the shape of the invariant set also changes in a continuous manner [8]: close parameter values will correspond to similar images. Then, it is reasonable to

suppose that the IFSs that approximate the images of a set of objects of a same class will be closer than those approximating sets of objects belonging to different classes. Despite these interesting properties, to our knowledge the Collage Theorem has *never* been applied before to extracting the characteristic information of an image for classification purposes. In this paper we try to fill this gap proposing and using the set of parameters of the IFSs as *descriptive* features. We call these features *fractal features*.

The conciseness of the representation and the reduced number of features (together with robustness and stability) allow the effective use of Machine Learning systems for implementing the classifier [12]. In the Image Understanding community there is currently a strong interest in the use of *adaptive classifiers* for building image recognition systems. An adaptive classifier can, in fact, be *automatically* trained to perform a classification task (i.e. to distinguish objects of different kinds) just by showing it a set of examples and counter-examples of each class. This is much simpler than the classical approach [13,14] consisting of, first, designing (by hand) a set of prototypes for each class and, next, writing a classification program which exploits the prototypes and a distance measure to perform the task. Besides being easier to produce, adaptive classifiers are more robust and flexible than the traditional ones.

Another advantage of fractal features is that they are simpler to extract than most of the features which are currently used to characterize images, in the sense that they can be extracted directly from the pixel matrix, whereas in the classical approach each image undergoes a sequence of processing steps, aimed at representing the information contained in it by means of more and more abstract descriptors [13,14].

Experiments show that fractal features have a *very high* discriminant power and that IFSs with few transformations are enough to distinguish among different classes even when many constraints are imposed on transformation selection.

3 XFF: automatic extraction of fractal features

In principle, the already existing PIFS-based fractal compression systems may be used to generate fractal features; this is why in our first attempt we used one of the better-known among them, *enc* by Fisher [4]. Soon, however, the need for a different approach emerged. The reason is that *enc* proved to be too *specialized* to the particular task of compression: intuitively speaking, the information it extracts does not correspond to the structure of the depicted object but is related to turning the texture of a piece of image into a close approximation of the texture of another piece of the same image. The consequence is that, as we will show in Section 4, the overall performance it allows to achieve in a task of image recognition is quite poor and we believe that the same would happen using any other encoder which, like *enc*, is taylored to image compression.

Indeed, the application that we tackle is pretty different from compression. In particular, we know that the object at hand belongs to a class out of a given set and all we need to capture in an image is the information sufficient to allow classification. For this reason rough approximations will do for our purposes. These considerations have encouraged us to search for *ad hoc* fractal feature extraction systems. In order to show that such systems can be developed we went back to the origin of IFS-based

encodings, i.e. the original formulation of the Collage Theorem, which our algorithm XFF (eXtraction of Fractal Features) operationalizes.

XFF extracts discriminant fractal encodings of 2-D isolated objects in a bitmap image and is implemented in C. To this aim it looks for an IFS of *predefined cardinality* r (IFS_{appr} in the following), whose invariant set *approximates* the object depicted in the image at hand, disregarding the background. The parameters of the transformations in IFS_{appr} are used as descriptive features of the processed image. XFF is very simple. It basically consists of two main processing phases: during the first a predefined number of contractions of the image are produced by scaling and properly rotating and/or flipping it (we call these contractions *proto-transformations* because they are parametric w.r.t. the translations); then, the proto-transformations are used to produce the transformations that constitute IFS_{appr} through an iterative process that stops after r transformations have been added to IFS_{appr}. The first phase is implemented by following standard image processing techniques, while the second consists of a search process in which IFS_{appr} is built by iteratively selecting and adding a transformation M_k to an initially empty set. At each iteration, a set of *candidates* is built by instantiating all the proto-transformations on a regular grid of points, whose step is decided by the user. The candidates are evaluated according to a *scoring criterion* and the one with the least score is selected.

The *scoring criterion* that we used is a variant of the *Hamming distance*. Briefly, it is designed so that the transformation with the least score together with those that are already in IFS_{appr} cover the maximum number of foreground pixels. More in detail, this procedure compares two matrices: one corresponding to the current candidate (say $transform$) and one corresponding to a part of the original image ($image$), which has the same size as $transform$ and whose top-left vertex is the point (x, y) of the grid at which the proto-transformation has been instantiated. Calling $w1$ the width of $transform$ and $h1$ its height, its score is the sum, over all the pixels that belong to its foreground, of the values

$$s_{ij} = ((image[i'][j'] - transform[i][j])^2 + 1) * punish_{ij}$$

where $i \in [1, w1]$, $j \in [1, h1]$, $i' = x + i$ and $j' = y + j$. The "+1" in the formula had to be added to take into account the information given by $punish_{ij}$ in the case of a perfect covering, i.e. when $\forall i, j(image[i'][j'] = transform[i][j])$. $punish_{ij}$ forces the covering operation to prefer pixels that belong to the foreground of the original image and, as a second requirement, that have not been covered yet. The value of $punish_{ij}$ varies pixel by pixel and changes over time to take into account the growing part of image being covered by the transformations added to IFS_{appr}.

The *output* of XFF is an IFS $\{M_1, ..., M_r\}$. Since the scaling factors and the flipping axes are fixed a priori and only a finite number of user-specified rotations is allowed, the set of possible transformations becomes finite modulo translations. As a consequence, only three parameters are needed to identify each transformation M_i: the first one encodes the applied proto-transformation and the other two encode the translations. Of course the drawback is that the transformations allowed might not lead to an IFS that captures in an optimal way distinctive features of any image.

Clearly these restrictions also influence the complexity of the algorithm inasmuch as they limit the search space. Consider an $n \times n$ pixel image. If f is the

total number of different transformations allowed (modulo translations) and s is the step used to build the candidates at each iteration, the number of such candidates is at most $\frac{n^2}{s^2}f$. The amount of operations needed by the scoring procedure for each transform is proportional (by a constant factor c_1) to the number of pixels in the transform, which is at most $(\ell n)^2$, where ℓ is the maximum scaling factor allowed. At each iteration additional $c_2(\ell n)^2$ operations (for some constant c_2) are needed to update the values of the *punish* variables. If r is the predefined size of the sought for IFS, the total number of operations is bounded above by $r(c_1\frac{\ell^2 n^4 f}{s^2} + c_2\ell^2 n^2)$.

4 Validation

A supervised Machine Learning system (or *adaptive classifier*) [15] can be thought of as a black box which, given a set of classes and a set of instances of each class (the *learning set*), exploits the similarities between objects belonging to the same class and the dissimilarities between objects belonging to different classes to induce a body of classification knowledge, i.e. to acquire the skill of correctly associating classes to instances. Of course, the most desired property of an adaptive classifier is the *generalization* capability, i.e. it should be able to build a classification knowledge which allows it to correctly classify unseen instances of the given classes. Since supervised learning is an induction process whose results depend on the learning instances, the validation of the classification knowledge obtained can be done only *empirically* using a set of instances that do not belong to the learning set (the *test set*). In order to verify the independence of the recognition performance obtained from the particular learning set that was used, the same classifier is trained many times (say q times) using different learning and test sets and the average performances are considered. This validation technique is known as (*q-fold*) *cross-validation*.

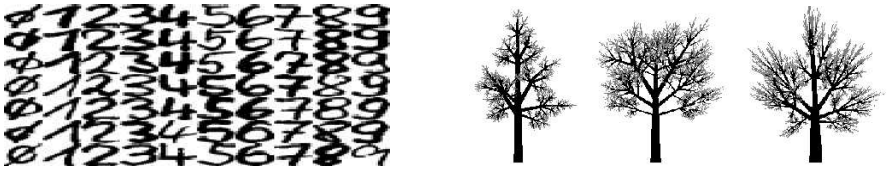
Neural networks [12] are a wide family of adaptive classifiers, which are generally used to tackle classification problems having a numerical nature. In this kind of problems each instance is described by an n-tuple of numbers, interpreted as a point in an n-dimensional space (the *input space*); in our experiments, for example, each instance is represented by the n-tuple of fractal features extracted by XFF. The learning rule that we exploited (i.e. the rule applied by our neural net to acquire the classification knowledge from the examples it was shown) is called *Scaled-conjugate Gradient* rule [16]. This rule has the advantage of learning faster and with less oscillations than many other neural network learning rules.

The often implicit assumption of most adaptive classifiers is that the input vectors encoding instances of a same class are somehow close to each other in the input space. Therefore, in the learning phase one aims at training the classifier to approximate a mapping that divides the input space in areas corresponding to the different classes (note that in general there may be different "clusters" in the input space corresponding to the same class, the important thing being the separation between classes). From this observation the key role played by the *features* emerges. Indeed there are many kinds of features which could be used to represent the same instances; the best ones are those which make the discrimination process easier. The fractal features show characteristics (e.g. the robustness, see Section 2) that make them a good candidate for image recognition tasks. In the following the results

Table 1: Digit recognition rates obtained using the features extracted by XFF and by *enc*.

	0	1	2	3	4	5	6	7	8	9	avg.
XFF	85.0	96.7	91.9	84.2	81.4	86.7	90.2	92.4	62.4	79.0	85.0
enc	79.6	78.0	71.6	60.8	65.6	81.6	72.4	69.2	26.0	47.2	65.2

obtained on two very different test-beds are reported to support this candidature.



(a) (b)

Figure 1: (a) instances of the digits; (b) instances of Elm, Oak and Lime-tree.

4.1 Digits

Briefly, this data set is a collection of hand-written instances of the ten digits (collected by the German Federal Post), acquired with a resolution of 16×16 pixels with 256 grey levels (Figure 1 (a)). In some experiments the instances were binarized, i.e., only those pixels whose value was higher than a certain threshold were considered as elements of the digit. The learning sets were made of 300 instances (30 per class) while the test sets were made of 700 instances (70 per class), all different than those shown during training. 3-fold cross-validation was used to check the independence of the results from the particular learning and test sets used. The percentages we report are the averages of those obtained in the various trials.

First of all, we tried to understand if IFSs capture characteristic information about (non-fractal) images; to this aim, we extracted the fractal features covering *by hand* each image by means of 4 self-affine transformations. In this case a 100 % recognition rate was always obtained.

Afterwards, we alternatively applied *enc* and XFF to make the encoding process fully automatic. The results are reported in Table 1. It must be noted first that the fractal features have a quite high discriminant power (for 7 classes out of 10 over 84 % of the test instances are correctly classified). Secondly, that this capability emerges only when *ad hoc* feature extraction systems are used. In fact, using the features produced by *enc* the average recognition performance is drastically reduced.

4.2 Trees

The classification problem consists of learning to distinguish three kinds of trees, namely *lime-tree*, *oak* and *elm* (Figure 1 (b)). The size of all images is 320×240 pixels. 150 instances were available, 50 per class and 6-fold cross-validation was

Table 2: Recognition rates obtained on the trees. The results of the first two columns were obtained using the setup, 100 hidden neurons, learning and test sets made of 75 examples. (a) perfect recognition was achieved in some of the runs; (b) averages obtained by cross-validation; (c) averages obtained using 3 transformations; (d) averages obtained using 5 transformations (200 hidden neurons had to be used in this case); (e) results obtained encoding the instances by means of *enc*.

class	(a)	(b)	(c)	(d)	(e)
elm	100	88.61	86.00	90.18	56.00
oak	100	91.38	95.34	83.64	76.00
lime-tree	100	75.38	67.34	61.45	60.00
average	100	85.12	82.89	78.42	64.00

applied. Depending on the experiment, each instance was encoded simply by using 3, 4 or 5 transformations, i.e. using alternatively 9, 12, or 15 features. Taking into account the extreme irregularity of the tree images, these encodings are very compact w.r.t. those that would be obtained by means of other common image encoding techniques, e.g. chain coding. About 450 experiments were carried out. The proto-transformations were obtained using only rotations that are multiple of $\pi/2$. The best results were obtained using 4 transformations with a side scaling factor equal to 0.35.

The outcome of the experiments are reported in Table 2. Columns (a) and (b) report the results obtained using the setup described above. The results of column (a) were reported just to show that in some cases a perfect recognition was actually obtained. Column (b), instead, reports the average results obtained by cross-validation. Columns (c) and (d) report the average results obtained when encoding the trees respectively by means of 3 and 5 transformations.

A first observation is that *12 numbers* resulted to be a sufficient encoding for a tree image of size 320×240, allowing an average 85.12% recognition rate to be achieved. We think that this result is a sufficient proof of the *power of discrimination* of IFS based features. The optimal performance obtained in some cases, then, strengthens this result. The second appealing aspect is that the IFSs were *extracted* in an *automatic* way.

5 Conclusions

Even though IFSs have been extensively studied and used for image compression, they were never taken into account as a means for characterizing visual representations of objects for recognition purposes. In the literature, indeed, there are very few examples of papers in which fractals have been used in image classification tasks. To the best of our knowledge, the only ones are those by Freisleben et al. [17], in which a neural network is trained to distinguish fractal images from non-fractal images, Erkmen et al. [18], in which fractals are used to build the belief functions of a Bayesian classifier, and Imiya et al. [19], in which a metric for planar self-similar forms is proposed. In the last paper mentioned [19], which is probably the closest to the topics we explored, a metric is defined for measuring the distance between IFSs: the smaller the distance the more similar the shapes represented. Note that

image classification is a natural application of such metrics. However, in order to use any metric in recognition tasks, some models are to be given for comparison. Furthermore, it is necessary to find a technique for producing IFS encodings for the learn/test instances.

The greatest novelty of our work is that we used the IFS representations themselves in order to acquire a classification knowledge. Thus, by attacking the problems of image feature extraction and classification knowledge acquisition, our method is to some extent *complementary* to the work by Imiya et al. [19]. We have shown that IFS encodings can be used to discriminate between images of different kinds of objects and therefore also in classification tasks. The only problem is to find good techniques for making fractal feature extraction automatic. Indeed when a perfect encoding is produced the recognition rates achieved are optimal [6,5]. In order to show that such algorithms *can* be developed, we implemented XFF, a simple fractal feature extraction system, that can be applied to any isolated shape. Experiments showed that recognition rates obtained using XFF are twenty percent better than those obtained using fractal compression systems (see Table 2). Furthermore, the encodings are generally much more compact.

Besides, the use of IFS-based fractal features reduces both the dimension of the input space and the range of each input feature, because of the *constraints* that are imposed on the transformations to be considered. Many learning algorithms *cannot* deal with applications having hundreds of dimensions while those that can take *too long* before converging [20]. Therefore the use of IFSs widens the range of learning systems that can be applied to image classification tasks, decreasing the time required by the training phase.

Of course, in order to optimize the classification process it is important to fully understand which kind of restrictions it is better to impose on the transformations and to determine an *optimal criterion* for guiding transformation selection. We regard these as very relevant open problems and hope for some hints from experts in the mathematics of fractal, that might help us towards their solution.

We feel that a promising direction of future research might in a sense mimick the evolution of image compression algorithms [21], by shifting the focus from IFSs to some more appropriate fractal technique.

Acknowledgements

We thank prof. Lorenza Saitta, prof. Charles Taylor and Giuseppe Lo Bello for the helpful discussions. We are grateful to the referees for their useful comments.

References

1. D. Forsyth, J. Malik, and R. Wilensky. Searching for digital pictures. *Scientific American*, pages 72–77, June 1997.
2. M. Barnsley and A. Sloan. A Better Way to Compress Images. *BYTE Magazine*, 13(1):215–223, 1988.
3. M. Barnsley and L. P. Hurd. *Fractal Image Compression*. AK Peters, Ltd., Wellesley, Massachusetts, 1993.

4. Y. Fisher. *Fractal Compression: Theory and Application to Digital Images.* Springer Verlag, New York, 1994.
5. M. Baldoni, C. Baroglio, D. Cavagnino, and G. Lo Bello. Extraction of Discriminant Features from Image Fractal Encoding. In M. Lenzerini, editor, *Proc. of AI*IA 97: Advances in Artificial Intelligence*, volume 1321 of *LNAI*, pages 127–138. Springer-Verlag, 1997.
6. G. Le Chiara and L. Saitta. Using Fractals to Learn Image Descriptions by means of Artificial Neural Networks. In *Proceedings of IEEE International Conference on Neural Networks*, pages 2952–2955, Orlando, USA, 1994.
7. M. Barnsley. *Fractals Everywhere.* Academic Press, San Diego, 1988.
8. K. Falconer. *Fractal Geometry, Mathematical Foundations and Applications.* John Wiley & Sons Ltd., Chichester, UK, 1990.
9. A. P. Pentland. Fractal-Based Description of Natural Scenes. *IEEE Trans. Pattern Analysis and Machine Intelligence*, (PAMI-6):661–674, 1984.
10. G. Lu. Fractal image compression. *Signal processing:Image communication*, (5):327–343, 1993.
11. T. Bedford, F.M. Dekking, M. Breeuwer, M.S. Keane, and D. van Schooneveld. Fractal coding of monochrome images. *Signal processing:Image communication*, (6):405–419, 1994.
12. D. E. Rumelhart and J. L. McClelland. *Parallel Distributed Processing : Explorations in the Microstructure of Coginition, Parts I & II.* MIT Press, Cambridge, Massachusetts, 1986.
13. L. O'Gorman. Basic techniques and symbol-level recognition – an overview. In K. Tombre R. Kasturi, editor, *Graphics Recognition, Methods and Applications*, volume 1072 of *LNCS*, pages 1–12. Springer-Verlag, 1995.
14. R. C. Gonzales and R. E. Woods. *Digital Image Processing.* Addison - Wesley Publishing Company, 1992.
15. J. G. Carbonell, R. S. Michalski, and T. M. Mitchell, editors. *Machine Learning: An Artificial Intelligence Approach*, volume I. Tioga Publishing Company, 1983.
16. M. F. Moller. A scaled conjungate gradient algorithm for fast supervised learning. *Neural Networks*, 6:525:533, 1993.
17. B. Freisleben, J. H. Greve, and J. Löber. Recognition of Fractal Images using a Neural Network. In J. Mira, J. Cabestany, and A. Prieto, editors, *New Trends in Neural Computation, IWANN '93*, volume 686 of *LNCS*, pages 632–637. Springer-Verlag, 1993.
18. A. M. Erkmen and H. E. Stephanou. Information Fractals for Evidential Pattern Classification. *IEEE Trans. on SMC*, 20(5):1103–1114, 1990.
19. A. Imiya, Y. Fujiwara, and T. Kawashima. A Metric of Planar Self-Similar Forms. In P. Perner, P. Wang, and A. Rosenfeld, editors, *Advances in Structural and Syntactical Pattern Recognition, SSPR'96*, volume 1121 of *LNCS*, pages 100–109. Springer-Verlag, 1996.
20. D. Michie, D. J. Spiegelhalter, and C. C. Taylor. *Machine Learning, Neural and Statistical Classification.* Ellis Horwood Series in AI. Prentice Hall, 1994.
21. N. Wadströmer. A Solution to the Inverse Problem of Iterated Function Systems. In *Proc. of the Picture Coding Symposium*, Melbourne, 1996.