Scalable Evolutionary Design of CA Pattern Classifier

Joy Deep Nath, Pabitra Mitra, and Niloy Ganguly

Department of Computer Science and Engineering,
Indian Institute of Technology, Kharagpur, India
{jnath,pabitra,niloy,}@iitkgp.ac.in

Abstract. The paper reports a scalable evolutionary design for pattern recognition using Multiple Attractor Cellular Automata ($MACA$). $MACA$ helps to impart non-linearity in the classifier using Hamming distance based attractors. Isomorphism in $MACA$ was exploited to make the method scalable to large classification problems involving non-linear boundaries. Extensive experimentation was performed on datasets with different topologies to establish the efficacy of the proposed method as compared to existing popular approaches like support vector machines. The classifier was applied to satellite image analysis problem. Experiments on different types of data sets were performed to discover the classifier's feature selection capabilities.

Key words: Cellular Automata, Pattern Classification, Genetic Algorithm, Feature Selection

1 Introduction

The classification problem may be again viewed as partitioning the feature space and mapping the corresponding regions to different classes. Machine learning methods provide techniques to determine the boundaries of the partitions in the features space and hence help in learning the classes. The most general techniques are based on Euclidean metric nearest neighbor rule or linear discrimination which are essentially linear classifiers and are not suitable for all problems. A data dependent non-linear metric is more versatile as it helps in capturing and imparting non-linearity to classifiers inherently.

Multiple attractor cellular automata ($MACA$), a special class of cellular automata has the inherent property of generating a non-linear partitioning of the feature space based on Hamming distance metrics [2].

In our present work, we show the results of a comprehensive set of experiments on $MACA$ based classifier. We demonstrate that identification of isomorphic $MACA$ leads to low computational complexity as compared to the scheme in [2, 3]. Lowering the complexity has helped us in making the classifier scalable. We have studied the performance of our classification scheme on a number of real life application problems. A number of interesting phenomena were observed which provides insight for designing more efficient classifiers. The basic design of the $MACA$ is stated in the next section.

2 Multiple Attractor Cellular Automata ($MACA$) Classifier

A $MACA$ belongs to the class of Linear CA [1, 2]. In this section, we state the basic features of the linear CA. The classifier and it's realization through evolutionary algorithms is stated next.

2.1 Characterization of a Linear CA

An n-cell one dimensional cellular automata (CA) is characterized by a linear operator $[T]_{n\times n}$ matrix. T is referred to as the *characteristic* matrix of the cellular automata. The i^{th} row of T corresponds to the neighborhood relation of the i^{th} cell, where

$$T_{ij} = \begin{cases} 1, \text{ if next state of the } i^{th} \text{ cell depends on the present state of } j^{th} \text{ cell} \\ 0, \text{ otherwise.} \end{cases}$$

Since the CA is restricted to three neighborhood dependency, $T[i, j]$ can have non-zero values for $j = (i-1)$, i, $(i+1)$. Thus, T becomes a tri-diagonal matrix. The next state s_{t+1} of a CA is given by

$$s_{t+1} = T \cdot s_t \quad that\ is, \quad s_{t+p} = T^p \cdot s_t$$

The complete characterization of the CA is reported in [1]. If all the states in the state transition diagram of a CA lie in some cycles, it is a group CA, otherwise it is a non-group CA. In this paper we concentrate on $MACA$ which is a special class of non-group CA.

2.2 Multiple Attractor Cellular Automata

The state transition graph of an $MACA$ consists of a number of *cyclic* and *non-cyclic* states. The set of non-cyclic states of an $MACA$ forms inverted trees rooted at the cyclic states. The *cycles* are referred to as *attractors*. *Fig.1* depicts the state transition diagram of a 5-cell $MACA$ with two attractors {00000,00100} having self loop.

With reference to the state transition diagram of a CA, the *depth* d of the CA is the number of edges between a non-reachable state and the attractor. The depth d of the 5-cell $MACA$ of *Fig.1* is 2.

The detailed characterization of $MACA$ is available in [1]. A few fundamental results for an n-cell $MACA$ having k number of attractors is next outlined.

Result I: The characteristic polynomial of the $MACA$ is $x^{n-m}(1+x)^m$, where $m = log_2(k)$.

Result II: The characteristic polynomial noted above can be also written in

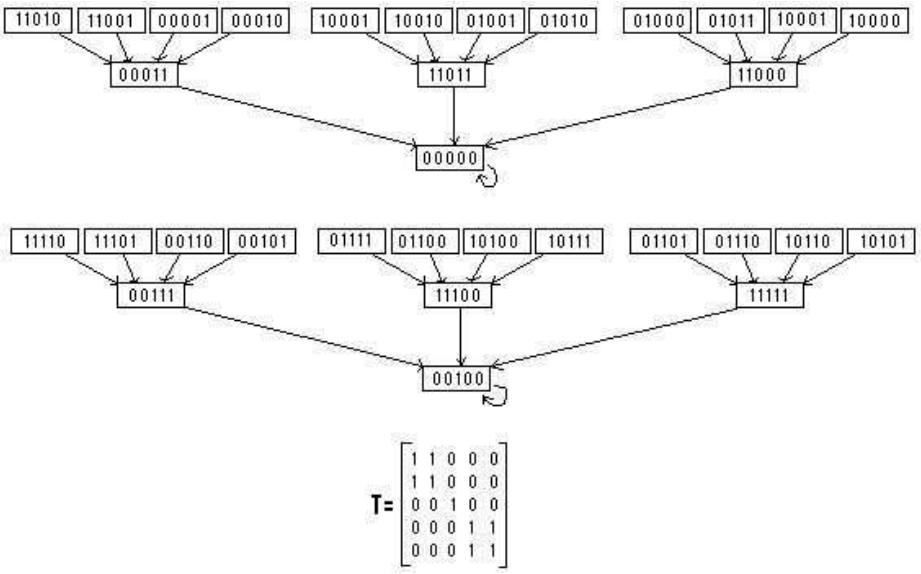


Fig. 1. State transition diagram of a 5-cell $MACA$ with Characteristic matrix T and Rule Vector $< 102, 60, 204, 102, 60 >$

elementary divisor form as
$(1+x)(1+x)\cdot\cdot m\ times\ \ x^{d_1}x^{d_2}\cdot\cdot x^{d_p}$ where $d_1 \geq d_2 \cdot\cdot \geq d_p$ and $d_1 + d_2 \cdot\cdot\cdot\cdot + d_p = n - m$.

Result III: The minimal polynomial of an $MACA$ is $x^{d_1}(1+x)$, where depth $= d_1$.

Definition 1. *[1] An m-bit field of an n-bit pattern set is said to be pseudo-exhaustive if all possible* 2^m *patterns appear in the set.*

Theorem 1. *[1] In an n cell MACA with* $k = 2^m$ *attractors, there exists* m-bit *positions at which the attractors give pseudo-exhaustive* 2^m *patterns.*

Theorem 2. *[1] The* modulo-2 *sum of two states is the non-zero predecessor of* 0-state *(pattern with all 0's) if and only if the two states lie in the same MACA basin.*

Theorem 3. *Given MACA T, and a state q,* $T^d \cdot q$ *yields the attractor of the basin which the state belongs to.*

The example $MACA$ of *Fig.1* is used to illustrate the above results.

Example 1 *– It is a* 5*-cell MACA having* 2 *number of attractors and the depth of the MACA is 2.*

– Result I: *The characteristic polynomial is* $x^4 \cdot (1+x)$*. Therefore, m=1. This is consistent with the result in the* Fig.1 *where attractor(k) is 2.*
– Result II: *The characteristic polynomial in elementary divisor form is* $x^4 \cdot (1+x) \cdot x^2$*.*
– Result III: *The minimal polynomial is* $x^2 \cdot (1+x)$*.*
– Result of Theorem 1: *In* Fig.1*, only one significant bit positions constitute the* PEF*.*
– Result of Theorem 2: *We take an attractor 00100 and any two states 11111, 11101 of the attractor basin. The* modulo-2 *sum of these two patterns is 00010 which is a state in* $0-basin$*. By contrast, if we take two states 00001 and 10100 belonging to two different attractor basins 00000 and 00100 respectively, their* modulo-2 *sum is 10101 which is a state in a non-zero attractor (00100) basin.*
– Result of Theorem 3: *We take a state 10001. The depth (d) of the* $MACA$ *is 2. Hence, Multiplying* $T^2 \cdot$*[10001] we get 00000 which is the attractor of the basin in which the state 10001 lies.*

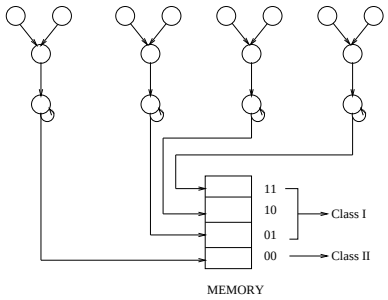


Fig. 2. $MACA$ based Classification Strategy

2.3 Classification Technique Using MACA

For an ideal classifier, to distinguish between two classes, we would need one bit. A k-attractor two class classifier needs $log_2(k)$ bits. The pseudo-exhaustive field (PEF) of an attractor provides the pointer to the class of states in the attractor basin. In order to identify the class of a state $\wp$, the $MACA$ is initialized with $\wp$ and operated for maximum of (depth) d number of cycles till it reaches an attractor. Next, the PEF bits can be extracted (as noted in [1]) to identify the class of $\wp$. In general, depth is defined as the number of time steps an $MACA$ needs to reach an attractor state when it is initialized with a non-reachable state as seed (*Theorem 3*).

***MACA* based Binary Classifier** The design of the $MACA$ based classifier for two pattern sets P_1 and P_2 should ensure that elements of one class (say P_1) are covered by a set of attractor basins that do not include any member from the class P_2. Any two patterns $a \in P_1$ and $b \in P_2$ should fall in different attractor basins. According to *Theorem 2*, the pattern derived out of *modulo-2* sum of a and b ($a \oplus b$) should lie in a non-zero attractor basin. Let X be a set formed from *modulo-2* sum of each member of P_1 with each member of P_2 that is, $X = \{x_l \mid x_l = (a_i \in P_1) \oplus (b_j \in P_2)\ \forall_{i,j}\}$. Therefore, all members of X should fall in non-zero basin. This implies that the following set of equations should be satisfied.

$$T^d \cdot X \neq 0 \tag{1}$$

where T is a valid $MACA$ to be employed for designing two class classifier. Hence, what we need is a scheme to get the correct $MACA$.

Hence, as shown in *Fig 2*, satisfying the above criteria, the $MACA$ is found / learned (in our case searched) using a GA evolution scheme proposed in [2]. Before we look into the merits and demerits of the proposed GA based evolution scheme, we introduce the pseudo-chromosome format and elaborate the steps of synthesis of the **T** matrix from the pseudo-chromosome. The former is used to represent the $MACA$ in the scheme in [2].

Pseudo-Chromosome Format It is a method of representing an $MACA$ with respect to the sequence in which its $x^{d_i}{}'s$ and $(1+x)'s$ are arranged. It is a string of n bits where

- d_i positions occupied by a x^{d_i} is represented by d_i followed by $(d_i - 1)$ zeros (for example, $x^3 = [300]$), and
- $(1 + x)$ is represented by -1.

The *pseudo-chromosome format* of the $MACA$ of Fig. 1 is illustrated in *Fig. 3*. **T** matrix being the generic way to represent an $MACA$ can be synthesized from the Pseudo-Chromosome as mentioned in the next section.

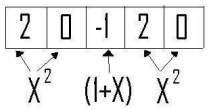
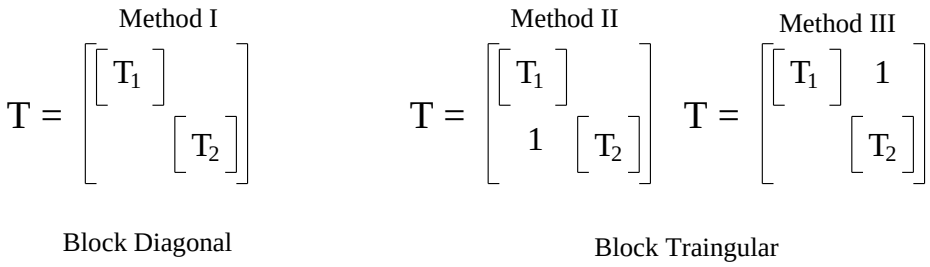


Fig. 3. MACA in Pseudo-Chromosome Format. Characteristic Poynomial $x^2(1+x)x^2$

Synthesis of T matrix from Pseudo-Chromosome From *Result II* of *Section 2.2*, the elementary divisor form of $MACA$ is $(1+x)(1+x)\cdot\cdot m\ times\ x^{d_1}x^{d_2}\cdot \cdot x^{d_p}$ & $d_1 + d_2 \cdots + d_p = (n - m)$, where the number of $(1 + x)$ determines the



[T_1]& [T_2] in Block Diagonal Form
Note: Method II (III) has '1' on lower (upper) diagonal position and so T_2 (T_1) has dependence on T_1 (T_2).

Fig. 4. Different methods to arrange two matrices

number of attractors. Each elementary divisor ($\phi_i(x)$) can be converted to a CA [1]. A tri-diagonal T matrix with characteristic polynomial $\phi_i(x)$ is accordingly synthesized. Let T_i matrices correspond to elementary divisors x^{d_i} (it will be a $d_i \times d_i$ matrix) and T_j matrices correspond to each elementary divisors $(x+1)$. If T_is and T_js are randomly arranged in block diagonal form, the characteristic polynomial of the resultant T is $x^{n-m} \cdot (1+x)^m$ and the minimal polynomial is $x^{d_p} \cdot (1+x)$ and it generates the $MACA$ [1].
We illustrate the synthesis of the **T** matrix using the $MACA$ example presented in *Fig. 1.*

Example 1. If we take Pseudo-Chromosome shown in *Fig. 3* and try to regenerate the **T** matrix of the MACA using the **Method I** as shown in *Fig. 4*, we obtain the **T** matrix and the basin distribution as shown in *Fig. 5.* However, if we took **Method III** followed by **Method II** as shown in *Fig. 4*, we obtain the **T** matrix and the basin distribution as shown in *Fig. 5.* Here, we notice a few interesting observations.

1. The attractors of the non-Zero basins obtained in both the cases are different.
2. However, the distribution of patterns in the non-Zero basins obtained in both the cases are same and they differ only in their position in the basin tree.
3. The PEF bit positions in both the cases remain the same.

The following observations have been made across various $MACA$'s and hence found to be true. This shows that the pattern classes shall remain same irrespective of the using Method I or a combination of Method I, II and III. The position of the PEF bit too remains the same and hence we can conclude that the usage of method II and III simply causes redundancy of information.

Having discussed about the Pseudo-Chromosome Format and it's conversion to **T** matrix, we discuss the GA based evolution scheme[2] next.

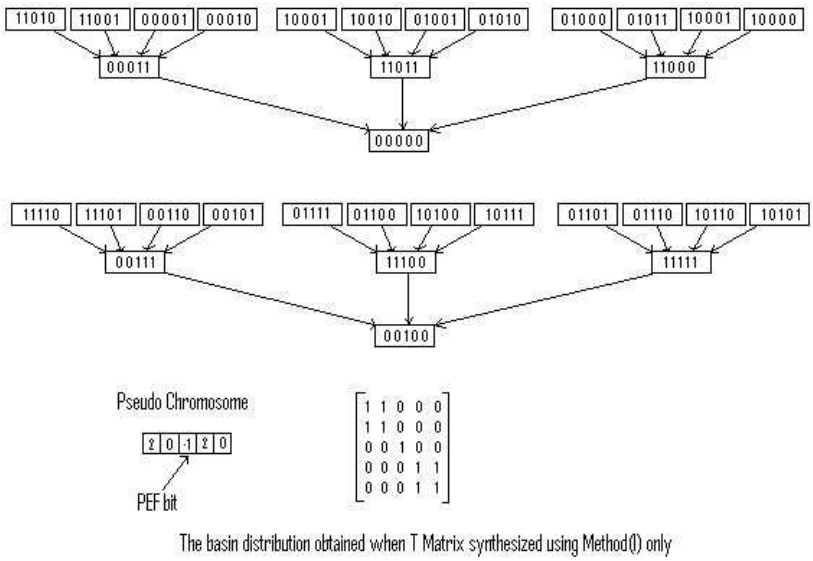


Fig. 5. An Example showing the effect of the solely using Method I during synthesis

2.4 *GA* based Evolution Scheme

The GA based evolution scheme [2], takes a pool of randomly generated $MACA$'s uniquely characterized by their **T** matrix and evolves the desired **T** matrix of the $MACA$. The evolution is guided by the fitness function[2] on the training samples. The former is based on the criteria mentioned in the *Section 2.3*. The training samples are the patterns or more specifically the states that the classification scheme needs to learn. The fitness function calculates the classification capacity of the $MACA$ corresponding to the **T** matrix.

We now perform a complexity analysis of this scheme. Assuming that there are k training samples on which the $MACA$ of 2^m attractors is to be learnt. The **T** matrix corresponding to the $MACA$ will be of the size $n \times n$ while the sample is of the size n. The fitness function tries to identify the attractors (*refer subsection 2.2*) to which the samples (patterns/states) belong. For this, each and every sample has to be multiplied by the **T** matrix d times to land in the attractor (*refer Theorem 3*) to ascertain it's attractor. The complexity of computing the attractor given a **T** matrix and a sample is $\mathrm{O}(n \cdot d)$. Doing so for all the training samples in $\mathrm{O}(n \cdot d \cdot k)$. Once the attractors of all the samples are ascertained, the samples can be grouped into their respective classes in $\mathrm{O}(2^m \cdot k)$ by comparison of attractors corresponding to each of samples against the 2^m attractors. If there are p $MACA$'s in the pool (population) then the total complexity of the GA based scheme is $p \cdot (O(2^m \cdot k) + O(n \cdot d \cdot k))$. Apart from this space complexity of $p \cdot (O(n^2) + O(2^m))$ for storing each of the $MACA$'s and their corresponding

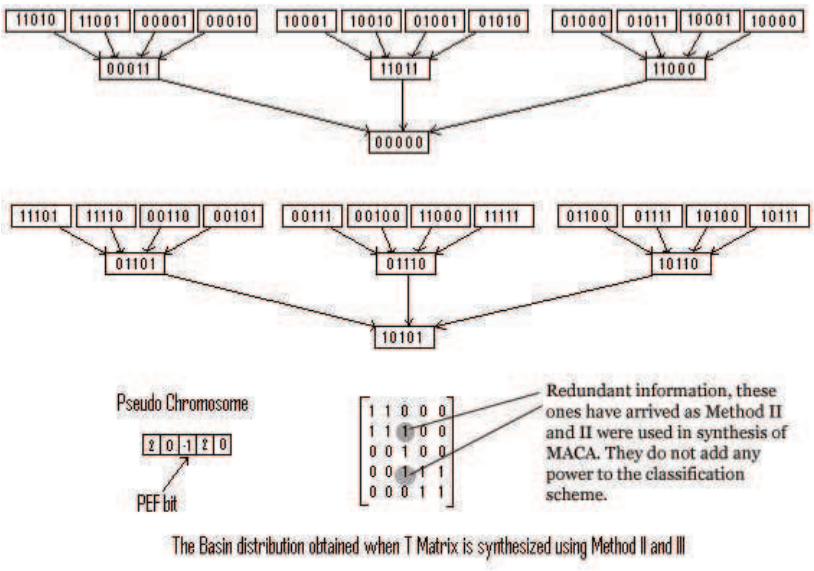


Fig. 6. An Example showing the effect of the using Method II and III during synthesis

attractors in additional to $O(n \cdot k)$ space required to store the samples. The space and time complexities of the scheme are given in *Table 2.4*.

Space Complexity	$p(O(n^2) + O(2^m)) + O(n \cdot k)$
Time Complexity	$p(O(2^m \cdot k) + O(n \cdot d \cdot k))$

Table 1. Space and Time Complexities of the GA based scheme

2.5 Reduction in Time Complexity : Our Approach

As discussed through example 1, all the **T** matrices corresponding to a pseudo-chromosomes are equivalent in terms of their basin configurations and pseudo-exhaustive fields. Since, the PEF bit(s) corresponding to the $MACA$ don't change for the patterns in a basin, we are done if we know values corresponding to the PEF bits in the pattern and the class membership of the basins (attractors). In this case, we do not require the **T** matrix. We can safely operate with the Pseudo-Chromosome. Considering the same parameters as mentioned in the *sub section 2.4*, the time complexity for determining the PEF bits of a pattern is $O(n)$, which is finding the positions of -1 in the Pseudo-Chromosome. Since, the attractors are now fixed as shown in example, the samples will be checked only on their PEF bits and we will hence determine the attractor basin to which a

sample belongs to in $O(m)$ and in $O(m \cdot k)$ for the whole training sample. The fitness of the whole population can now be found in $p \cdot O(m \cdot k) + O(n)$. This clearly is an improvement over the time complexity mentioned in *Table 2.4*.

Since we did not use Method II and III, which used to cause an interdependence between the blocks [2], we had the advantage of not storing the $MACA$'s **T** matrix at every crossover and mutation [2] as would have been required. Thus, we have a space complexity of $p \cdot (O(n) + O(2^m)) + O(m \cdot k)$ and have saved space in our current classification scheme. Summarizing the complexities we have the *Table 2.5*.

Space Complexity	$p(O(n) + O(2^m)) + O(n \cdot k)$
Time Complexity	$p(O(m \cdot k) + O(n))$

Table 2. Space and Time Complexities of Our GA based scheme

3 Experiments

With the modification of doing away with the synthesis as well as the use of **T** matrix in classification scheme altogether, we experiment comprehensively to see the performance of our new scalable $MACA$ Classifier. We perform experiments for checking the classification strengths using the functionality tests. Having reduced space and time complexities, we perform scalability tests. We also perform test on a real world data to validate the claim that our current classification scheme is ideal for developing a inferencing schemes in future.

4 Datasets Considered

To test for the functionality of our new *Scalable $MACA$ Classifier*, we have first artificially created four datasets which represent four different binary classification problems. The $MACA$'s ability to classify the four diverse datasets is tested. Each of the datasets is briefly explained one by one next.

1. **Linear Classification Problem**
   This classification problem has two classes which can be separated using a linear hyperplane. A two dimensional linear classification problem would look like *Fig 7.a*.
2. **Concave Classification Problem**
   This classification problem has two classes which cannot be separated using a linear hyperplane, but can be separated by a quadratic hyperplane. A two dimensional concave classification problem would look like *Fig 7.b*.

3. **Double Spiral Classification Problem**
   The two-spiral problem was first proposed by Alexis Wieland of MITRE Corporation and now forms one the important benchmarks at the Carnegie Mellon repository [6]. The task requires the pattern classifiers to learn a mapping that distinguishes between points on two intertwined spirals as shown in *Fig. 7.c*. The classifier is provided with two inputs, which represents the x and y coordinates on the 2-D plane and should output a one if the point falls on one spiral and a zero if it falls on the other spiral. This particular problem is difficult for most current algorithms because it requires the classifier to learn the highly nonlinear separation of the input space. The two-intertwined spirals problem has been used by some researchers as a benchmark for neural networks [5, 7].
4. **Annular Classification Problem**
   This classification problem has two classes one of which is enclosed in circle and the other lies on an annular ring which encompasses the first circle. A two dimensional annular classification problem would look like *Fig 7.d*.

The datasets were prepared artificially by placing points of *dark* and *lighter* shades on an image file of dimension 256×256 as shown in *Fig.* 7. The *darker points* represented one class and the *lighter points* the other. The two features extracted were the x and y co-ordinates of the points (*dark or light*). Hence, we had a two-dimensional binary classification problem.

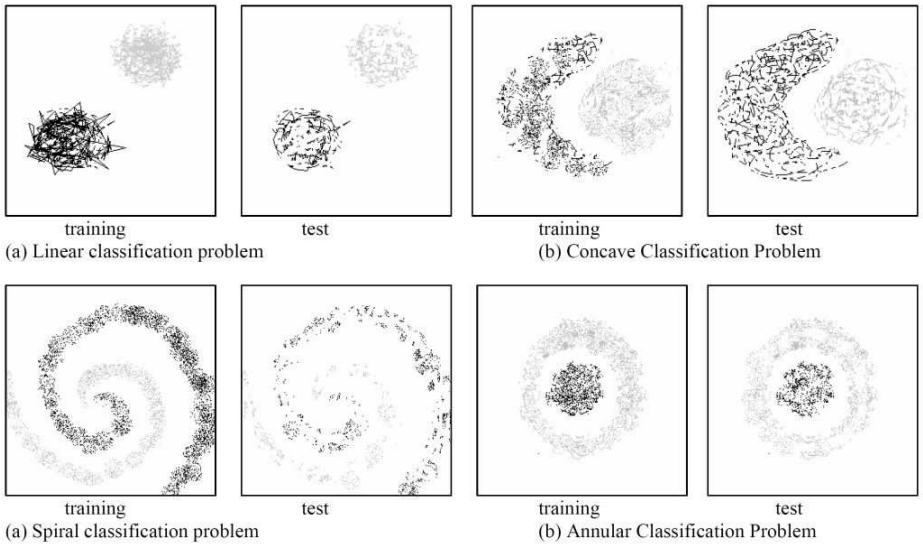


Fig. 7. The pictorial view of datasets used to represent the different types of classification problems

4.1 Implementation Details

Input: Since the range of the feature (the x and y co-ordinates) was between 0 and 255, each feature was represented using 8 bits. Both the feature vectors were then concatenated to yield a 16 bit vector, which is the input vector to our classifier. Hence, in our case, the $MACA$ is of length $n = 16$.

Training: For our GA formulation, we started off with an $m = 4$ and an initial population of 50 $MACA$, each of which has $m=4$. The *top ten* fittest $MACA$s go into the next round (*new population)* and they perform crossover and mutation [2] to generate the rest 40 of the *new population*. The training data consisted around 2000-3000 samples of each of the classes for each classification problem. The $MACA$ ran on the average for 6-7 rounds (of generating *new population*) before all the top-ten fittest $MACA$s ended up having the same fitness values. This is when we stopped the training and took the first of the $MACA$s as our solution $MACA$.

Testing: The test dataset had again typically around 2000-3000 samples of each of the classes for each problem. Using the PEF bit values and their corresponding class association, we tested the test dataset to obtain the results mentioned in the next sub section.

4.2 Classification Accuracy

The performance of our classifier was compared against a linear classifier based on *SVM Light* [4] on the datasets shown earlier. The results are shown below

Dataset	Scalable $MACA$ classifier accuracy on		$SVMLight$ Accuracy
	Training Data	Test Data	results
Linear	99.24%	99.61%	99.71%
Concave	92.77%	91.99%	95.44%
Spiral	83.88%	77.45%	82.46%
Annular	73.8%	75.94%	75.95%

Table 3. Accuracy test results across different Classification problems

We find that the $MACA$ Classifier performs at par with the SVM over the Linear and Concave Datasets. There is no discrepancy between the training and test data accuracy results. The $MACA$ slightly lags behind the SVM on the Spiral Dataset, however performs at par over the Annular Dataset again. This

shows that the $MACA$ Classifier performs as good as the linear SVM which has been widely used as a general purpose classifier across various benchmark problems in classification.

5 Scalability

The purpose of this was to ascertain that the classification scheme is scalable. Hence, we picked the linear classification problem and scaled it up. In the previous experiment, we had the $MACA$ of size n=16. We, then performed experiment for higher values of n.

5.1 Large Datasets

The datasets were prepared by assuming that the dimension of the figure (grid) was $2^n \times 2^n$, where $2n$ is the required size of $MACA$ which we intend to scale to. The examples (points in our case) belonging to Class 1 were chosen randomly from the points inside the circle with radius $2^n/5$ and center at ($2^n/4$,$2^n/4$), while the points belonging to Class 2 were chosen randomly inside the circle with center at ($3{\cdot}2^n/4$,$3{\cdot}2^n/4$) and radius $2^n/5$. The net effect was that the grid looked similar to *Fig.* 7a, and we intended to show that we can fare well by scaling the size of the problem.

5.2 Computational Time

Dataset	Training Data		Test Data		Time taken
Length of $MACA$ (2n)	Accuracy	Examples	Accuracy	Examples	in seconds
16	99.24%	4311	99.61%	2313	2
32	98.42%	12000	97.66%	2000	3.0
64	96.22%	50000	96.33%	8000	28
100	100.00%	60000	98.54%	10000	32
100	100.00%	4000	99.69%	2000	3

Table 4. Scalability Test results

The time values in the scalability test results are obtained when the program was run on a Java Virtual Machine (JVM) on a system with Pentium© III processor and 512MB RAM. The behaviour is in accordance with the complexity analysis of *Table 2.5*. Here, we can particularly see that there is *almost no time dependance on the size of the size of $MACA$ but only on the number of samples* (refer Row 1 & 5 in *Table 4*). Hence, we find that the proposed classification scheme scales extremely well with the size of the input.

6 Application to Remote Sensing Image Classification

We test our scheme on a real-world classification example. We have chosen a binary classification problem. A four band image of the city of Kolkata, India taken by Indian Remote Sensing (IRS) Satellite and a known classification of the pixels (ground truth) into two classes is taken. One class consists of pixels that correspond to manmade structures and water bodies while the other class doesn't. The goal of the classification scheme is to correctly classify unknown pixels (samples) into the class representing manmade structures and water bodies or otherwise.

6.1 IRS Dataset

The IRS image is a four band image in 512×512 grid. Hence, corresponding to each pixel, we have we have four intensity values (or features / attributes). The values of each of the band was found to be between 14 and 80, so we could therefore represent each of the values using 7 bits. Hence, for a pixel, we would have 28 bits. Therefore, the MACA would be of the size 28. Ground truth is available from a pre-classified image in [8] and 120000 examples are chosen randomly as training samples from the aforementioned tagged examples and 20000 as test examples. Also, the classified image obtained using $MACA$ classifier is shown in *Fig. 8*. It was found to tally well with the ground truth.

Training Data		Test Data	
accuracy	samples	accuracy	samples
97.76%	120000	97.26%	20000

Table 5. Accuracy results on IRS Dataset

The results int *Table 5* show that performance of the classifier is fairly good in terms of accuracy in classifying the pixels.

6.2 Observations on the Classifier

Apart from the accuracy, we also made some interesting observations on the final pseudo-chromosome of the $MACA$ which was the fittest. It was known to us *a priori* that band 4 of the image held key to the classification. The threshold for the classification was set on the value of the attribute representing the band 4. In the initial population of the $MACA$ the PEF bits (denoted by -1 in the pseudo-chromosome) were scattered randomly across pseudo-Chromosome. Our observation (*Fig.* 9) shows that the PEF bits had gravitated towards the bits that represented band 4, i.e. the last 7 bits in the pseudo-chromosome shown in *Fig.* 9. The pseudo-chromosome presented corresponds to the $MACA$ which classifies the IRS dataset based on our classification scheme. This suggests that

Fig. 8. Image of Kolkata, India showing water bodies and man-made structures regenerated using our Classifier

our classifier is able to identify the important attributes (the attributes which leads to the decision about the class membership) and hence it paves way for us to devise inferencing mechanism for classification based on the distribution of the dataset. The inferencing mechanism shall exploit our encoding scheme which enables us to maintain a direct correlation between the attributes and the representation in the final decision making step (the position and value of the PEF bit in our case) of the classifier. This can be of remarkable consequence because unlike the neural systems, where the final classification decision is based on the firing of certain neurons, which is too complex to correlate the neural weights and the input attributes to devise a inferencing mechanism on it. In our next set of experiments we have successfully attempted towards the latter.

7 Feature Selection Using $MACA$ Classifier

For pattern recognition task all features present in a data set may not have the same importance, i.e., some features may be redundant and also some may have derogatory influence on the classification task. Thus selection of a proper subset of features from the available set of features is important for designing

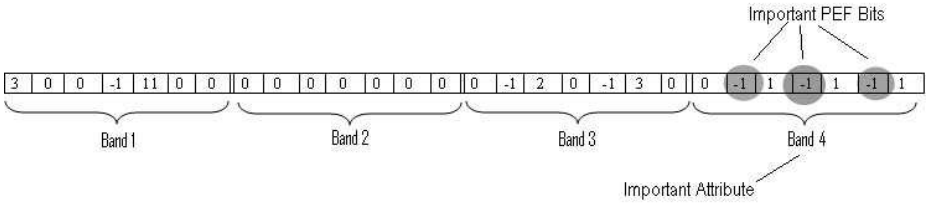


Fig. 9. The 28-bit Pseudo-Chromosome corresponding to the $MACA$ that classifies the IRS dataset best by our classification scheme. Note the increased density in the PEF bits corresponding to Band 4

of efficient classifiers. But most of the existing feature selection methods [**?**,10] perform feature analysis in a separate phase, offline with the main classification process. Independent domain specific schemes for automatic feature selection have also been proposed in [12, 11], but they remain very domain specific. In the work of [13], a hybrid feature selection algorithm that uses three different statistical measurements to evaluate features: between-class pairwised distance, linear separability and overlapped feature histogram was proposed. Their hybrid feature selection algorithm applied the Bayesian EM (Expectation Maximization) to the features ranked by the three measurements alluded to above to select a sub-optimal feature set. This was a general approach but again, done as a prelude to the actual classification.

A recent approach has been reported in [9] was a neuro-fuzzy scheme for classification with online feature selection was proposed using a four-layered feed-forward network for fuzzy rule based classification. Based on the observations in *Section 6*, we find that, it is possible for us to perform feature selection while training. GA based scheme points to the relevant features using the PEF bits. We clarify this using two simple experiments.

7.1 Experiment Feature Selection

We artificially create two sets of datasets (training data and testing data) in which the classes are linearly separable. One dataset will have the classes which show large variance in only one feature while the other will show variance in both the features. It is expected that the classifier should show more number of PEF bits in the bits corresponding to the vector with large variance while the it would show more or less equal number of PEF bits corresponding to both the features in the second case.

7.2 Datasets

Variance in Single Feature *Figure 1* shows two classes which vary along the x-axis alone. These classes have their means around $(50, 150)$ and $(150, 150)$

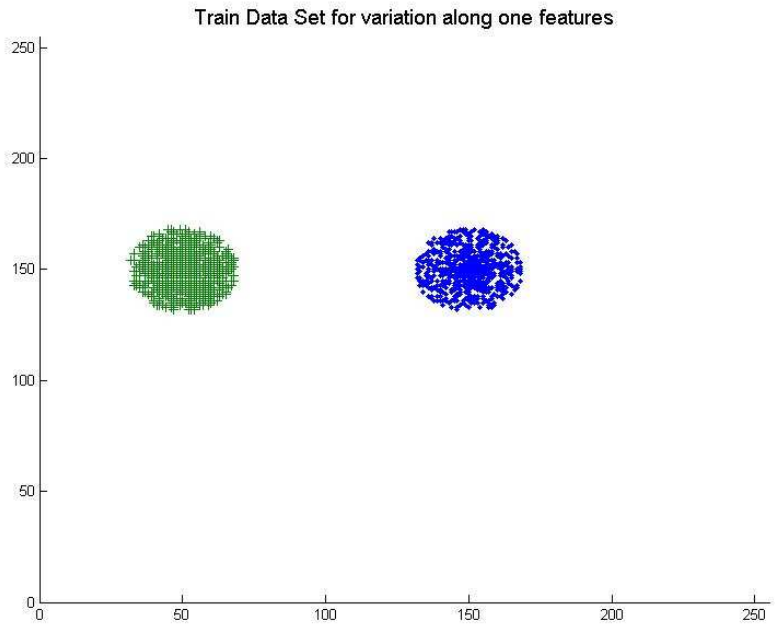


Fig. 10. Datsets with variance along one feature only

on the (x, y) plane with a class spread of 20 units about their mean on both axes. The classifier (in our case it is the pseudo-chromosome) that was learnt for this dataset is depicted in *Figure 2*. Here, we notice that the PEF bits are concentrated on the x feature since the variance is shown along the x axis and it remains true to our expectations.The classifier gave a 0% error in classifying a similar test dataset.

Another interesting observation can be made looking at the pseudo-chromosome. The value of the bit corresponding to the PEF bit in x corresponds to 128 which looks justifiable as the class with mean as $(150, 150)$ in it's x feature vector will have the bit corresponding to 128 *"on* while the other class with mean as $(50, 150)$ in it's x feature vector will have the bit corresponding to 128 *"off*. Hence, the bit corresponding to 128 is truly a discriminating bit and has been rightly captured by our classifier. The attractors corresponding to the class with mean $(150, 150)$ have been found to have the 128's bit position on while it was off in the other class's attractors.

A rule extracted out of this information could be extracted from the like- *if any feature vector has* $x_{128} == 1$ *then it is in Class A, else, it is in Class B* could be useful. If semantics are formally attached to the feature representation scheme which is pretty straightforward in our case, we could optimistically make a scheme which would infer from these rules meaning like- *A member of Class*

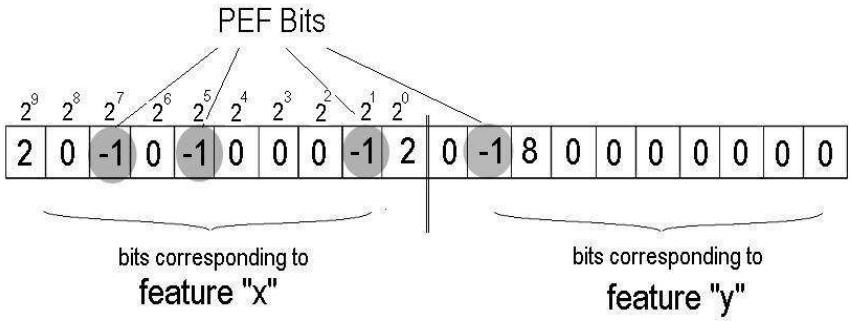


Fig. 11. Pseudo-chromosome corresponding to the classifier that classifies dataset with variation along x-axis

A has feature $X >= 128$ *while a member of Class B has feature* $X < 128$.

Variance in Both Features *Figure 3* shows two classes which vary along both the x-axis and y-axis. These classes have their means around $(50, 50)$ and $(150, 150)$ on the (x, y) plane with a spread of 20 units about their mean on both axes. The classifier (in our case it is the pseudo-chromosome) that was learnt for this dataset is depicted in *Figure 4*. Here, we notice that the PEF bits are distributed both on the x feature and the y feature as the variance is shown along both the axes; true to our expectations. The classifier also gave a 0% error in classifying a similar test dataset. A similar observation can be made in this case too. Here both axes will show some properties and the semantics shown will be applied in a slightly different manner. The bottom line is that our classification scheme can be extended to have rule generation. Next, we move onto a real life example where feature selection holds importance.

8 Classification and Feature Selection in Molecular Cancer Study

Although cancer classification has improved over the past 30 years, there has been no general approach for identifying new cancer classes (class discovery) or for assigning tumors to known classes (class prediction). A generic approach

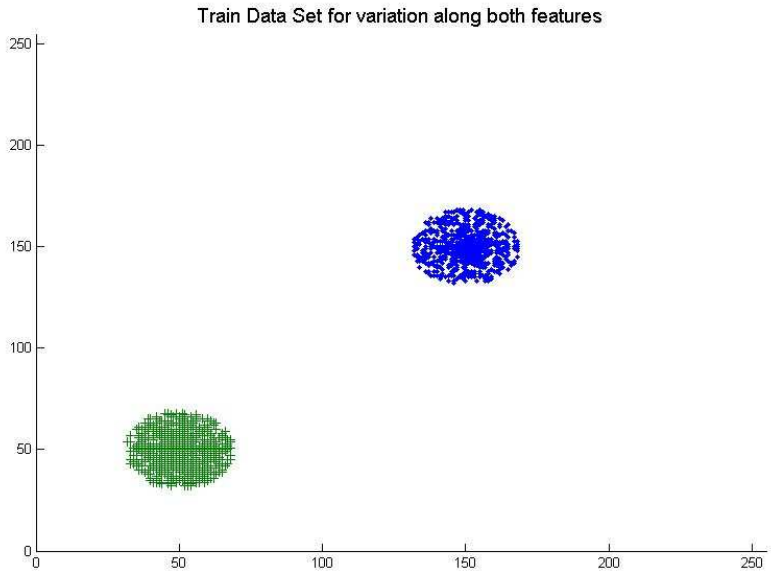


Fig. 12. Datsets with variance along both features

to cancer classication based on gene expression monitoring by DNA microarrays is described and applied to human acute leukemias as a test case in [14]. A class discovery procedure automatically discovered the distinction between acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL) without previous knowledge of these classes. The automatically derived class predictor was able to determine the class of new leukemia cases. The results in [14] demonstrate the feasibility of cancer classification based solely on gene expression monitoring and suggest a general strategy for discovering and predicting cancer classes for other types of cancer, independent of previous biological knowledge.

In this experiment, the task was to classify the test data after learning the gene expressions from the training set and also to find the genes which are important in characterizing the class of Leukemia. The important genes had been medically identified and hence our task was to tally that the features (genes) identified by our scheme with the ones medically identified.

8.1 Dataset

Blood samples were taken from the patients suffering from both ALL and AML (the two forms of Leukemia). 7129 genes in the blood were isolated by clinical methods described in [14]. By procedures which are beyond the scope of our current domain, an integer value corresponding to each of the 7129 genes were obtained. After appropriately offsetting the features to bring them to a common

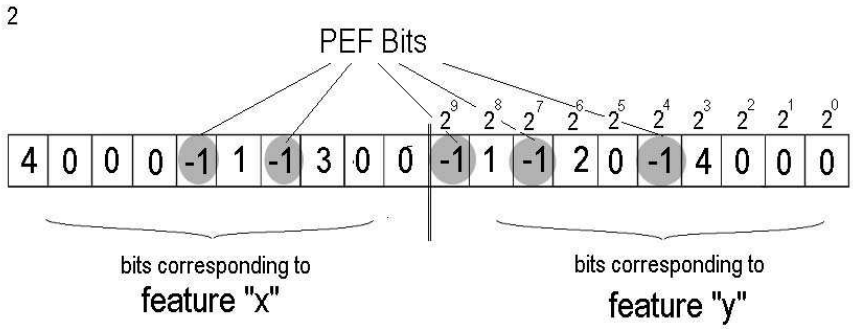


Fig. 13. Pseudo-chromosome corresponding to the classifier that classifies dataset with variation along both axis

scale, we had 7129 features. For every patient, each features lay in the range of 0-265,234. Next, each feature was binarized (so that each feature now had a size of 19 bits) and all these features were concatenated to form the feature vector of size 135451 (19 times 7129) bits.

The training data consisted of 38 patients in which there were 26 patients infected with ALL and 12 patients infected with AML. The testing data consisted of 34 patients which had 20 ALL cases and 14 AML cases.

8.2 Results

The training data was classified with full accuracy using just 10 PEF bits. The 7129 genes were pruned to some 500 genes based on expert opinion and then they had been used for classification in [14]. Top 4 genes were identified in [14] and proved medically too. In our trained classifier, majority of the PEF bits were corresponding to those genes mentioned. Moreover, the class prediction in our case was at par with results of [14] as 32 of the 34 patients were correctly identifed in their respective classes.

Hence, this experiment re-affirms the belief that the proposed classification scheme is capable of feature selection.

9 Conclusion

In our present work, we have exhaustively experimented on the $MACA$-based binary classifier and proved it's functional capabilities. We have developed a fast $MACA$ generation scheme and reduced the computational complexity of the $MACA$ based classification system. This has enabled us to show that the

classification system is scalable. Current classification scheme has also enabled us to draw some conclusions on the importance of certain attributes. This shall enabled us to devise an inferencing system on top of the $MACA$ classifier which is of great importance to the data mining community.

The current $MACA$ based classifier shows a promise of building a simple binary classifier and an inferencing mechanism on the dataset on which the classifier is being learnt. Our future work includes building a multi-class classification based on this and theoretical work which gives us a deeper understanding on the dynamics of the classifier.

References

1. P. P. Chaudhuri, D. R. Choudhury, S. Nandi and S. Chattopadhyay, *"Additive Cellular Automata Theory and Applications"*, IEEE Computer Society Press, California, USA, 1997.
2. N. Ganguly, *"Cellular Automata Evolution : Theory and Applications in Pattern Recognition and Classification"*, Ph.D thesis CST Dept. BECDU India.
3. P. Maji C. Shaw, N. Ganguly, B. K. Sikdar and P. P. Chaudhuri, *"Theory and Application of Cellular Automata For Pattern Classification"*, Fundamenta Informaticae pp 321354, Vol. 58, 2003.
4. T. Joachims, *"Learning to Classify Text Using Support Vector Machines"*, Dissertation, Kluwer, 2002.
5. K. J. Lang and M. J. Witbrock, *"Learning to Tell Two Spirals Apart"*, Proceedings of the 1988 Connectionist Models Summer School, Morgan Kaufmann (Ed.), pp. 52-59, 1988.
6. S. Singh, *"2D spiral pattern recognition with possibilistic measures"*, Pattern Recognition Letters, pp. 141-147, Vol. 19, 1998.
7. S. E. Fahlman and C. Lebiere, *"The Cascade- Correlation Leaming Architecture"*, Advances in Neural Information Processing Systems, pp. 524-532, Vol. 2, D. Touretzky (Ed.), 1993.
8. S. K. Pal and P. Mitra, *"Multispectral image segmentation using the rough-set-initialized EM algorithm"*, IEEE Transactions on Geoscience and Remote Sensing, pp. 2495-2501 Vol. 40, No. 11, November 2002.
   bibitemDe97 R. De, N. R. Pal and S. K. Pal, *"Feature analysis: neural network and fuzzy set theoretic approaches"*, Pattern Recognition, Vol. 30, 1997.
9. D. Chakraborty and N. R. Pal, *"Designing Rule-Based Classifiers with On-Line Feature Selection: A Neuro-fuzzy Approach"*, Lecture notes in computer science, ISSN 0302-9743.
10. D.W. Ruck , S. K. Rogers and M. Kabrisky, *"Feature selection using a multilayered perceptron"*, Journal of Neural Network Computing, 1990.
11. S. Zhou and J. Jin, *"Automatic feature selection for unsupervised clustering of cycle-based signals in manufacturing processes"*, IIE Transactions, Vol. 37, 2005.
12. C. D. Huang, C. T. Lin and N. R. Pal, *"Hierarchical learning architecture with automatic feature selection for multiclass protein fold classification"*, IEEE Trans Nanobioscience, Vol.2, Dec 2003.
13. H. Guo, Y. L. Murphey, *"Automatic Feature Selection - A Hybrid Statistical Approach"*, 15th International Conference on Pattern Recognition (ICPR'00), 2000.
14. T. R. Golub, *et al*, *"Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring"*, Science, Vol. 286, 1999.