An Evolutionary Approach to Agent-based Pattern Classification

Federico Bergenti

Dipartimento di Matematica, Università degli Studi di Parma
Viale G. P. Usberti 53/A, 43100 Parma, Italy
Email: federico.bergenti@unipr.it

Abstract: This paper presents an agent-based approach to pattern classification that intensively uses evolutionary techniques to support supervised learning of classifiers. The paper first provides a formalization of the peculiar sort of agents and multi-agent systems that we consider. Then, a brief description of the adopted evolutionary technique is provided together with a justification for its use. Finally, the proposed approach is embedded into a prototypical, real-world application that offers a solid ground for the concrete assessment of classification performances. The preliminary results that we obtained clearly show that the proposed approach realizes a good trade off between accuracy and speed in a fully distributed and decentralized manner.
Keywords: pattern classification, evolutionary method, multi-agent system.

1. Introduction

The use of agents and Multi-Agent Systems (MASs) in various tasks related to pattern recognition has gained interest in recent years mainly because the decentralized and autonomous character of agents provides a very natural means to concretize the scheme of multiple experts cooperatively working in a recognition process. There is a notable body of research that apply the metaphor of agent to nearly all aspects of combining multiple experts to enhance the performances of single experts in the intent to realize high quality, efficient pattern recognition (see, e.g., [1] for one of the most recent works on the topic).

In this paper, we take a different perspective and we study the possibility of using agents to realize single experts of classification. We see agents at a micro scale and we feed them with input patterns that we want to classify into predeterminated categories. Each agent is provided with a part of the input pattern and the result of classification—the estimated class of the pattern—is a-priori, one-to-one associated with a property of the MAS that is required to hold at the end of the computation. For example, the benchmark discussed in Section 3 comprises a set of MASs—at least one for each category—each of which is made of 104 agents; each agent is fed with a single bit of the input pattern. Each MAS is an expert of classification for a particular category and its output, a binary answer in this case, is determined by the state that a specific representative agent holds at the end of the computation.

The proposed approach allows exploiting the decentralized computation of agents in the early stage of pattern classification, i.e., in the realization of single experts, and it also provides a uniform, agent-based view of the whole process when the outputs of various experts—holons, we may say—are combined together using one of the available agent-based approaches to multi-expert classification.

The micro-scale view of MASs that we adopt in this work, together with the inherent difficulties of pattern classification, call for learning techniques to support the concrete realization of agents. We consider the classification problem in its most general form and, unfortunately, most of the proposals thought to combine (micro-scale) agents and learning are designed to deal with specific problems only; just a few of them are truly general purpose (see, e.g., [15] for an approach that shares the objectives with the present work). This is the reason why we decided to adopt a very general evolutionary approach, which is well appreciated for providing good performances with very few—if any—assumptions about the problem at hand.

Before describing the proposed evolutionary approach, we need to formally define which peculiar sort of MAS we consider for the realization of classifiers. This is done in Section 2 which provides a well-founded definition of MAS meant to concretize the micro-scale view of agents that we assume in this paper and to give some results needed to support the proposed evolutionary approach.

Agents that contribute to implement a single classifier are co-evolved using the general-purpose supervised approach described in Section 3. The same section also describes a real-world problem of low-level vision where we successfully benchmarked the proposed approach for the synthesis of classifiers. We ran our agent-based, co-evolved classifiers against a set of images of digits taken from real car plates in the intent to perform on-the-fly character recognition.

The preliminary results that we obtained from the presented benchmark are still far from real applicability (which is 99.9% in such a scenario) and they are briefly discussed in Section 4 together with the description of some future development of this research.

2. Which Multi-Agent Systems?

The longstanding debate about a generally agreed definition of MAS seems definitely abandoned now and the net result of it is that no formal, accepted definition of MAS exists—and probably will ever exist. Anyway, we need to formally ground our approach for agent-based classification in order to give provable warranties of its characteristics. This is the reason why we present here a formalization of the idea of MAS and we prove some property needed to support our approach to agent-based classification. The definition of MAS that we give extends the

well-known definition of Cellular Automaton (CA) to incorporate a more agent-oriented view of computation that relaxes some assumptions, i.e., homogeneity, locality and synchronicity, that are core features of CAs.

2.1 Cellular Automata

von Neumann [13] first introduced CAs in the early fifties as computing systems capable of self-reproduction and self-organization. As an informal definition, a CA is a system made of a grid of cells each of which performs a simple computation. Each cell is connected to a set of neighbours in order to facilitate information exchange across the grid. In the conventional understanding of CAs, all cells are equipped with a single rule that drives all their computations, i.e., they all compute the same function synchronously and the complex behaviour of the system emerges from *(i)* the synchronous application of the same rule to different data, and *(ii)* the flow of information across the grid.

The homogeneous behaviour that characterizes CAs makes them ideal for the simulation of the dynamics of complex physical systems (see, e.g., [10]) and they find in such an application domain their most common use. In recent years, CAs have also been used in many other applications where fast and parallel computation is required, e.g., low-level, real-time vision [2]. From a theoretical point of view, CAs are an interesting model for massively parallel computation [14].

In order to extend CAs to model MASs, we need to drop various features that characterize them, as follows.

Homogeneity. All cells in a CA are assumed to compute the same function over different data. This is not sufficient to model MASs because we want to equip each and every single agent with a peculiar rule. This leads to the model of Non-Uniform CAs (NCAs), in which different cells are possibly assigned different rules. NCAs gained some interest in the past decade for the realization of flexible hardware [11, 12] and we already applied them to real-world classification problems [4].

Locality. Agents are typically interested in communicating with local neighbours, but we cannot accept this limitation as a general characteristic of MASs. Agents may need to communicate with remote agents and every single agent may have its peculiar network of acquaintances.

Synchronicity. Agents autonomously interact with their environment in a completely asynchronous manner. No central clock or source of control can be assumed in a general definition of MAS.

It is worth noting that we do not incorporate in our model of MAS other possible generalizations of CAs. For example, we assume that local rules of agents do not change over time and we assume steady networks of acquaintances.

We start here from a common definition of CA in order to extend it in the mentioned directions to come to a definition of MAS which will form the basis of the proposed approach for agent-based classification.

Definition 2.1 *A* Cellular Automaton *A in $d \in \mathbf{N}^+$ dimensions is a quadruple:*

$$A =< S,\ d,\ V,\ f >$$

where S is a finite set of state symbols, $V \subseteq L$ is a finite neighbourhood structure over the lattice $L = \mathbf{Z}^d$, and f is a transition function known as rule. *A* cell *is a point x in the lattice L.*

A CA associates a state $s \in S$ to each cell and a *global configuration of states* c is defined in the space:

$$S^L = \{c \mid c : L \to S\}.$$

The neighbourhood structure $V \subseteq L$ is a set of $m \in \mathbf{N}^+$ vectors used to build the local neighbourhood of each cell:

$$V = \{v_i \in L \mid i = 1,\ 2,\ \ldots,\ m\}.$$

Given V, the neighbourhood V_x of each cell x is created by means of T, the Abelian group of translations of L into itself[1]:

$$V_x = \{(x + v) \in L \mid v \in V,\ x \in L\}.$$

We can easily map each cell x into a *local configuration of states* using the global configuration of states, as follows:

$$c_{V_x} : V_x \to S$$
$$c_{V_x}(v) = c(x + v) \quad \text{with} \quad x \in L,\ v \in V.$$

The *local computation* of a CA is the mapping:

$$f : S^V \to S$$
$$f(c_V) = s \quad \text{with} \quad c_V \in S^V,\ s \in S.$$

We can now say that given a global state c_0 at time t_0, a CA computes in terms of repeated, synchronous applications of f to all cells. This is better captured if we rewrite f as:

$$f(c_{V_x}) = f(s_1,\ s_2,\ \ldots,\ s_m)$$
$$\text{with} \quad s_i = c_{V_x}(v_i),\ v_i \in V$$

$s_i \in S$ being the states of the cells in the neighbourhood of x.

We can finally define the function that a CA globally computes at each step as:

$$G_f : S^L \to S^L$$
$$[G_f(c)](x) = f(c_{V_x}) \quad \text{with} \quad c_{V_x} \in S^V,\ x \in L$$

which is then generalized to the global function computed after n steps as a repeated composition G_f^n.

Given an initial global configuration of states c_0, we can now compute the final global configuration of states after n steps as: $c_n = G_f^n(c_0)$.

Given such definitions, an outstanding result is that CAs are equivalent to Universal Turing Machines [7].

Finally, it is worth mentioning that normally we are not interested in CAs that span the entire lattice L; rather we often deal with CAs that exhibit a periodic behaviour. Before introducing so called *Periodic Cellular Automata*, we need the following definition:

Definition 2.2 *A global configuration of states $c \in S^L$ is said* periodic *with period $l \in L$ if and only if:*

$$\forall x \in L,\ c(x + l) = c(x).$$

[1] T is defined in the usual manner as $(L,\ +)$ where $+ : L \to L$ is the vector sum between the elements of L.

So that we can now properly define periodic CAs in terms of a characterization of their computations, as follows:

Definition 2.3 *A cellular automaton A is said* periodic *with period $l \in L$ if and only if for any initial configuration $c_0 \in S^L$ with period l:*

$$\forall n \geq 0,\ \forall x \in L,\ c_n(x+l) = c_n(x).$$

Such a definition is of notable importance because the following result holds:

Proposition 2.1 *A CA $A =< S,\ d,\ V,\ f >$ is periodic with period $l \in L$ if and only if its initial configuration c_0 is periodic with period l, regardless of V and f.*

Such a classic result of the theory of CAs allows considering finite grids of cells wrapping their neighbourhoods across their boundaries as periodic CAs.

2.2 Multiagent Systems

In order to generalize the previous definition of CA to incorporate a more agent-oriented view of computation, we need to address the three issues already mentioned, i.e., homogeneity, locality and synchronicity. We start from the first two because the latter does not require changes to definitions and results.

Definition 2.4 *A* Multi-Agent System *A in $d \in \mathbf{N}^+$ dimensions is a quadruple:*

$$A =< S,\ d,\ I,\ M >$$

where S is a finite set of state symbols, $I \in \mathcal{P}(L)^L$ is a map of neighbourhood structures over the lattice $L = \mathbf{Z}^d$, and M is a map of rules*. An* agent *is a point x in the lattice L.*

For each agent x, the MAS assigns a state, a local rule via M and a local neighbourhood by means of I, which maps each and every single agent to a local neighbourhood structure:

$$I : L \to \mathcal{P}(L) \quad I(x) = V.$$

For an agent x, we can define its neighbourhood $V_x \subseteq L$ as the set of cells that it can reach using its local neighbourhood structure, as follows:

$$V_x = \{y = x + v \mid v \in I(x),\ y \in L\}.$$

The local configuration of states of an agent x is the set of states of the agents in its local neighbourhood.

In order to complete the definition of a MAS, we now need to consider V, the union of all local neighbourhood structures:

$$V = \bigcup_{x \in L} I(x)$$

which allows defining:

$$F = \{f : S^V \to S\}$$

the set of all possible rules in the MAS. Having said this, we can give a precise meaning to M, the map of local rules in the MAS, as $M : L \to F$.

Given M ed I it is now possible to define the global function G that the MAS computes at each step exactly like we did for CAs:

$$G : S^L \to S^L \quad [G(c)](x) = [M(x)](c_{V_x})$$

which finally *(i)* allows defining the global function that a MAS computes in n steps and, notably, *(ii)* supports the definition of periodic MAS.

The given definition of MASs has two important properties that allow formally connecting MASs with CAs. The demonstration of such properties is out of the scope of this paper and interested readers can consult a self-contained discussion on the topic that also provides detailed proofs at [3].

First, we provide a result about the computational power of MASs and CAs in their most general form:

Proposition 2.2 *Given a MAS A_M it is not always possible to define a CA A_C capable of performing the same computation of A_M in the same number of steps.*

Then, we refine the relationship between the computational power of MASs and CAs in a more practical, i.e., periodic, situation:

Proposition 2.3 *Given a periodic MAS A_M with period $p \in L$, it is always possible to find a periodic CA with the same period capable of performing the same computation of A_M in the same number of steps.*

As previously noted, the definition of MAS that we have just formalized does not consider the mentioned issue of synchronicity at all, i.e., a MAS is still a synchronous system and altogether agents compute their local function synchronously. This is beneficial because it allows talking about steps of computation and defining G^n, the global function that the MAS computes in n steps. Unfortunatelly, this is not what we want from a general definition of MAS.

In order to cope with this anticipated problem of the definition that we introduced, we first simply relax synchronicity and let agents apply their local rules over the states of their local neighbours. This forbids to talk about a global computation step of the entire MAS and we need to assume local computation steps with no temporal relations.

Fortunately, the provided definition of MAS does not allow local rules and local neighbourhoods to change over time, which permits to exploit some common assumption that we normally take for granted in MASs:

1. Agents access the states of their neighbours using messages sent to explicitly ask for information about such states, i.e., agents do not read the states of their neighbours;
2. Agents are normally supported by a mailbox that store messages before actual processing;
3. The messages that an agent sends to another agent in its local neighbourhood are assumed to reach the receiver in the same order they were sent.

These common assumptions allows defining a local step of computation of agent x in terms of the application of its local rule to the states that it requested to its neighbours after having

completed a previous step. Independently of the actual temporal relationship between the applications of local rules within the MAS, an agent x applies its local rule for the n-th time only after having received n messages from its neighbours. This is of crucial importance for our model—and also for our final objective of implementing classification—because we can work as if MASs were synchronous systems because we can still rely on the previous well-founded definitions of computation step and of global configuration of states after n steps of computation. The demonstration of this result, and its actual formal statement, are out of the scope of this paper.

3. Agent-based Classifiers

The definition of MAS formally stated in the previous section allows giving a precise meaning to the terminology computation step, initial configuration and final configuration when applied to MASs. This permits to rephrase the approach to agent-based classification that we propose in this paper as follows. An input pattern that needs to be classified into one of a number of predetermined categories is loaded as the initial configuration of a MAS, distributing parts of the pattern to different agents. The MAS is left computing for a known number of steps and the output of classification is made available in terms of a specific property of the final configuration, which is normally the state of a chosen representative agent.

The problem now is to design a MAS capable of globally performing a complex classification task. This is described as a supervised learning process in the following subsection.

3.1 Co-Evolving Agent-based Classifiers

The design of a MAS capable of performing some required global computation is commonly considered hard task because it requires mapping a global problem onto a model that inherently relies on local interactions only. Sipper [9] showed that global problems can be addressed using NCAs in conjunction with a particular co-evolutionary algorithm capable of learning the rules needed at each cell to have the NCA perform the requested task. He called such a co-evolutionary algorithm Cellular Genetic Algorithm (CGA) [11] and his approach followed a line of research that used evolutionary techniques to synthesize CAs with various purposes (see, e.g., [8, 6, 5]).

Here, we adopt the CGA with a minor extension that MASs require, i.e., we encode both the local rule and the structure of the local neighbourhood into a single chromosome. In our experiments, we always used a binary, fixed-length string representation of chromosomes and we predetermined the largest neighbourhood allowed in order to have a fixed-length header of the chromosome encoding the actual structure of the local neighbourhood. Moreover, we decided to exploit the intended use in classification to simplify the evolution process. We assumed populations made of single individuals and we allowed migrations only in the scope of the local neighbourhood of each agent. The adaptation of the CGA to MASs is applied in this work as an evolutionary supervised learning algorithm for pattern classification: a MAS is fed with a training set of pre-classified patterns along with the corresponding categories. The local fitness of each agent is computed as the frequency of correct classification over the whole training set, which extimates the success probability of classification:

Fig. 1. (top) original grey-scale images of car plates digits and (bottom) preprocessed patterns.

$$\text{fitness} = \frac{\text{number of patterns classified correctly}}{\text{number of patterns in the training set}}$$

This fitness is local, as it is measured only on the final state of each agent; therefore the CGA is used to evolve the MAS as a whole to maximise the fitness of all agents. This training phase results in the co-evolution of a set of rules and local neighbourhoods driven by the local fitness computed by each agent at each step of the evolution. The resulting MAS is generally organised in clusters of agents characterised by the same rule—quasi-uniform MAS—and giving similar performances.

To exploit co-evolved MASs for classification, the final state of the agent with the highest fitness, or a suitably chosen set of well-performing agents, is used. An ideal classification is achieved by a set of MASs, each trained to recognize a specific class of patterns, with unit fitness. In this ideal situation, each input pattern activates one and only one classifier. As this is never the real case, the output of all classifiers must be taken into account using some multi-expert technique

3.2 Experiments and Results

In order to benchmark the proposed agent-based classification technique on a non-trivial classification task, a database of digits obtained from the acquisition of real car plate images from different viewpoints and in different lighting conditions was considered. Each digit consists of an 8×13 pixel gray-scale image.

A simple preprocessing phase was applied to each digit to obtain binary patterns that were classified by means of binary—two states only—agents. Such a phase consisted of histogram stretching followed by thresholding. Examples of the preprocessed patterns together with their original images are shown in Figure 1.

In our prototype implementation, each bidimensional MAS consisted of a grid—i.e., periodic lattice—of the same size of the input pattern and computation was considered completed after $n = 24$ steps, which we consider a sufficient number of steps to facilitate information propagation throughout the grid. After some experiments that involved the co-evolution of both the local rules and the local neighbourhoods, we decided to restrict on homogeneous von Neumann neighbourhoods to ease the supervised learning phase.

The initial architecture consisted of 10 MASs, one for each category, that were separately trained to detect different target patterns, i.e., different digits. During the training phase, each MAS was co-evolved through 5 runs of the CGA, iterated for

Table 1. Performances of *(a)* single-digit and *(b)* pairwise classifiers

Digit	Sensitivity	Specificity	Fitness
0	0.964	0.995	0.980
1	0.873	0.980	0.927
2	0.932	0.945	0.939
3	0.894	0.968	0.931
4	0.937	0.950	0.944
5	0.925	0.963	0.944
6	0.958	0.925	0.942
7	0.989	0.977	0.983
8	0.840	0.976	0.908
9	0.880	0.967	0.924

(a)

Digit Pair	Sensitivity	Specificity	Fitness
0/9	1.000	0.988	0.994
7/1	0.966	0.988	0.977
2/3	0.977	1.000	0.989
5/6	0.966	0.972	0.969
9/6	0.976	0.986	0.981
6/8	0.958	1.000	0.979
4/1	0.989	0.977	0.983
1/3	1.000	0.990	0.995
3/8	0.981	1.000	0.991

(b)

$20,000$ generations with a mutation rate of 10^{-3} on a training set of 892 patterns. The best MAS for each digit was chosen for classification using the final state of the classifier's highest-fitness agent as a binary output.

Table 3.2a shows the results obtained for each classifier on a test set of 884 patterns in terms of sensitivity and specificity computed on the whole training set as *(i)* the frequency of correct classification of the target pattern, and *(ii)* the frequency of correct classification of non-target patterns, respectively.

As a preliminary test aimed at evaluating the discrimination capabilities of the obtained set of classifiers we estimated the maximum achievable performance compatible with the presence of collisions. As the estimated performance were too low—about 90%—we decided to extend this simple architecture to improve quality. In this regard, a number of classifiers trained to solve the most critical cases were added. It was expected that such new classifiers could achieve rather good performances, as a binary classifier generally works better if the features of the two classes it has to discriminate are somehow uniform within each class. Based on these assumptions, we extended the initial architecture by adding a set of classifiers trained on patterns belonging to pairs of categories that produced the highest number of collisions, as shown in Table 3.2b.

4. Conclusions

This paper presents a general-purpose framework intended to support the realization of pattern classifiers in terms of co-evolved MASs. First, we formally defined the peculiar sort of MAS that we consider in terms of a generalization of CAs. This opened the question of synthesizing the rules of each and every single agent to make a whole MAS behave like a classifier. We adapted a technique originally developed for NCAs to our purposes and we employed our formalization to show that the proposed approach is well founded. The net result of this activity is a general purpose approach that use local genetic algorithms to co-evolve the rules and neighbourhood structures of all agents in the MAS.

In order to provide a quantitative measure of the characteristics of the proposed approach, we applied it against a real-world problem that originates from a car plate recognition application. The results of conducted experiments show that the agent-based classifiers behave reasonably, even if the results demonstrate that our co-evolved, agent-based classifiers are not yet ready for an adoption in the field.

The presented work has been originating multiple directions of development. Mainly, we see interesting developments in understanding the properties of the formal models that we briefly sketched and in comparing such models with others available in the literature. In particular, we are in the process of understanding how the proposed model of MAS can characterize emergent computation and how it relates to techniques commonly used to understand emergent phenomena in complex systems.

References

[1] A Addis, G Armano and E Vargiu, From a generic multiagent architecture to multiagent information retrieval systems. Procs 6^{th} Int'l Workshop "From Agent Theory to Agent Implementation" 2008, LNCS, Springer.

[2] G Adorni, A Broggi, G Conte and V D'Andrea, A self tuning system for real-time optical flow detection. Procs IEEE System, Man, and Cybernetics Conference 1993, Vol. 3, pp. 7–12.

[3] F Bergenti, Metodologia di progetto di classificatori basati su automi cellulari evolutivi e sviluppo di una applicazione per il riconoscimento di simboli. Master's thesis, Università degli Studi di Parma, 1998.

[4] S Cagnoni, F Bergenti, M Mordonini and G Adorni, Evolving binary classifiers through parallel computation of multiple fitness cases. IEEE Transactions on Systems, Man and Cybernetics Part B–Cybernetics, Vol. 3, No. 35, 2005, pp. 548–555.

[5] M Chady and R Poli, Evolution of cellular-automaton-based associative memories, Tech. Rep. CSRP-97-15, University of Birmingham, School of Computer Science, 1997.

[6] P Chaudhuri, D Chowdhury, S Nandi and S Chattopadhyay, Additive Cellular Automata: Theory and Applications, IEEE Computer Society Press, 1997.

[7] E F Codd, Cellular Automata, Academic Press, 1968.

[8] M Mitchell, J Crutchfield and R Das, Evolving cellular automata with genetic algorithms: A review of recent work. Procs 1^{st} Int'l Conference on Evolutionary Computation and its Applications 1996.

[9] M Sipper, Evolution of Parallel Cellular Machines: the Cellular Programming Approach, Springer-Verlag, 1997.

[10] T Toffoli and N Margolus, Cellular Automata Machines: A New Environment for Modeling, MIT Press, 1987.

[11] M Tomassini, Evolutionary algorithms. Towards Evolvable Hardware 1996, E Sanchez and M Tomassini, Eds., LNCS, Springer-Verlag, pp. 19–47.

[12] M Tomassini and E Sanchez, Towards Evolvable Hardware, Springer-Verlag, 1996.

[13] J von Neumann, Theory of Self-Reproducing Automata, University of Illinois Press, 1966.

[14] S Wolfram, Theory and applications of cellular automata, World Scientific, 1986.

[15] Q F Zhao, A general framework for cooperative co-evolutionary algorithms: a society model. Procs IEEE Int'l Conference on Evolutionary Computation 1998, IEEE Computer Society Press, pp. 57–62.