Using Cellular Automata for feature construction – preliminary study

Matej Mertik[1], Mykola Pechenizkiy[2], Gregor Štiglic[1], Peter Kokol[1]

[1]Department for Computer Science, University of Maribor,
matej.mertik @uni-mb.si
[2]Department for Informatic and computer Systems, University of Jyväskylä
mpechen@cs.jyu.fi

Abstract— **When first faced with a learning task, it is often not clear what a good representation of the training data should look like. We are often forced to create some set of features that appear plausible, without any strong confidence that they will yield superior learning. Beside, we often do not have any prior knowledge of what learning method is the best to apply, and thus often try multiple methods in an attempt to find the one that performs best. This paper describes a new method and its preliminary study for constructing features based on Cellular Automata (CA). Our approach uses self-organisation ability of cellular automata by constructing features being most efficient for making predictions. We present and compare the CA approach with standard Genetic Algorithm (GA) which both use Genetic Programming (GP) for constructing the features. We show and discuss some interesting properties of using CA approach in our preliminary experimental study by constructing features on synthetically generated dataset and benchmark datasets from the UCI machine learning repository. Based on the interesting results, we conclude with directions and orientation of the future work with ideas of applicability of CA approach in the feature.**

Index Terms— **Feature Construction (FC), Datamining, Classification, Cellular Automata (CA)**

I. INTRODUCTION

Design of appropriate data representations in the KDD is as important as selection of the method that suits best for the task. Better performance is often achieved using features derived from the original input.

There are two different goals which may be pursued for feature construction: achieving best reconstruction of the data or being most efficient for making predictions. The first problem is closely related to the data compression and it is unsupervised learning problem. The second problem is supervised learning problem and it deals with finding such transformation of the training data to a new representation space so that the performance of the classification technique can be improved.

The approaches reported in the literature within Feature Construction term can be roughly divided into three categories [1]:

- Feature selection methods (also referred to as variable selection), where the resulting representation is a subset of the original one, i.e. $F_{Selected} \subseteq F_{Original}$
- Feature weighting methods, where the transformation method assigns weights to particular attributes (thus, the representation does not change here, i.e. $F_{Selected} = F_{Original}$). The weight of attributes reflects relative importance of an attribute and may be utilized in the process of inductive learning.
- Feature construction methods. Here, new features are invented and defined (in some language) as expressions, which refer to the values of original ones. What about feature extraction, feature transformation concepts?

Principally, each of approaches listed here encompasses its predecessors. For example, feature selection may be regarded as a special case of feature weighting with constrained domains of weights (e.g. Boolean {0,1} set). Analogously, feature construction usually does not forbid creating features being 'clones' of the original attributes (i.e. $F_{Original} \subseteq F_{Constructed}$), what is essentially equivalent to feature selection.

For encountered optimization problems Brute Force technique can be applied. However, because of higher dimensional spaces, the computational complexity of Brute Force is growing exponentially, therefore techniques based on heuristic searches such as Forward Selection, Backward Elimination, Hill Climbing, and lately Genetic Algorithm (GA) and Genetic Programming (GP) becomes popular in DM/ML community.

In this work we present a preliminary study of the new approach for constructing features based on Cellular Automata (CA) model. The CA model uses Genetic Programming methodology for constructing the features during its transitions. We compare this novel approach with classical GA on some synthetically and benchmark datasets.

The rest of the paper is organized as follow: In Section 2 we present classical GA approach with GP for feature construction task. We continue with brief description of ideas behind CA in Section 3, and, in Section 4 we introduce our CA-based approach for feature construction, emphasizing its differences with the commonly used GA approach. In section 5 we present the results of our experimental study on several synthetically generated datasets and benchmark datasets. We conclude our work with the brief summary and present the main directions for future research.

1-4244-0470-3/07/$20.00 ©2007 IEEE

II. Genetic Algorithms and Genetic Programming for Feature Construction

GAs are adaptive heuristic search methods that may be used to solve all kinds of complex search and optimization problems [2], for which no efficient heuristic method has been developed. They are based on the genetic ideas of natural selection and genetic processes of biological organisms. Simulating the principles of natural selection and "survival of the fittest" first laid down by Charles Darwin, GAs are able to evolve solutions to real-world problems. They are often capable of finding optimal solutions even in the most complex search spaces or at least they offer significant benefits compared to other search and optimization techniques. Fig. 1 presents typical structure of genetic algorithm.

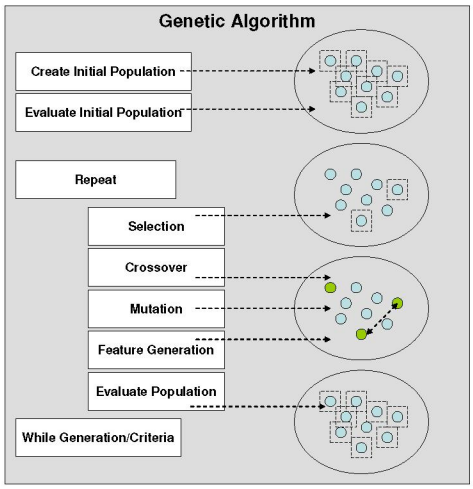

Fig.1 Structure of genetic algorithm

The technique of GP was proposed by Koza [3]. GP has been successfully applied to several applications like symbolic regression, the robot control programs, and classification. Although GP uses the same principles as GA [2], there are some important differences. Basically GA has the genome of string structure, however while the genome in GP is the tree structure. In such a way GP may overcome some problems of GA by searching local optimum.

GP begins with a population that is a set of random created individuals, where each individual represents a potential solution in the form of a binary tree, which is constructed by all possible compositions of the sets of terminals and functions. Then a fitness value of each tree is calculated by a suitable fitness function. Individuals with better fitness are selected to generate new population in next generation with genetic operators, which generally include reproduction, crossover, mutation, and others that are used to evolve functional expressions.

GP is strong in its ability to discover the underlying data relationships and express them mathematically. Therefore is common used in the feature construction problems [4,5]. In our work we will show the use of GP within the model of CA. We will compare the results with standard GA approach for constructing the features (see Fig. 1).

III. Cellular Automata

CA are massively parallel systems [6] consisting of a lattice of cells (in any number of dimensions) each of which has a number of associated discrete states. The state of these cells is updated at discrete time steps, the resultant state dependent upon local state rules. In spite of their simplicity, the dynamics of CA is potentially rich, and ranges from attracting stable configurations to spatio-temporal chaotic features and pseudo-random generation abilities [7]. Those abilities enable the diversity that can possibly overcome local optima by solving engineering problems.

From the computational viewpoint, they are universal, one could say, as powerful as Turing machines and, thus, classical Von Neumann architectures [8]. These structural and dynamical features make them powerful; CA-based algorithms are developed to solve engineering problems in cryptography for instance, and theoretical CA-based models are built in ecology, biology, physics and image-processing. These powerful features make CA difficult to analyze. On the other side almost all long-term behavioural properties of dynamical systems and cellular automata in particular, are hard to predict [7].

However in this paper the aim was not to analyze the process of CA but to use it for feature constriction in the supervised learning problem for improvement of classification accuracy of the learner and compare it with well known approach of GA.

IV. Cellular Automata as Model for Feature Construction

Our novel idea is to exploit benefits of self-organization abilities of cellular automata in feature construction task and its potential in the extraction of new knowledge. Our first task was therefore to define the basic elements (content of the cells) and transaction rules that would result in a constructing features ability of cellular automata.

Most obvious choice is to define a cell FCC (feature construction cell) in the feature construction cellular automata as an operator. In a simplified view we can look at cellular automata as on a population of such operators. In each cell we then have to build an operator that has different and eventually better abilities than already produced ones. If all FCC cells would have the same operator or similar operators then there would be no diversity and it would be quite unreasonable to use any transaction rule on such automata.

Constructing the automata

The most desirable initial situation in a CA is where there is unique operator in each of its FCC cells. We can ensure this to some extend by randomly using different features and operators trees at the beginning of the process. Each FCC in CA represents a new generated feature, which could improve the classification accuracy. The next step is to evaluate the feature set based on original features and recently generated feature. In a similar way we estimate the first population in GA approach.

However, in GA the selection process with mutation and

crossover operators are followed (Fig 1), while in our CA model mentioned steps of genetic operators are implemented within transaction rule of CA. Figure 2 below represent a model of Feature Construction cellular Automata FCCA.

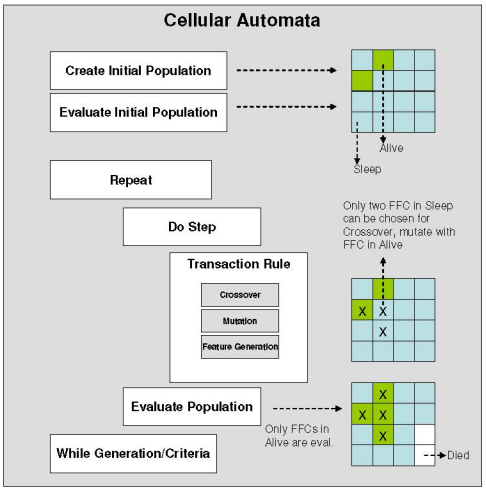

Fig.2 FCCA Automata

Transaction rules and process of building the features

Transaction rule must be defined in a way to enforce the searching process that should lead to better abilities of FCC to improve classification accuracy of classifier. Successful cells should therefore be rewarded, but on the other hand cells with not so good operators shouldn't be punished too much at once, for preserving the diversity of the CA. Therefore three different possible states of FFC cells in CA were defined:

- FCC is alive, if operator in a cell can improve accuracy of CAs classificator
- FCC fall in sleep mode whenever cannot improve the accuracy of classifier and its energy is already minimal.
- FCC dies, if nothing happened during its sleeping for some steps.

In each generation of CA, FCCs cells are then treated by transaction rule. They are selected by the transaction rule and applied by crossover or mutate operators. However they can be in crossover only within its neighborhood which satisfied transaction rule conditions. To guide the search process to the better constructed features, three important conditions were defined. First, we try to crossover operators of better FCC with their neighbors only if there are in a state of sleeping. Second, if there are more FCC cells sleeping, then we are trying to preserve CA dynamics by randomly choosing of only two operators of FFCs in the neighborhood what is adopted from famous Game Of Live [9]. However with such a limitation we are trying to preserve its dynamics and diversity. Finally, the third condition is that in the evaluation phase only living cells are evaluated that may lead to significant decrease in the number of computations as is explained below.

With transaction rule (Fig 3) dependable of specific neighborhood state and encountered conditions, the new cell's state is then calculated. Because of different possible state of FCC cells and described transaction rule (Fig 3) we are building better synthetic features from transition to transition in FCCA, whereby we are applying genetic operators only on maximal two neighbor's cells in sleeping mode and where only live cells are evaluated.

FFC CELL DEFINITION
State: *Alive, Sleep, Died*
Operator: *GPF*
Energy: *E*
Max steps of sleeping: *MaxSleep*

TRANSACTION RULE
IF cell is in ***Alive*** mode **THEN**
  CLASSIFY
  IF a cell with new *feature GPF* improves classificator C accuracy on *SO* **THEN**
    Increase the energy *E* of the cell
  ELSE
    Decrease the energy *E* of the cell
    IF cell energy *E* < 0 **THEN** put cell in *Sleep* state **ENDIF**
  ENDIF
ENDIF

IF cell is in the ***Sleep*** state **THEN**
  IF cell has less or equal than 3 neighbours in state *Alive* **THEN**
    Calculate probability for crossing *CP*
    IF *CP<Threshold* **THEN**
      Cross Cell(*GPF*) with **Best**(*Alive*(*GPF*))
    ELSE
      Calculate probability for mutate *MP*
      IF *MP< Threshold* **THEN**
        mutate Cell(*GPF*)
      ELSE
        Copy Best(*Alive*(*GPF*)) to this Cell
      ENDIF
    ENDIF
    Put Cell in state *Alive*
  ELSE
    SleepCount = *SleepCount* + *1*
    IF *SleepCount* < *MaxStep* **THEN** put cell in state *Die* **ENDIF**
  ENDIF
ENDIF

Fig.3 Transaction rule of FCCA

In general this last conditions of transaction rule has showed up that less classifying tasks (training of classifier, respectively) by searching process is used than by classical GA approach. The unpredictability of behavior of CA, different states of cells and selection of two in the neighborhood showed that in each generation we verify fewer individuals (FCC is individual for CA, respectively) than by GA. In GA each individual is verified by a classifier in each generation. In Fig. 3 the transaction rule of FFCA is presented. Described approach of Feature Constructing Cellular Automata was implemented in Java programming language within the support of Yale, a free open-source environment for KDD and machine learning [10].

V. EMPIRICAL RESULTS

Evaluation methodology

To test the presented idea of the CA for feature construction we implemented simple cellular automata and used simple *if-then* rules in the cells within transaction rule as is presented on Fig 2 and 3.

We have synthetically generated eight datasets (varying the number of features) to verify the usability of CA model for FC. Each dataset has been constructed by a function F(p):

$$F(p) = a_j^{\,3} + a_{j+1}^{\,2} + a_{j+2}$$

where,

$$\begin{cases} n < k; j = 1 \\ n > k; j = \left\{1, \downarrow \frac{n}{k}, \ldots \right. \end{cases} \qquad (1)$$

where,

$n = number\ of\ attributes$

$k = redudance\ faktor$

$$\begin{cases} \sum_{i=1}^{i=n-1}; & a_i = Random \\ i = n; & F(p) \end{cases}$$

For purpose of the analysis we also made put some restrictions to ease the evaluation methodology at the beginning. We selected linear regression classifier as a learner, and tried to improve the classification accuracy with generating only one new additional feature using only operator of multiplying. And the end, because of learning of concept, which fuzziness growth with randomness in additional attributes by larger datasets, we added some redundancy. When FCCA algorithm has shown success with the synthetically generated datasets on polynomial function (1), we tested the CA approach on four well-known datasets from UCI repository.

Synthetically generated datasets

The following datasets were generated by (1) for evaluating purposes:

TABLE 1: SYNTHETICALLY GENERATED DATASETS

Dataset name	Num. of attributes /population	Redundancy
Set 1	5 / 9	No
Set 2	10 / 25	No
Set 3	15 / 25	No
Set 4	20 / 49	No
Set 5	25 / 49	No
Set 6	50 / 64	Applied
Set 7	75 / 81	Applied
Set 8	100 / 81	Applied

For each datasets both approaches of feature construction (GA and CA) were applied on the same initial population. Their size is growing with the increasing the number of attributes because of diversity maintenance. All experiments were done with the generation criteria of 5 generations. We made ten runs on every dataset with both of the methods.

TABLE 2: CA RUNS EXPERIMENTS ON SET 2

Run	Constructed feature	Regression	Num. of C
1	*(*(att1,att1),att5)	104,23	59
2	*(*(att4,att1),*(att3,att6))	112,61	106
3	*(*(att1,att7),*(att1,att1))	96,47	107
4	*(att2,*(att10,att6))	112,35	50
5	*(att1,*(att2,att1))	99,88	114
6	*(*(att5,att1),*(att1,att9))	108,14	95
7	*(*(att3,att3),att1)	111,24	102
8	*(att8,*(att1,att2))	113,22	99
9	*(att1,*(att2,att1))	104,23	116
10	*(*(att6,att6),att1)	112,61	50
Avg		107,26	89,8
Original feature set regression		113,1	

In Table 2 there is one example of experiment on the second presented, where there were 25 individuals in initial. As we see from the table, constructed features on the dataset reduced the error from 113,10 to 107,26 in average. For such solution in average there was 90 computations needed by the CA.

TABLE 3: GA RUNS EXPERIMENTS ON SET 2

Run	Constructed feature	Regression	Num. of C
1	*(*(att1,att1),att5)	104,22	125
2	*(*(att4,att1),*(att3,att6))	112,61	125
3	*(*(att1,att7),*(att1,att1))	96,46	125
4	*(att2,*(att10,att6))	112,34	125
5	*(att1,*(att2,att1))	99,87	125
6	*(*(att5,att1),*(att1,att9))	108,13	125
7	*(*(att3,att3),att1)	111,23	125
8	*(*(att8,att1),*(att7,att1))	106,05	125
9	*(att4,att4)	137,61	125
10	*(*(att6,att6),att1)	112,23	125
Avg	*(*(att1,att1),att5)	104,22	125
Original feature set regression		113,1	

Constructed features from the table 3 were generated on the same dataset with same initial population by GA approach. They improves the regression factor from 113,1 to 104,22 in average. From the both tables we can see that both approaches found similar syntactical features. However for such solutions by GA approach 125 computations were needed, because whole population was evaluated during every evaluation step.

In Fig. 4 average constant of regression factor by original set, GA approach and CA approach on all synthetically datasets are represented.

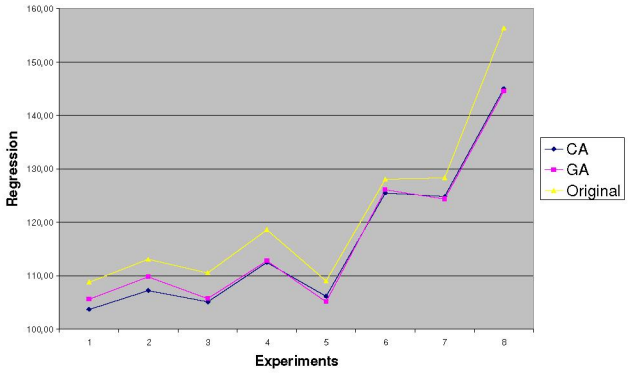

Fig.4 Comparison of accuracies for CA and on the second dataset GA averaged over 10 runs

It has showed up that the accuracies of improved learner by both approaches where similar. We can see that some times the CA behaves little better. But the main point is obviously. We can clearly see that both approaches found solutions, which improve learner accuracy (both approaches are under regression of original dataset, respectively).

In Fig. 5 the average computation of CA and GA approach are shown normalized by initial size of population of experiment (Table 1)

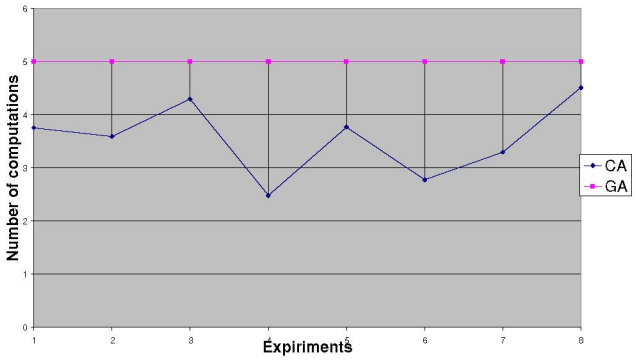

Fig.5 Comparison of average number of computation CA/GA

We can see that CA approach because of the differences by the transition rule needs less computational power than GA. This is potentially interesting benefit from the CA that leaded us to perform further tests on the real datasets.

Test on real datasets

With experiments on real datasets, we try to improve classification ability of different C5.0 classifier. Some datasets from UCI machine learning repository were used for this purpose. We used the parameters from the experiments on synthetically dataset; therefore our size of initial population and number of generations was defined by previous experiences in different size of attributes in synthetically datasets. In Table 4 real dataset for the evaluation purposes are presented.

TABLE 4: SYNTHETICALLY GENERATED DATASETS

Dataset name	Number of attributes	Population /Generation
wdbc	31	81 / 50
wpbc	32	81 / 50
iris	4	25 / 50
Pima	8	25 / 50

In the table 5 average results of accuracies with additional generated feature by both approaches are shown.

TABLE 5: RESULTS ON SYNTHETICALLY GENERATED DATASETS

Dataset name	CA accuracy	GA accuracy	CA comp.	GA comp.	Original accuracy
wdbc	**0,9430**	**0,9431**	325,1	405	0,9420
wpbc	**0,7235**	**0,7240**	350,2	405	0,7123
iris	**0,9567**	**0,9560**	94,90	125,00	0,9533
pima	**0,7287**	**0,7287**	102,50	125,00	0,7213

It can be seen from the table that the improvements of the learner could be less significant on real datasets than on synthetically generated. However this can not be compared directly because of different learner in both cases and distribution of the data. We can see also that with the real dataset the CA approach needed fewer computations as GA because of its structure.

As we can see from the results above, the both approaches were successful by constructing synthetically features. We can also notice that CA approach by our experiments showed up little better accuracies in average than GA approach. However this is only preliminary study therefore additional experiments for prove the significance should be needed. Because all experiments were limited by defined number of generation, there were some experiments that did not provide solutions. The reason lies in the initial population which is very important and depends from the size of attributes and distribution of the data and their concepts. Although so, there were not a lot of unsuccessful experiments, and they were equally distributed between both approaches. Therefore further analysis will be done especially by parameters of transaction rule, which have influence on the dynamics of CA.
Considering preliminary results, we can therefore say that CA is appropriate model for feature construction which brings some of new possibilities that can be used by such a task.

VI. FINAL REMARKS AND CONLCUSION

In this paper we presented a novel model of constructing features design by the model of Cellular Automata. We presented FCCA approach and compare it with well known optimization method of GA. From the preliminary results we can say that conclusions are quite promising. We showed by the examples that CA and GA approaches both found appropriate solutions during the search process for improving the classifier (learner) and that in average the accuracy of CA was slightly better than by GA, however we can not generalize this fact yet. For this purposes further analysis should be done. More important, we showed that because of the design of the transition rule, CA uses less computational power than GA. This conclusion brings us fresh motivation to research the conditions and kind of problems where this advantage could bring positive effect. Because of design of transaction rule, it is quite obvious that the problems with large initial populations are appropriate for such intention; there is no significant difference within small population. But there will be further analysis and research needed to confirm this better.

Additionally, there is another interesting point that CA model brings with by the task of constructing features. With comparison to GA, the research area of Cellular Automata brings us innovative and interesting knowledge in the field of research of complexity and dynamics of CA [6]. This knowledge and experiences may be used for meta-guidance of the learning process in generally; therefore this is interesting area where we are planning to do further analysis on behavior of the described FCCA. This also argues that it is reasonable to continue on research work with using CA model for such purposes.

Not at last, there are many of ways and ideas to define new transaction rule for feature construction in the future, one of them could be searching for more features during the same transaction rule, or even more, from this study we believe that is possible to define the transaction rule also for feature selection or extraction tasks. More generally Cellular Automata could be also appropriate mechanism for genetic algorithms, where we could use proposed transaction rules, which are defining selection and GA operator's behavior in

a different way than by classical approach. As we showed some less computation power is needed in this way as showed in this preliminary study of using cellular automata for feature construction task.

VII. REFERENCES

[1] Guyon I, Elisseeff A: An Introduction to Variable and Feature Selection, Journal of Machine Learning Research 3 (2003) 1157-1182

[2] D.E. Goldberg: Genetic Algorithms in Search, Optimization, and Machine Learning, Addison Wesley, Reading, MA, 1989

[3] Koza, J. R., Genetic Programming: On the Programming of Computers by Means of Natural Selection. Cambridge, MA, MIT Press, 1992.

[4] Muharram M., Smith G.D.: Evolutionary Constructive Induction, November 2005 (Vol. 17, No. 11) pp. 1518-1528

[5] Ritthoff O., Klinkenberg R., Fischer S, Mierswa I: A hybrid approach to feature selection and generation using an evolutionary algorithm, John A. Bullinaria, editor, Proceedings of the 2002.

[6] Flocchini, P., Geurts, F., Santoro, N.: Compositional experimental analysis of cellular automata: Attraction properties and logic disjunction. Technical Report TR-96-31, School of Computer Science, Carleton University (1996)

[7] C. Svozil: Constructive Chaos and generation of algorithmic complexity by Cellular automata, Theory & Experiment, CNLS Los Alamos, September 9-12, 1989

[8] Rendell P.: Turing universality of the game of life, Collision-based computing, Springer-Verlag, 2001, 513-539

[9] M. Gardner: Mathematical games: The fantastic combinations of John Conway's new solitaire game "life", Scientific American 223, October, 1970, 120-123

[10] Mierswa I., Wurst M., Klinkenberg R., Scholz M.: YALE: Rapid Proto-typing for Complex Data Mining Tasks, in Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-06), 2006.