USING CELLULAR AUTOMATA TO PREDICT RELIABILITY OF MODULES

Mitja Lenič*, Petra Povalej*, Peter Kokol*, Ana Isabel Cardoso**

*University of Maribor, FERI, Smetanova 17
2000 Maribor, Slovenia
mitja.lenic@uni-mb.si

** University of Madeira, Campus Universitario da Penteada,
9000-390 Funchal, Portugal

ABSTRACT

Software development is complex process that incorporates many different aspects and is therefore hard to control. For that reason many different metrics are used to capture different aspects of software artefact. On other hand, data-mining from such database should also contain different aspects and try to express nature of software development process. For that reason we use an ensemble of classifier that have the ability to boost classification accuracy comparing to single classifiers and are a commonly used method in the field of machine learning. However in some cases ensemble construction algorithms do not improve the classification accuracy. Mostly ensembles are constructed using specific machine learning method or a combination of methods, the drawback being that the combination of methods or selection of the appropriate method for a specific problem must be made by the user. To overcome this problem we invented a novel approach where ensemble of classifiers is constructed by a self-organizing system applying cellular automata (CA).

KEY WORDS

Software development, cellular automata, classification, ensembles

1. Introduction

Software development is a complex and complicated process in which software faults are inserted into the code by mistakes during the development process or maintenance. In development process usually different metrics are used to measure different aspects of software artefacts and determine next steps of development. By monitoring development process many valuable data is gathered that can be used in future project to determine critical artefact and improve quality of produced software. Rules in gathered data are expected to be complex, because the nature of software development process that incorporates many aspects of software artefacts like for example reliability. Of course that statement is also true for a large variety of problems arising in engineering, physics, chemistry and economics can be formulated as classification problems [1, 2, 8]. Although classification represents only a small part of the machine learning field, there are a lot of methods available. To develop a general method that at the same time exposes different aspects of a problem and produces “optimal” solution is impossible, due to the so called “no free lunch” theorem [6]. Standard local optimization technique is to use multi-starts, by trying different starting points and running the processes of classification independently from each other and selecting the best result from all trials.

Moreover ensembles [4] of classifiers have the ability to boost classification accuracy comparing to single classifiers. However the single classifiers like ensembles of classifiers can often find solutions near local optima, on the other hand they are not so successful by global exploration of search space. Therefore the popular methods for global exploration of the search space are used like simulated annealing, genetic algorithms and lately also theory of self-organization systems that could gain some diversity leading to possible better solutions.

During the last few years, the scientific area of self-organizing systems became very popular. The field seeks general rules about the growth and evolution of systemic structure, the forms it might take, and finally methods that predict the future organization that will result from changes made to the underlying components. A lot of work has been done using Cellular Automata (CA), Boolean Networks, with Alife, Genetic Algorithms, Neural Networks and similar techniques. In this paper we propose a new method for nonlinear ensembles of classifiers that exploit the ability of self-organization applying CA in search for an optimal solution. The aim is to combine single classifiers in an ensemble represented by a CA. We show how improved accuracy can be obtained by interaction of neighbourhood cells containing different kind of classifiers.

The paper is organized as follows. In Sec. 2 we introduce in short the theory of CA. In Sec. 3 we discuss how to design a CA as classifier. In Sec. 4 we present experiment setup and in Sec. 5 we discuss results.

2. Cellular automata

CA are massively parallel systems [3] obtained by composition of myriads of simple agents interacting locally, i.e. with their closest neighbours. In spite of their simplicity, the dynamics of CA is potentially very rich, and ranges from attracting stable configurations to spatio-temporal chaotic features and pseudo-random generation abilities. Those abilities enable the diversity that can possibly overcome local optima by solving engineering problems. Moreover, from the computational viewpoint, they are universal, could say, as powerful as Turing machines and, thus, classical Von Neumann architectures. These structural and dynamical features make them very powerful: fast CA-based algorithms are developed to solve engineering problems in cryptography and microelectronics for instance, and theoretical CA-based models are built in ecology, biology, physics and image-processing.

On the other hand, these powerful features make CA difficult to analyze, almost all long-term behavioural properties of dynamical systems, and cellular automata in
particular, are unpredictable. However in this paper the aim was not to analyze the process of CA rather to use it for superior classification tasks.

3. Cellular automata as a classifier

Our novel idea is to exploit benefits of self-organization abilities of a cellular automata in classification and its possible use in the extraction of new knowledge. Our first task was to define the basic elements (content of the cells) and transaction rules that would result in a learning ability of cellular automata and representing gained classification knowledge in its structure. Most obvious choice is to define a cell in the classification cellular automata (CCA) as classifier.

In a simplified view we can look at a cellular automata as on an ensemble of classifiers. In each cell we have to build a classifier that has different and eventually better abilities than already produced ones. Contrary if all cells would have the same classifier there would be no diversity and it would be quite unreasonable to use any transaction rules on such automata, because all cells would classify in the same way. Thus we must ensure the most possible diversity of classifiers used in different automata’s cells to benefit from the idea of CCA.

3.1. Initializing automata

The most desirable initial situation in a CCA is having a different classifier in each of its cells. We can ensure that by different techniques. The most obvious one is to use a different kind of classifier for each cell. That can be problematic, because the number of methods are limited, especially if the CCA has a lot of cells. But even if there is only one method available we can still use so called fine-tuning. Namely, almost all machine-learning methods used for classification have multiple parameters that affect classification. By changing those parameters different classifiers can be obtained. Parameters can be randomly selected or defined by using for example evolutionary algorithms for determining most appropriate parameters. Another possibility is by changing expected probability distributions of input samples, which may result in different classifiers, even by using same machine learning method with same parameters. Still another approach is to feature reduction/selection. That technique is recommended when a lot of features are presented.

3.2. Transaction rules

Transaction rules must be defined in such way to enforce the learning process that should lead to generalization abilities of the CCA. Successful cells should therefore be rewarded, but on the other hand cells with not so good classifiers shouldn’t be punished to much, to preserve the diversity of the CCA. Each classifier in a cell can classify a single learning sample or can generate unknown tag to signal that it cannot assign any class to current sample. From the transaction rules point of view a classification result can have three outcomes: same class as the learning sample, different class or cannot classify. Cannot classify tag plays important role when combining partial classifiers that are not defined on the whole learning set.

In that case CCA becomes a classifier system i.e. when using IF THEN rules in CCAs cells. Therefore cell with unknown classification for current learning sample should be treated differently as misclassification.

Beside the cells classification ability also the neighbourhood ability plays a very important role in the self-organization ability of a CCA. Transaction rules depend on the specific neighbourhood state to calculate new cell state. In general we want to group classifiers that support similar hypothesis and have therefore similar classification on learning samples. Even if sample is wrongly classified, the neighbourhood can support a cell classifier by preventing it’s elimination form automata. With that transaction rule we encourage creation of decision centres for a specific class and in this way we can overcome the problem of noisy learning samples – another CCA’s advantage.

3.3. Ensuring diversity

It is very important to ensure CCA classifier diversity. If all of classifiers classify in the same manner induction algorithm and self organizing features of CA would not take place. One way to ensure diversity is by changing expected distribution probability of learning samples in the same manner as in boosting algorithms. This assures that hard to classify instances become more attention and different aspects are exposed.

Another possibility is to use different induction methods for classifier construction. For example decision trees with different purity measure can be used to obtain different structures [10]. Like all ensemble methods our approach uses loose coupling, what enables to use different knowledge representation. Therefore we can use neural networks, SVMs, decision trees, etc. in the same ensemble eg. cellular automata to obtain diverse individual classifier cells.

Evolutionary approaches to construction of classifiers can present good source of diverse classifiers. Classifiers can be taken from different steps of population in evolution. It is not important to take only good classifiers, because not so good ones should not be included in final automata. That must be assured careful selection of transaction rules.

The third possibility of generating classifier is random creation. As in all ensemble method it has to be assured that all classifiers produce at least weak hypothesis (error is less than 0.5). If that condition is not satisfied ensemble is not going to improve classification accuracy.

3.4. The learning algorithm

Once the CCAs classifier diversity is ensured, transaction rules are continuously applied. Beside its classifier information, each cell contains also statistical information about its successfullness which in a form of the cells energy. Transaction rules can increase or decrease energy level dependent on the successfullness of classification and cells neighbours. If energy drops below zero the cell is terminated. New cell can be created dependent on learning algorithm parameters with its initial energy state and a classifier used from pool or newly generated classifier dependent on neighbourhood classification abilities and modifying expected probability distribution and/or used features. When using second approach we create local boosting algorithm to assure that misclassified samples are correctly classified by the new cell. Of course if cell is too different from the neighbourhood it will decently die out and return to the pool.

The learning of a CCA is done incrementally by supplying samples from the learning set. Transaction rules are executed first on the whole CCA with a single sample and then continued with the next until the whole problem is learned by using all samples - that is a similar technique than used in neural networks [5, 7].
Transaction rules do not directly imply learning, but the iteration of those rules create the self-organizing ability. Of course this ability depends on classifiers used in CCA cells, and its geometry. Stopping criteria can be determined by defining fixed number of iterations or by monitoring accuracy.

3.5. Inference algorithm

Inference algorithm differs from learning algorithm, because it does not use self-organization. Simplest way to produce single classification would be to use the majority of the cell voting. On other hand some votes can be very weak from the transaction rule point of view. If transaction rules are applied using only neighbourhood majority vote as sample class those weak cells can be eliminated. After several irritations of transaction rules only cells with strong structural support survive on which majority voting can be executed. Weighted voting algorithm can be used by exploiting energy state of classifier in a cell.

4. Experiment setup and results

To test the usability of proposed method we want to predict potentially dangerous modules. First the modules have been identified by reliability either as OK or DANGEROUS. By applying the model developed by Pighin [9]. The purpose of this model is to use the software complexity metrics to define complexity risk thresholds. The modules are divided into two classes – bellow and above some fault threshold value (5 faults in our case).

A set of 168 attributes, containing various software complexity measures, have been determined for each software module and the Alpha - complexity based metrics [12], has been calculated for each software module. From all 217 modules 2/3 have been randomly selected for the learning set, and the remaining 1/3 modules has been selected for the testing set. The goal was to predict potentially dangerous modules and in this manner designing reliable software. Study was performed on software from real world medical environment. We tested a medical software system consisting of 217 modules representing more than 20000 lines of code.

To evaluate our hypothesis, that self organisation of cellular automata can improve ensemble classification accuracy mainly on exploiting geometric relations and transaction rules we made following setup.

Diversity of classifiers was achieved with use of AdaBoost [11] algorithm based on ID3 decision tree induction with 10 individual classifiers. Ensemble of classifier was constructed and evaluated. Then individual classifiers were from ensemble and used as source of diverse classifiers for cellular automata. If we can achieve better classification accuracy that can be directly seen as impact of voting/connection of cellular automata.

For transaction rules we used the following rules:

- If a cell for any reason cannot classify learning sample leave the energy state of the cell unchanged;
- For each iteration all cells use one point of energy (to live). If energy level drops below zero cell is terminated.
- In our experiment cellular automata had 10x10 cells with Taurus layout and impact radius 3 with Euclidian distance influence reduction.

For evaluation criteria we used following measures were used:

\[
\text{accuracy} = \frac{\text{num. of correctly classified objects}}{\text{num. of all objects}}
\]

\[
\text{accuracy}_c = \frac{\text{num. of correctly classified objects in class } c}{\text{num. of all objects in class } c}
\]

\[
\text{average class accuracy} = \frac{\sum \text{accuracy}_c}{\text{num. of classes}}
\]

In table 1 accuracy A(T) and average class accuracy ACC(T) on test set of Fault database are presented. In all boosting and non pruned methods accuracy and average class accuracy on learning set were at 100%.

<table>
<thead>
<tr>
<th>Method</th>
<th>A(T)</th>
<th>ACC(T)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID3</td>
<td>83.56%</td>
<td>82.45%</td>
</tr>
<tr>
<td>Pruned ID3</td>
<td>82.19%</td>
<td>67.50%</td>
</tr>
<tr>
<td>AdaBoost ID3</td>
<td>90.41%</td>
<td>87.17%</td>
</tr>
<tr>
<td>CCA ensemble</td>
<td>93.15%</td>
<td>89.05%</td>
</tr>
<tr>
<td>Gini</td>
<td>79.45%</td>
<td>71.84%</td>
</tr>
<tr>
<td>AdaBoost Gini</td>
<td>87.67%</td>
<td>82.17%</td>
</tr>
</tbody>
</table>

Table 1: Results on Fault database

It can be clearly observed, that boosting improves both A(T) and ACC(T) compared to single method. On other hand CCA ensemble outperformed all other method. It is even better than AdaBoost ID3 from which individual classifiers were taken. Because both method use same individual classifier base we can conclude, that improvement of classification accuracy results from only organisation and transaction rules of cellular automata.

It was also interesting to observe final layout of CCA ensemble (Figure 1). Not all classifiers form AdaBoost ID3 were used in final automata.

First impression of resulting CCA is that it is relatively sparse. But it should be kept in mind that affect radius of single classifier is set to 3. From presented result we can say that the ability of self-organization, is very promising and that more elaborate CCAs can produce even better results.

CCA also inherits some drawbacks of ensembles. Produced result, although in symbolic form is complex and usually involves more attributes than simpler but less accurate classifiers. But on the other hand classification

![Table 1: Results on Fault database](image)

![Figure 1: Layout of CCA ensemble for Fault database](image)
with CCA can be even cheaper, as shown in our experiment, because not all classifiers are required in final CCA. From computation point of view CCA approach uses additional power to apply transaction rules which can be expensive in the learning process, but its self organizing feature can result in better classification, that can also mean less costs.

As many other method, our approach requires some additional parameters, like size of automata, transaction rules parameters (like radius) which can affect final result. That can be either seen as drawback or good feature to obtain various results.

5. Concluding remarks and future work

The aim of this paper was to present the use of self-organization ability of a cellular automata in classification. In this way we combined classifiers in an ensemble that is eventually better than individual classifiers or traditional ensembles generated by boosting or bagging methods. We empirically showed that CCA can give superior results as more complex classifiers or ensembles of those classifiers and those improvements can be clearly seen as direct result of using cellular automata for voting and learning algorithm and not its individual classifiers. This is very promising and taking into account the simplicity of the tested CCA we can foresee a lot of room for improvements. The advantages of the resulting self-organizing structure of cells in CCA is the problem independence, robustness to noise and no need for the user input. The important research direction in the future are to analyze the resulting self-organized structure, impact of transaction rules on classification accuracy, introduction other social aspect for cells survival.

References