Proceedings of the 6th Győr Symposium and 3rd Hungarian-Polish
and 1st Hungarian-Romanian Joint Conference on Computational Intelligence

Genetic Programming Based Rule Classifier for Data Streams with Heterogeneous Features

K. Ząbkiewicz[1,2], S. Łukasik[1,3], P. Kulczycki[1,3]

[1]Polish Academy of Sciences, Systems Research Institute
Newelska 6, 01-447 Warsaw, Poland
E-mail: {zabkie,slukasik,kulpi}@ibspan.waw.pl

[2]University of Bialystok, Faculty of Economics and Informatics in Vilnius
Kalvariju 135, LT-08221 Vilnius, Lithuania
E-mail: K.Zabkiewicz@uwb.edu.pl

[3]Cracow University of Technology, Department of Automatic Control and IT
Warszawska 24, 31-155 Krakow, Poland
E-mail: {szymonl,kulczycki}@pk.edu.pl

Abstract: Classification of data streams is currently a very important task. Datasets characterized by constant influx of data are predominantly massive and often have various types of features. Even more challenging is to classify evolving streams. Various approaches have been proposed to deal with this problem. In this paper we will present a new method based on mixture of genetic programming and rule-based classifier designed for that task. The motivation for choosing this approach is provided by the fact that tree-like structures are easy to understand and interpret. What is more continuous generation and optimization of rules reminds real-life scenario, where changing conditions cause natural selection and adaptation. The approach was preliminarily tested on data sets that are changing in time and have heterogeneous features.

Keywords: data streams, rule-based classifier, genetic programming, heterogeneous features, concept drift

1. Introduction

Machine learning became nowadays one of the important, though not always obvious, part of our life. A variety of methods using this paradigm are employed, e.g. in weather forecasting, financial markets analysis, intrusion detection, medical images segmentation [15]. All of those problems are inherently associated with the analysis of data streams, that will be the main interest of this paper. We define data stream as a sequence of data items that can be read once or by making small number of passes by an algorithm, in the prescribed sequence. It is also assumed that computing and storage

capabilities for this type of data are limited [1]. Latest predictions leave no illusions that amounts of data streams will increase in future, so this topic is very actual [16].

Classification constitutes a very important part of machine learning process. This technique belongs to supervised learning category, i.e. learning with data that is labeled. It uses various algorithms to build a model based on known examples of one or more classes labeled with name of the class. The model is a function that maps properties (attributes, features) of examples to the appropriate class labels. When new example, whose class is unknown, arrives the learned model gives it a label based on properties of this example [2]. Among popular classification algorithms one can name, e.g. decision trees, k-nearest neighbors, support vector machines, rule-based learners, neural networks, Naive Bayes, etc. [15].

Data sets can be described as single-type feature(homogeneous) sets and mixed-type feature (heterogeneous) sets. Types of features can belong to one of the following categories:

a) numerical feature,

b) nominal feature,

c) ordinal feature.

Numerical features are simply integers or real (floating point) numbers. Nominal features are used to describe objects' semantical properties. They are mostly expressed by words e.g.: *small, large, sunny, rainy, true, false*, etc. Ordinal features describe some order or scale mostly expressed by numbers. We must note that nominal and ordinal attribute values cannot be quantitatively compared among themselves using inequality operators, as in case of numerical features.

Building classifier for data streams is even more challenging than classifying static data. Dynamics of data stream brings up phenomenon called concept drift. In short it constitutes changes of statistical properties of data set over time.

Finally, it is important to note that one of the important requirements for classifiers is interpretability and understandability of results – especially when they are to be used by non-specialists in the machine learning field. Tree-like and rule-based classifiers are known to fulfill these conditions.

The main goal of this paper is to demonstrate a new method for classifying evolving data streams with diverse types of features. This paper is organized as follows. The next section presents the main subject of this contribution and related work in this area. Section 3 describes proposed method. Section 4 contains brief presentation of preliminary results obtained using our algorithm. Lastly, Section 5 concludes this paper and discusses areas of further studies.

2. Online Classification

The problem of online classification of data streams is very significant. The main reason for that is a common need to analyze data and make decisions in real time. Gaber et al. [7] provide examples of using this technique in solving real world challenges such as:

biosensor measurements' analysis around a city for security reasons, analysis of web logs and web clickstreams, real-time analysis of data streams generated from stock markets, analysis of simulation results and on-board sensor readings in scientific applications.

2.1. Classification under concept drift

Dynamic streams have tendency to evolve, i.e. their data distribution changes over the time. This evolution is also called concept drift. This term was used for the first time in Schlimmer and Granger work [8]. The main consequence of this occurrence is that the classifier will be outdated after some period of time and performance of classification will eventually decrease.

More formal definition of concept drift is given in Žliobaitės paper [8]. Let $X \in R^n$ be an instance in n-dimensional feature space. $C = \{c_i\}$, where $c_1, c_2, \ldots, c_k$ is set of class labels. The optimal classifier used to classify, that is to perform an assignment $X \rightarrow c_i$, is completely determined by the prior probabilities for the classes $P(c_i)$ and the class-conditional probability density function $p(X|c_i), i = 1 \ldots, k$. According to Bayesian decision theory the classification decision for instance X at equal costs of mistake is made on a basis of maximal a posteriori probability defined as

$$p(c_i|X) = \frac{P(c_i)p(X|c_i)}{p(X)}$$

Concept drift in essence occurs when:

a) class priors $P(c_i)$ change over time,

b) the distributions of one or several classes $p(X|c)$change,

c) the posterior distributions of the class memberships $p(c|X)$might change.

Author of [8] points out four concept drift types: sudden, gradual, incremental, and reoccurring. Later on he also provides 4 tasks that concept drift learner should solve:

- future assumption – the assumption about future data source,
- change type – the possible change patterns,
- learner adaptivity – the mechanisms which make the learner adaptive, chosen using assumed change type,
- model selection – choosing particular parameter set for the selected learner at every time step (e.g. the weights for an ensemble member, the window size, etc.)

Learners which can deal with concept drift can be generally divided into two groups: evolving and the ones with triggers. Rule-based classifier are members of the second

group. It is characterized by ability to detect changes, determine training window size and instance selection.

Recent overview on classifier adaptation – when concept drift occurs – is given by Gama et al. [9]. Authors formulate main requirements for classification models in changing environments: detect the concept drift (and adapt if needed) as soon as possible, distinguish drifts from noise and be adaptive to changes, but robust to noise. Finally it is required to operate in time intervals smaller than example arrival time and use not more than fixed amount of memory for any storage.

They also provide basic parts of online adaptive learning procedure:

- **Prediction**: after getting new example, a prediction is made using current model.
- **Diagnosis**: after some time true label for new example is received and the estimation of the loss can be made
- **Updating**: new example and its true label can be used for model update.

Due to the requirement of those three components a generic schema for an online adaptive learning algorithm was proposed in [9]. It consists of four modules: Memory, Learning, Change Detection, Loss Estimation. Rule-based classifiers can be implemented using first three of them.

2.2. Rule-based classifiers for non-stationary data streams

A decision rule is defined as a logic predicate taking the form IF antecedent THEN class label. Antecedent is a conjunction of conditions written in a form *Attribute Operator Value*, where operator means relation between certain attribute and value of its domain. The big advantage of rule sets is that they are not hierarchically structured, so descriptions of the concept can be updated or removed when becoming outdated without hardly affecting the learning efficiency [13].

Although learning under concept drift is now very popular, the approaches based on rule sets are not so widespread. Deckert in her paper [11] made detailed review of four already known rule-based methods for classifying non-stationary data streams: Floating Rough Approximation (FLORA, 1996), AQ11-Partial Memory+Window Adjustment Heuristic (AQ11-PM+WAH, 2003), Fast and Adaptive Classifier by Incremental Learning (FACIL, 2006), Very Fast Decision Rules (VFDR, 2011). In addition to that new approach called Rule-based Incremental Learner (RILL, 2014) was recently proposed [12].

2.3. Genetic Programming in Data Stream Classification

Genetic Programming (GP) is an interesting evolutionary method for solving real life problems. First described by Koza [4] it was applied in solving different problems such as: optimal control, planning, sequence induction, symbolic regression, automatic programming, classification, etc. In essence it brings genetic algorithms to a higher level of abstraction. Instead of chromosomes encoded as binary strings "programs"

based on tree-like structures are proposed. The main principles used in genetic algorithms are inherited, but are appropriately modified due to new structure of data. More information on Genetic Programming is given in [5,6].

Espejo et al. [3] provide detailed overview of using GP to solve classification problems. One of paper's subsections describe its use in generating rule classifiers. There is only a short note on using Genetic Programming in data stream classification albeit the problem is still open for further research. Authors of [3] point out advantages of using this technique however such as flexibility, ability to perform explicit and automatic feature selection and extraction, interpretability and comprehensibility.

2.4. Our contribution

In this work we propose new method based on practices described in previous subsections. It will consist of the following components: time window, data samples kept in memory, rule set that is generated and optimized using Genetic Programming.

Proposed solution is designed to work on data streams with mixed-type features.

3. Rule-based classifier optimized by using Genetic Programming

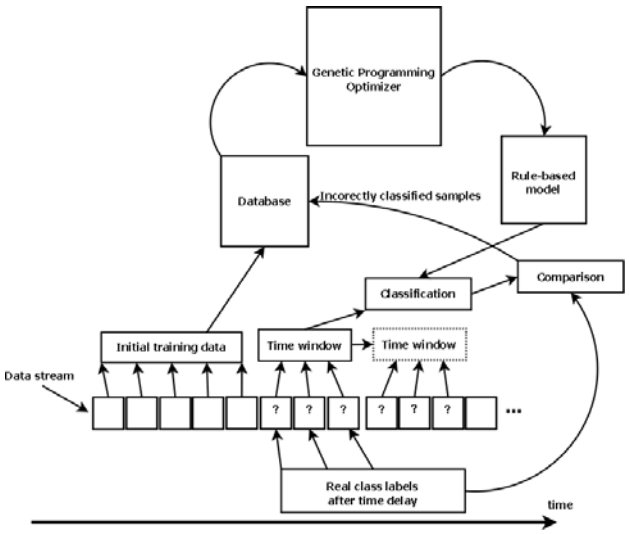

Figure 1. Illustration of the algorithm

3.1. Model of proposed approach

In this subsection we will show formulated model of classifier for dynamic data streams with heterogeneous features. The main concept of the model is shown on Figure 1.

Now let us have detailed look on the presented approach. First some basic knowledge is needed, so first *d* data samples are kept in memory. We shall call it a database which

will be updated during the process of online classification. Data window - having size *l* - is a container for appearing data samples. Elements in time windows are classified. If object is incorrectly classified then it is added to database (at the same time its class identifier is set to correct one). Of course limited amount of memory is available so the mechanism of forgetting has to be implemented. We assume that each class has balanced number of records in the database. If this number is exceeded current database records of class that violate this postulation are replaced with new ones.

First step of our algorithm is to fill the database with incoming samples. We assume at that time that all classes are being exemplified - we select representative subset of initial data having examples of different classes. Subsequent part of data will be used for classification. Before using classifier on incoming data we must build initial rule set. As for any successive step of the algorithm it constructed using Genetic Programming – which was indicated on Figure 1.

The main task of Genetic Programming is to generate tree structures describing computer program. In this case computer program will be single classification rule. It is a member of entire population of rules, therefore number of rules is exactly equal to the number of members of the population S. Classification rules are written in conjunctive normal form. The disjunctions are the leaves of the tree. They consist of a feature, relation operator(<, >, <=, >=, ~=, =) and value. Conjunctions are the nodes of the tree. If we consider nominal or ordinal features we can only use ~= and = operators. The example rule looks like this:

((x26~=0.378)||(x8==0.000))&&(((x11>=4.095)||(x29>=0.759)))->1

Before selecting members for crossover or mutation we must evaluate each one of them. To achieve this goal we use fitness function assessing classification rules which consists of true positive rate (TPR). In short it is a ratio of number of times when rule gave the correct answer (true positive, TP) to the number of times when rule had "fired" (P). This measure is also called sensitivity or recall and its expressed by the following equation:

$$fitness = TPR = \frac{TP}{P} = \frac{TP}{TP + FN},$$

where FN is number of times when rule had "fired" but gave wrong answer(also called false negative). Although this type of performance measure is used in binary classification, we can convert multiple class classification problem into two class by aggregating other classes into one (let us call it the “other” class). As a result we have rule that can classify example as expected class or as the “other” class.

Member selection for crossover or mutation – used to obtain new population – is performed using tournament selection. Process of generating new population stops when the assumed number of generations *K* is exceeded. When performing crossover of parents the connecting point of two subtrees is a conjunction operator. If both parent rules have only one feature then offspring will be generated by connecting both of them with conjunction. During mutation we can change values of leaf, i.e. operator and value. We also can generate new subtree, which will be connected to selected conjunction.

After generating initial rule set we can perform classification of new data. We are using static size time window to simulate time delay of getting true class values. After period of time symbolized by time window size classified examples are compared with true class values – it can represent a process of expert's evaluation. Examples, which are misclassified are added to database. We can omit putting correctly classified elements to database, because knowledge about them is stored in rule set. Because size of database is fixed, some examples will be overwritten or in other words forgotten. After these steps new data stream chunk can be classified. This in turn will trigger the process of updating database and optimizing rule set.

4. Experimental Results

For the purpose of experiments we implemented proposed model of classifier in Matlab. The tests were performed on KDD Cup 1999 dataset (10 percent subset used in competition) [14]. This dataset has multiple classes - 4 different attack types and 1 representing normal behavior. Distribution of classes is not uniform. Records belonging to one class of attack and to normal cover about 99 percent of whole data, the rest is divided between other classes of attacks.

Experiment was performed in such way. First subset of first 100000 data samples was chosen. New set was split into two parts: one for randomly selecting representative subset for initial database, another for classification. Initial rule set was generated from this database by using Genetic Programming. Number of generations K was set to 150, probability of crossover p_{cross} to 0.9, probability of mutation p_{mut} to 0.1. The tournament size was set to 3. We kept our database and window size constant. Initial database size d was set to 100. Maximum number of examples of each class is set to the number of members of the most popular class in the initial database, window size l is 25.

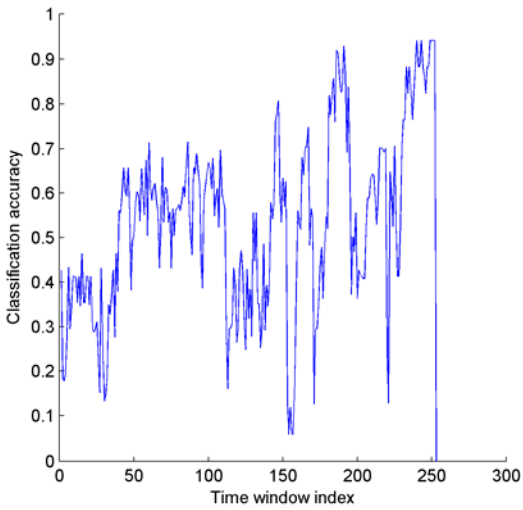

Figure 2. Average change of classification accuracy as time window moves forward

We performed 20 experiments. In each one we used the same initial database. We measured classification performance for each window slide. The results are presented in Figure 2

As we can see classification accuracy is fluctuating, but has tendency to improve. The places of deep decrease demonstrate the appearance of class which was poorly represented in the rule base. It is also worth to notice that such a fall, classification accuracy starts to grow very rapidly. This is caused by gradual rules reconstruction performed by Genetic Programming.

5. Summary

In this paper we proposed new model of classifier for data streams with heterogeneous features based on rule set generated and optimized using Genetic Programming. Experimental results are promising and have significant potential for further improvement. The dataset which was used in experiments has imbalanced classes, so one of the enhancements could be using misclassification costs. Another solution could be modification of Genetic Programming parameters, such as probability of mutation or crossover when potential concept drift occurs. For now we do not have formulated proper forgetting strategy. In future we plan to investigate it to keep optimal size of the database. Problem of window size and manner of its change will be under consideration as well. Finally we plan to compare performance of proposed classifier with other known methods based on rule sets and dealing with mixed-type features and concept drift.

Acknowledgement

This research was supported in part by PL-Grid Infrastructure.

References

[1] Henzinger, M., Raghavan, P., Rajagopalan, S.: *Computing on data streams* , Technical Note 1998-011, Digital Systems Research Center, Palo Alto, CA, May 1998
[2] Drummond, C.: *Classification*, in Encyclopedia of Machine Learning, Editors: Sammut, C., Webb, G.I., Springer Science+Business Media, pp. 168-171, 2011
[3] Espejo, P.G., Ventura, S., Herrera F.: *A Survey on Application of Genetic Programming to Classification*, IEEE Transactions on systems, man and cybernetics - Part C: Applications and reviews, Vol. 40, No 2, March 2010
[4] Koza, J.R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, 1st edn. Bradford Books, MIT Press , 1992
[5] Koza, J.R., Poli, R., Genetic Programming: *Search Methodologies. Introductory Tutorials in Optimization and Decision Support Techniques*, Chapter 6. Springer, 2014
[6] Poli, R., Langdon W.B., McPhee N.F.: *A Field Guide to Genetic Programming*, 2008
[7] Gaber, M.M., Zaslavsky, A., Krishnaswamy, S.: *Data Stream Mining*, in Data Mining and Knowledge Discovery Handbook, 2nd ed., Editors: Maimon, O., Rokach, L., Springer Science+Business Media, pp. 759-787, 2010

[8] Schlimmer, J., Granger, R.: *Incremental learning from noisy data*. Machine Learning 1(3), pp. 317-354, 1986.
[9] Žliobaitė, I., *Learning under Concept Drift: an Overview*. Technical Report, Faculty of Mathematics and Informatics, Vilnius University, Vilnius, Lithuania, 2009
[10] Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: *A Survey on Concept Drift Adaptation*, ACM Computing Surveys, Vol.1, No. 1, 2013
[11] Deckert, M.: *Incremental Rule-based Learners for Handling Concept Drift: An Overview*, Foundations of Computing and Decision Sciences, Vol. 38, No. 1, pp. 35-65, 2013
[12] Deckert, M., Stefanowski, J.: *RILL: Algorithm for Learning Rules from Streaming Data with Concept Drift*, in Foundations of Intelligent Systems, 21st International Symposium, ISMIS 2014, Roskilde, Denmark, June 25-27, 2014. Proceedings, Editors: Andreasen, T., Christiansen, H., Cubero, J.C., Raś, Z.W., Springer International Publishing, pp. 20-29, 2014
[13] Gama, J., Kosina, P.: *Learning decision rules from data streams*, in Proceedings of the Twenty-Second international joint conference on Artificial Intelligence IJCAI'11 - Volume Two, AAAI Press, pp. 1255-1260, 2011
[14] Bache, K., Lichman, M.:UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science,2013, https://archive.ics.uci.edu/ml/machine-learning-databases/kddcup99-mld/kddcup99.html
[15] Alpaydin, E.: *Introduction to Machine Learning, second edition*, MIT Press, 2010.
[16] Gantz, J., Reinsel, D*.: The Digital Universe In 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East* , Sponsored by EMC Corporation, 2012