2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

A First Attempt on Evolutionary Prototype Reduction for Nearest Neighbor One-Class Classification

Bartosz Krawczyk, Isaac Triguero, Salvador García, Michał Woźniak and Francisco Herrera

Abstract— **Evolutionary prototype reduction techniques are data preprocessing methods originally developed to enhance the nearest neighbor rule. They reduce the training data by selecting or generating representative examples of a given problem. These algorithms have been designed and widely analyzed in standard classification providing very competitive results. However, its application scope can be extended to many other specific domains, such as one-class classification, in which its way of working is very interesting in order to reduce computational complexity and sensitivity to noisy data. In this contribution, we perform a first study on the usefulness of evolutionary prototype reduction methods for one-class classification. To do so, we will focus on two recent evolutionary approaches that follow very different strategies: selection and generation of examples from the training data. Both alternatives provide a resulting preprocessed data set that will be used later by a nearest neighbor one-class classifier as its training data. The results achieved support that these data reduction techniques are suitable tools to improve the performance of the nearest neighbor one-class classification.**

I. Introduction

One-class classification (OCC) is a specific area of machine learning, dealing with situations, in which not all of the classes are available at the training step [1]. It assumes that the classifier is built on the basis of samples coming only from a single class (known as the target class or the target concept), while it must discriminate between the known examples and new, unseen examples (known as outliers) that do not meet the assumption about the concept.

In case of OCC we often face the problem of big data size. Although we cannot estimate any information about the size of the outlier class, we usually have abundant access to the target class examples. As they in most cases represent the proper and desirable working scheme of the examined system, they are easy to obtain. However, not all of these target samples are relevant to the decision criteria forming procedure [2]. They do not contribute any new knowledge to the description of the target class and only increase the computational complexity of the classifier training procedure. Therefore, they can be discarded without any harm to the classification procedure [3].

Data preprocessing techniques [4] simplify training data, aiming to enable data mining algorithms to be applied in a faster and more accurate way, by removing noisy or redundant data. In the literature, we can find techniques that are focused on the attribute space and others that take into consideration the instance space. This latter is commonly known as the family of instance reduction methods. Those instance reduction methods that are designed to enhance the classification capabilities of the nearest neighbor (NN) rule [5] are named as prototype reduction (PR) methods.

PR methods look for a reduced set of prototypes that better adjust the decision boundaries between classes in NN classification. They are usually categorized into prototype selection [6] and prototype generation or abstraction [7]. The former only selects a subset of instances from the training data set [8]. The latter may select or generate new artificial prototypes. Thus, PR can be seen as a combinatorial and optimization problem. Among the existing PR methods, evolutionary algorithms have been stressed as the most promising methods [6], [7].

In this paper, we investigate the usefulness of evolutionary PR methods for one-class classification. As PR has proven itself very useful for distance-based classifiers in multi-class problems, we examine their performance in connection with Nearest Neighbor Data Description [9], that is, a one-class minimal distance classifier. To do this, we will treat OCC as a binary classification problem in which the majority class is the target concept and the rest of prototypes are considered outliers. So far, there were some preliminary studies done on PR for OCC [10]. In our proposal, we use compound evolutionary methods for an effective prototype reduction.

To analyze which kinds of PR methods are more appropriate for OCC, in our experiments we will pick up two different PR methods: a memetic prototype selection algorithm, called SSMA [11], and a differential evolution-based technique for prototype generation [12], denoted as SFLSDE. We will conduct experiments involving 18 classification data set that will be statistically validated. The presented results show that both alternatives allow us to maintain an acceptable accuracy, while reducing significantly the size of the training set.

The paper is organized as follows. Section II provides background information about one-class classification. Section III explains the use of PR techniques for the one-class problem. Section IV presents and discusses the empirical results. Finally, the conclusions achieved in this contribution are shown in Section V.

Bartosz Krawczyk and Michał Woźniak are with the Department of Systems and Computer Networks, Wrocław University of Technology, Poland (email: {bartosz.krawczyk,michal.wozniak}@pwr.wroc.pl).

Isaac Triguero and Francisco Herrera are with Department of Computer Science and Artificial Intelligence, University of Granada, Spain (email: {triguero,herrera}@decsai.ugr.es).

Salvador García is with Computer Science Department, University of Jaen, Spain (email: sglopez@ujaen.es).

This work was supported by The Polish National Science Centre under the grant PRELUDIUM number DEC-2013/09/N/ST6/03504 and by the Spanish Ministry of Education and Science under Project TIN2011-28488 and the Andalusian Research Plan P10-TIC-6858, P11-TIC-7765.

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

II. One-Class Classification

In this section we briefly introduce the OCC problem (Section II-A) and we detail the NN algorithm used for OCC (Section II-B).

A. Definition

OCC is a specific area of machine learning. It distinguish two classes - target concept ω_T and outliers ω_O. Hence, it can be seen as a binary classification problem. However, the main difference lies in how the classifier is being constructed. OCC assumes, that during the training phase only objects coming from the target class ω_T are available. However, new and unseen objects may appear during the classifier exploitation phase. The main principle of OCC is building such pattern recognition models using only examples from ω_T , that will be able to dichotomize between ω_T and ω_O. In the standard dichotomy problems we may expect objects from the other classes to predominantly come from one direction. Here, the available class should be separated from all the possible outliers - this leads to a situation in which a decision boundary should be estimated in all directions in the feature space around the target class.

A good OCC model should aim at minimizing two types of errors. The first one is false acceptance rate and reflect the degree in which the used one-class classifiers fails to detect outliers ω_O. High false acceptance rate will mean, that the used model cannot distinguish between the target concept and unwanted objects, and will allow too many negative samples to be accepted. In practical applications this can be especially dangerous, as it may fail to recognize a malicious behavior or to detect a failure of the system. Another type of error that is of high importance in OCC is the false rejection rate. In this case an object from the target concept will be labeled as an outlier and rejected by the system. This happens in cases, when used classifier is fitted strongly to the training data (as overfitting with only single class available is very common problem) and it loses generalization abilities. In practical applications this may be a case, when a valid financial transaction is classified as a fraud or when a standard traffic in the network is labeled as an intrusion.

Then, an efficient OCC algorithm requires a trade-off between high outlier rejection rate and generalization over the target class. In OCC during the classifier training step only objects from ω_T are at hand, but during the exploitation step both objects from ω_O and ω_T may appear.

OCC is a solution to many real-life problems where data from a single class is abundant but is hard or even impossible to obtain for other objects. This is often the case in problems such as intrusion detection [13], machine fault diagnosis [14], or solid-state fermentation [15].

OCC has gained an increasing interest over the last decade, which lead to introduction of several types of OCC algorithms. We can distinguish four main approaches:

- **Density estimation methods**: they are based on the idea of estimating the density of ω_T. This is the most straightforward approach for OCC, but at the same time it can be a very effective tool. However, this approach has limited applications, as it requires a high number of available samples and the assumption of a flexible density model [16]. Among the most popular density methods for OCC the Gaussian model, the mixture of Gaussians [17], and the Parzen density [18] can be mentioned. Also is a popular method, lifting some of the drawbacks of other density estimators.
- **Reconstruction methods**: algorithms from this group were originally introduced as a tool for data modeling [19]. These one-class classifiers are based on making assumptions about the target class structure and topology. Use of reconstruction methods for OCC is based on the idea that possibly the unknown outliers do not satisfy those assumptions about the structure of objects under consideration. The most popular techniques are the k-means [20], the self-organizing maps [21] and the auto-encoder networks [22].
- **Boundary methods**: these classifiers were introduced due to the fact, that estimating the complete density or structure of a target concept in one-class problems may very often be burdened with a high computational cost or even impossible. This group of algorithms concentrate on estimating only the enclosing boundary for ω_T. Such a boundary should be an efficient description of the target class distribution. The main aim of these methods is to find the optimal size of the volume enclosing given training points [23], because one that is too small can lead to an overtrained model, while one that is too big may lead to an extensive acceptance of outliers into the target class. Boundary one-class classifiers can efficiently work with a lower number of training examples than methods from two previous groups. The most popular boundary algorithms include the Support Vector Data Description [24], the One-class Support Vector Machine [25] and the Nearest Neighbor Data Description [1].
- **Ensemble methods**: combining one-class classifiers can lead to a significant improvement in their stability and outlier detection rate [9], [2]. Ensembles allow us to train less complex individual classifiers, thus reducing the risk of model overfitting, which is one of the major concerns in using the OCC. It was shown that carefully designed ensembles for OCC may outperform a single-model approach [26], [27]. Dedicated diversity measures have been introduced to tackle the specific nature of OCC and allow an efficient ensemble pruning [28].

B. Nearest Neighbor Data Description

In this work, we concentrate on the usage of a minimal distance classifier, adjusted to the one-class problems: Nearest Neighbor Data Description (NN-dd) [1], [29]. Our choice comes from two different reasons: firstly, NN-dd is an effective one-class predictor with no parameters to set; secondly, PR works exceptionally well for multi-class NN

methods and we wanted to check if this holds for one-class cases. Let us now describe shortly the NN-dd algorithm.

NN-dd avoids the direct density estimation (which can be a complex problem as mentioned earlier) and instead uses only distance to the first nearest neighbor. In this classification approach a cell, often represented as a d-dimensional hypersphere, is located around the given test object x. The volume of this cell is grown until it encompasses k objects from the training set. This allows to estimate the local density with:

$$p_{NN}(x) = \frac{k/N}{V_k(dist(x, NN_k^{tr}(x)))}, \quad (1)$$

where N is the size of the training set, $NN_k^{tr}(x)$ is the k nearest neighbor of x in the training set, V_k is the volume of the cell, that contains this object, and $dist(\cdot,\cdot)$ is a distance value between two points.

In the one-class case of NN-dd, a given unknown test object x is accepted as ω_T only if its local density is equal or larger than the local density of its nearest neighbor in the training set. For local density estimation, $k = 1$ is used.

Assuming that we use a d-dimensional cells, we can write the decision scheme of the considered NN-dd as:

$$f_{NN^{tr}}(x) = I\left(\frac{dist(x, NN^{tr}(x))}{dist(NN^{tr}(x), NN^{tr}(NN^{tr}(x)))} \leq 1\right). \quad (2)$$

This can be understood as the classification rule used by the NN-dd. It works by comparing the distance from the new object x to its nearest neighbor from the training set (which consist of labeled target class examples) with the distance from this nearest neighbor $NN^{tr}(x)$ to its nearest neighbor. This means, that new object is accepted, if it satisfies the local density of objects in the target class.

NN-dd is a simple, but effective one-class classifier. It combines advantages of density estimation methods (good description of the target class) with advantages of boundary methods (do not require a large training set to work properly). Additionally, NN-dd does not have any free parameters, that require a careful tuning from the user. It relies completely on the given set of data. This is a very important feature. Most of OCC methods are very sensitive to parameter tuning, and bad settings can make them ineffective. As OCC is used in many areas of applications [14], methods that do not require an expert knowledge about machine learning from the end-user are of high importance.

III. Prototype Reduction Methods

In this section we present a formal description of the PR problem (Section III-A), the methods used in this work (Section III-B) and how to apply PR for OCC (Section III-C).

A. Definition

A formal notation of the PR problem is the following: Let TR be a training data set and TS a test set, they are formed by a determined number **n** and **t** of samples, respectively. Each sample $\mathbf{x}_p$ is a tuple $(\mathbf{x}_{p1}, \mathbf{x}_{p2}, ..., \mathbf{x}_{pD}, \omega)$, where, $\mathbf{x}_{pf}$ is the value of the f-th feature of the p-th sample. This sample belongs to a class ω, given by $\mathbf{x}_{p\omega}$, and a d-dimensional space. For the TR set the class ω is known, while it is unknown for TS.

The purpose of PR is to provide a reduced set RS which consists of **rs**, **rs** $<$ **n**, prototypes, which are either selected or generated from the examples of TR. The prototypes of RS should be calculated to efficiently represent the distributions of the classes and to discern well when they are used to classify the training objects. The size of RS should be sufficiently reduced to deal with the storage and evaluation time problems of the NN classifier.

As we stated before, PR is usually divided into those approaches that are limited to select instances from TR, known as prototype selection, and those that may generate artificial examples (prototype generation). Both strategies have been deeply studied in the literature. Most of the recent proposals are based on evolutionary algorithms to select [11] or generate [12] an appropriate RS. Recent reviews about these topics are [6] and [7]. More information about PR can be found at the SCI2S thematic public website on *Prototype Reduction in Nearest Neighbor Classification: Prototype Selection and Prototype Generation* [1].

B. Selected PR methods

As we commented above, in our experiments we will focus on two different PR techniques: SSMA [11] and SFLSDE [12]. In the following we will introduce their main characteristics.

- **SSMA**: It was proposed in [11] to overcome one of the most relevant weaknesses of prototype selection methods, that is, the lack of convergence to face large-scale problems. SSMA utilizes of a local search or meme specifically developed for the prototype selection problem. This interweaving of the local and global search phases allows the two to influence each other.
  As an evolutionary prototype selection method, this steady-state memetic maps the TR onto a chromosome structure that is composed by genes, each one with two possible states (a binary codification: 0,1), meaning which instances will composed the resulting RS. Its fitness function is computed as follows: the instances in the training set TR are classified with the prototypes that are selected in the current chromosome, by using the NN rule with a leave-one-out validation scheme. The resulting fitness value is measured as the number of successful hits in comparison with the total number of classifications
  Its performance has been evaluated in different works [11], [6] and it is remarkable its good trade-off between high accuracy and reduction rates.
- **SFLSDE**: This method was proposed in [12] as a differential evolution algorithm for prototype generation. As such, it uses real parameter codification to encode

[1] http://sci2s.ugr.es/pr/

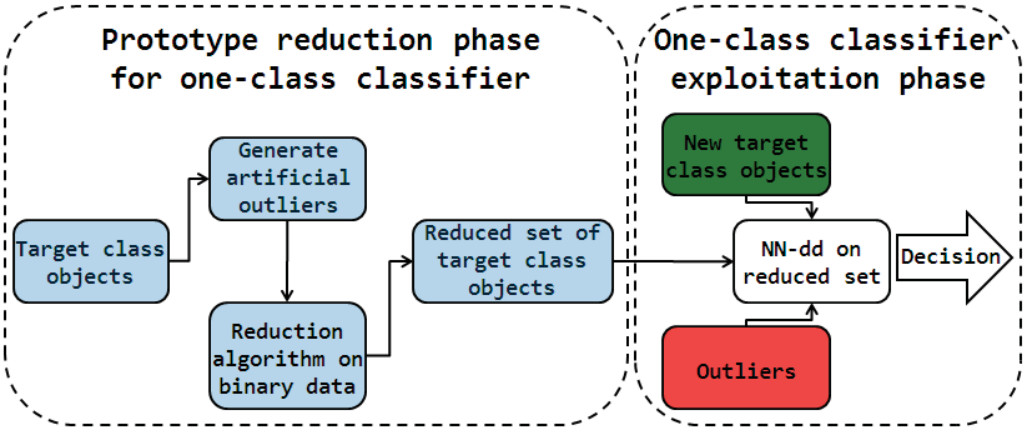

Fig. 1. Flowchart of the application of PR methods on OCC

solutions. Each individual is composed by a complete solution RS that are generated with the respective mutation and crossover operators. As in the previous algorithm, the fitness function is computed as the success in training classification.

The SFLSDE approach is a memetic differential evolution [30] that integrates an adaptive mechanism to choose appropriate mutation and crossover parameters during the evolutionary cycle. Thus, it avoids to fix these important parameters for differential evolution, which crucially depends on the addressed problem.

In terms of performance, it has shown to perform well in terms of accuracy, while the reduction rate keeps fixed as an initial parameter.

Comparing both approaches, they have different advantages and disadvantages that are inherent to way they conceive the PR problem:

- Prototype selection assumes that existing input data perfectly delimits the decision boundaries between classes.
- Prototype generation relates to a more complex problem, that is, the search space can be much more difficult to be explored. Thus, finding solution by using SFLSDE requires a higher cost than a prototype selection method.

C. Applying PR to OCC

In this section we explain how to use PR methods to OCC. As we stated above, PR techniques are preprocessing techniques, so that, they are applied first over the exiting data. Then, a classifier is trained with the resulting RS.

For OCC, we have proceeded as follows:

1) Initially, the training set TR is only formed by examples of the target class ω_T.
2) Since PR methods were designed for multi-class problems, we need to have example outliers. As in real-life scenarios outliers are not available during the training phase, one need to find another way to have access to them. We follow the approach proposed in [31], where an efficient procedure for generating artificial outliers around the target class was proposed. We assume an uniform distribution of outliers around the target concept and we generate number of outliers equal to the target class objects in the training set - to have a balanced classification problem. Please note, that artificial outliers are used only as an input for PR methods - the actual evaluation is carried out on "real" outliers from the testing set, that have not been used in any way during the training procedure.
3) Thus, PR methods are run on a binary data set, consisting of target class examples and artificial outliers. As an output, we receive a reduced set of prototypes for target class and artificial outlier class. Then, artificial outliers are discarded and we will use only prototypes for the target class.
4) Finally, we conduct the testing phase with the usage of reduced set of target class prototypes. They are used to label new samples according to the NN-dd procedure. The testing set consists of both new target class objects and outliers, so we can establish the accuracy of our proposed methods.

Figure 1 draws a flowchart of the followed approach.

IV. Experimental Investigations

The aim of this experimental study was to check the performance of the used PR methods for nearest neighbor one-class classification. We examined, to how extent datasets are reduced by each method and how it affects the final classification accuracy. Section IV-A provides details of the problems selected for the experimentation. Section IV-B defines the measures employed to evaluate the performance of the algorithms, the configuration parameters utilized and the statistical test conducted.

A. Datasets

In total we chose 18 datasets from the UCI Repository. Most of them are binary ones, where the majority class was used as the target concept and the minority class as outliers. In case of multi-class datasets (such as Iris), we merged all but one classes as target concept and used the remaining one

(with smallest number of samples) as outliers. Details of the chosen datasets are given in Table I.

TABLE I
DETAILS OF DATASETS USED IN THE EXPERIMENTS.

No.	Name	Objects	Features	Classes
1.	Climate Model Simulation Crashes	540	18	2
2.	Congressional Voting Records	435	16	2
3.	Contraceptive Method Choice	1473	9	3
4.	Credit Approval	690	15	2
5.	Fertility	100	10	2
6.	Haberman's Survival	306	3	2
7.	Hepatitis	155	19	2
8.	Hill-Valley	606	101	2
9.	ILPD (Indian Liver Patient Dataset)	583	10	2
10	Iris	150	4	3
11.	Mammographic Mass	961	9	2
12.	Musk (Version 2)	6598	168	2
13.	Ozone Level Detection (One Hour)	2536	73	2
14.	Parkinsons	197	23	2
15.	Pima Indians Diabetes	768	8	2
16.	Sonar, Mines vs. Rocks	208	60	2
17.	Statlog (Heart)	270	13	2
18.	Wisconsin Breast Cancer (Original)	699	10	2

B. Experimental set-up

Experiments were carried out with the use of 5x2 fold-cross validation, where the training set consisted only of a portion of examples from the training class, while the testing fold consisted of outliers and target class samples.

To compare the performance of the algorithms, we use the following measures:

- *Accuracy:* It counts the number of correct classifications regarding the total number of instances.
- *Reduction rate:* It measures the reduction of storage requirements achieved by a PR algorithm.

$$ReductionRate = 1 - size(RS)/size(TR) \quad (3)$$

  Reducing the stored instances in the TR set will yield a time reduction to classify a new input sample.
- *Accuracy·Reduction rate:* It measures the trade-off between accuracy and reduction rates.
- *Runtime classification:* It refers to the time needed to classify all the instances of the test set TS regarding a given training data.

Results are presented with the respect to the test accuracy (to check, how the PR methods affect the performance of the NN-dd), the reduction rate (to check, how much the used methods reduces the volume of the training set), the product of these two terms (to evaluate the trade-off), and the average runtime classification (to analyze the complexity reduction in terms of seconds).

Table II presents all the parameters involved in our experimental study. These parameters have been fixed according to the recommendation of the corresponding authors of each algorithm. In this study, all the methods use the Euclidean distance to compute distances between examples.

Finally, to validate statistically the results, we use the Shaffer post-hoc test [32] to find out which of the tested methods are distinctive among an n x n comparison. The post-hoc procedure is based on a specific value of the significance level α. Additionally, the obtained p-values should be examined in order to check how different given two algorithms are. We fix the significance level $\alpha = 0.05$ for all comparisons. Two separate Shaffer procedures are used: one with respect to the accuracy and one with respect to the accuracy * reduction rate (to give a trade-off between the classification quality and training set minimization). The second Shaffer test is carried out only between the reduction methods (as standard NN-dd has no reduction rate).

TABLE II
PARAMETER SPECIFICATION FOR PR METHODS

Algorithm	***Parameters***
SSMA	Population Size=30, Evaluations=10 000, Cross=0.5
	Mutation=0.001
SFLSDE	Population Size = 40, Iterations = 500
	iterSFGSS =8, iterSFHC=20, Fl=0.1, Fu=0.9
	Mutation operator: RandToBest/1/Bin, Reduction Rate=0.98

C. Results

Results of the experiments, presented according to the accuracy and reduction rate of the examined methods, are given in Table III. Outputs of Shaffer post-hoc test over accuracy are given in Table IV and over accuracy·reduction ratio in Table V.

TABLE III
RESULTS OF THE EXPERIMENTS WITH THE RESPECT TO ACCURACY [%] AND REDUCTION RATIO [%].

No.	NN-dd	SSMA + NN-dd		SFLSDE + NN-dd	
	Accuracy	Accuracy	Reduction	Accuracy	Reduction
1.	75.54	71.47	97.42	74.82	94.31
2.	94.30	92.37	99.54	94.02	98.23
3.	89.76	83.97	99.22	89.06	97.63
4.	79.63	79.31	97.83	78.59	97.11
5.	86.89	86.24	98.00	85.98	97.00
6.	69.98	67.23	99.02	68.82	98.72
7.	67.43	65.98	97.42	65.34	98.87
8.	86.94	86.65	97.03	86.38	97.53
9.	93.41	89.32	99.15	92.59	97.60
10.	96.21	94.03	97.33	95.87	97.33
11.	87.37	82.94	99.17	86.17	97.51
12.	71.38	69.42	99.69	71.12	99.75
13.	78.49	72.53	99.77	76.84	97.17
14.	67.39	67.05	96.95	66.28	97.94
15.	90.52	90.01	98.70	90.26	97.39
16.	82.66	81.79	98.07	82.66	96.16
17.	68.06	66.93	98.15	67.70	97.41
18.	91.36	89.37	99.29	91.36	97.72

TABLE IV
SHAFFER TEST FOR EXAMINED METHODS WITH THE RESPECT TO THEIR ACCURACY. SYMBOL '=' STANDS FOR CLASSIFIERS WITHOUT SIGNIFICANT DIFFERENCES, '+' FOR SITUATION IN WHICH THE METHOD ON THE LEFT IS SUPERIOR AND '-' VICE VERSA.

hypothesis	p-value
NN-dd vs SSMA + NN-dd	+ (0.0193)
NN-dd vs SFLSDE + NN-dd	= (0.3120)
SSMA + NN-dd vs SFLSDE + NN-dd	- (0.0364)

TABLE V

SHAFFER TEST FOR EXAMINED METHODS WITH THE RESPECT TO THEIR ACCURACY * REDUCTION RATIO. SAME SYMBOLS AS IN TABLE IV.

hypothesis	p-value
SSMA + NN-dd vs SFLSDE + NN-dd	- (0.0408)

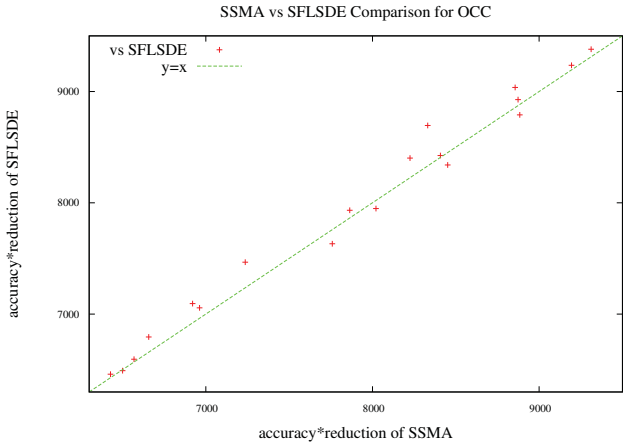

Fig. 2. Accuracy·Reduction comparison between SSMA and SFLSDE for OCC

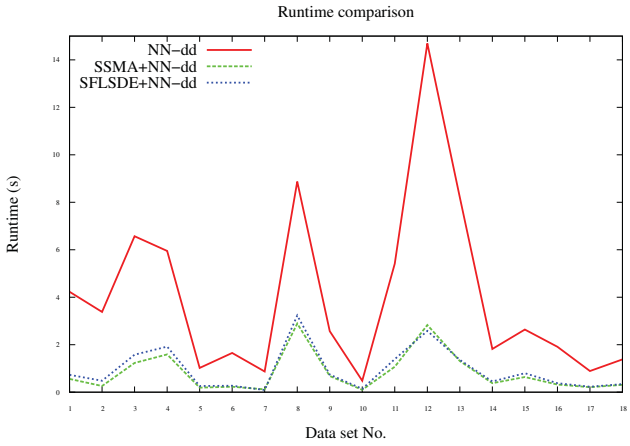

Fig. 3. Runtime comparison

In the scatterplot of Figure 2, each point compares the accuracy·reduction ratio of SSMA and SFLSDE a single dataset. The x-axis position of the point is the accuracy of SSMA, while the y-axis position is the accuracy of SFLSDE. Therefore, points below the $y = x$ line correspond to datasets for which SSMA performs better than SFLSDE.

Finally, Figure 3 depicts a runtime comparison between all the methods used. It shows for each data set the average runtime classification in seconds.

D. Discussion

The obtained results allow us to draw several interesting conclusions on the applicability of the prototype reduction methods in one-class classification tasks:

- Firstly, let us notice that in all of 18 cases both of the methods always achieved reduction rate greater than 95%. This is a very significant reduction of the training volume size, which leads to a lower computational complexity of NN-dd.
- The NN-dd classifier tends to work well in conjunction with the used reduction methods. Regardless of the used reduction method, we were able to greatly lower the training set size, while maintaining acceptable accuracy.
- Out of the two used PR methods, SSMA returned greater reduction rate, very often around 98-99 %. However, this was connected with some drop in the accuracy. Most significant drops occurred for three datasets (Contraceptive Method Choice, Mammographic Mass and Ozone Level Detection), where we lost around 5-6% in accuracy performance. It means that SSMA has removed an excessive number of instances for these data sets, so that, the resulting reduced set does not represent adequately the original problem in order to NN-dd can classify test data properly. The Shaffer post-hoc test shows, that when considering the accuracy measure over all 18 datasets SSMA performs statistically worse than simple NN-dd without any reduction.
- Second used method, SFLSDE, did not reduce the training set in such extent as SSMA. Typically the volume returned was 2-3 times bigger. Only for four datasets (Hepatitis, Hill-Valley, Musk and Parkinsons) did SFLSDE achieved higher reduction ratio than its predecessor. However, in this case we did not observed a drop of accuracy after the reduction. Shaffer test confirms, that there does not exist any statistically significant difference between the SFLSDE and simple NN-dd. This means that we can efficiently apply this method for data set reduction without sacrificing the accuracy of the one-class minimal distance predictor.
- When comparing SSMA with SFLSDE we can see that the first method offers greater reduction, while the second does not drop in accuracy. Shaffer test over accuracy results proves, that SFLSDE is superior according to this criterion. Another Shaffer test was carried out on a criterion being a combination of accuracy and reduction rate, in order to check if greater reduction may compensate for a loss in recognition rate. SFLSDE emerges as a superior method from this test. Figure 2 supports this statement by showing that most of the points are upper the line $y = x$.
- In terms of runtime, we can observe in Figure 3 that the data reduction process has implied in a wide reduction in the time spent by NN-dd when SSMA or SFLSDE are previously applied. In this figure, we can check that the bigger is the original training the greater is the runtime decrement (For example, see Musk data set, No. 12).

To sum it all up, one may see that data reduction can be effectively applied for OCC, especially those schemes that are based on prototype generation. Nevertheless, a too big reduction leads to a drop of the accuracy of one-class

predictor. This can be explained by the nature of used NN-dd, which uses a distance between two neighbors in the training set as a threshold for accepting new objects. In case of too reduced set, the estimated threshold becomes ineffective and allows for an increased acceptance of outliers.

V. Conclusions

In this contribution we have performed a first investigation about how evolutionary prototype reduction methods behave with one-class nearest neighbor classification. Concretely, we have studied two prototype reduction methods with different philosophies: selection and generation of prototypes. We have combined these PR methods with the NN-dd algorithm for one-class classification. The experimental study carried out has allowed us to establish several differences between the used reduction methods. We have observed that the SFLSDE algorithm (prototype generator) obtains, in general, a better trade-off between accuracy and reduction rates. However, both alternatives have shown to perform properly in combination with NN-dd, providing a great complexity reduction that results in a faster classification stage without significant accuracy losses. All the results have been statistically validated, which reinforces the conclusions achieved. Thus, the use of evolutionary prototype reduction methods is a promising research line to improve one-class classification.

References

[1] D. M. J. Tax, "One-class classification," 2001.

[2] B. Krawczyk, M. Woźniak, and B. Cyganek, "Clustering-based ensembles for one-class classification," *Information Sciences*, vol. 264, pp. 182–195, 2014.

[3] F. Zhu, N. Ye, W. Yu, S. Xu, and G. Li, "Boundary detection and sample reduction for one-class support vector machines," *Neurocomputing*, vol. 123, pp. 166 – 173, 2014.

[4] D. Pyle, *Data Preparation for Data Mining*, ser. The Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, 1999.

[5] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[6] S. García, J. Derrac, J. R. Cano, and F. Herrera, "Prototype selection for nearest neighbor classification: Taxonomy and empirical study," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 3, pp. 417–435, 2012.

[7] I. Triguero, J. Derrac, S. García, and F. Herrera, "A taxonomy and experimental study on prototype generation for nearest neighbor classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 42, no. 1, pp. 86–100, 2012.

[8] F. Angiulli and G. Folino, "Distributed nearest neighbor based condensation of very large datasets," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 19, no. 2, pp. 1593–1606, 2007.

[9] D. M. J. Tax and R. P. W. Duin, "Combining one-class classifiers," in *Proceedings of the Second International Workshop on Multiple Classifier Systems*, ser. MCS '01. London, UK: Springer-Verlag, 2001, pp. 299–308.

[10] G. G. Cabral and A. L. I. de Oliveira, "A novel one-class classification method based on feature analysis and prototype reduction," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Anchorage, Alaska, USA, October 9-12, 2011*, 2011, pp. 983–988.

[11] S. García, J. R. Cano, and F. Herrera, "A memetic algorithm for evolutionary prototype selection: A scaling up approach," *Pattern Recognition*, vol. 41, no. 8, pp. 2693–2709, 2008.

[12] I. Triguero, S. García, and F. Herrera, "Differential evolution for optimizing the positioning of prototypes in nearest neighbor classification," *Pattern Recognition*, vol. 44, no. 4, pp. 901–916, 2011.

[13] G. Giacinto, R. Perdisci, M. Del Rio, and F. Roli, "Intrusion detection in computer networks by a modular ensemble of one-class classifiers," *Inf. Fusion*, vol. 9, pp. 69–82, January 2008.

[14] A. Bartkowiak and R. Zimroz, "Outliers analysis and one class classification approach for planetary gearbox diagnosis," *Journal of Physics: Conference Series*, vol. 305, no. 1, 2011.

[15] H. Jiang, G. Liu, X. Xiao, C. Mei, Y. Ding, and S. Yu, "Monitoring of solid-state fermentation of wheat straw in a pilot scale using ft-nir spectroscopy and support vector data description," *Microchemical Journal*, vol. 102, 2012.

[16] S. Sonnenburg, G. Rtsch, C. Schfer, and B. Schlkopf, "Large scale multiple kernel learning," *Journal of Machine Learning Research*, vol. 7, pp. 1531–1565, 2006.

[17] H. Zuo, O. Wu, W. Hu, and B. Xu, "Recognition of blue movies by fusion of audio and video," in *2008 IEEE International Conference on Multimedia and Expo, ICME 2008 - Proceedings*, 2008, pp. 37–40.

[18] G. Cohen, H. Sax, and A. Geissbuhler, "Novelty detection using one-class parzen density estimator. an application to surveillance of nosocomial infections," in *Studies in Health Technology and Informatics*, vol. 136, 2008, pp. 21–26.

[19] K. F. Cheung, "Fuzzy one-mean algorithm: Formulation, convergence analysis, and applications," *Journal of Intelligent and Fuzzy Systems*, vol. 5, no. 4, pp. 323–332, 1997.

[20] B. Chen, A. . Feng, S. . Chen, and B. Li, "One-cluster clustering based data description," *Jisuanji Xuebao/Chinese Journal of Computers*, vol. 30, no. 8, pp. 1325–1332, 2007.

[21] O. Taylor and J. MacIntyre, "Adaptive local fusion systems for novelty detection and diagnostics in condition monitoring," in *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 3376, 1998, pp. 210–218.

[22] L. Manevitz and M. Yousef, "One-class document classification via neural networks," *Neurocomputing*, vol. 70, no. 7-9, pp. 1466–1481, 2007.

[23] D. M. J. Tax, P. Juszczak, E. Pekalska, and R. P. W. Duin, "Outlier detection using ball descriptions with adjustable metric," in *Proceedings of the 2006 joint IAPR international conference on Structural, Syntactic, and Statistical Pattern Recognition*, ser. SSPR'06/SPR'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 587–595.

[24] D. M. J. Tax and R. P. W. Duin, "Support vector data description," *Machine Learning*, vol. 54, no. 1, pp. 45–66, 2004.

[25] Y. Chen, X. S. Zhou, and T. S. Huang, "One-class svm for learning in image retrieval," in *IEEE International Conference on Image Processing*, vol. 1, 2001, pp. 34–37.

[26] B. Krawczyk and M. Woźniak, "Accuracy and diversity in classifier selection for one-class classification ensembles," in *Proceedings of the 2013 IEEE Symposium on Computational Intelligence and Ensemble Learning, CIEL 2013 - 2013 IEEE Symposium Series on Computational Intelligence, SSCI 2013*, 2013, pp. 46–51.

[27] T. Wilk and M. Woźniak, "Soft computing methods applied to combination of one-class classifiers," *Neurocomput.*, vol. 75, pp. 185–193, Jan. 2012.

[28] B. Krawczyk and M. Woźniak, "Diversity measures for one-class classifier ensembles," *Neurocomputing*, vol. 126, pp. 36–44, 2014.

[29] G. Cabral, A. L. I. Oliveira, and C. Cahu, "A novel method for one-class classification based on the nearest neighbor data description and structural risk minimization," in *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*, 2007, pp. 1976–1981.

[30] F. Neri and V. Tirronen, "Scale factor local search in differential evolution," *Memetic Computing*, vol. 1, no. 2, pp. 153–171, 2009.

[31] D. M. J. Tax and R. P. W. Duin, "Uniform object generation for optimizing one-class classifiers," *Journal of Machine Learning Research*, vol. 2, pp. 155–173, 2001.

[32] N. Garcia-Pedrajas, J. Maudes-Raedo, C. Garca-Osorio, and J. J. Rodriguez-Diez, "Supervised subspace projections for constructing ensembles of classifiers," *Information Sciences*, vol. 193, pp. 1–21, 2012.