Soft Comput (2015) 19:3387–3400
DOI 10.1007/s00500-014-1492-5

FOCUS

One-class classifiers with incremental learning and forgetting for data streams with concept drift

Bartosz Krawczyk · **Michał Woźniak**

Published online: 21 October 2014

Abstract One of the most important challenges for machine learning community is to develop efficient classifiers which are able to cope with data streams, especially with the presence of the so-called concept drift. This phenomenon is responsible for the change of classification task characteristics, and poses a challenge for the learning model to adapt itself to the current state of the environment. So there is a strong belief that one-class classification is a promising research direction for data stream analysis—it can be used for binary classification without an access to counterexamples, decomposing a multi-class data stream, outlier detection or novel class recognition. This paper reports a novel modification of weighted one-class support vector machine, adapted to the non-stationary streaming data analysis. Our proposition can deal with the gradual concept drift, as the introduced one-class classifier model can adapt its decision boundary to new, incoming data and additionally employs a forgetting mechanism which boosts the ability of the classifier to follow the model changes. In this work, we propose several different strategies for incremental learning and forgetting, and additionally we evaluate them on the basis of several real data streams. Obtained results confirmed the usability of proposed classifier to the problem of data stream classification with the presence of concept drift. Additionally, implemented forgetting mechanism assures the limited memory consumption, because only quite new and valuable examples should be memorized.

Communicated by E. Lughofer.

This work was supported by the Polish National Science Center under the Grant No. DEC-2013/09/B/ST6/02264.

B. Krawczyk (✉) · M. Woźniak
Department of Systems and Computer Networks, Wrocław University of Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland
e-mail: bartosz.krawczyk@pwr.edu.pl

M. Woźniak
e-mail: michal.wozniak@pwr.edu.pl

Keywords Pattern classification · One-class classification · Data stream classification · Concept drift · Incremental learning · Forgetting

1 Introduction

Contemporary computer systems manage and store enormous amount of data. It is predicted that the volume of stored information will be doubling every two years. People send 14 billion e-mails and more than 350 million tweets per day. The huge chains of discount department stores (as Walmarkt Inc.) register more than 1 million transactions per hour. Therefore, the marked leading companies desire to develop smart analytic tools based on machine learning approach, which can analyze such enormous amount of data. Additionally, designing such analytical tools should take into a consideration that most of data arrives continuously in the form of so-called data stream (Gama 2010). Furthermore, the relation within the data, i.e., statistical dependencies characterizing a given phenomenon (such as client behavior), may change (Gama 2012). This observation requires a special analytical model which can cope with such non-stationary characteristics. In the beginning, the data streams originated in the financial markets. Today, data streams can be found everywhere—in the Internet, monitoring systems, sensor networks and other domains (Hulten et al. 2009). Data streams differ from the traditional static data, because they can be viewed as an infinite amount of data that arrives continuously, where memory and computational complexity play the crucial roles. Due to this mining data stream poses many new challenges to the contemporary machine learning systems (Aggarwal et al. 2004).

In this paper, we will mainly focus on the classification task, which is a widely used analytical approach (Duda et al. 2001). Basically, it aims at assigning a given observation to one of the predefined categories. Such situation can be found, e.g., in spam filtering, biometrics, medical decision support, or fraud detection to enumerate only a few. The concept drift in the classification model mean that the statistical dependencies between attributes describing an object and its predefined label could change over time.

To explain the possible types of the changes, let us shortly introduce a statistical classification model. This theory assumes (Duda et al. 2001) that both the attributes describing an object $x \in \mathcal{X} \subseteq \mathcal{R}^d$ and its correct classification (class label) $j \in \mathcal{M} = \{1, 2, \ldots, M\}$ are observed values of a pair of random variables $(\mathbf{X}, \mathbf{J})$. The probability distribution of them is given by the *prior* class probabilities

$$p_j = P(\mathbf{J} = j), \quad j \in \mathcal{M} \tag{1}$$

and class-conditional probability density function of $\mathbf{X}$

$$f_j(x) = f(x|j), \quad x \in \mathcal{X}, \ j \in \mathcal{M}. \tag{2}$$

The classification algorithm Ψ maps the feature space $\mathcal{X}$ to the set of defined class labels $\mathcal{M}$

$$\Psi : \mathcal{X} \to \mathcal{M}. \tag{3}$$

If the probability characteristics given by Eqs. (1) and (2) are known, then the optimal classifier Ψ^*, minimizing the misclassification probability, makes decisions according to the following rule:[1]

$$\Psi^*(x) = i \quad \text{if } p_i(x) = \max_{k \in \mathcal{M}} p_k(x), \tag{4}$$

where $p_j(x)$ stands for the *posterior* probability

$$p_i(x) = \frac{p_i f_i(x)}{f(x)} \tag{5}$$

and $f(x)$ denotes a probability density function

$$f(x) = \sum_{k=1}^{M} p_k \ f_k(x) \tag{6}$$

Reverting to the notion of a concept drift, we can distinguish two types of such an event, according to its influence on the probability density function Eq. (6) or on the *posterior* probability Eq. (5) (Gama et al. 2013):

[1] We assume so-called "zero-one" lost function, i.e., if the cost of the misclassification error is the same and equals 1 and cost of correct classification is 0. Otherwise the classification rule points on the class for which the expectation value of the lost function is the smallest (Duda et al. 2001).

– virtual concept drift means that changes do not impact the *posterior* probabilities, but affect the conditional probability density functions (Widmer and Kubat 1993).
– real concept drift means that changes affect the *posterior* probabilities and may impact unconditional probability density function (Schlimmer and Granger 1986; Widmer and Kubat 1996).

From the classification point of view, the real concept drift is important because it can strongly affect the shape of the decision boundary. The virtual drift does not affect the decision rule, especially taking into consideration the Bayes decision rule Eq. (4). Another drift taxonomy depends on the drift impetuosity and here we can distinguish:

– slow changes, i.e., gradual or incremental drift.
– abrupt changes, i.e., sudden drift.

The presence of a concept drift can lead to serious deterioration of classifier's accuracy (Lughofer and Angelov 2011). This is depicted in Fig. 1, where two types of concept drift are shown.

Additionally, we can consider a reoccurring concept drift. It may occur in cases of, e.g., seasonal phenomena as weather prediction or client preferences of clothes or sport stores (Widmer and Kubat 1996). Therefore, developing efficient methods which are able to deal with this type of change in data stream is nowadays the focus of intense research.

The main aim of this paper is to introduce an efficient method of incremental data stream classification, i.e., we will consider the task where the concept drift is rather smooth and the classifier model will try to follow the model's changes. As implementation of the classifier, we propose a novel modification of the weighted one-class classifier which tunes the shape of its decision boundary on the basis of weights assigned to the training objects. The main contribution of this work is a novel one-class classifier applied to this task with the built-in adaptation and forgetting mechanisms. The evaluation of the proposed method is carried out on the basis of the computer experiments on real and semi-synthetic data sets. The outline of the work is as follows. First, the related works on data stream classification and one-class classifiers will be presented. Then, the original algorithm will be described. The following section is focusing on the results of the experimental research. The last part concludes the paper.

2 Related works

In this section, a short overview on the related fields will be given.

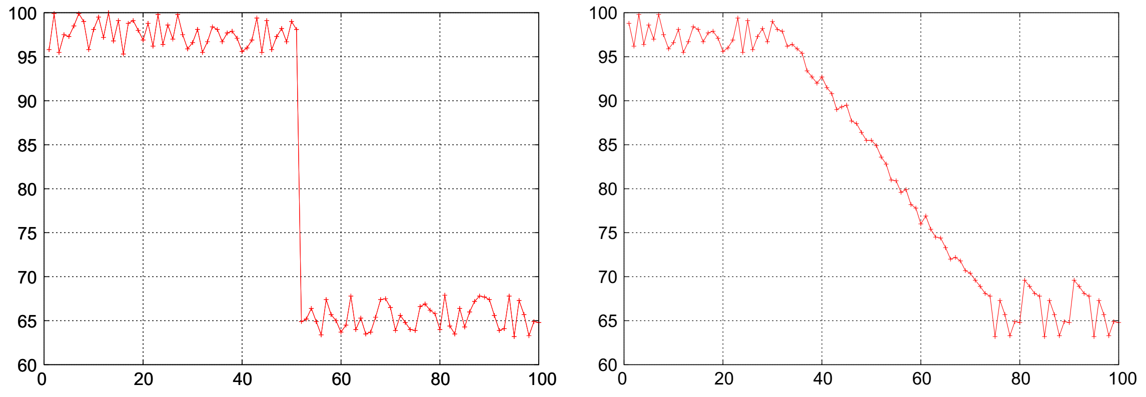

Fig. 1 Exemplary classifier accuracy deterioration—sudden drift versus incremental one

2.1 Data stream classification in the presence of concept drift

In general, the following approaches can be considered to cope with drifting data stream.

- Training new classifier every time when new data are available. Such an approach is impractical and very expensive, especially if drift occurs rapidly.
- Detecting concept drift in new data, and if these changes are significant enough then train the classifier on the basis of new data gathered after a drift occurred.
- Adopting an incremental learning algorithm for the classification model to smoothly adapt to changing nature of incoming objects.

The first algorithms designed to deal with a drifting data were STAGGER (Schlimmer and Granger 1986), IB3 (Aha et al. 1991), and the suite of FLORA algorithms (Widmer and Kubat 1996). Now, there are plethora of methods proposed to cope with the concept drift phenomenon. Basically, we can organize them into the following groups:

1. Drift detection algorithms.
2. Online learners.
3. Sliding window-based solutions.
4. Ensemble approaches (Kuncheva 2004; Ouyang et al. 2011).

The drift detectors try to alarm the recognition system in case changes occur (Sobolewski and Wozniak 2013), but this mechanism is not used by all classifiers dedicated to data stream. Some evolving systems continuously adjust the model to incoming data (Zliobaite 2010), what is called implicit drift detection (Kuncheva 2008) as opposed to explicit drift detection methods that raise a signal to indicate change. The detector can be based on changes in the probability distribution of the instances (Gaber and Yu 2006; Markou and Singh 2003) or classification accuracy (Klinkenberg and Joachims 2000; Baena-García 2006). Many detection algorithms are based on a knowledge of object labels after the classification to detect the presence of a concept drift, however as pointed out in Zliobaite (2010), such approach is not useful from a practical point of view.

Online learners relate to classification algorithms that continuously update their classifier parameters while processing the incoming data. According to Domingos and Hulten (2003), such methods should meet some basic requirements:

- Each object must be processed only once in the course of training.
- The memory and computing time are limited.
- The classifier training could be interrupted several times and its quality should not be lower than the classifier trained using batch mode.

Such classifiers work fast and are able to adapt their model in a very flexible manner. Among the others, the following are the most popular online learners: Naïve Bayes, Neural Networks (Polikar et al. 2001), and Nearest Neighbor (Aha et al. 1991). It is worth noting the Concept-Adapting Very Fast Decision Tree (CVFDT) algorithm (Hulten et al. 2009), which is an extended version of the ultra fast decision tree, which ensures consistency with incoming data by maintaining alternative subtrees.

The last groups of algorithms based on sliding windows incorporate the forgetting mechanism. This approach is based on the assumption that the recently arrived data have higher relevancy, because they contain characteristics of the current context. Usually three strategies are used:

- selecting the instances by means of a sliding window that cuts off older instances (Widmer and Kubat 1996);
- weighting the data according to their relevance;

– applying bagging and boosting algorithms that focus on misclassified instances (Bifet et al. 2009; Chu and Zaniolo 2004).

When dealing with the sliding window, the main question is how to adjust the window size. On the one hand, a shorter window allows focusing on the emerging context, though data may not be representative for a longer lasting context. On the other hand, a wider window may result in mixing the instances representing different contexts.

Therefore, certain advanced algorithms adjust the window size dynamically depending on the detected state [e.g., FLORA2 (Widmer and Kubat 1996) and ADWIN2 (Bifet and Gavaldà 2007)] or multiple windows may even be used (Lazarescu et al. 2004).

The last group consists of algorithms that incorporate a set of classifiers (Wang et al. 2003; Stanley 2003; Tsymbal et al. 2008). It has been shown that a collective decision can increase classification accuracy because the knowledge that is distributed among the classifiers may be more comprehensive. This premise is true if the set consists of diverse members (Hulten et al. 2009; Shipp and Kuncheva 2002). In static environments, diversity may refer to the classifier model, the feature set, or the instances used in training.

In a changing environment, diversity can also refer to the context. This makes ensemble systems interesting for researchers dealing with concept drift. We can distinguish three main approaches related to classifier ensembles for data stream:

1. Dynamic combiners, where individual classifiers are trained in advance and their relevance to the current context is evaluated dynamically while processing subsequent data (Jacobs et al. 1991; Littlestone and Warmuth 1994).
2. Updating the ensemble members, where each ensemble consists of a set of online classifiers that are updated incrementally based on the incoming data (Fern and Givan 2003; Kolter and Maloof 2007; Bifet et al. 2011; Rodríguez and Kuncheva 2008).
3. Dynamic changes of the line-up of ensemble, namely, individual classifiers are evaluated dynamically and the worst one is replaced by a new individual trained on the most recent data (Jackowski 2013; Kolter and Maloof 2003).

The Streaming Ensemble Algorithm (SEA) (Street and Kim 2001) or the Accuracy Weighted Ensemble (AWE) (Wang et al. 2003) keeps a fixed-size set of classifiers. Data are collected in data chunks, which are used to train new classifiers. The SEA uses a majority voting, whereas the AWE makes decision on the basis of weighted voting. Dynamic Weighted Majority (DWM) algorithm (Kolter and Maloof 2003) reduces the weight when the classifier makes an incorrect decision. Eventually, the classifier is removed from the ensemble when its weight falls below a given threshold. Independently, a new classifier is added to the ensemble when the committee makes a wrong decision. Jackowski proposes a classifier ensemble training methods dedicated to a so-called recurring context (Jackowski 2013). To select an ensemble for the current model, the ensemble pruning method based on evolutionary programming is employed. Woźniak et al. propose the dynamic ensemble model called Weighted Aging Ensemble (WAE) (Woźniak et al. 2014). It can modify the line-up of a classifier ensemble on the basis of three factors: diversity measure, overall ensemble accuracy, and time that given classifier has spent as a member of the ensemble pool.

In this paper, we present a novel adaptive weighted one-class classifier that is able to change itself according to the nature of received data streams. We propose to use principles of incremental learning and forgetting to adjust shape of the decision boundary according to the concept changes in new data chunks. The learning and forgetting in data streams is realized by modifying weights assigned to objects—we propose how to calculate weights for new incoming objects to use their information to change the classifier and how to forget the old objects to prevent the overfitting of the classifier and uncontrolled increase in the volume of the stored data. Thus, let us shortly present the idea of one-class classifier.

2.2 One-class classification

One-class classification (OCC) is one of the most challenging areas of machine learning. It is assumed that during the classifier training stage we have at our disposal only objects coming from a single class distribution—the target concept (Koch et al. 1995). During the exploitation phase of such a classifier, there may appear new objects originating from different distributions than the target class. They are known as outliers and should be rejected by the trained model. Therefore, one-class classification aims at deriving a classification boundary that may separate the known target objects from possible outliers that may appear. No assumptions about the nature of outliers should be made.

OCC is an attractive approach for data stream classification, because it can be used for binary classification without an access to counterexamples, decomposing a multi-class data stream, outlier detection or novel class recognition. Most of data stream methods focus on supervised learning, which require a fully labeled data set for training. However, labeling an entire chunk of data stream can be expensive or hard to obtain, which limits the real-life applications of such methods (Kurlej and Woźniak 2012). Additionally, let us consider the intrusion detection for computer system (IDS/IPS)

(Noto et al. 2012). The target class covering the normal and safe messages is unchanged but the malicious messages as intrusion are still changing, because the malicious users are trying to lead security systems on. Therefore, they are inventing a variety of attacks. If we concentrate on normal messages only, as target class, then we can believe that we are able to train classifiers which are capable of distinguishing normal messages from malicious messages without knowledge about the outlier class. It could protect our security system against so-called "zero day attack" as well. Furthermore, OCCs require only a small number of positively labeled examples for initial training, which can be a valuable property in case of non-stationary classification.

Most of the existing works on OCC have not explicitly dealt with the changing nature of the input data (Masud et al. 2011). They are based on an underlying assumption that the training data set does not contain any uncertainty information and properly represents the examined concept. However, in many real-world applications, data change its nature over time—which is a vital problem for data stream analysis (Hashemi et al. 2009). For example, in environmental monitoring applications, data may change according to the examined conditions and what once was considered an outlier may in near future become a representative of the target concept. This kind of dynamic information (typically ignored in most of the existing one-class learning methods) is critically important for capturing the full picture of the examined phenomenon. Therefore, there is a need for introducing novel, adaptive techniques for dealing with non-stationary data sets (Gomez-Verdejo et al. 2011). On the other hand, OCC can be very useful for a variety of real-life data stream applications. Such an practical example would include a process of monitoring the nuclear power plant (Jackowski and Platos 2014). Here, the data arrive as a stream of sensor outputs and can be subject to changes due to the different energy demand ratios. It is easy to gather labeled examples of proper behavior of such a plant, but counterexamples are obviously dangerous to collect. This is an example of real-life situation in which an incremental OCC model, that can adapt to changes in data, can be of a high demand.

So far the OCC problem for data streams, especially in the presence of concept drift, has not been investigated thoroughly. One should mention several proposal of on-line One-Class Support Vector Machines (Zhang et al. 2009), an one-class modification of very fast decision trees (OcVFDT) (Li et al. 2009) and uncertain one-class classifier for summarizing concepts in the presence of uncertainty applying a generated bound score into a one-class SVM-based learning phase (Liu et al. 2014). Few ensemble techniques, based on chunks of data and standard one-class classifiers, have also been recently introduced (Zhang et al. 2010; Zhu et al. 2011).

2.3 Weighted one-class support vector machine

One-class classification aims at distinguishing between the available objects coming from the target distribution i_T and unknown outliers i_O that are unavailable during the classifier training step but may appear in the process of classifier exploitation. One-class support vector machine (OCSVM) (Schölkopf and Smola 2002) achieves this goal by computing a closed boundary in a form of a hypersphere enclosing all the objects from i_T. During the exploitation phase, a decision made about the new object is based upon checking whether it falls inside the hypersphere. If so, the new object is labeled as one belonging to i_T. Otherwise it belongs to i_O. The center a and a radius R are the two parameters that are sufficient for describing such a decision hypersphere. To have a low acceptance of the possible outliers, the volume of this d-dimensional hypersphere, which is proportional to R^d, should be minimized in such a way that tightly encompasses all available objects from i_T. The minimization of R^d implies minimization with respect to R^2. Following this, the minimization functional may be formulated as follows:

$$\Theta(a, R) = R^2, \tag{7}$$

with respect to the constraint:

$$\forall_{i \in \{1,\dots,N\}} \quad \|x_i - a\|^2 \leq R^2, \tag{8}$$

where x_i are objects from i_T, and N stands for the quantity of training objects. Additionally, to allow the fact that there may have been some outliers in the training set and to increase the robustness of the trained classifier, some objects with distance to a greater than R are allowed in the training set, but associated with an additional penalty factor. This is done identically as in a standard SVM by the introduction of slack variables ξ_i.

This concept can be further extended to a weighted one-class support vector machine (WOCSVM) (Bicego and Figueiredo 2009) by the introduction of weights w_i that allows for an association of an importance measure to each of the training objects. This forces slack variables ξ_i, to be additionally controlled by w_i. If with object x_i there is associated a small weight w_i, then the corresponding slack variable ξ_i indicates a small penalty. In effect, the corresponding slack variable will be larger, allowing x_i to lie further from the center a of the hypersphere. This reduces an impact of x_i on the shape of a decision boundary of WOCSVM.

Using the above-mentioned ideas, we can modify the minimization functional:

$$\Theta(a, R) = R^2 + C \sum_{i=1}^{N} w_i \xi_i, \tag{9}$$

with the modified constraints that almost all objects are within the hypersphere:

$$\forall_{i\in\{1,\ldots,N\}} \quad \|x_i - a\|^2 \leq R^2 + \xi_i, \tag{10}$$

where $\xi_i \geq 0$, $0 \leq w_i \leq 1$. Here, C stands for a parameter that controls the optimization process—the larger the C, the less the outliers are allowed with the increase of the volume of the hypersphere.

For establishing weights, we may use techniques dedicated to a weighted multi-class support vector machines (Cyganek 2012). In this paper, we propose to use a following formula:

$$w_i = \frac{|x_i - x_{\text{mean}}|}{R+\delta}, \tag{11}$$

where $\delta > 0$ is used to prevent the case of $w_i = 0$. The value of x_{mean} is computed with the usage of all available learning samples. In case of data streams, this would mean that samples from the incoming chunk are used together with the samples from previous chunks that have not been discarded in the forgetting process.

3 Adaptive WOCSVM for incremental data stream

We assume that the classified data stream is given in a form of data chunks. At the beginning, we have at our disposal an initial data set $\mathcal{DS}_0$ that allows to train the first version of the classifier. Then, for each i-th iteration, we receive an additional chunk of data labeled as $\mathcal{DS}_i$. We assume that there is a possibility of concept drift presence in the incoming chunks of data. Therefore, it would be valuable to adjust the one-class classifier to changes in the nature of data. The idea is depicted in Fig. 2. Later, we will train the classifier on the incoming chunk, but it will be employed (evaluated) on the following one.

In case when we are using a WOCSVM trained on $\mathcal{DS}_0$ for all new incoming data, we notice a significant drop in performance—and after few new chunks of data, it is possible that this model will not be able to handle the new objects, as objects in the stream have shifted significantly from their initial distributions. To prevent this from happening, we propose to adapt an one-class classifier incrementally with the new incoming data to deal with the presence of concept drift and allow for a more efficient novelty detection in data streams. We propose to apply the classifier adaptation in a changing environment via modification of weights assigned to objects from the data set. We introduce an incremental learning procedure, meaning that the data set $\mathcal{DS}$ will consist of all available chunks of data at the given i-th moment. Additionally, we augment our model with a forgetting mechanism to discard objects that are no longer relevant to the current state of the analyzed concept. The proposed algorithm is summarized in a pseudo-code manner in Algorithm 1.

Algorithm 1 Adaptive WOCSVM with forgetting

```
Require: input data stream,
    data chunk size,
    classifier training procedure()
    treshold $\varepsilon$
1: $i \leftarrow 0$
2: repeat
3:     collect new data chunk $DS_i$
4:     $DS = DS \cup DS_i$
5:     calculate the weights of the examples according to Eq. (11)
6:     correct the weights of the data from $DS_i$ (according to Eq. (12) if applicable)
7:     correct the weights of the data to simulate forgetting (according to Eq. (13),(14), or (15))
8:     remove from $DS$ all objects with weights smaller than $\varepsilon$
9:     classifier $\leftarrow$ classifier training procedure($DS$, weights assigned to objects in $DS$)
10:    $i \leftarrow i + 1$
11: until end of the input data stream
```

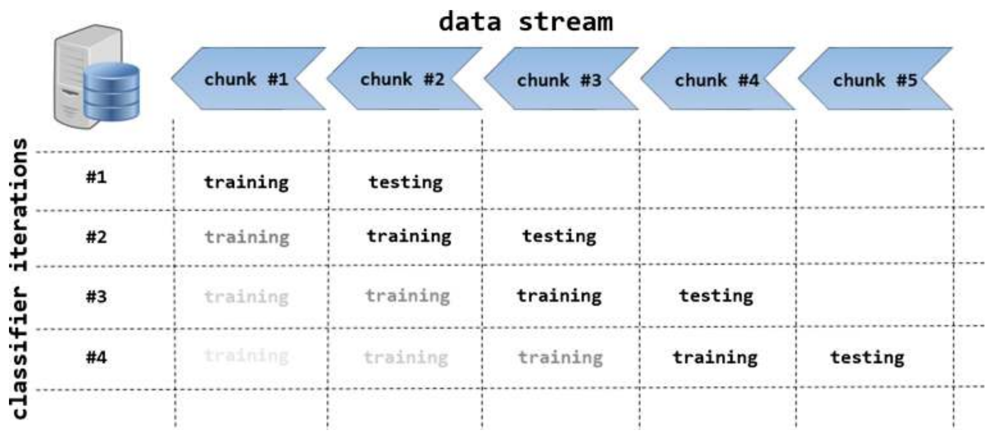

Fig. 2 Algorithm training and testing phases on the successive data chunks

Let us present the details of the crucial algorithm's phases.

3.1 Incremental learning

We propose to extend the WOCSVM concept by adding an incremental learning principle to it (Ross et al. 2008). We use the passive incremental learning. In this method, new data are added without considering its importance. Incremental learning should allow to change the shape of the previously learned decision boundary. Here, we propose two strategies for updating our classifier with new, incoming data by:

- assigning weights to objects from $\mathcal{DS}_k$ according to Eq. (11). This is motivated by the fact that in the incoming data chunk not all objects should have the same impact on the shape of the new decision boundary,
- assigning the highest weights to objects coming from the new data chunks:

  $$\forall_{x_i \in \mathcal{DS}_k} \quad w_i = 1. \tag{12}$$

  This is motivated by the fact that in the presence of the concept drift objects from a new chunk, therefore, should have a top priority in forming the new decision boundary.

3.2 Incremental forgetting

If we apply only the incremental learning principle to the WOCSVM, the decision boundary will become more and more complex with each additional chunk of data. This enlarges our data set and the memory requirements, what is impractical and leads to a poor generalization ability. This can be avoided by forgetting unnecessary, outdated examples (Gent et al. 2012). It seems natural that the degree of importance of data reduces as the time passes.

The simplest way is a removal of objects coming from the previous (or oldest iteration). Yet in this way, we discard all the information they carried—while they still may have some valuable influence on the classification boundary (e.g., in case of a gradual concept drift where the changes of the data distribution are not rapid). A method that allows for a gradual decrease of the object influence over time seems a far more attractive idea.

Identically as in incremental learning, we modify the weights assigned to objects to change the influence of the data on the shape of the decision boundary. In this case, we propose to reduce weights of objects from previous chunks of data in each iteration.

We propose three methods for calculating new weights for objects delivered in previous iterations:

- gradual decrease of weights with respect to their initial importance—here, we introduce a denomination factor τ that is a user-specified value used to decrease the weights in each iteration:

  $$w_i^{k+1} = w_i^k - \tau. \tag{13}$$

  This is motivated by the fact that if an object had initially assigned a higher weight, it had a bigger importance for the classifier. As such, these objects can be valuable for a longer period of time than objects with initial low weights. In this approach, their weights will sooner approach the 0 value and they will be removed in a fewer iterations than objects with high initial weights.
- aligned decrease of weights without considering their initial importance—here, we introduce a time factor κ that is a user-specified value standing for a number of iterations after which the object should be removed:

  $$w_i^{k+1} = w_i^k - (w_i^a/\kappa), \tag{14}$$

  where w_i^a stands for the initial value of the weight assigned to ith object. As we can see, the weights of objects are reduced with each iteration till they are equal to 0 (and removed from $\mathcal{DS}$)—the main difference is that this method does not consider the initial importance of data. This means that all the objects from the k-th data chunk will be removed in the same moment, after κ iterations. This is motivated by the fact that changes in the dynamic environment can be unpredictable and quickly move from the original distribution—therefore, data from previous steps may quickly loose its importance.
- decrease of weights according to the following sigmoidal function:

  $$w_i^{k+1} = w_i^a 2\exp(-\beta(k+1))(1+\exp(-\beta(k+1)) \tag{15}$$

  where β is responsible for the forgetting rapidity. Its value should be determined experimentally. This method allows for a smooth forgetting of previous data with the rapidity of changes controlled by user.

4 Experimental investigations

Our experimental aims were as follow:

- to establish, if applying principles of incremental learning and forgetting in one-class classification will allow to handle data streams with the presence of concept drift efficiently;
- to examine the effectiveness of proposed incremental learning and forgetting schemes that were introduced in this paper.

Table 1 Details of data stream benchmarks used in the experiments

Data set	Objects	Features	Classes	Drift type
RBF	1,000,000	20	2	Gradual
LED	1,000,000	24	10	Gradual
COV	581,012	53	7	Unknown
ELEC	45,312	7	2	Unknown
AIR	539,383	7	2	Unknown

4.1 Data sets

There is an abundance of benchmarks for comparing machine learning algorithms working in static environments. However, for non-stationary data streams, there is still just a few publicly available data sets to work with.[2] Most of them are artificially generated ones, with only some real-life examples. Following the standard approaches found in the literature, we decided to use both artificial and real-life data sets. Additionally, to the best knowledge of authors , there are no data stream benchmarks for one-class classification problems. Therefore, we need to apply a transformation changing existing multi-class streams into one-class data. To do this, we need to select a single class to serve as the target concept and use remaining class(es) as outliers. Details of used benchmarks can be found in Table 1.

Below, we provide a short description of each data set:

- **RBF**: the radial basis function generator outputs a user-defined number of drifting centroids. Each of such centroids is described by a class label, position, weight, and standard deviation. The created data set is defined as a two-class problem with 1,000,000 objects in each class. Four gradual recurring concept drifts were defined. To change this data set into a one-class problem, we use positive class as the target class and negative class as outlier class.
- **LED**: this is an artificial data stream generator, described by 24 features that describe an output of seven-segment LED display. We generate a data set consisting of 1,000,000 objects and with the presence of gradual concept drift. We use digit 0 as the target class and remaining digits as outliers.
- **COV**: forest cover type is a real-life data set that deals with different cover types in four wilderness areas. Each example is described by 53 cartographic features that allow to categorize cover type into one out of seven classes. There are 581,012 examples in this data set. We use Lodgepole Pine (cover type 2) class as target class (as it is the most numerous one, with 283,301 examples) and remaining six classes as outliers.
- **ELEC**: electricity is a real-life data set describing fluctuations in energy prices from the electricity market. This benchmark consists of 45,312 objects, each described by seven features. We use the positive class as the target concept and negative class as outliers.
- **AIR**: airlines in a real-life data set dealing with the problem of predicting flight delays. This data stream consists of 539,383 objects, each described by seven features. We use the positive class (flight on time) as the target concept and negative class (flight delay) as outliers.

[2] http://en.wikipedia.org/wiki/Concept_drift.

4.2 Set-up

For the purposes of experimental analysis, we use as a base model WOCSVM with RBF kernel, $\sigma = 0.1$ and $C = 10$.

We use seven different models of the proposed one-class classifiers in our experiments—each with applied different combination of incremental learning and forgetting mechanisms. The details and abbreviations of these algorithms are given in Table 2. As reference methods, we apply an Incremental and On-line One-Class Support Vector Machine (IOCSVM, with RBF kernel, $\sigma = 0.1$ and $C = 10$) (Zhang et al. 2009) and One-Class Very Fast Decision Tree (OcVFDT) (Li et al. 2009).

The data block size used for creating data chunks was $d = 2{,}500$ for all the data sets. For all experiments, we set the threshold for discarding old objects $\varepsilon = 0.05$

As we deal with one-class task, for training purposes we utilize only objects belonging to the target class in given chunks. For testing, we use all of objects from the incoming chunk (both target class and outliers).

An important aspect of designing an experiment for non-stationary data was choosing an appropriate metric for evaluating examined methods. We decided to measure classification accuracy and time efficiency. We use the data chunk-based evaluation method, described earlier in this paper.

4.3 Results and discussion

The average accuracies achieved by tested algorithms are presented in Table 3, their average training times on data chunk are presented in Table 4, and their average memory usage is presented in Table 5. Additionally, we present detailed accuracy behavior of the proposed incremental learning and forgetting schemes on each of analyzed data sets in Figs. 3, 4, 5, 6 and 7 to allow for a visual inspection of their performance.

The experimental analysis allows us to draw several interesting conclusions about the performance of the proposed schemes for one-class classification in non-stationary environments. First of all, analysis of performance of a standard

Table 2 Details of used versions of the proposed one-class classification model with different incremental learning and forgetting procedures

Model	Description
L_1F_0	Standard WOCSVM without any adaptation mechanism
	Trained on a single incoming chunk, without any previous data
L_1F_1	Incremental learning: assigning weights to new objects according to Eq. (11)
	Forgetting: gradual decrease of weights according to Eq. (13) with $\tau = 0.15$
L_2F_1	Incremental learning: assigning highest weights to new objects
	Forgetting: gradual decrease of weights according to Eq. (13) with $\tau = 0.15$
L_1F_2	Incremental learning: assigning weights to new objects according to Eq. (11)
	Forgetting: aligned decrease of weights without considering their initial importance according to Eq. (14) with $\kappa = 0.1$
L_2F_2	Incremental learning: assigning highest weights to new objects
	Forgetting: aligned decrease of weights without considering their initial importance according to Eq. (14) with $\kappa = 0.1$
L_1F_3	Incremental learning: assigning weights to new objects according to Eq. (11)
	Forgetting: decrease of weights according to proposed function Eq. (15) with $\beta = 9$
L_2F_3	Incremental learning: assigning highest weights to new objects
	Forgetting: decrease of weights according to proposed function Eq. (15) with $\beta = 9$

Table 3 Average classification accuracies (%)

Data set	OcVFDT	IOCSVM	L_1F_0	L_1F_1	L_2F_1	L_1F_2	L_2F_2	L_1F_3	L_2F_3
RBF	62.82	71.06	59.67	71.34	75.69	73.47	74.12	73.82	**77.12**
LED	56.94	64.48	51.98	64.24	68.49	62.21	65.06	64.98	**70.36**
COV	55.90	71.08	57.32	70.35	**75.73**	68.23	72.38	73.11	**75.76**
ELEC	68.39	70.42	62.08	70.15	73.49	70.03	72.94	72.67	**74.04**
AIR	62.72	64.51	61.12	63.84	**66.13**	61.98	64.22	63.87	**66.14**

Bold values indicate the most accurate methods for a given dataset

Table 4 Average chunk training time in seconds [s]

Data set	OcVFDT	IOCSVM	L_1F_0	L_1F_1	L_2F_1	L_1F_2	L_2F_2	L_1F_3	L_2F_3
RBF	1.87	4.63	6.12	6.43	5.14	6.32	5.24	6.33	5.13
LED	1.92	4.62	5.98	6.19	5.02	6.23	6.10	6.28	5.04
COV	2.38	4.06	6.01	6.12	5.00	6.14	5.02	6.14	4.99
ELEC	1.32	3.74	4.56	4.73	4.26	4.68	4.21	4.65	4.22
AIR	0.97	2.10	4.43	4.61	3.19	4.51	3.16	4.38	3.17

Table 5 Average memory consumption in megabytes [MB]

Data set	OcVFDT	IOCSVM	L_1F_0	L_1F_1	L_2F_1	L_1F_2	L_2F_2	L_1F_3	L_2F_3
RBF	1.12	4.62	3.54	4.04	2.78	3.97	2.65	3.82	2.34
LED	0.23	3.17	2.54	2.33	1.25	2.04	1.19	2.12	1.08
COV	0.11	2.03	1.37	2.02	1.09	1.97	1.00	1.92	0.93
ELEC	0.04	1.08	0.53	0.92	0.51	0.91	0.50	0.89	0.46
AIR	0.15	1.76	2.63	1.21	1.54	2.04	1.59	1.97	1.45

WOCSVM shows us that canonical one-class classifiers display a very poor adaptation abilities in the presence of changing concepts. We tested the performance of this classifier without any incremental learning or forgetting scheme, so it was trained each time from a scratch on incoming data chunks. This limited its adaptation, as it did not have any access to previous objects—which can be very important in case of gradual drifts. Without any mechanism to incorporate new observations, while storing the previous ones, standard WOCSVM could not adapt sufficiently its deci-

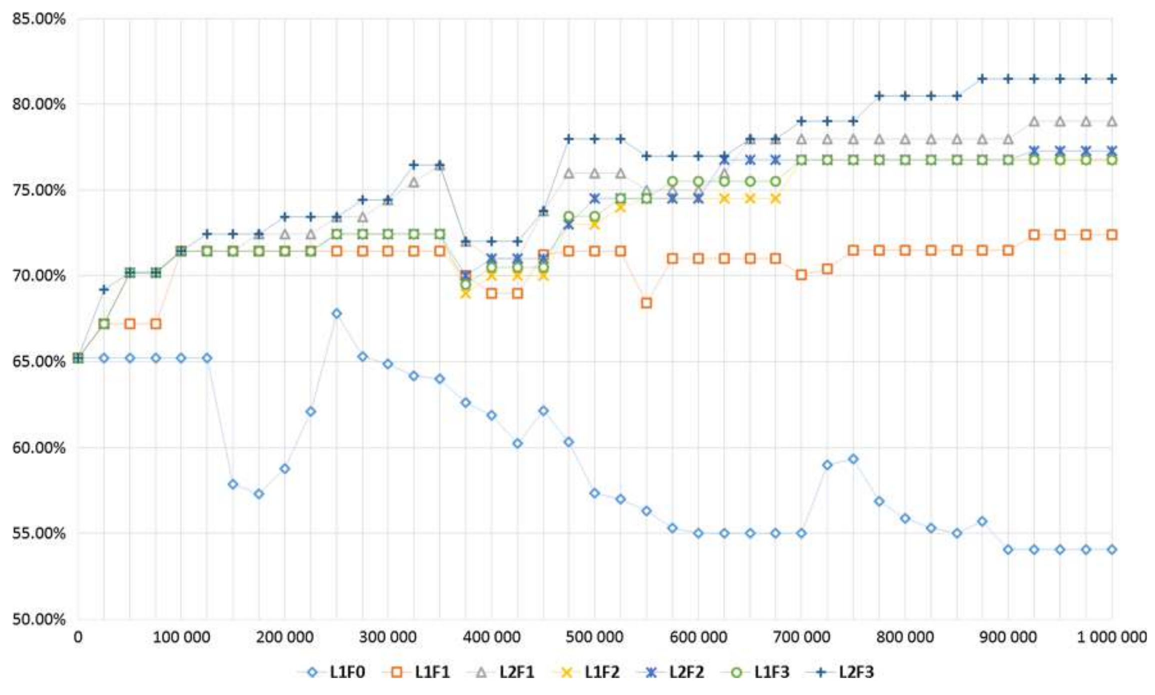

Fig. 3 Accuracies of proposed incremental learning and forgetting methods for RBF data set

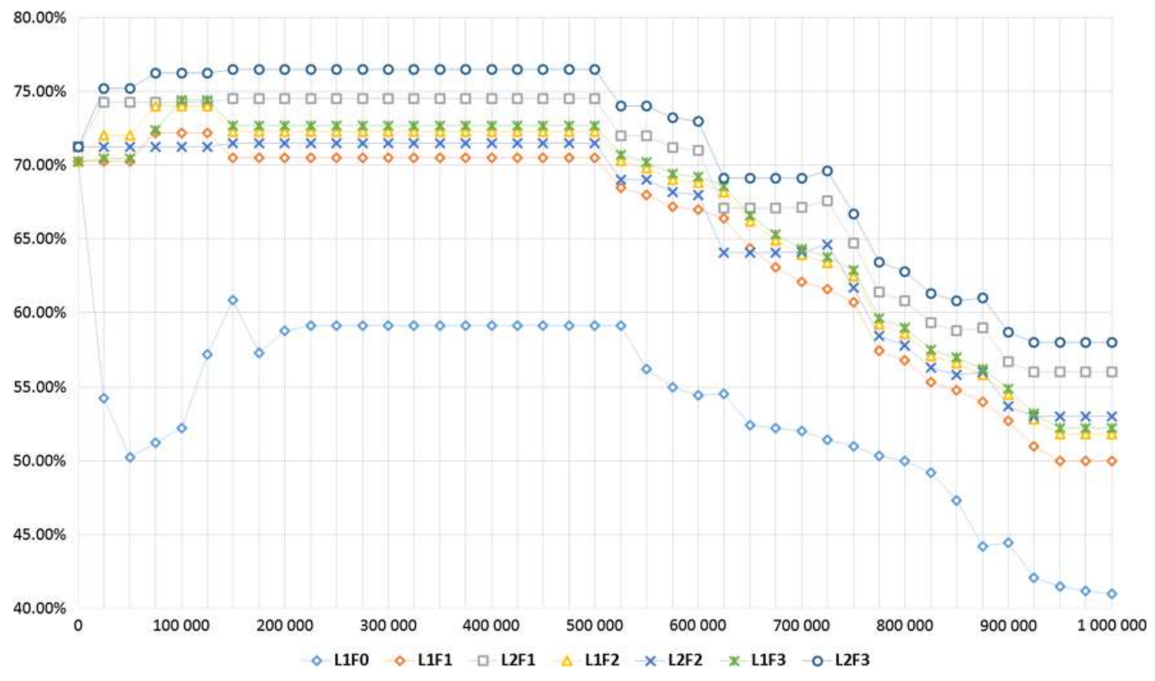

Fig. 4 Accuracies of proposed incremental learning and forgetting methods for LED data set

sion boundary, which resulted in an extended acceptance of outliers.

When comparing our approaches to the reference methods (IOCSVM and OcVFDT), one may see that they outperform these state-of-the-art classifiers for non-stationary single class streams. This is because most used one-class methods for data streams (such as the reference ones) do not have embedded methodologies for dealing with shifting concept—they just work in an incremental way. Our method is able to efficiently adapt to the changing context by forgetting the no longer relevant samples, and thus increasing its overall accuracy. One should note that OcVFDT has much lower training time than our methods, but this is at the cost of a much lower overall accuracy.

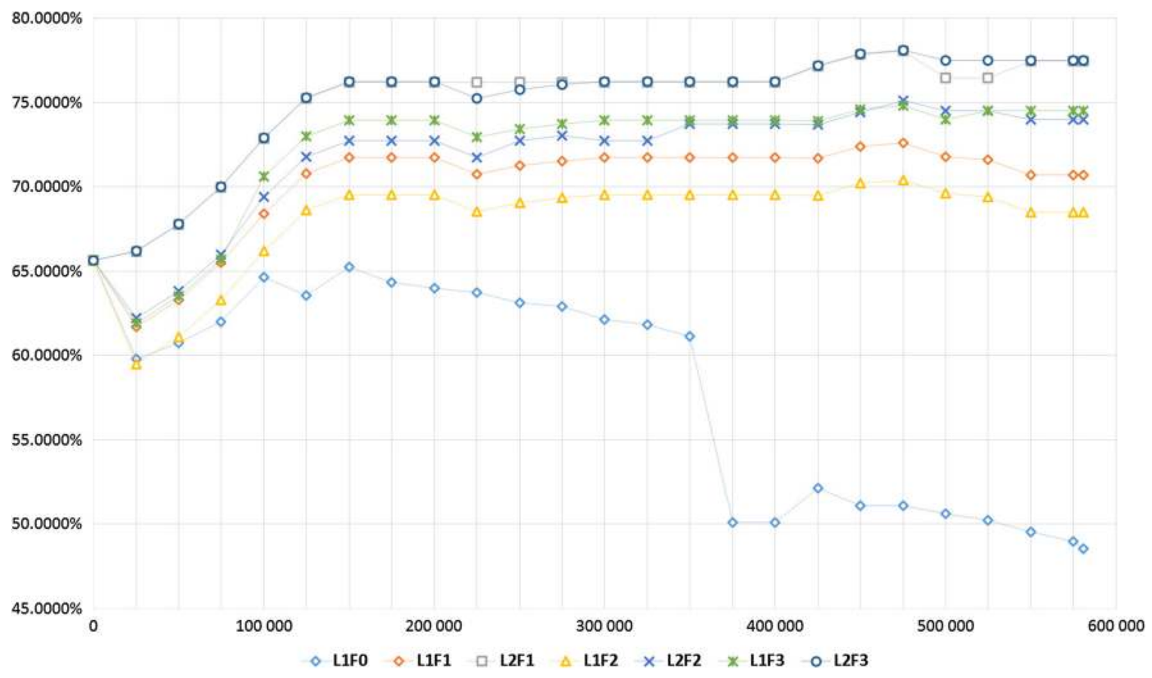

Fig. 5 Accuracies of proposed incremental learning and forgetting methods for COV data set

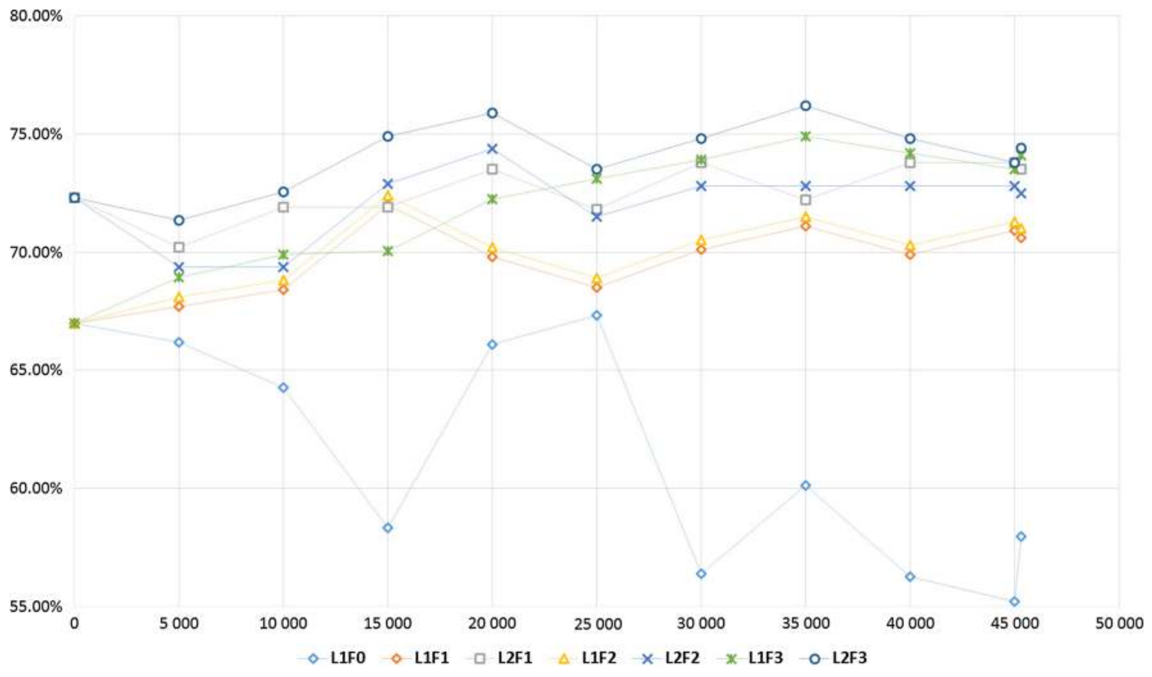

Fig. 6 Accuracies of proposed incremental learning and forgetting methods for ELEC data set

When examining the proposed schemes for incremental learning, one may see two major trends. First, calculating new weights based on the distance returns in most cases an inferior performance to approach based on assigning highest weights for incoming objects. This can be explained by the fact that weight calculation in the first scheme is based on the distance between the new objects and the center of the one-class classifier hypersphere. As the center of the classifier is moving with each new training data chunk (as it is adapting to changes in data), weights quickly lose their meaning with the change of inner-class data distribution. Additionally, data from incoming chunk represent the current state of the classified target concept and its influence on the shape of the decision boundary should be boosted. That is why the

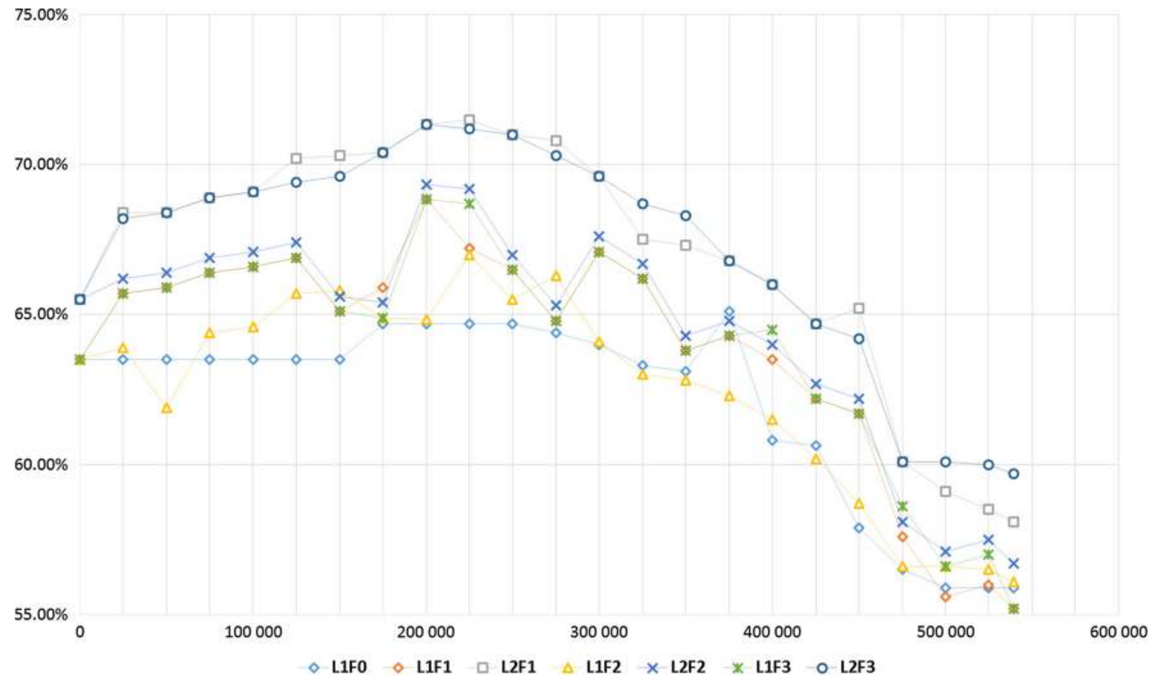

Fig. 7 Accuracies of proposed incremental learning and forgetting methods for AIR data set

simpler method based on assigning highest weights to new object returns more accurate classifiers. Furthermore, this approach significantly reduces the computational time and memory consumption of proposed methods, as can be seen in Tables 4 and 5. This is due to the fact that calculating distances and new weights for incoming examples requires additional processing time, while simply assigning highest weight alleviates the computational load connected with processing new chunks.

As for the forgetting mechanism, we can see their important influence on the proposed method. They allow to store useful information for some time, but allow to gradually reduce their level of influence on the shape of the decision boundary. This is of high importance for constructing efficient models for cases with gradual or recurring drifts. From the three proposed methods, we can see that gradual decrease of weights with respect to their initial importance and decrease of weights according to proposed sigmoidal function returns the best performance. This can be explained by the fact that these methods take into consideration the initial weight of examples and remove them according to their relevance. We can assume that if a given object was important in previous chunk, it can still be of some use in few incoming partitions of data stream (especially in case of slow changes in the environment). Out of these two methods, the one based on our sigmoidal function returns superior results in three out of five data sets. For the remaining two benchmarks, all proposed forgetting mechanisms output similar accuracies. The memory and time consumptions of these methods are quite similar, but one may see that sigmoidal-based forgetting tends to return slightly less complex models.

Taking the mentioned factors into consideration, one may conclude that the best one-class classification model for data streams with the presence of concept drift would be returned by combining incremental learning by assigning highest weights to objects coming from the new data chunk with forgetting by sigmoidal-based function. Experimental results show us that such a combination returns superior results to all other algorithms presented in this paper. Additionally, it has the lowest memory consumption and one of the lowest time complexities. This allows us to draw a conclusion that this model is a good choice for tackling one-class classification for non-stationary data in changing environments both in terms of overall accuracy for dichotomizing between target concept and outliers, and in terms of overall computational complexity and ability to work in real-time pattern classification systems.

5 Conclusions

In this work, we introduced a modified Weighted One-Class Support Vector Machine augmented with the principles of incremental learning and forgetting. These techniques allowed to adapt the decision boundary of the classifier to changes in the incoming data.

Our proposition focused on the modifications of weights used by WOSVM. We employed the incremental training of the weights which was additionally supported by forgetting mechanism. The forgetting boosts classifier's ability to generalization and strongly reduces the amount of stored data, because outdated examples are removed from the memory.

We conclude with suggestion that the efficient one-class classification model for data streams with the presence of concept drift would be returned by combining incremental learning by assigning highest weights to objects coming from the new data chunk with forgetting by sigmoidal-based function. Experimental results show us that such a combination returns superior results to all other algorithms presented in this paper. Additionally, it has the lowest memory consumption and one of the lowest time complexities.

As the achieved results are very promising, then we decided to continue our work with them in the future. The following research direction will be explored:

- implementing and testing new forgetting mechanisms,
- testing our approach on different types of concept drift, especially on sudden shift which requires drift detector and shorter restoration time of the adaptation strategies,
- implementing active learning mechanisms which do not require labels of each examples in a chunk,
- implementing classifier ensemble based on the proposed incremental WOCSVM.

These topics can lead to a new propositions of efficient classifiers for one-class classification in data streams with changing concept.

References

Aggarwal CC, Han J, Wang J, Yu PS (2004) On demand classification of data streams. In: KDD-2004, Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining, pp 503–508 (2004)

Aha DW, Kibler D, Albert MK (1991) Instance-based learning algorithms. Mach Learn 6(1):37–66

Baena-García M, del Campo-Ávila J, Fidalgo R, Bifet A, Gavaldá R, Morales-Bueno R (2006) Early drift detection method. In: Fourth international workshop on knowledge discovery from data streams

Bicego M, Figueiredo MAT (2009) Soft clustering using weighted one-class support vector machines. Pattern Recognit 42(1):27–32

Bifet A, Gavaldà R (2007) Learning from time-changing data with adaptive windowing. In: Proceedings of the seventh SIAM international conference on data mining, Minneapolis

Bifet A, Holmes G, Pfahringer B, Kirkby R, Gavaldà R (2009) New ensemble methods for evolving data streams. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '09ACM, New York, pp 139–148

Bifet A, Holmes G, Pfahringer B, Read J, Kranen P, Kremer H, Jansen T, Seidl T (2011) Moa: a real-time analytics open source framework. In: Proceedings of European conference on machine learning and principles and practice of knowledge discovery in databases (ECML PKDD 2011), Athens. Springer, Heidelberg, pp 617–620

Chu F, Zaniolo C (2004) Fast and light boosting for adaptive mining of data streams. The 5th Pacific-Asia conference on knowledge discovery and data mining (PAKDD), pp 282–292

Cyganek B (2012) One-class support vector ensembles for image segmentation and classification. J Math Imaging Vision 42(2–3):103–117

Domingos P, Hulten G (2003) A general framework for mining massive data streams. J Comput Graph Stat 12:945–949

Duda RO, Hart PE, Stork DG (2001) Pattern classification, 2nd edn. Wiley, New York

Fern A, Givan R (2003) Online ensemble learning: an empirical study. Mach Learn 53(1–2):71–109

Gaber MM, Yu PS (2006) Classification of changes in evolving data streams using online clustering result deviation. In: Proceedings of international workshop on knowledge discovery in data streams

Gama J (2010) Knowledge discovery from data streams. Chapman & Hall/CRC, New York

Gama J (2012) A survey on learning from data streams: current and future trends. Prog AI 1(1):45–55

Gama J, Zliobaite I, Bifet A, Pechenizkiy M, Bouchachia A (2013) A survey on concept drift adaptation. ACM Comput Surv (2013, in press)

Gent IP, Miguel I, Moore NCA (2012) An empirical study of learning and forgetting constraints. AI Commun 25(2):191–208

Gomez-Verdejo V, Arenas-Garcfa J, Lazaro-Gredilla M, Navia-Vazquez A (2011) Adaptive one-class support vector machine. IEEE Trans Signal Process 59(6):2975–2981

Hashemi S, Yang Y, Mirzamomen Z, Kangavari M (2009) Adapted one-versus-all decision trees for data stream classification. IEEE Trans Knowl Data Eng 21(5):624–637

Hulten G, Spencer L, Domingos P (2001) Mining time-changing data streams. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '01ACM, New York, pp 97–106

Jackowski K (2013) Fixed-size ensemble classifier system evolutionarily adapted to a recurring context with an unlimited pool of classifiers. Pattern Analysis and Applications, pp 1–16. doi:10.1007/s10044-013-0318-x

Jackowski K, Platos J (2014) Application of adass ensemble approach for prediction of power plant generator tension. In: de la Puerta JG, Ferreira IG, Bringas PG, Klett F, Abraham A, de Carvalho AC, Herrero l, Baruque B, Quintin H, Corchado E (eds) International joint conference SOCO14-CISIS14-ICEUTE14, Advances in Intelligent Systems and Computing, vol 299. Springer International Publishing, pp 207–216

Jacobs RA, Jordan MI, Nowlan SJ, Hinton GE (1991) Adaptive mixtures of local experts. Neural Comput 3:79–87

Klinkenberg R, Joachims T (2000) Detecting concept drift with support vector machines. In: Proceedings of the seventeenth international conference on machine learning. ICML '00Morgan Kaufmann Publishers Inc., San Francisco, pp 487–494

Koch MW, Moya MM, Hostetler LD, Fogler RJ (1995) Cueing, feature discovery, and one-class learning for synthetic aperture radar automatic target recognition. Neural Netw 8(7–8):1081–1102

Kolter J, Maloof M (2003) Dynamic weighted majority: a new ensemble method for tracking concept drift. In: Third IEEE international conference on data mining, 2003. ICDM 2003, pp 123–130

Kolter JZ, Maloof MA (2007) Dynamic weighted majority: an ensemble method for drifting concepts. J Mach Learn Res 8:2755–2790

Kuncheva LI (2004) Classifier ensembles for changing environments. In: Roli F, Kittler J, Windeatt T (eds) Multiple classifier systems. In: Proceedings of the 5th international workshop, MCS 2004, Cagliari, Lecture Notes in Computer Science, vol 3077. Springer, pp 1–15

Kuncheva LI (2008) Classifier ensembles for detecting concept change in streaming data: overview and perspectives. In: 2nd workshop SUEMA 2008 (ECAI 2008), pp 5–10

Kurlej B, Woźniak M (2012) Active learning approach to concept drift problem. Logic J IGPL 20(3):550–559

Lazarescu MM, Venkatesh S, Bui HH (2004) Using multiple windows to track concept drift. Intell Data Anal 8(1):29–59

Li C, Zhang Y, Li X (2009) Ocvfdt: one-class very fast decision tree for one-class classification of data streams. In: Proceedings of the third international workshop on knowledge discovery from sensor data, SensorKDD '09. ACM, pp 79–86

Littlestone N, Warmuth MK (1994) The weighted majority algorithm. Inf Comput 108(2):212–261

Liu B, Xiao Y, Yu P, Cao L, Zhang Y, Hao Z (2014) Uncertain one-class learning and concept summarization learning on uncertain data streams. IEEE Trans Knowl Data Eng 26(2):468–484

Lughofer E, Angelov P (2011) Handling drifts and shifts in on-line data streams with evolving fuzzy systems. Appl Soft Comput 11(2):2057–2068

Markou M, Singh S (2003) Novelty detection: a review-part 1: statistical approaches. Signal Process 83(12):2481–2497

Masud M, Gao J, Khan L, Han J, Thuraisingham BM (2011) Classification and novel class detection in concept-drifting data streams under time constraints. IEEE Trans Knowl Data Eng 23(6):859–874

Noto K, Brodley C, Slonim D (2012) Frac: a feature-modeling approach for semi-supervised and unsupervised anomaly detection. Data Mining Knowl Discov 25(1):109–133

Ouyang Z, Gao Y, Zhao Z, Wang T (2011) Study on the classification of data streams with concept drift. In: FSKD, IEEE, pp 1673–1677

Polikar R, Upda L, Upda SS, Honavar V (2001) Learn++: an incremental learning algorithm for supervised neural networks. Trans Syst Man Cybern Part C 31(4):497–508

Rodríguez JJ, Kuncheva LI (2008) Combining online classification approaches for changing environments. In: Proceedings of the 2008 joint IAPR international workshop on structural, syntactic, and statistical pattern recognition, SSPR & SPR '08. Springer, Berlin, pp 520–529

Ross DA, Lim J, Lin R, Yang M (2008) Incremental learning for robust visual tracking. Int J Comput Vis 77(1–3):125–141

Schlimmer JC, Granger RH Jr (1986) Incremental learning from noisy data. Mach Learn 1(3):317–354

Schölkopf B, Smola A (2002) Learning with kernels: support vector machines, regularization, optimization, and beyond. Adaptive computation and machine learning. MIT Press, Massachusetts

Shipp CA, Kuncheva L (2002) Relationships between combination methods and measures of diversity in combining classifiers. Inf Fusion 3(2):135–148

Sobolewski P, Wozniak M (2013) Concept drift detection and model selection with simulated recurrence and ensembles of statistical detectors. J Univ Comput Sci 19(4):462–483

Stanley KO (2003) Learning concept drift with a committee of decision trees. In: Artificial intelligence: a modern approach

Street WN, Kim Y (2001) A streaming ensemble algorithm (sea) for large-scale classification. In: Proceedings of the seventh ACM SIGKDD international conference on knowledge discovery and data mining, KDD '01ACM, New York, pp 377–382

Tsymbal A, Pechenizkiy M, Cunningham P, Puuronen S (2008) Dynamic integration of classifiers for handling concept drift. Inf Fusion 9(1):56–68

Wang H, Fan W, Yu PS, Han J (2003) Mining concept-drifting data streams using ensemble classifiers. In: Proceedings of the ninth ACM SIGKDD international conference on knowledge discovery and data mining, KDD '03ACM, New York, pp 226–235

Widmer G, Kubat M (1993) Effective learning in dynamic environments by explicit context tracking. In: Brazdil P (ed) Machine learning: ECML-93, vol 667., Lecture Notes in Computer ScienceSpringer, Berlin, pp 227–243

Widmer G, Kubat M (1996) Learning in the presence of concept drift and hidden contexts. Mach Learn 23(1):69–101

Woźniak M, Cal P, Cyganek B (2014) The influence of a classifiers diversity on the quality of weighted aging ensemble. In: Nguyen N, Attachoo B, Trawinski B, Somboonviwat K (eds) Intelligent information and database systems, Lecture Notes in Computer Science, vol 8398. Springer International Publishing, pp 90–99

Zhang D, Cai L, Wang Y, Zhang L (2010) A learning algorithm for one-class data stream classification based on ensemble classifier. In: 2010 international conference on computer application and system modeling (ICCASM), vol 2, pp V2-596–V2-600 (2010)

Zhang Y, Meratnia N, Havinga P (2009) Adaptive and online one-class support vector machine-based outlier detection techniques for wireless sensor networks. In: International conference on advanced information networking and applications workshops, 2009. WAINA '09, pp 990–995

Zhu X, Ding W, Yu P, Zhang C (2011) One-class learning and concept summarization for data streams. Knowl Inf Syst 28(3):523–553

Zliobaite I (2010) Change with delayed labeling: when is it detectable? In: Proceedings of the 2010 IEEE international conference on data mining workshops, ICDMW '10IEEE Computer Society, Washington, DC, pp 843–850