



Increasing pattern recognition accuracy for chemical sensing by evolutionary based drift compensation

S. Di Carlo ^{a,*}, M. Falasconi ^{b,c}, E. Sanchez ^a, A. Scionti ^a, G. Squillero ^a, A. Tonda ^a

^a Politecnico di Torino, Control and Computer Engineering Department, Torino, Italy

^b SENSOR CNR-IDASC, Brescia, Italy

^c University of Brescia, Department of Chemistry and Physics for Engineering and Materials, Brescia, Italy

ARTICLE INFO

Article history:

Received 8 July 2010

Available online 6 June 2011

Communicated by R.C. Guido

Keywords:

Sensor drift

Evolutionary strategy

Classification systems

ABSTRACT

Artificial olfaction systems, which mimic human olfaction by using arrays of gas chemical sensors combined with pattern recognition methods, represent a potentially low-cost tool in many areas of industry such as perfumery, food and drink production, clinical diagnosis, health and safety, environmental monitoring and process control. However, successful applications of these systems are still largely limited to specialized laboratories. Sensor drift, i.e., the lack of a sensor's stability over time, still limits real industrial setups. This paper presents and discusses an evolutionary based adaptive drift-correction method designed to work with state-of-the-art classification systems. The proposed approach exploits a cutting-edge evolutionary strategy to iteratively tweak the coefficients of a linear transformation which can transparently correct raw sensors' measures thus mitigating the negative effects of the drift. The method learns the optimal correction strategy without the use of models or other hypotheses on the behavior of the physical chemical sensors.

© 2011 Elsevier B.V. All rights reserved.

1. Introduction

The human sense of smell is a valuable tool in many areas of industry such as perfumery, food and drink production, clinical diagnosis, health and safety, environmental monitoring and process control (Gobbi et al., 2010; Vezzoli et al., 2008). Artificial olfaction mimics human olfaction by using arrays of gas chemical sensors combined with pattern recognition (PaRC) methods (Pearce et al., 2003). When a volatile compound comes into contact with the surface of the array, a set of physical changes modifies the electrical properties of the material from which each sensor is composed. This electronic perturbation can be measured, digitalized and used as a feature for the specific compound. A preliminary calibration phase can therefore be used to train a selected PaRC algorithm in order to map each concentration of gas to the responses from the sensor array. The trained model is then used for identification during later measurements. The classification rate of the PaRC system determines the final performance of the electronic olfaction system.

Gas sensor arrays represent a potentially low-cost and fast alternative to conventional analytical instruments such as gas chromatographs. Considerable research into new technologies is underway, including efforts to use nano-engineering to enhance

the performance of traditional resistive Metal Oxide (MOX) sensors. However, successful applications of gas sensor arrays are still largely limited to specialized laboratories (Pardo and Sserveglieri, 2004). Lack of stability over time and the high cost of recalibration are factors which still limit the widespread adoption of artificial olfaction systems in real industrial setups (Padilla et al., 2010).

The *gas sensor drift* consists of small and non-deterministic temporal variations of the sensor response when it is exposed to the same analytes under identical conditions (Padilla et al., 2010). This problem is generally attributed to sensors aging (Sharma et al., 2001), but it could also be influenced by a variety of sources including environmental factors or thermo-mechanical degradation and poisoning (Ionescu et al., 2000). The main result is that the sensors' selectivity and sensitivity decrease. The gas sensor drift changes the way samples distribute in the data space, thus limiting the ability to operate over long periods. PaRC models built in the calibration phase become useless after a period of time, in some cases weeks or a few months. After that time the artificial olfaction system must be completely re-calibrated to ensure valid predictions (Aliwell et al., 2001). It is still impossible to fabricate chemical sensors without drift. In fact, drift phenomena afflict almost all kinds of sensors (Polster et al., 2009; Chen and Chan, 2008; Owens and Wong, 2009). Sensor drift must be therefore detected and compensated to achieve reliable measurements from a sensor array.

Algorithms to mitigate the negative effect of gas sensor drift are not new in the field; the first attempt to tackle this problem dates

* Corresponding author.

E-mail address: stefano.dicarlo@polito.it (S. Di Carlo).

back to the early 90s (Pearce et al., 2003, chap. 13). Nevertheless, the study of sensor drift is still a challenging task for the chemical sensor community (Pearce et al., 2003; Padilla et al., 2010). Solutions proposed in the literature can be grouped into three main categories: (i) periodic calibration, (ii) attuning methods, and (iii) adaptive models.

Retraining the PaRC model by using a single calibrant or a set of calibrants is perhaps the only robust method to mitigate drift effects even in the presence of sensor drift over an extremely long period of time (Sisk and Lewis, 2005). However, calibration is the most time-intensive method for drift correction since it requires system retraining and additional costs. Hence, it must be used sparingly. Moreover, while this approach is quite simple to implement for physical sensors, where the quantity to be measured is exactly known, chemical sensors pose a series of challenging problems. Indeed, in chemical sensing, the choice of the calibrant strongly depends on the specific application especially when the sensing device is composed of a considerable number of cross-correlated sensors (Hines et al., 1999; Haugen et al., 2000). This leads to loss of generalization and lack of standardization which, on the contrary, is an important requirement for industrial systems.

Attuning methods aim to separate and reject drift components from real responses. They can provide significant improvements in the classification rate over a fixed time period, and may also make it possible to obtain real responses to be used in gas quantitative analysis. Within the sensor community, considerable attention has been given to attuning methods performing component correction based on Principal Component Analysis (PCA) (Artursson et al., 2000; Tomic et al., 2004), Independent Components Analysis (ICA) (Natale et al., 2002), Canonical Correlation Analysis (CCA), or Partial Least Squares (PLS) (Gutierrez-Osuna, 2000). Orthogonal Signal Correction (OSC) was recently demonstrated to be one of the best methods to attune PaRC models, and to compensate drift effects (Padilla et al., 2010). However, such techniques do not completely solve the problem. One of the main drawbacks is the need for a set of calibration data containing a significant amount of drift making it possible to precisely identify the set of components to be corrected or rejected. This may not be possible in industrial setups where calibration data are collected over a short period of time. Moreover, the addition of new analytes to the recognition library represents a major problem since rejected components are usually necessary to robustly identify these new classes. Finally, these methods contain no provisions for updating the model and thus may ultimately be invalidated by time evolving drift effects.

Adaptive models aim to adapt the PaRC model by taking into account pattern changes due to drift effects. Neural networks such as self-organizing maps (SOMs) (Marco et al., 1998; Zuppa et al., 2004) or adaptive resonance theory (ART) networks (Vlachos et al., 1997; Llobet et al., 2002) have frequently been used in the past. They have the advantage of being simple because no recalibration is required. Yet, two main weaknesses can be identified. First, a discontinuity in response between two consecutive exposures (regardless of the time interval between the exposures) immediately invalidates the PaRC model and prevents adaptation. Second, one way to obtain reliable results is to set appropriate thresholds for choosing the winning neuron, and this typically requires a high number of training samples increasing the complexity of the network topology. Moreover, they are limited to gas classification applications. Whenever both classification and gas quantitative analysis are required, it would be difficult for current adaptive methods to be applied to obtain reliable gas concentration measurements (Hui et al., 2003).

In this paper we present and discuss an evolutionary based adaptive unsupervised drift-correction methodology designed to work with state-of-the-art classification systems. The term *unsupervised* refers to the fact that drift correction is obtained without

considering any specific drift model. Drift effects are learnt directly from the set of unlabeled raw measures obtained from the sensor array. This work improves our previous attempt to apply evolutionary methods in the drift correction process (DiCarlo et al., 2010). A linear transformation is applied to raw sensor's features to compensate for the effect of the sensor drift. The linear transformation is slowly and continuously evolved to follow the drift modification over time. Evolution is achieved through a covariance matrix adaptation evolution strategy (CMA-ES), perfectly suited for solving difficult optimization problems in the continuous domain. Compared to existing adaptive solutions, the proposed approach adapts to changes in the sensors' responses even when the number of available samples is not high and new classes of elements are introduced in the classification process in different time frames. Experimental results demonstrate that the suitability of the proposed methodology does not depend on the classifier exploited.

2. Methods and theory

Fig. 1 summarizes the main steps and concepts of the proposed drift correction process.

As common in artificial olfaction systems a preliminary calibration phase collects a set of training samples associated to m different classes of gas compounds denoted as y_i ($i \in [1, m]$) (step 0, Fig. 1). Training samples make it possible to train a classifier able to map a generic sample $\mathbf{x} \in \mathbb{R}^n$ (where n is the number of sensors in the array) into one of the m available classes:

$$\mathcal{C} : \mathbf{x} \rightarrow \{y_1, y_2, \dots, y_m\} \quad (1)$$

Any type of classification algorithm can be in principle plugged into this system. The idea behind the proposed drift correction method is to reduce variations caused by the drift in the sensors' response, thus augmenting the validity window of the classification model.

Once the calibration phase is concluded, the system is ready to analyze and classify unknown samples. The drift correction is an iterative process (steps 1–8, Fig. 1). The analysis considers windows of samples. A *window* (W) is a collection of k consecutive measurements obtained from the same sensor array, in which the drift can be assumed as linear (step 1, Fig. 1). Windows are not necessarily associated to measurement sessions. A single measurement session may be split into multiple windows, and multiple sessions may be grouped into a single window depending on the specific application and measurement setup. We denote with $\mathbf{x}_{i,j} \in \mathbb{R}^n$ the j th sample of the i th window W_i . Windows and samples within a window are ordered with ascending sampling time.

Following (Artursson et al., 2000), we assume that sensor drift causes changes in the sensors' response slowly over the time, and that both its direction and intensity for each sample are not randomly distributed.

For each window W_i the drift correction process performs six computational steps (steps 2–7, Fig. 1):

- step 2, Fig. 1: each sample $\mathbf{x}_{i,j} \in W_i$ is corrected by applying a *correction factor* (cf) able to mitigate the effect of the drift (see Section 2.1). The result is a set of *corrected samples* denoted as $\mathbf{xc}_{i,j} \in \mathbb{R}^n$;
- step 3, Fig. 1: each corrected sample $\mathbf{xc}_{i,j}$ is classified using the classifier \mathcal{C} of Eq. (1) trained during the calibration phase (see Section 2.2);
- step 4, Fig. 1: corrected samples and classification results are used in an evolutionary process to adapt the current correction factor to sensor drift changes observed in the current window (see Section 2.3);
- step 5, Fig. 1: each sample $\mathbf{x}_{i,j} \in W_i$ is corrected again by applying the updated correction factor computed during step 4;
- step 6, Fig. 1: corrected samples are classified again;

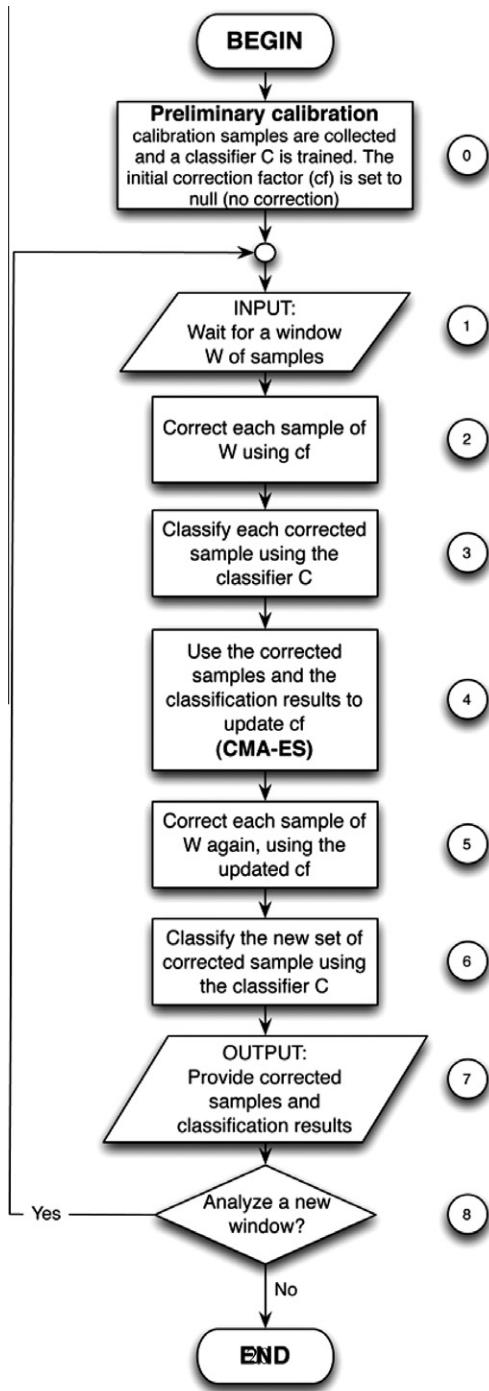


Fig. 1. Conceptual steps of the drift correction process.

- step 7, Fig. 1: the final classification results and corrected samples are provided as outcome of the system.

The same process is repeated until all windows of samples have been analyzed (step 8, Fig. 1).

The following subsections provide details on how the different steps are implemented.

2.1. Correction factor

By considering a sample $\mathbf{x}_{i,j} \in \mathbb{R}^n$ as a point in the n-dimensional space of the sensor array features, the sensor drift effect is a translation of the point along a preferred direction.

In general, although it is well known that for long-term observations the law that governs the sensor drift is non-linear, several experiments show that, in the short/mid term (days or weeks), sensor responses drift in a linear manner, except for the very early time in which sensors are not yet stabilized (Artursson et al., 2000; Nelli et al., 2000; Hui et al., 2003; Llobet et al., 2002). In some cases, depending on the specifications of the sensor's manufacturer, linear models of the sensor response with time have also been used to simulate long-term drift (Marco et al., 1998). This observation represents the basis of most of the attuning methods based on component correction discussed in Section 1.

Under the hypothesis that, in the very short term (samples of a window), the variation imposed by the drift can be considered linear in time, we can envision to apply a correction exploiting a linear transformation. Although this assumption is a limit for previous drift counteractions that do not allow adaptation, it is a good approximation in our case. In fact, when no adaptation is possible, the correction capability worsens in the long term when the hypothesis of linearity does not hold any more. In our specific case, the linearity is assumed only within a single window of samples whose size can be adapted in order to respect this constraint. Moreover, even if a certain portion of non linear drift does exists in short term measurements, according to (Zuppa et al., 2007) we can still assume that it represents a negligible part of the overall phenomena whose main component is linear in nature.

Given a sample $\mathbf{x}_{i,j} \in W_i$ the corrected sample $\mathbf{xc}_{i,j}$ is therefore computed as:

$$\mathbf{xc}_{i,j} = \mathbf{x}_{i,j} + \underbrace{\mathbf{x}_{i,j} \times \mathbf{M}_i}_{\text{correction factor}} \quad (2)$$

where $\mathbf{M}_i \in \mathbb{R}^{n \times n}$ is the *correction matrix* for the window W_i . \mathbf{M} provides the coefficients to compute each corrected feature as a linear combination of all features in the sample. It makes it possible to take into account correlations among sensors in the drift phenomena. The correction factor of feature i of a sample \mathbf{x} can therefore be computed as:

$$cf_i = x[1] \cdot M[1][i] + \cdots + x[n] \cdot M[n][i] \quad (3)$$

The correction matrix for the first window (\mathbf{M}_1) is initially set to the null matrix, i.e., no correction is applied immediately after the calibration.

2.2. Classification

Once sensor drift has been compensated, corrected samples can be classified. State-of-the-art classifiers (e.g., k-NN, Random Forests, etc. (Duda et al., 2000)) can be used in this phase without modifications to their standard implementation. The possibility of choosing different classifiers represents one of the strengths of the proposed method, which allows to select the best PaRC model based on the specific application.

2.3. Correction factor optimization

The correction matrix \mathbf{M}_i , used to correct samples of a window W_i , is continuously adapted when passing from a window to the next one. The overall goal of this optimization process is to update \mathbf{M}_i on the basis of the information provided by samples of W_i . This process makes it possible to follow the evolution of the sensor drift and therefore prepare the system for the analysis of the next window W_{i+1} . The adaptation is obtained using the CMA-ES, a stochastic population-based search method in the continuous domain, which aims at minimizing an objective function $f : \mathcal{S} \subseteq \mathbb{R}^p \rightarrow \mathbb{R}$ in a black-box scenario (see Section A for specific details).

In our specific application the solution computed by the CMA-ES during the elaboration of the window W_i identifies the candidate correction matrix for the window W_{i+1} (\mathbf{M}_{i+1}). We denote with \mathbf{M}^s the correction matrix obtained from the solution $\mathbf{s} \in \mathcal{S} \subseteq \mathbb{R}^{p \times n}$ by computing each element of the matrix as follows:

$$M^s[i][j] = s[(i-1) \cdot n + j], \quad (i \in [1, n], j \in [1, n]) \quad (4)$$

The objective function exploited in this paper is expressed as:

$$f_i(\mathbf{s}) = \sum_{j=0}^{|W_i|-1} D(\mathbf{x}_{i,j} + \mathbf{M}^s \times \mathbf{x}_{i,j}, \mu_c(\mathbf{x}_{i,j})) \quad (5)$$

It sums the distances (D) of each corrected sample in W_i ($\mathbf{x}_{i,j} + \mathbf{M}^s \times \mathbf{x}_{i,j}$) from the centroid of the related class in the training set ($\mu_c(\mathbf{x}_{i,j})$). The centroid of a class y is computed as follows:

$$\mu_y = \frac{\sum_{i=1}^{|y|} \mathbf{t}_i^y}{|y|} \quad (6)$$

where $|y|$ is the number of training samples of class y and \mathbf{t}_i^y is the i th training sample of the class. The function $f_i(\mathbf{s})$ aims to measure how corrected samples tend to deviate from the class distributions learnt by the classifier during the calibration phase.

The evolutionary process terminates on the bases of the following stop conditions:

1. The optimum value of the objective function has been reached. Depending on the considered distance function D in Eq. (5) (see Section 2.4), this value can be set to zero or to a lower bound indicating that all corrected samples have been collapsed into a region closed to the centroid of the related class. Due to the complexity of the optimization process this condition cannot be always reached;
2. During the optimization process, the value of the objective function of each possible couple of candidate solutions in the current population P_c differs less than a predefined threshold ω_{min} :

$$f_i(\mathbf{s}_x) - f_i(\mathbf{s}_y) < \omega_{min} \quad \forall x, y \in P_c \quad (7)$$

3. the step size σ_{cur} of the CMA-ES (see Section A):
 - increases more than a predefined threshold $\bar{\sigma}_{max}$ with respect to its initial value σ_{ini} , i.e., the optimization process is trying to explore an area in the search space that is too large,
 - σ_{cur} decreases more than a predefined threshold $\bar{\sigma}_{min}$, i.e., the optimization process is trying to explore a local minimum

$$\begin{cases} |\sigma_{ini} - \sigma_{cur}| > \bar{\sigma}_{max} \\ |\sigma_{ini} - \sigma_{cur}| < \bar{\sigma}_{min} \end{cases} \quad (8)$$

The initial step size σ_{ini} is used to sample the search space around an initial search point (i.e., a randomly chosen value or a previous solution).

Together with the three defined stop conditions, the optimization is also interrupted if a maximum number of generations has been reached.

Table 1
Parameters resulting from the tuning of each classifier.

Classifier	Parameter	Description	Art. Dataset	Art. Dataset
kNN	k	Number of nearest neighbors	37	21
PLS	Ncomp	Number of components one wishes to fit	4	21
NNET	size	Number of units in the hidden layer	3	5
	Decay	Parameter of weight decay	0.1	0.03
RF	Mtry	Number of variables randomly sampled as candidates at each split	2	4

2.4. Distance functions

Four types of distances have been used in this work to compute the objective function of Eq. (5):

- **Mahalanobis distance:** the Mahalanobis distance computes the distance between two samples by taking into account how samples distribute in the space. It makes it possible to overcome problems deriving from non spherical distributions of samples:

$$D_m(\mathbf{x}, \mu_c) = \sqrt{(\mathbf{x} - \mu_c) \cdot \text{Cov}^{-1} \cdot (\mathbf{x} - \mu_c)^T} \quad (9)$$

where Cov^{-1} is the inverse of the covariance matrix for samples of the training set of class c .

- **Exponential distance:** the Mahalanobis distance of the sample \mathbf{x} is exponentially scaled, as follows:

$$D_x(\mathbf{x}, \mu_c) = e^{D_m(\mathbf{x}, \mu_c)} \quad (10)$$

It exponentially penalizes samples that are moved far from the related centroid.

- **Linear step distance:** the distance of the sample \mathbf{x} from the centroid of its class is computed with a step function as follows:

$$D_{ls}(\mathbf{x}, \mu_c) = \begin{cases} 0 & 0 \leq D_m(\mathbf{x}, \mu_c) \leq D_{m_{max}}^c \\ \frac{D_m(\mathbf{x}, \mu_c)}{D_{m_{max}}^c} - 1 & D_{m_{max}}^c < D_m(\mathbf{x}, \mu_c) \leq 2D_{m_{max}}^c \\ 10^3 & 2D_{m_{max}}^c < D_m(\mathbf{x}, \mu_c) \end{cases} \quad (11)$$

where $D_{m_{max}}^c$ is the maximum Mahalanobis distance of samples of the training set of the class c from the related centroid. This step function gives maximum importance to samples close to the centroid of the related class ($D_{ls}(\mathbf{x}, \mu_c) = 0$) while strongly penalizes samples that are moved far from the centroid of the related class ($D_{ls}(\mathbf{x}, \mu_c) = 10^3$). In the region between the two cases the distance is increased linearly.

- **Exponential step distance:** similarly to D_{ls} the distance is computed with a step function as follows:

$$D_{xs}(\mathbf{x}, \mu_c) = \begin{cases} 0 & 0 \leq D_m(\mathbf{x}, \mu_c) \leq D_{m_{max}}^c \\ e^{D_m(\mathbf{x}, \mu_c)} & D_{m_{max}}^c < D_m(\mathbf{x}, \mu_c) \leq 2D_{m_{max}}^c \\ e^{2D_{m_{max}}^c} & 2D_{m_{max}}^c < D_m(\mathbf{x}, \mu_c) \end{cases} \quad (12)$$

the main difference w.r.t. D_{ls} is the way samples far from the centroid are penalized.

The choice of the best distance function depends on the considered dataset. This represents a degree of freedom that makes it possible to tune the drift correction system for the specific application.

3. Case studies and experimental results

The proposed methodology has been validated on a set of experiments performed on two datasets: the first composed of

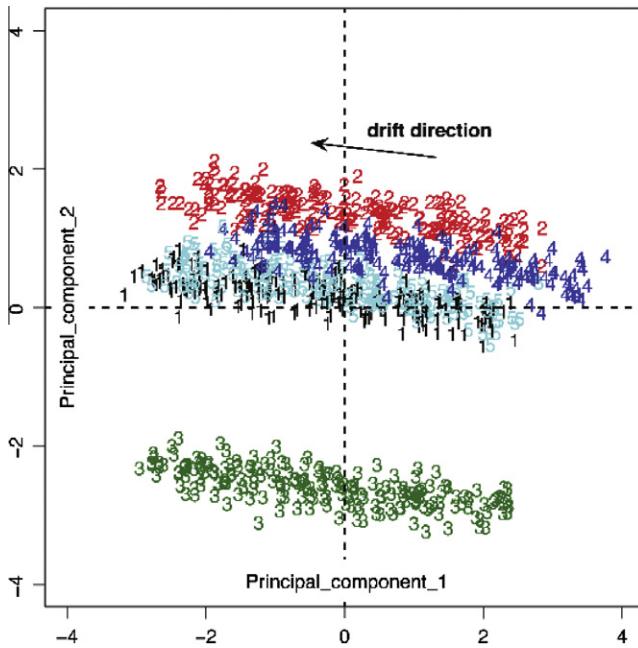


Fig. 2. Projection of the first two principal components of the PCA computed for the artificially generated dataset.

simulated data artificially generated, while the second composed of samples obtained from a real application.

The full correction system has been implemented as a combination of Perl and C code. A pool of four classifiers has been considered: k-Nearest Neighbors (*k*NN), Partial Least Square discriminant analysis (PLS), feed-forward Neural NETwork with a single hidden layer (NNET) and Random Forests (RF). All classifiers have been implemented using the Classification And REgression Training (CARET) package of *R*, a free and multi-platform programming language and software environment widely used for statistical software development and data analysis. Details on the implementation of each classifier are available in Kuhn (2008). The performance of the prediction model of each classifier has been tuned and optimized by performing leave-group-out-cross-validation (LGOCV). For each classifier 50 folds of the training set have been generated with 95% of samples used to train the model while the remaining ones used as test data. The size of the grid used to search the tuning parameters space for each classifier (e.g., *k* for *k*NN) has been set to 5. This represents a good compromise in terms of computational time of the training phase and optimization results. The optimal parameters obtained from the tuning phase are reported in Table 1.

Table 2

Performance of the drift correction system in terms of classification rate on the artificial data set.

Classifier		<i>W</i> ₁	<i>W</i> ₂	<i>W</i> ₃	<i>W</i> ₄	<i>W</i> ₅	<i>W</i> ₆	<i>W</i> ₇	<i>W</i> ₈	<i>W</i> ₉	<i>W</i> ₁₀	<i>W</i> ₁₁	<i>W</i> ₁₂	<i>W</i> ₁₃	<i>W</i> ₁₄	<i>W</i> ₁₅	<i>W</i> ₁₆	<i>W</i> ₁₇	<i>W</i> ₁₈	T.Cr.
<i>Classifiers without drift correction</i>																				
kNN		1.00	0.96	0.98	0.88	0.88	0.80	0.90	0.78	0.78	0.68	0.66	0.62	0.68	0.60	0.60	0.62	0.60	0.58	0.75
NNET		1.00	1.00	1.00	0.96	1.00	0.98	1.00	0.98	0.98	0.94	0.94	0.82	0.92	0.90	0.86	0.86	0.86	0.82	0.93
PLS		1.00	0.96	0.98	0.88	0.92	0.82	0.86	0.80	0.64	0.64	0.56	0.52	0.38	0.32	0.26	0.30	0.26	0.24	0.63
RF		0.86	0.88	0.86	0.80	0.80	0.80	0.78	0.76	0.76	0.74	0.64	0.62	0.62	0.56	0.48	0.48	0.44	0.46	0.68
<i>Drift correction using the Mahalanobis distance D_m</i>																				<i>ΔT.Cr</i>
kNN	Avg	1.00	1.00	0.96	0.99	0.96	0.97	0.95	0.95	0.97	0.95	0.94	0.91	0.89	0.86	0.84	0.81	0.79	0.78	+0.17
	C.I.	.006	.015	.012	.015	.016	.020	.019	.021	.024	.026	.026	.027	.027	.028	.030	.029	.029	.030	
NNET	Avg	0.98	1.00	0.99	0.96	0.97	0.99	0.98	0.99	0.99	0.95	0.97	0.92	0.94	0.91	0.89	0.84	0.84	0.85	+0.02
	C.I.	.002	.000	.003	.003	.003	.002	.004	.002	.002	.013	.012	.015	.015	.015	.017	.020	.026	.029	
PLS	Avg.	1.00	1.00	1.00	0.96	0.99	0.97	0.98	0.98	0.99	0.98	0.99	0.96	0.94	0.94	0.90	0.84	0.80	0.80	+0.31
	C.I.	.001	.000	.001	.005	.004	.005	.006	.005	.002	.004	.006	.013	.015	.017	.020	.025	.030	.031	
RF	Avg	0.99	0.97	1.00	0.99	0.91	0.88	0.84	0.93	0.87	0.85	0.84	0.81	0.83	0.82	0.81	0.82	0.80	0.79	+0.19
	C.I.	.002	.002	.001	.002	.007	.006	.005	.011	.008	.007	.006	.005	.006	.006	.006	.009	.010	.011	
<i>Drift correction using the linear step distance D_{ls}</i>																				<i>ΔT.Cr</i>
kNN	Avg	0.98	0.92	0.94	0.88	0.92	0.89	0.88	0.86	0.88	0.85	0.83	0.81	0.79	0.78	0.76	0.76	0.73	0.73	+0.09
	C.I.	.006	.015	.012	.015	.016	.020	.019	.021	.024	.026	.026	.027	.027	.028	.030	.029	.029	.030	
NNET	Avg	0.97	0.92	0.94	0.88	0.89	0.87	0.87	0.85	0.86	0.82	0.82	0.78	0.79	0.77	0.76	0.76	0.73	0.71	-0.10
	C.I.	.005	.013	.011	.015	.019	.022	.021	.024	.025	.025	.029	.028	.031	.029	.031	.032	.034	.036	
PLS	Avg	0.98	0.91	0.93	0.88	0.91	0.87	0.84	0.84	0.82	0.78	0.76	0.74	0.70	0.67	0.66	0.66	0.62	0.61	+0.16
	C.I.	.006	.016	.013	.016	.020	.023	.025	.026	.032	.034	.037	.039	.040	.041	.043	.044	.045		
RF	Avg	0.98	0.92	0.94	0.92	0.87	0.85	0.81	0.81	0.79	0.78	0.77	0.75	0.74	0.73	0.72	0.69	0.68	+0.12	
	C.I.	.006	.008	.009	.012	.015	.016	.014	.018	.016	.017	.017	.017	.020	.019	.021	.024	.026		
<i>Drift correction using the exponential distance D_x</i>																				<i>ΔT.Cr</i>
kNN	Avg	1.00	0.99	0.98	0.92	0.95	0.92	0.90	0.88	0.88	0.84	0.83	0.80	0.77	0.77	0.76	0.75	0.70	0.69	+0.10
	C.I.	.000	.002	.003	.015	.015	.016	.018	.022	.024	.028	.031	.032	.033	.033	.035	.035	.035	.035	
NNET	Avg	1.00	1.00	1.00	0.94	0.96	0.92	0.80	0.75	0.77	0.72	0.70	0.66	0.66	0.64	0.64	0.62	0.60	0.59	-0.16
	C.I.	.000	.002	.002	.007	.014	.020	.034	.040	.038	.039	.039	.037	.038	.035	.037	.035	.035	.037	
PLS	Avg	1.00	1.00	0.99	0.93	0.95	0.89	0.83	0.79	0.77	0.74	0.72	0.67	0.65	0.66	0.63	0.60	0.57	+0.14	
	C.I.	.000	.000	.003	.013	.014	.024	.031	.034	.037	.038	.037	.040	.038	.041	.041	.040	.040	.038	
RF	Avg	0.92	0.90	0.91	0.90	0.84	0.82	0.80	0.83	0.78	0.76	0.74	0.72	0.70	0.69	0.66	0.67	0.64	0.62	+0.09
	C.I.	.006	.008	.011	.016	.015	.016	.013	.021	.020	.024	.022	.028	.029	.030	.031	.032	.031	.032	
<i>Drift correction using the exponential step distance D_{xs}</i>																				<i>ΔT.Cr</i>
kNN	Avg	1.00	0.99	0.98	0.92	0.92	0.90	0.91	0.88	0.89	0.87	0.86	0.81	0.80	0.79	0.78	0.76	0.73	0.74	+0.11
	C.I.	.000	.003	.006	.011	.016	.016	.016	.019	.021	.023	.024	.027	.025	.028	.028	.026	.027	.028	
NNET	Avg	0.98	0.98	0.98	0.93	0.94	0.94	0.94	0.93	0.92	0.89	0.88	0.82	0.82	0.81	0.80	0.80	0.77	0.76	-0.05
	C.I.	.001	.004	.005	.007	.014	.016	.015	.016	.016	.019	.025	.026	.028	.030	.032	.033	.037	.037	
PLS	Avg	0.99	0.99	0.99	0.94	0.93	0.91	0.91	0.90	0.88	0.87	0.85	0.82	0.79	0.75	0.73	0.72	0.69	0.67	+0.22
	C.I.	.002	.003	.003	.013	.017	.020	.019	.023	.026	.026	.032	.032	.034	.037	.036	.038	.041	.041	
RF	Avg	0.99	0.96	0.99	1.00	0.95	0.95	0.92	0.95	0.91	0.89	0.87	0.83	0.83	0.83	0.80	0.81	0.79	0.76	+0.21
	C.I.	.002	.002	.003	.002	.008	.011	.013	.012	.017	.017	.015	.016	.015	.019	.019	.017	.019	.019	

3.1. Artificial dataset

For a preliminary evaluation we tested the proposed drift correction methodology with a dataset of simulated data. Simulated data allow us to control the parameters that influence the drift correction capability such as feature space dimensionality n , number of classes m , separation among clusters α (given in standard deviation units) and drift direction/intensity.

3.1.1. Experimental setup

Chemical sensor data can be simulated in several ways and there are many factors affecting the actual pattern distribution in the feature space including type of sensor, measured analyte and concentration, cross-correlation among sensors and environmental conditions. General sensors' models that include every mentioned factor are still not available.

In absence of specific information, for a preliminary evaluation of the proposed method, we simulated 1,000 measurements of five independent analytes ($m = 5$) with an array of six gas chemical sensors ($n = 6$) by initially distributing (undrifted) samples according to a Gaussian statistics, as already done in previous published works (Tibshirani et al., 2005; Falasconi et al., 2010). The centroid of each class c is randomly drawn according to a multivariate Normal distribution in n dimensions $\mu_c = \mathcal{N}(\mathbf{0}, \frac{\alpha^2}{2n} \mathbf{I})$ ($\alpha = 12$ in our specific case). Using the term $\frac{\alpha^2}{2n}$ as scaling factor of the variance, the expectation value of the square distance between any two centroids is equal to α^2 independently of n . This makes it possible to have enough separation among classes to build efficient classifiers. In order to control the minimum separation of clusters, we discarded simulations where, due to the randomness of the process, any two centers were closer than $\alpha/2$.

For each class, we generated 250 Gaussian distributed samples with unit variance affected by a drift linear in time according to the following equation:

$$\mathbf{x}(c, t) = \mathcal{N}(\mu_c, \mathbf{I}) + \underbrace{\left(\frac{t}{h} \cdot \mathbf{u}_d \right)}_{\text{drift effect}} \quad (13)$$

where h represents a scaling factor for the discrete time t (h has been set to 40 in our specific case to guarantee a significant amount of drift). The term \mathbf{u}_d represents a randomly generated unitary vector in the n -dimensional space describing the direction of the drift applied to each sample of the dataset. In our simulated data all classes are linearly drifted in the same direction, and samples of the different classes are uniformly distributed in time to present similar drift conditions. The effect of the drift is evident by looking at the projection over the first two principal components of the PCA reported in Fig. 2.

While the initial simulated data are generated according to a pure Gaussian statistics, the introduction of the drift phenomenon that spreads the classes in the feature space introduces a correlation factor among the sensors obtaining a set of data that cannot be considered as pure Gaussian. Following previous publications in the field (Artursson et al., 2000), we assumed that the drift has a preferred direction in the multidimensional feature space, which is similar to assume randomly distributed drift coefficients for each individual sensor.

The reliability of the assumptions on simulated data can be qualitatively observed by comparing the distribution of our simulated data reported in Fig. 2 with the real experimental data reported in Fig. 5. The similarity is noticeable.

The main advantage of this procedure for generating simulated data, when compared to other methods that take into account specific sensor models, is the possibility of easily controlling parameters such as number of sensors, number of features, drift amount and direction, data distribution. This is an important instrument when performing preliminary validation of a drift correction system.

The experimental session included 100 runs of the drift-correction process for each of the four objective functions based on the

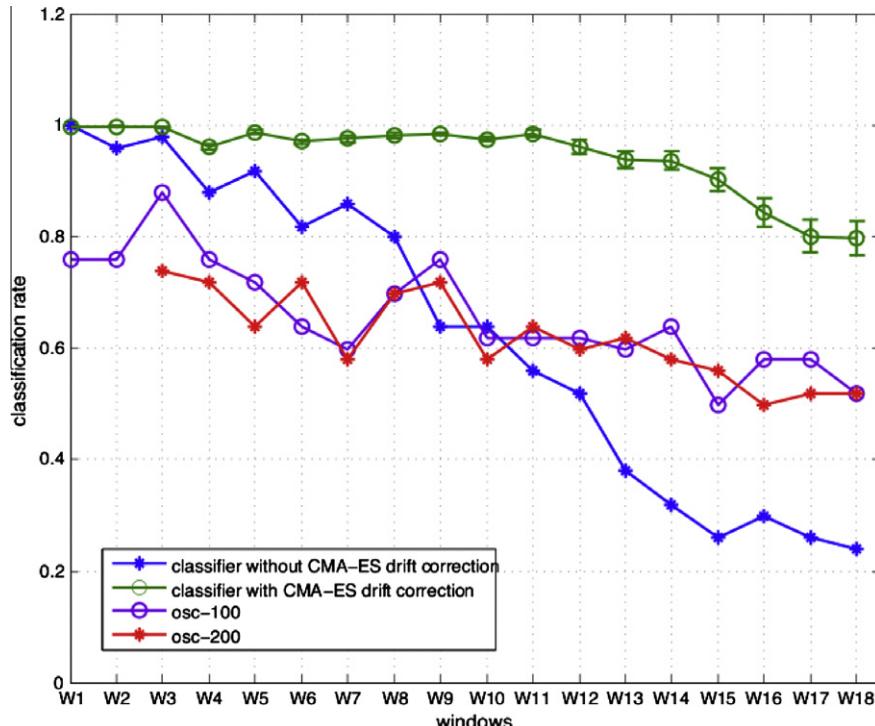


Fig. 3. Comparison of the proposed drift correction systems with the OSC for the PLS classifier with the objective function using the Mahalanobis distance D_m .

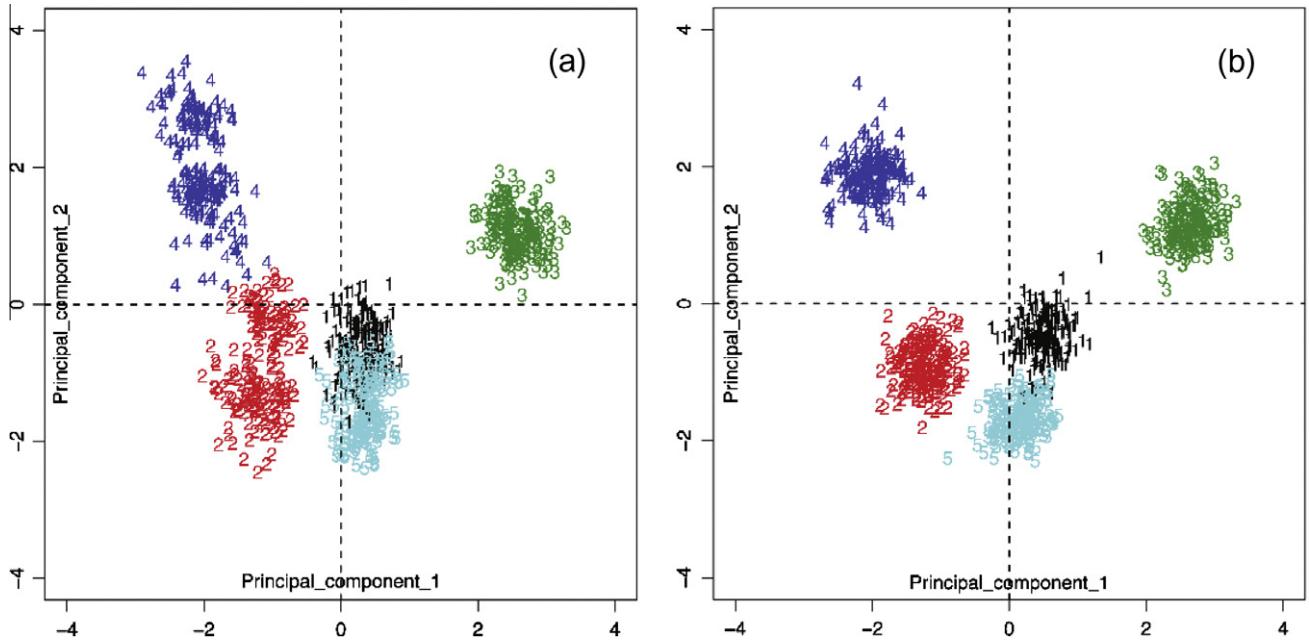


Fig. 4. Comparison of the corrected data set (a) with the original data without drift for the artificial data set (b), using PLS classifier.

distances introduced in Section 2.4. The first 100 samples of the data set have been used as training data for the PaRC model, while the remaining 900 samples have been used as test set to be analyzed. The test set has been processed splitting the data in windows of 50 samples.

3.1.2. Results and discussion

Table 2 shows the performance of the proposed system for the four considered classifiers and the four considered objective functions. Results are provided in terms of classification rate on each of the 18 windows and total classification rate (T. Cr.). To better highlight the benefits of the correction process, **Table 2** reports both the classification rate of each classifier when no correction is applied and the one considering the correction system. Results for the correction system are produced in terms of average classification rate over the 100 considered runs (Avg). In order to evaluate the stability of the results over the different runs, for each average value, the related confidence interval (C.I.), computed considering 95% level of confidence, is reported. The total classification rate is expressed in this case as the variation w.r.t. the one of the classifier without correction.

Results provided in **Table 2** confirm that, in general, for all considered classifiers, the drift correction process improves the classification rate with results that are quite stable over different runs. In particular, the two objective functions based on the Mahalanobis distance (D_m) and the exponential step distance (D_{xs}) seem to provide better results. NNET gained lower improvement due to a quite high original classification rate. On the contrary, the most significant improvement was obtained for PLS corrected with the objective function based on the Mahalanobis distance.

Fig. 3 graphically compares the performance of the proposed drift correction method with the Orthogonal Signal Correction (OSC) that, as introduced in Section 1, represents a state-of-the-art attuning method to perform drift correction. OSC has been implemented using the `osccalc.m` function of the PLS toolbox package (ver. 5.5) for the MATLAB environment (64 bit, ver. 7.9). For the experiments we chose to remove one orthogonal component. Results are evaluated considering the PLS classifier corrected with the objective function based on the Mahalanobis distance.

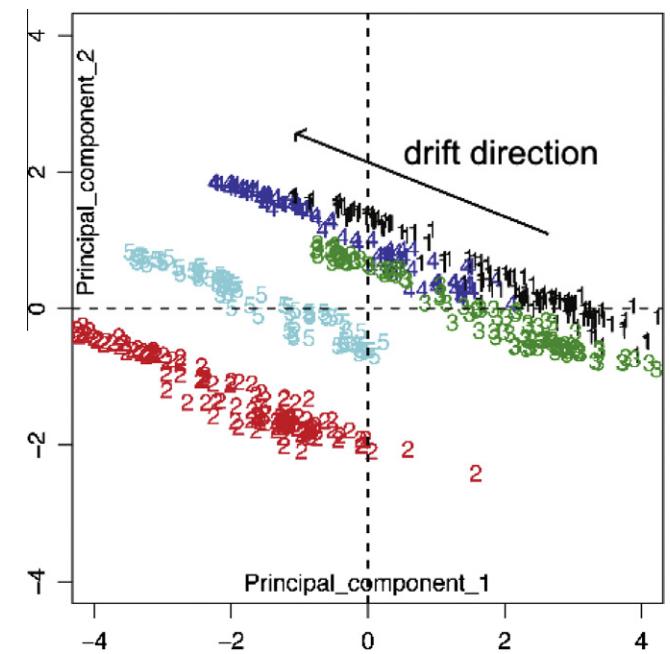


Fig. 5. Projection of the first two principal components of the PCA computed for the real dataset.

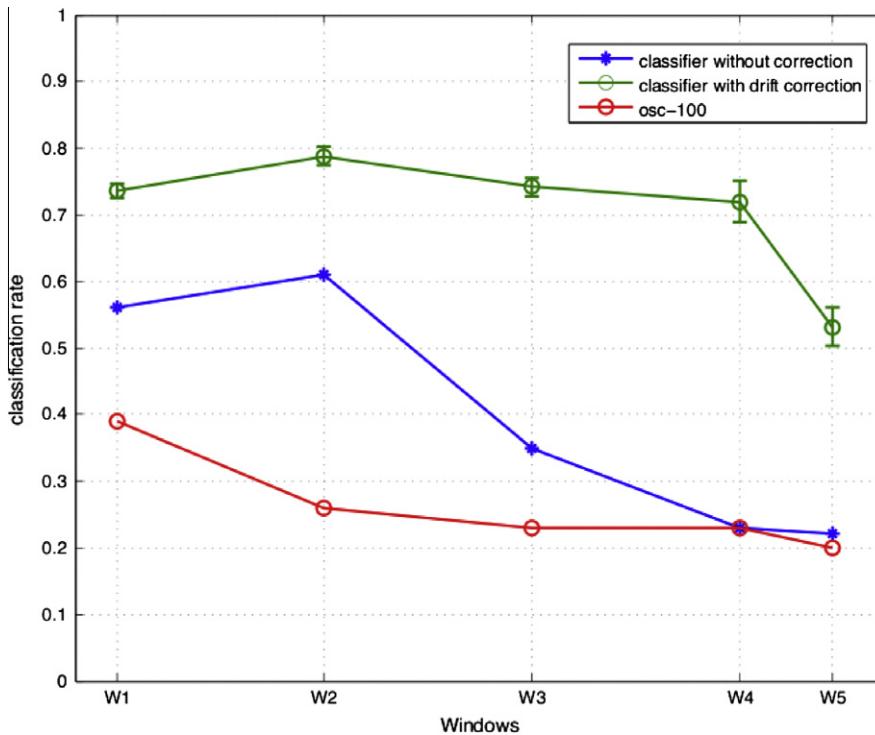
Since the size of the training set strongly influences the effectiveness of this approach, we provided results considering different values for the training set size (100 samples for osc-100 and 200 samples for osc-200) (Padilla et al., 2010). The proposed results clearly show how the proposed method outperforms the OSC requiring a reduced set of training data.

Finally, to show the ability of the correction process to actually remove the drift component from the considered samples, **Fig. 4** graphically shows the projection over the first two principal components for the corrected dataset for one of the runs performed with the PLS classifier using the Mahalanobis distance (**Fig. 4-a**) and for the original data without drift (**Fig. 4-b**). The last set of data

Table 3

Performance of the drift correction system in terms of classification rate on the real set.

Classifier	W_1	W_2	W_3	W_4	W_5	T.Cr	Classifier	W_1	W_2	W_3	W_4	W_5	T.Cr		
<i>Classifiers without drift correction Dm</i>															
kNN	0.63	0.54	0.35	0.32	0.31	0.45									
NNET	0.56	0.65	0.63	0.47	0.36	0.55									
PLS	0.56	0.61	0.35	0.23	0.22	0.42									
RF	0.86	0.86	0.82	0.70	0.69	0.80									
<i>Drift correction using the Mahalanobis distance D_m</i>															
kNN	Avg	0.66	0.62	0.53	0.55	0.52	$\Delta T.Cr$	kNN	Avg	0.20	0.24	0.28	0.22	0.27	$\Delta T.Cr$
	C.I.	.004	.025	.015	.027	.032	+0.13		C.I.	.000	.009	.021	.017	.020	-0.21
NNET	Avg	0.28	0.54	0.52	0.50	0.49	-0.09	NNET	Avg	0.02	0.41	0.26	0.31	0.28	-0.30
	C.I.	.009	.009	.016	.022	.026			C.I.	.010	.022	.018	.029	.022	
PLS	Avg	0.40	0.41	0.28	0.30	0.37	-0.07	PLS	Avg	0.20	0.26	0.29	0.26	0.30	-0.16
	C.I.	.016	.015	.018	.021	.024			C.I.	.000	.015	.021	.020	.022	
RF	Avg	0.90	0.78	0.80	0.80	0.80	+0.02	RF	Avg	0.60	0.64	0.22	0.28	0.40	-0.36
	C.I.	.003	.003	.002	.000	.000			C.I.	.004	.023	.020	.029	.037	
<i>Drift correction using the linear step distance D_{ls}</i>															
kNN	Avg	0.89	0.72	0.55	0.56	0.49	$\Delta T.Cr$	kNN	Avg	0.71	0.74	0.53	0.53	0.51	$\Delta T.Cr$
	C.I.	.006	.022	.023	.033	.033	+0.21		C.I.	.013	.013	.012	.018	.021	+0.16
NNET	Avg	0.81	0.70	0.71	0.68	0.54	+0.16	NNET	Avg	0.70	0.78	0.72	0.82	0.64	+0.19
	C.I.	.013	.023	.030	.046	.038			C.I.	.002	.006	.007	.022	.034	
PLS	Avg	0.86	0.67	0.53	0.55	0.41	+0.21	PLS	Avg	0.74	0.79	0.74	0.72	0.53	+0.31
	C.I.	.007	.017	.025	.031	.029			C.I.	.011	.013	.014	.031	.029	
RF	Avg	0.95	0.91	0.86	0.85	0.83	+0.08	RF	Avg	0.88	0.77	0.81	0.81	0.90	+0.02
	C.I.	.002	.012	.010	.023	.031			C.I.	.004	.002	.003	.005	.019	
<i>Drift correction using the exponential step distance D_{xs}</i>															

**Fig. 6.** Comparison of the proposed drift correction systems with the OSC for the PLS classifier with objective function using the exponential step distance D_{xs} .

was stored during the generation of the artificial dataset before inserting the drift component (see Eq. 13). Both plots have been generated using the same projection to allow comparison. The figure confirms how the drift observed in Fig. 2 has been strongly mitigated, producing a distribution of samples that approximates the one without drift.

This is an important result that makes it possible to perform quantitative gas analysis and further examinations on the corrected data, overcoming one of the main problems of previous adaptive correction methods (see Section 1).

3.2. Real dataset

To additionally validate the proposed approach we also performed a set of experiments on a real data set collected at the *SENSOR Lab*, an Italian research laboratory specialized in the development of chemical sensor arrays.¹ All data have been collected using an EOS835 electronic nose composed of 6 chemical

¹ <http://sensor.ing.unibs.it/>.

MOX sensors. Further information on sensors and used equipments can be found in (Pardo and Sberveglieri, 2004) and its references. The goal of the experiment is to determine whether the EOS835 can identify five pure organic vapors: ethanol (class 1), water (class 2), acetaldehyde (class 3), acetone (class 4), ethyl acetate (class 5). All these are typical chemical compounds to be detected in real-world applicative scenarios.

3.2.1. Experimental setup

A total of five different sessions of measurements were performed over one month to collect a dataset of 545 samples, a high value compared to other real datasets reported in the literature. While the period of time was not very long, it was enough to obtain data affected by a certain amount of drift. Not all classes of compounds have been introduced since the first session, mimicking a common practice in real-world experiments. Samples of classes 1 and 2 have been introduced since the beginning; class 3 is first introduced during the second session, one week later; first occurrences of classes 4 and 5 appear only during the third session, 10 days after the beginning of the experiment. Classes are not perfectly balanced in terms of number of samples, with a clear predominance of classes 1, 2 and 3 over classes 4 and 5. All these peculiarities make this dataset complex to analyze allowing us to stress the capability of the proposed correction system. The effect of the drift is evident by looking at the projection over the first two principal components of the PCA reported in Fig. 5.

As for the artificial dataset the experimental session included 100 runs of the drift-correction process for each of the four considered objective functions. The first 20 samples of each class have been used as training data for the ParC model, while the remaining 445 samples have been used as test set. The drift correction process has been applied to windows of 100 samples, with the last one of 45 samples. The bigger size of the windows compared to the artificial dataset is required to tackle the additional complexity of the real data.

3.2.2. Results

Table 3 summarizes the performance of the drift correction system on the real data set.

Results immediately highlight how the correction process for this particular experiment is harder when compared to the case study with artificial data. The main difficulty is connected to the fact that samples from different classes are introduced non homogeneously over the time and the initial interclass distance among the centroids is not enough to avoid partial overlapping of the classes. Moreover, the use of bigger windows increases the effort required by the CMA-ES to compute the appropriate correction matrices. However, the exponential step distance still produces interesting improvements in the classification rate. Looking also at the results of the artificial dataset this distance seems the best compromise to work with generic data.

PLS corrected with the objective function based on the exponential step distance is the classifier that gained better improvements. **Fig. 6** compares again the results for this case with the correction obtained applying the OSC. This time, due to the limited amount of samples, a single case with 100 samples of training has been considered. Again the proposed drift correction approach performs better than the OSC.

4. Conclusions

In this paper, we proposed an evolutionary based approach able to counteract drift phenomenon affecting gas sensor arrays. The presented methodology is based on a computational flow that corrects and classifies drifted samples applying a linear correction fac-

tor and then using state-of-the-art classification methods. Samples are elaborated in windows after a preliminary calibration phase required to build the initial prediction model to perform classifications.

The correction factor is continuously adapted exploiting an evolutionary process. Corrected samples and classification results feed the evolutionary process that updates the correction factor mitigating the effects caused by the sensor drift accumulated during the current classification window.

As experimentally demonstrated, the proposed approach is flexible enough to work with different state-of-the-art classification algorithms and does not rely on complex drift models to perform its correction. In fact, the evolutionary process applied periodically alleviates the effects caused by the sensor drift.

In order to experimentally assess the method, we performed different experiments with artificial and real data sets. In the first case, drift was artificially produced following a predetermined trend in the samples. In the second case, a real data set considering five pure organic vapors was exploited. Results collected on both cases experimentally corroborate that the proposed methodology performs better than state-of-the-art methods, such as OSC.

The critical element of the proposed system is represented by the employed fitness function, which is one of the most difficult problems that arise when an evolutionary algorithm is exploited in a complex environment. Several fitness functions based on different concepts of distance among samples have been experimentally tested with good results in this paper. However, selecting the best function for a given experimental setup is not trivial.

We are currently working on designing a more robust and generic fitness function able to provide reliable results in a wide range of use cases. Additionally, future works also include new experiments on real data sets affected by different models of both linear and non-linear drift.

Appendix A. CMA-ES

The covariance matrix adaptation evolution strategy (CMA-ES) is an optimization method first proposed by Hansen, Ostermeier, and Gawelczyk (Hansen et al., 1995) in mid 90s, and further developed in subsequent years (Hansen and Ostermeier, 2001; Hansen et al., 2003).

Similar to quasi-Newton methods, the CMA-ES is a second-order approach estimating a positive definite matrix within an iterative procedure. More precisely, it exploits a *covariance matrix*, closely related to the inverse Hessian on convex-quadratic functions. The approach is best suited for difficult non-linear, non-convex, and non-separable problems, of at least moderate dimensionality (i.e., $n \in [10, 100]$). In contrast to quasi-Newton methods, the CMA-ES does not use, nor approximate gradients, and does not even presume their existence. Thus, it can be used where derivative-based methods, e.g., Broyden–Fletcher–Goldfarb–Shanno or conjugate gradient, fail due to discontinuities, sharp bends, noise, local optima, etc.

In CMA-ES, iteration steps are called *generations* due to its biological foundations. The value of a generic algorithm parameter y during generation g is denoted with $y^{(g)}$. The mean vector $\mathbf{m}^{(g)} \in \mathbb{R}^n$ represents the favorite, most-promising solution so far. The step size $\sigma^{(g)} \in \mathbb{R}_+$ controls the step length, and the covariance matrix $\mathbf{C}^{(g)} \in \mathbb{R}^{n \times n}$ determines the shape of the distribution ellipsoid in the search space. Its goal is, loosely speaking, to fit the search distribution to the contour lines of the objective function f to be minimized. $\mathbf{C}^{(0)} = \mathbf{I}$

In each generation g , λ new solutions $\mathbf{x}_i^{(g+1)} \in \mathbb{R}^n$ are generated by sampling a multi-variate normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{C})$ with mean $\mathbf{0}$ (see Eq. (A.1)).

$$\mathbf{x}_k^{(g+1)} \sim \mathcal{N}(\mathbf{m}^{(g)}, (\sigma^{(g)})^2 \mathbf{C}^{(g)}), \quad k = 1, \dots, \lambda \quad (\text{A.1})$$

Where the symbol \sim denotes the same distribution on the left and right side.

After the sampling phase, new solutions are evaluated and ranked. $\mathbf{x}_{i:\lambda}$ denotes the i th ranked solution point, such that $f(\mathbf{x}_{1:\lambda}) \leq \dots \leq f(\mathbf{x}_{\lambda:\lambda})$. The μ best among the λ are selected and used for directing the next generation $g + 1$. First, the distribution mean is updated (see Eq. A.2).

$$\mathbf{m}^{(g+1)} = \sum_{i=1}^{\mu} w_i \mathbf{x}_i^{(g)}, \quad w_1 \geq \dots \geq w_{\mu} > 0, \quad \sum_{i=1}^{\mu} w_i = 1 \quad (\text{A.2})$$

In order to optimize its internal parameters, the CMA-ES tracks the so-called *evolution paths*, sequences of successive normalized steps over a number of generations. $\mathbf{p}_{\sigma}^{(g)} \in \mathbb{R}^n$ is the conjugate evolution path. $\mathbf{p}_{\sigma}^{(0)} = \mathbf{0}$. $\sqrt{2} \frac{\Gamma(\frac{n+1}{2})}{\Gamma(\frac{n}{2})} \approx \sqrt{n} + \mathcal{O}(\frac{1}{n})$ is the expectation of the Euclidean norm of a $\mathcal{N}(\mathbf{0}, \mathbf{I})$ distributed random vector, used to normalize paths. $\mu_{\text{eff}} = (\sum_{i=1}^{\mu} w_i^2)^{-1}$ is usually denoted as *variance effective selection mass*. Let $c_{\sigma} < 1$ be the learning rate for cumulation for the rank-one update of the covariance matrix; $d_{\sigma} \approx 1$ be the damping parameter for step size update. Paths are updated according to Eqs. (A.3) and (A.4).

$$\mathbf{p}_{\sigma}^{(g+1)} = (1 - c_{\sigma}) \mathbf{p}_{\sigma}^{(g)} + \sqrt{c_{\sigma}(2 - c_{\sigma}) \mu_{\text{eff}}} \mathbf{C}^{(g)-\frac{1}{2}} \frac{\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}} \quad (\text{A.3})$$

$$\sigma^{(g+1)} = \sigma^{(g)} \exp \left(\frac{c_{\sigma}}{d_{\sigma}} \left(\frac{\|\mathbf{p}_{\sigma}^{(g+1)}\|}{\sqrt{2} \frac{\Gamma(\frac{n+1}{2})}{\Gamma(\frac{n}{2})}} - 1 \right) \right) \quad (\text{A.4})$$

$\mathbf{p}_c^{(g)} \in \mathbb{R}^n$ is the evolution path, $\mathbf{p}_c^{(0)} = \mathbf{0}$. Let $c_c < 1$ be the learning rate for cumulation for the rank-one update of the covariance matrix. Let μ_{cov} be parameter for weighting between rank-one and rank- μ update, and $c_{\text{cov}} \leq 1$ be learning rate for the covariance matrix update. The covariance matrix \mathbf{C} is updated (Eqs. (A.5) and (A.6)).

$$\mathbf{p}_c^{(g+1)} = (1 - c_c) \mathbf{p}_c^{(g)} + \sqrt{c_c(2 - c_c) \mu_{\text{eff}}} \frac{\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}} \quad (\text{A.5})$$

$$\begin{aligned} \mathbf{C}^{(g+1)} = & (1 - c_{\text{cov}}) \mathbf{C}^{(g)} + \frac{c_{\text{cov}}}{\mu_{\text{cov}}} \times \left(\mathbf{p}_c^{(g+1)} \mathbf{p}_c^{(g+1)\top} + \delta(h_{\sigma}^{(g+1)}) \mathbf{C}^{(g)} \right) \\ & + c_{\text{cov}} \left(1 - \frac{1}{\mu_{\text{cov}}} \right) \sum_{i=1}^{\mu} w_i \text{OP} \left(\frac{\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}} \right) \end{aligned} \quad (\text{A.6})$$

where $\text{OP}(\mathbf{X}) = \mathbf{X}\mathbf{X}^T = \text{OP}(-\mathbf{X})$.

Most noticeably, the CMA-ES requires almost no parameter tuning for its application. The choice of strategy internal parameters is not left to the user, and even λ and μ default to acceptable values. Notably, the default population size λ is comparatively small to allow for fast convergence. Restarts with increasing population size have been demonstrated (Auger and Hansen, 2005) to be useful for improving the global search performance, and it is nowadays included an option in the standard algorithm.

References

- Aliwell, S.R., Halsall, J.F., Pratt, K.F.E., O'Sullivan, J., Jones, R.L., Cox, R.A., Utembe, S.R., Hansford, G.M., Williams, D.E., 2001. Ozone sensors based on WO_3 : A model for sensor drift and a measurement correction method. *Meas. Sci. Technol.* 12 (6), 684–690.
- Artursson, T., Eklov, T., Lundström, I., Mårtensson, P., Sjöström, M., Holmberg, M., 2000. Drift correction for gas sensors using multivariate methods. *J. Chemom. Special Issue: Proc. SSC6* 14 (5–6), 711–723.
- Auger, A., Hansen, N., 2005. A restart cma evolution strategy with increasing population size. In: Proc. IEEE Congr. Evol. Comput. vol. 2. pp. 1769–1776.
- Chen, D., Chan, P., 2008. An intelligent ISFET sensory system with temperature and drift compensation for long-term monitoring. *Sens. J. IEEE* 8 (12), 1948–1959.
- Di Carlo, S., Falasconi, M., Sánchez, E., Scionti, A., Squillero, G., Tonda, A., 2010. Exploiting evolution for an adaptive drift-robust classifier in chemical sensing. *Appl. Evol. Comput.*, 412–421.
- Duda, R.O., Hart, P.E., Stork, D.G., 2000. *Pattern Classification*, 2nd ed. Wiley-Interscience.
- Falasconi, M., Gutierrez, A., Pardo, M., Sberveglieri, G., Marco, S., 2010. A stability based validity method for fuzzy clustering. *Pattern Recognit.* 43 (4), 1292–1305.
- Gobbi, E., Falasconi, M., Concina, I., Mantero, G., Bianchi, F., Mattarozzi, M., Musci, M., Sberveglieri, G., 2010. Electronic nose and alicyclobacillus spp. spoilage of fruit juices: An emerging diagnostic tool. *Food Control* 21 (10), 1374–1382.
- Gutierrez-Osuna, R., 2000. Drift reduction for metal-oxide sensor arrays using canonical correlation regression and partial least squares July 20–24. Proc. 7th Internat. Symp. Olfaction and Electronic Nose. Institute of Physics Publishing, p. 147.
- Hansen, N., Müller, S.D., Petrosnff, P.K., 2003. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evol. Comput.* 11, 1–18.
- Hansen, N., Ostermeier, A., 2001. Completely derandomized self-adaptation in evolution strategies. *Evol. Comput.* 9, 159–195.
- Hansen, N., Ostermeier, A., Gawelczyk, A., 1995. On the adaptation of arbitrary normal mutation distributions in evolution strategies. In: The Generating Set Adaptation. Proc. 6th Internat. Conf. Genetic Algorithms. Morgan Kaufmann, pp. 312–317.
- Haugen, J.-E., Tomic, O., Kvål, K., 2000. A calibration method for handling the temporal drift of solid state gas-sensors. *Anal. Chim. Acta* 407 (1–2), 23–39.
- Hines, E., Llobet, E., Gardner, J., 1999. Electronic noses: A review of signal processing techniques. *Circuits Devices Syst. IEE Proc.* 146 (6), 297–310.
- Hui, D., Jun-Hua, L., Zhong-Ru, S., 2003. Drift reduction of gas sensor by wavelet and principal component analysis. *Sens. Actuators B: Chem.* 96 (1–2), 354–363.
- Ionescu, R., Vancu, A., Tomescu, A., 2000. Time-dependent humidity calibration for drift corrections in electronic noses equipped with sno2 gas sensors. *Sens. Actuators B: Chem.* 69 (3), 283–286.
- Kuhn, K., 2008. Building predictive models in r using the caret package. *J. Stat. Softw.* 28 (5), 1–26.
- Llobet, E., Brezmes, J., Ionescu, R., Vilanova, X., Al-Khalifa, S., Gardner, J.W., Bârsan, N., Correig, X., 2002. Wavelet transform and fuzzy armap-based pattern recognition for fast gas identification using a micro-hotplate gas sensor. *Sens. Actuators B: Chem.* 83 (1–3), 238–244.
- Marco, S., Ortega, A., Pardo, A., Samitier, J., 1998. Gas identification with tin oxide sensor array and self-organizing maps: Adaptive correction of sensor drifts. *IEEE Trans. Instrum. Meas.* 47 (1), 316–321.
- Natale, C.D., Martinelli, E., D'Amico, A., 2002. Counteraction of environmental disturbances of electronic nose data by independent component analysis. *Sens. Actuators B: Chem.* 82 (2–3), 158–165.
- Nelli, P., Faglia, G., Sberveglieri, G., Cereda, E., Gabetta, G., Dieguez, A., Romano-Rodriguez, A., Morante, J.R., 2000. The aging effect on sno2-au thin film sensors: electrical and structural characterization. *Thin Solid Films* 371 (1–2), 249–253.
- Owens, W.B., Wong, A.P.S., 2009. An improved calibration method for the drift of the conductivity sensor on autonomous ctd profiling floats by theta-s climatology. *Deep-Sea Res. Part I-Oceanogr. Res. Papers* 56 (3), 450–457.
- Padilla, M., Perera, A., Montoliu, I., Chaudry, A., Persaud, K., Marco, S., 2010. Drift compensation of gas sensor array data by orthogonal signal correction. *Chemom. Intell. Lab. Syst.* 100 (1), 28–35.
- Pardo, M., Sberveglieri, G., 2004. Electronic olfactory systems based on metal oxide semiconductor sensor arrays. *MRS Bull.* 29 (10), 703–708.
- Pearce, T.C., Shiffman, S.S., Nagle, H.T., Gardner, J.W., 2003. *Handbook of machine olfaction*. Wiley-VHC, Weinheim.
- Polster, A., Fabian, M., Villinger, H., 2009. Effective resolution and drift of parascientific pressure sensors derived from long-term seafloor measurements. *Geochem. Geophys. Geosyst.*, 10.
- Sharma, R.K., Chan, P.C.H., Tang, Z., Yan, G., Hsing, I.-M., Sin, J.K.O., 2001. Investigation of stability and reliability of tin oxide thin-film for integrated micro-machined gas sensor devices. *Sens. Actuators B: Chem.* 81 (1), 9–16.
- Sisk, B.C., Lewis, N.S., 2005. Comparison of analytical methods and calibration methods for correction of detector response drift in arrays of carbon black-polymer composite vapor detector. *Sens. Actuators B: Chem.* 104 (2), 249–268.
- Tibshirani, R., Walther, G., 2005. Cluster Validation by Prediction Strength. *J. Comput. Graph. Stat.* 14 (3), 511–528.
- Tomic, O., Eklöv, T., Kvål, K., Haugen, J.-E., 2004. Recalibration of a gas-sensor array system related to sensor replacement. *Anal. Chim. Acta* 512 (2), 199–206.
- Vezzoli, M., Ponzioni, A., Pardo, M., Falasconi, M., Faglia, G., Sberveglieri, G., 2008. Exploratory data analysis for industrial safety application. *Sensors and Actuators B: Chemical* 131 (1), 100–109, special Issue: Selected Papers from the 12th International Symposium on Olfaction and Electronic Noses - ISOEN 2007. In: International Symposium on Olfaction and Electronic Noses.
- Vlachos, D., Fragoulis, D., Avaritsiotis, J., 1997. An adaptive neural network topology for degradation compensation of thin film tin oxide gas sensors. *Sens. Actuators B: Chem.* 45 (3), 223–228.
- Zuppa, M., Distante, C., Persaud, K.C., Siciliano, P., 2007. Recovery of drifting sensor responses by means of dwt analysis. *Sens. Actuators B: Chem.* 120 (2), 411–416.
- Zuppa, M., Distante, C., Siciliano, P., Persaud, K.C., 2004. Drift counteraction with multiple self-organising maps for an electronic nose. *Sens. Actuators B: Chem.* 98 (2–3), 305–317.