Signal Processing Techniques Based on Adaptive Radial Basis Function Networks for Chemical Sensor Arrays

Hyung-Gi Byun

Abstract

The use of a chemical sensor array can help discriminate between chemicals when comparing one sample with another. The ability to classify pattern characteristics from relatively small pieces of information has led to growing interest in methods of sensor recognition. A variety of pattern recognition algorithms, including the adaptive radial basis function network (RBFN), may be applicable to gas and/or odor classification. In this paper, we provide a broad review of approaches for various types of gas and/or odor identification techniques based on RBFN and drift compensation techniques caused by sensor poisoning and aging.

Keywords: Chemical sensor array, Pattern recognition, Gas and/or odor classification, Radial basis function networks, Drift compensation

1. INTRODUCTION

The human nose contains a large array of chemical sensors. Information patterns are processed in the olfactory brain of an animal in order to discriminate between gases and/or odors based on previous learning experiences. Given this instrumental means of odor measurement, the human user interface needs to be considered very carefully, as the results need to be presented in a form that can be easily interpreted by user. The signals produced by an array of chemical sensors consist of measurements of responses to odors producing different patterns that are shown in multidimensional space.

A variety of pattern recognition techniques may be applied to classify different odors, for quantitative prediction, and for the recognition of unknown gases and odors. These include neural networks that take the input patterns generated by the array of sensors and may be trained to associate these patterns with particular classes of chemicals that may be of interest to the user. The use of neural networks within artificial sensory analysis has been growing in momentum in recent years. The ability to recognize pattern characteristics from relatively small pieces of information has led to growing interest in possible applications and the development of sensory recognition.

We have investigated the characteristics of radial basis function networks (RBFNs) applied to gas and/or odor classification problems. Radial basis function networks train rapidly—usually orders of magnitude faster than other neural network techniques such as backpropagation—while exhibiting none of backpropagation’s training pathologies such as paralysis or local minima problems. An RBFN is a two-layer network where the output units form a linear combination of the basis functions computed by hidden units [1-5].

To use an RBFN as a classifier, parameters such as centers, widths, and weights must be optimized from the training (learning) stage period to the classification (testing) stage, in which classification is carried out using the RBFN with optimized and fixed parameters. Along with how well sensor mechanism is used, how accurately (close to optimum) parameters are acquired is one of the most important factors in classification performance. The performance of an RBF network is highly dependent on the choice of centers and widths in a basis function. For a minimum number of nodes, the selected centers should accurately represent the training data for acceptable classification. The centers and widths of the RBFN are calculated by a fuzzy c-means algorithm and the distribution of the input patterns. The weights of the network are calculated by the singular value decomposition (SVD) method in a single-shot process. This RBFN-SVD is
considered superior to other learning algorithms, particularly in terms of processing speed and the solvability of nonlinear pattern responses in gas and/or odor analysis.

However, since the centers and widths are fixed after they are chosen, this method often results in an unsatisfying performance when input patterns are not particularly clustered. Thus, fine-tuning of the centers and widths is needed, and the stochastic gradient (SG) method is successfully applied for this purpose. The adaptive RBFN-SG algorithm has shown good classification performance for complex chemical patterns. Fine-tuning of all network parameters, centers, widths, and weights can be carried out by SG (RBFN-SG) with initial weight values of zero. The learning speed of an algorithm employing the stochastic-gradient descent method is dependent on its convergence coefficients. To avoid a long training time, we applied Taylor’s expansion in a similar way to that of the normalized least means square (LMS) method to obtain the optimum convergence coefficients as quickly as possible. For convenience, this normalized SG (NSG) method will be referred to as RBFN-NSG.

Owing to sensor drift, sometimes a trained (learned) RBFN is no longer effective for gas and/or odor classification. We have suggested enhanced signal processing techniques for readjusting the RBFN weights in the test stage using a probability distribution function (PDF) construction and the cross-correntropy concept based on Information Theoretical Learning (ITL). The RBFN weights are readjusted in the test stage by using proposed techniques for sensor drift compensation.

In this contribution, we review a theoretical analysis of training techniques for RBFN parameters to improve recognition performance and to test phase-readjustment techniques to compensate for sensor drift. The performances of the proposed techniques were evaluated using an odor-sensing system.

## 2. TRAINING RBFN PARAMETERS

### 2.1 RBFN Based on Singular Value Decomposition (SVD) Method

The architecture of an RBFN is simple and consists of input, hidden, and output layers. The basis function in the hidden layer produces a localized response to the input, and typically uses hidden layer neurons with Gaussian response functions. The output of the \( k \)-th input feature \( x_k \) is determined by

\[
z_k = \sum_{j=1}^{H} w_{kj} \exp\left(-\frac{(x_k - c_j)^2}{s_j^2}\right)
\]

(1)

where \( c_j \) is the center associated with hidden unit \( j \), \( s_j \) is the width coefficient for hidden unit \( j \), and \( w_{kj} \) is the weight from hidden unit \( j \) to the output.

Processing using an RBFN is mainly carried out in two sessions: training and testing. Training an RBF network is conducted by finding the centers, widths, and weights connecting hidden nodes to output nodes.

Most of the training algorithms for an RBF network have been divided into the two stages of processing. First, a clustering method such as the fuzzy c-means algorithm is applied to the input patterns in order to determine the centers for hidden layer nodes. After the centers are fixed, the widths are determined in a way that reflects the distribution of the centers and input patterns. Once the centers and widths are fixed, the weights between the hidden and output layers are trained by a single-shot process using the Singular Value Decomposition (SVD) method. This method is called the RBFN-SVD method in the review contribution.

### 2.2 RBFN Based on Stochastic Gradient (SG) Method

For a given set of input patterns measured using a chemical sensor array, a fuzzy c-means algorithm with random initial conditions is carried out to determine the locations of the cluster’s centers, which are then fed into the hidden-layer units of the RBF network. The Euclidean distance between the input patterns and the cluster’s centers is evaluated. Then a Gaussian basis function with the initial widths is applied. Once the centers and widths are fixed, the weights between the hidden and output layers are determined by a single-shot process using SVD. This RBFN-SVD method provides some useful solutions for training the RBFN for pattern classification problems. However, this method often results in unacceptable classification performance when input patterns are not particularly clustered.

Although the fuzzy c-means algorithm provides relatively good performance compared with other center selection methods, because of random initial conditions, it is still difficult to select optimum centers. We call this method of determining RBFN parameters a rough-tuning stage. As a second stage to fine-tune the widths, centers, and weights previously determined by pattern distributions, the fuzzy c-means algorithm and SVD can be employed for parameters closer to their optimum values. This idea of employing a fine-tuning stage was proposed in [6]. The weights are also updated again based on the finely tuned centers and widths [7]. In the fine-tuning stage, the centers and widths are repeatedly updated by the stochastic gradient (SG) method introduced in the [8] as described below, and then the weights are
updated again by the same SVD.

We define system error $e$ as the difference between the target values and RBFN output values, and assume that the mean squared error (MSE) $E[e^2]$ is a convex function of the parameters. The current parameters after the initial tuning stage are adjusted to be closer to the minimum point of the MSE by utilizing the gradient of the MSE function. Using the instant error power $e^2$ instead of the statistically averaged, the RBFN parameters of centers and widths can be finely adapted by some iterations. Employing iteration number $n$ and defining $s_j^{(n)}$, $c_j^{(n)}$, and $w_j^{(n)}$ as the width, center vector, and weight for node $j$, respectively, the update equations based on SG can be described as

$$s_j^{(n+1)} = s_j^{(n)} - \mu_s \frac{\partial e^2}{\partial s_j^{(n)}}$$

$$c_j^{(n+1)} = c_j^{(n)} - \mu_c \frac{\partial e^2}{\partial c_j^{(n)}}$$

$$w_j^{(n+1)} = w_j^{(n)} - \mu_w \frac{\partial e^2}{\partial w_j^{(n)}}$$

where $\mu_s$ and $\mu_c$ are convergence coefficients for the widths and centers, respectively, and they control the speed of adaptation. In addition, the weights between the hidden and output layers are also determined again by SVD calculation after finishing the fine-tuning of the widths and centers by (2) and (3).

With the use of the SG method for fine-tuning the centers and widths and the use of SVD once more for weights, the fine-tuning scheme for an RBF network yielded a better training performance.

### 2.3 RBFN Based on Normalized Stochastic Gradient (NSG) Method

In SG methods, convergence coefficients play an important role in training speed and error performance. One method of lowering MSE involves using a very small value of $\mu_s$. However, a small $\mu_s$ makes the convergence speed of the SG algorithm slow. We want to use methods that acquire a fast learning speed and low MSE performance. The use of a time-varying convergence parameter can be a solution to the problem of the RBF-SG algorithm. In [9] a modified version of a raised-cosine function is used as a time-decreasing function and is applied to a convergence coefficient for center adaptation.

Inspired by the improved training performance resulting from the fine-tuning approach based on the SG method, the researchers proposed to employ the SG method for the fine-tuning of all network parameters (centers, widths, and weights), in which initial weights were set to be zero [10]. Furthermore, the researchers in [10] found that the adaptation coefficients $\mu_s$, $\mu_c$ and $\mu_w$ for widths, centers, and weights, respectively, can be optimized by normalizing the convergence parameter as a further step toward a time-varying convergence coefficient.

For convenience, the activation level of hidden unit $j$ at iteration time $n$ is defined as

$$v_j^{(n)} = \exp\left(\frac{-L_j^{(n)} - L_{j+1}^{(n)}}{s_j^{(n)2}}\right)$$

Then the RBFN output $Z_n$ can be rewritten as

$$Z_n = \sum_{j=1}^{M} w_j^{(0)} v_j^{(n)}$$

or $Z_n = W^T V_n$, where $W_n = [w_1^{(0)}, w_2^{(0)}, ..., w_M^{(0)}]^T$ and $V_n = [v_1^{(0)}, v_2^{(0)}, ..., v_M^{(0)}]^T$.

By defining $u_j^{(n)} = v_j^{(n)} - v_j^{(n-1)}$ as $Q_j^{(n)}$ and using Taylor’s expansion of system error $e^{(n)}$ at iteration $n$, the following form of error power is obtained:

$$e^{(n+1)2} = e^{(n)2}(1 + \sum_{j=1}^{M} 2\mu_s^{(0)} ||Q_j^{(n)}||^2)^2$$

Differentiating (5) with respect to the convergence coefficient, and letting $\partial e^{(n+1)2} / \partial \mu_s^{(0)} = 0$, yields the normalized convergence coefficient for centers $* \mu_s^{(0)}$ as

$$* \mu_s^{(0)} = \frac{1}{\alpha + \sum_{j=1}^{M} 2 ||Q_j^{(n)}||^2}$$

where a positive constant $\alpha$ is inserted for stability control.

Viewing $Q_j^{(n)}$ as an input vector to the center-updating process in (3), the variance of input $Q_j^{(n)}$ to the center-updating process varies wildly in practical circumstances. When the instant power of the input vector $Q_j^{(n)}$ is large, the time-varying coefficient $* \mu_s^{(0)}$ becomes small, and it lowers the steady-state MSE. On the other hand, when the instant power is weak, the process is still stable with the aid of the constant $\alpha$. Thus, the tracking dynamics of the normalized form (6) appear to be less sensitive to a variety of signal power distribution aspects than in the RBFN-SG.

The resulting center adaptation algorithm becomes

$$c_j^{(n+1)} = c_j^{(n)} + \frac{e^{(n)}Q_j^{(n)}}{\alpha + \sum_{j=1}^{M} 2 ||Q_j^{(n)}||^2}$$

Similarly, the normalization approach to the adaptation of...
weights leads to
\[
W_j^{(n+1)} = W_j^{(n)} + \frac{\varepsilon_j^{(n)}}{\alpha + \sum_j \varepsilon_j^{(n)2}} \cdot v_j^{(n)} \cdot \mu_c^{(n)}
\]  

(8)

For convenience, this normalized SG (NSG) method will be referred to in this paper as RBFN-NSG.

2.4 Performance Comparison in Training RBFN Parameters for Odor Classification

To evaluate the training performance of RBFN parameters for the classification of odors, data was collected from an odor-sensing system that had an array of conducting polymer sensors mounted on a ceramic substrate with associated electronics. The system was developed by Krishna Persaud of the University of Manchester [11]. For solvent vapors with associated concentrations of 1% acetonitrile (ac1), 10% acetonitrile (ac10), 1% acetone (ae), 1% butanone (bu), 10% methanol (me), 1% propanol (pr1), 10% propanol (pr10), and water (wa), measurements of chemicals were repeated to collect patterns over periods of four weeks. The collected response patterns are depicted in Fig. 1.

We notice that there is some variability in the responses of the sensor array during the four-week measurement. This may result from drifts, among other possible causes.

For network training, eight centers (H = 8) for each class were chosen from 528 patterns in data sets for weeks 1 and 2. The predefined target values (classes) are \(A_1 = 1\) for ac1, \(A_2 = 2\) for ac10, ..., \(A_8 = 8\) for wa, respectively.

During the training phase, the RBFN is trained by the above two methods (RBFN-SVD and RBFN-NSG) with 528 patterns (66 patterns for each gas) from the data sets of weeks 1 and 2 obtained from the solvent vapors. Then, during the testing phase, the trained RBFN was applied to the unseen data of 412 patterns from weeks 3 and 4 to evaluate the odor prediction capability under drift effects. This is shown in Fig. 3.

In Fig. 2, the trained result of RBFN-SVD with input patterns from the data sets of weeks 1 and 2 is shown. The result for RBFN-NSG is depicted in Fig. 3, where the values of constant \(\alpha\) for \(*\mu_c^{(n)}\) and \(*\mu_w^{(n)}\) are 1400 and 400, respectively. For width adaptation in RBFN-NSG, the static convergence coefficient \(\mu_s = 0.001\) is used. These convergence coefficients and constants were determined experimentally to have minimum a steady-state MSE. The initial weights for RBFN-NSG are all set to zero.

From Fig. 2, it is observed that in the training phase RBFN-
NSG yields a superior classification performance compared with RBFN-SVD. However, in Fig. 3, it is revealed that even a well-trained RBFN can produce poor classification results when it is placed in the test phase for drifted patterns from significantly aged or contaminated sensors.

As shown in Fig. 3, difficulties remain owing to the nature of the sensors. The sensors may drift or suffer from aging and poisoning. In biological systems, the sensors have a limited lifetime, but a biological system maintains an invariant pattern-recognition capability even though the characteristics of the sensors involved may have changed. This ability may be possible partly because the biological sensors are continually replaced.

By contrast, electronic odor-sensing systems sometimes show pattern variations over long time periods even if identical odors are presented. Such drifts in the signals of electronic odor-sensing systems are usually classified into two categories: short-term drift caused by memory effects, and long-term drift caused by sensor poisoning and aging. After a certain time, the initial learned capability of pattern recognition can be destroyed, so that predicting odors after some period of time can be difficult.

3. TEST PHASE READJUSTMENT FOR SENSOR DRIFT COMPENSATION

A number of possible approaches have been suggested to compensate for the drift effect caused by aging and poisoning of sensor materials combined with various circumstances after some period of time [12]. Adaptive neural networks are an active area of research and promise even more sophisticated neural networks that can automatically compensate for the drift effect [13].

To use the RBFN as a classifier, parameters such as centers, widths, and weights must be optimized during the training phase and then during the test phase. Classification is carried out using the optimized RBFN on unseen unlabeled input data from the sensor electronics.

Even with optimized and fixed weights in the RBFN, the output of the system during the test phase is often shifted and dispersed as shown in Fig. 3 so that the weights optimized in the training phase are no longer effective for odor classification. This may require some recalibration for the entire odor-sensing system, including sensors and associated electronics, or replacement of the aged or contaminated sensors with new ones. However, this hardware-oriented recalibration is time-consuming, costly, and sometimes impossible in field-test situations.

By using software to avoid this impractical hardware-oriented recalibration, it can be considered that whenever the system is placed in test phases, the RBFN parameters are retrained with the aid of the same odor samples used in the previous training phase. However, this requires that the odor samples are preserved in a pristine state and that enough of them are somehow stored to be provided whenever classification is needed. In addition, automatic mechanical processes for injecting and cleaning the gases for recalibration must be installed if the system is used in unmanned operating situations.

These obstacles to somewhat hardware-oriented recalibration have led researchers to search for all software-dependent methods for sensor-drift compensation. One such method is to readjust the RBFN weights even during the test phase with the aid of the output distribution information obtained in the training phase without the use of the stored odor samples [14]. The method is based on the ITL method introduced by Principe [15].

Unlike the use of an MSE criterion, as in SG methods for learning by adaptive systems, ITL methods are built on a combination of a nonparametric probability density function (PDF) estimator and entropy or information potential (IP). Estimating the data PDF nonparametrically is carried out by a Parzen window method using a Gaussian kernel [15,16]. The study in [15] demonstrated that the error samples of ITL-trained systems exhibit a more concentrated density function than MSE. Based on ITL methods, researchers in [17] introduced the Kullback-Leibler (KL) divergence for training adaptive systems. The KL divergence estimates mutual information that is capable of quantifying the entropy between pairs of random variables, but it is not quadratic in PDFs. Thus, it is not easy to integrate it with the IP. As another divergence measure, the Euclidian distance (ED) between two PDFs contains only quadratic terms [15].

Based on minimization of ED, the authors in [18] proposed using the ED criterion to train the adaptive system in order to match two PDFs from different output patterns, and successively applied it to classification problems with a real biomedical data set [13]. As a similar approach, [14] applied an ED minimization method to compensate for sensor drift in which the RBFN weights are readjusted even during the test phase, but only with the use of output data samples that were obtained during the training phase. This approach can be considered the opening stage of an all-software-based and unsupervised sensor-drift compensation method based on RBFN.
3.1 ED Minimization for Weight Readjustment Using Training-Phase Output Samples

The Euclidian distance between the PDF \( f_y \) of \( N \) trained output samples in the training phase defined as \( Y^N = \{y_1, y_2, \ldots, y_N\} \), and the PDF \( f_y \) of \( M \) current system output data during the test phase as \( Z^M = \{z_1, z_2, \ldots, z_M\} \), can be defined as

\[
ED = \int \frac{1}{2\sigma^2} \exp\left(-\frac{(y-y_j)^2}{2\sigma^2}\right) dy - 2 \int f_y(\alpha) f_y(\alpha) d\alpha
\]

We define a zero-mean Gaussian kernel with kernel size \( \sigma \) as

\[
G_\sigma(\alpha) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{\alpha^2}{2\sigma^2}\right)
\]

Parzen kernel density estimation can be expressed as

\[
f_y(\alpha) = \frac{1}{N} \sum_{i=1}^{N} G_\sigma(\alpha - y_i) \quad \text{By inserting it into (9), the Euclidian distance ED becomes}
\]

\[
ED = \frac{1}{M} \sum_{j=1}^{M} \sum_{i=1}^{M} G_\sigma(z_j - y_i) + \frac{1}{M} \sum_{j=1}^{M} \sum_{i=1}^{M} G_\sigma(z_j - z_i) - 2 \frac{1}{M} \sum_{j=1}^{M} \sum_{i=1}^{M} G_\sigma(z_j - z_i)
\]

Each term of (11) corresponds to an IP defined in [15]. The first term \( \frac{1}{M} \sum_{j=1}^{M} \sum_{i=1}^{M} G_\sigma(z_j - y_i) \), indicating the summed interactions among output sample pairs produced and stored in the training phase, is not controllable with respect to weights in the test phase.

Therefore, the resulting cost function \( P(W) \) becomes

\[
P(W) = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} G_\sigma(z_j - z_i) - 2 \frac{1}{MN} \sum_{j=1}^{M} \sum_{i=1}^{M} G_\sigma(z_j - z_i)
\]

To minimize \( ED \) (MED), we minimize \( P(W) \) with respect to the RBFN weights in the test phase.

As a minimization of the MSE, the minimization of \( P(W) \) can also be implemented by using the following gradient. The gradient at the iteration time \( k \) is evaluated from

\[
\frac{\partial P(W)}{\partial W} = \frac{1}{2\sigma^2} \sum_{i=1}^{L} \sum_{j=1}^{L} (y_i - z_j) G_{\sigma^2}(y_i - z_j) (V_i - V_j)
\]

Through a gradient descent, the updated equation for readjusting weights in the test phase can be expressed with the convergence coefficient \( \mu_\text{med} \) as

\[
W_{\text{med}} = \mu_\text{med}\left[ \frac{1}{2\sigma^2} \sum_{i=1}^{L} \sum_{j=1}^{L} (y_i - z_j) G_{\sigma^2}(y_i - z_j) (V_i - V_j) \right]
\]

For convenience, this weight readjustment method by MED in the test phase will be called RBFN-MED in this paper.

3.2 ED Minimization for Weight Readjustment Using a Set of Dirac Delta Functions

In the RBFN-MED method, the minimization of ED forces the test-phase output samples \( Z^M = \{z_1, z_2, \ldots, z_M\} \) to be placed immediately following the distribution pattern of the trained output samples \( Y^N = \{y_1, y_2, \ldots, y_N\} \).

In this RBFN-MED method, a problem can be observed in that the trained output samples are well gathered at each level but are still dispersed and biased to some extent from each target level, as shown in Fig. 3. That is, the PDF \( f_y \) is very similar to, but not the same as, the PDF of the actual target levels for each class \( A_i \) defined as \( A_1 = 1, A_2 = 2, \ldots, A_8 = 8 \). This property limits the capability of sensor-drift-effect compensation and inspires researchers to find ways that the test phase output samples can be gathered to match the PDF of the actual target levels.

One proposed solution is to construct a PDF of target values using a set of Dirac delta functions instead of using \( f_y \) constructed with a trained output data set [19].

Assuming that \( L \) target values \( (A_1, A_2, \ldots, A_L) \) are equally likely with a probability of \( \frac{1}{L} \), the PDF \( f_y(\alpha) \) of the desired target values can be constructed as a simple equation:

\[
f_y(\alpha) = \frac{1}{L} [\delta(\alpha - A_1) + \delta(\alpha - A_2) + \ldots + \delta(\alpha - A_L)]
\]

Then \( ED = \int f_y(\alpha) d\alpha + \int f_y(\alpha) d\alpha - 2 \int f_y(\alpha) f_y(\alpha) d\alpha \) has terms expressed as

\[
\int f_y(\alpha) d\alpha = \frac{1}{L} \sum_{i=1}^{L} \int \delta(\alpha - A_i) \delta(\alpha - A_i) d\alpha = \frac{1}{L}
\]

Accordingly,

\[
\int f_y(\alpha) d\alpha = \frac{1}{L} \sum_{i=1}^{L} \delta(\alpha - A_i)
\]

\[
\int f_y(\alpha) f_y(\alpha) d\alpha = \frac{1}{L} \sum_{i=1}^{L} f_y(A_i) = \frac{1}{L} \sum_{i=1}^{L} \sum_{j=1}^{L} \delta(\alpha - z_j)
\]
Thus the Euclidian distance to be minimized becomes

\[ ED = \frac{1}{L} + \frac{1}{L} \sum_{j=1}^{L} \sum_{i=1}^{L} G_{\sigma_j^2}(z_j - z_i) \]

\[ -2 \frac{1}{L} \sum_{j=1}^{L} \sum_{i=1}^{L} G_{\sigma_j^2}(z_j - z_i) \]

As in (13), the gradient using iteration time index \( k \) is derived as

\[ \frac{\partial ED}{\partial W} = -\frac{2}{LN_{\sigma^2}} \sum_{j=1}^{L} \sum_{i=1}^{L} (z_j - z_i) G_{\sigma_j^2}(z_j - z_i) (V_j - V_i) \]

The weight update equation using a set of Dirac delta functions becomes

\[ W_{ij} = W_{ij} - \mu_{\text{new}} \left[ \frac{1}{2N_{\sigma^2}} \sum_{j=1}^{L} \sum_{i=1}^{L} (z_j - z_i) G_{\sigma_j^2}(z_j - z_i) (V_j - V_i) \right] \]

This weight readjusting method in the test phase without the use of trained output samples is referred to as RBFN-MED2 [19].

### 3.2.1 Performance Comparison for Weight Readjustment in Test Phase

To examine sensor-drift compensation performance, the same experimental system was used as in Fig. 3. The drift output in Fig. 3 is now compared with the compensated output by the weight readjustment algorithm (14) operating in the test phase, where \( N \) trained output samples are used as target samples that are unknown in the test phase. For the RBFN-MED2 method, the number of block data \( N = M = 66 \), kernel size \( \sigma = 0.5 \), and convergence coefficient \( \mu_{\text{new}} = 0.00001 \) were used.

The test phase results are depicted in Fig. 4. It is noticeable that the RBFN-MED2 has significantly compensated for the sensor drift effect, producing output samples more dense at each target level for eight gases.

For a closer performance comparison between compensation methods, distributions of output samples for drifted and compensated results are compared in Fig. 5 and 6, respectively. It is observed that the distributions for most of the gases have sharper and narrower spikes in Fig. 6 than in Fig. 5. In particular, the distribution of drifted gas 1, which is considered the most difficult to compensate for the drift effect, is widely dispersed even beyond target value 3. The distribution of compensated gas 1 does not exceed 2.5, and its spike is centered exactly at 1. Although the compensated distribution for gas 8 shows a little more dispersion, the center (mean) of...
the distribution is shifted closer to or above level 8 than in the drifted distribution.

### 3.3 Readjustment Method for Centers in RBFN in the Test Phase

As discussed above, the method RBFN-MED2 could compensate for the sensor drift effects to a great extent, however, this method also showed some deteriorated distributions, particularly in gas 8 as shown in Fig. 6. For better compensation performance, there has been a demand for techniques that control other RBFN parameters for the performance enhancement of sensor drift compensation.

In a recent work [20], a new readjustment method for centers of an RBFN based on ED in (18) is proposed and experimented for sensor drift compensation.

To control the centers of an RBFN, the cost function (19) is minimized through a gradient descent with respect to centers.

The gradient of (19) with respect to the center vector of node \( n \), \( \mathbf{c}^{(n)} \), is calculated as

\[
\frac{\partial \text{ED}}{\partial \mathbf{c}^{(n)}} = -2N^2 \sigma^2 \sum_{i=k-N,1}^{k} \sum_{j=k-N,1}^{k} (z_i - z_j) G_{\sigma^2}(z_i - z_j) \left( \frac{\partial c_n}{\partial \mathbf{c}^{(n)}} - \frac{\partial c_n}{\partial \mathbf{c}^{(n)}} \right)
\]

\[
\frac{2}{LN \sigma^2} \sum_{i=k-N,1}^{k} \sum_{j=k-N,1}^{k} (z_i - z_j) G_{\sigma^2}(z_i - z_j) \left( \frac{\partial c_n}{\partial \mathbf{c}^{(n)}} - \frac{\partial c_n}{\partial \mathbf{c}^{(n)}} \right)
\]

(22)

On the other hand, the derivative \( \frac{\partial Z_i}{\partial \mathbf{c}^{(n)}} \) can be obtained as

\[
\frac{\partial Z_i}{\partial \mathbf{c}^{(n)}} = \frac{\partial}{\partial \mathbf{c}^{(n)}} \sum_{n=1}^{N} w^{(n)} \exp\left( -\frac{||x_i - \mathbf{c}^{(n)}||^2}{(s^{(n)})^2} \right)
\]

\[
= 2 w^{(n)} \exp\left( -\frac{||x_i - \mathbf{c}^{(n)}||^2}{(s^{(n)})^2} \right) \frac{x_i - \mathbf{c}^{(n)}}{(s^{(n)})^2}
\]

(23)

Inserting (23) into (22) leads to

\[
\frac{\partial \text{ED}}{\partial \mathbf{c}^{(n)}} = \frac{1}{-2N^2 \sigma^2} \sum_{i=k-N,1}^{k} \sum_{j=k-N,1}^{k} (z_i - z_j)
\]

\[
G_{\sigma^2}(z_i - z_j) \cdot 2 w^{(n)} \exp\left( -\frac{||x_i - \mathbf{c}^{(n)}||^2}{(s^{(n)})^2} \right)
\]

\[
\frac{x_i - \mathbf{c}^{(n)}}{(s^{(n)})^2} - \frac{x_j - \mathbf{c}^{(n)}}{(s^{(n)})^2}
\]

(24)

\[
\frac{2}{LN \sigma^2} \sum_{i=k-N,1}^{k} \sum_{j=k-N,1}^{k} (z_i - z_j) G_{\sigma^2}(z_i - z_j) \cdot 2 w^{(n)}
\]

\[
\cdot \exp\left( -\frac{||x_i - \mathbf{c}^{(n)}||^2}{(s^{(n)})^2} \right) \frac{x_i - \mathbf{c}^{(n)}}{(s^{(n)})^2}
\]

Finally, the center vector at iteration time \( k \) for node \( n \), \( \mathbf{c}_{k+1}^{(n)} \), can be readjusted according to the following equation:

This center-readjusting equation (25), together with the weight-readjusting equation (21), take one step further in improving the capability of an RBFN to solve the sensor drift problems. This is done using a software approach based on an ED criterion between probability density functions.

### 3.3.1 Performance Comparison for Center and Weight Readjustment in Test Phase

Using the same input pattern data for eight solvent vapors as in Fig. 3 (gas 1: 1% acetonitrile, gas 2: 10% acetonitrile, gas 3: 1% acetone, gas 4: 1% butanone, gas 5: 10% methanol, gas 6: 1% propanol, gas 7: 10% propanol, and gas 8: water), three results of RBFN parameter control during the test phase are compared:

- Case 1: Without compensation in test phase
- Case 2: After readjustment of weight only by equation (21)
- Case 3: After readjustment of weight and center by equations (21) and (25)

Unlike the pattern numbers of each gas that are followed consecutively by the number of the next gas on the horizontal axis...
as shown in Fig. 2-4, Fig. 7-9 show output samples for each gas placed separately on the vertical axis for a clearer comparison of the dispersion. The test-phase output without compensation is depicted in Fig. 7, and the compensated output through weight readjustment is shown in Fig. 8. The output after the readjustment of weights and centers is shown in Fig. 9.

It is noticeable that output samples of most gases in case 1 are tightly concentrated in case 2 and even more in case 3. For a stricter performance comparison, variance (a measure of dispersion) is depicted in Fig. 10.

Compared with the case of weight readjustment only, the additional center readjustment yields a decreased dispersion for most gases at the expense of a small increased dispersion for gas 1. In the case of gas 8, where the dispersion of the weight readjustment increased, we can observe a conspicuous performance enhancement. While the variance of output produced without compensation is 0.06247, the variance after additional center readjustment is 0.03364, which is a reduction of nearly half.

The fact that reduced dispersion for most gases (and especially for gas 8) deteriorated even after weight readjustment proves that the unsupervised method of weight and center readjustment by an ED minimization approach is highly effective in compensating for sensor drift.

### 3.3 Cross-Correntropy Method for Readjustment of RBFN Weights in Test Phase

From section 3-1 to 3-3, unsupervised methods of RBFN parameter readjustment for sensor drift compensation were developed based on an ED between two PDFs.

In this section, a new concept using a combination of kernel transformation and ITL—cross-correntropy—is introduced [21]. We will show how the cross-correntropy concept can be applied to signal processing for sensor-drift compensation during the test phase, as it appeared in [22].

When two scalar random processes $X(t)$ and $X(u)$ separated by a certain time delay $t-u$ are given, kernel algorithms can transform the data $X(t)$ and $X(u)$ from the input space to a high-dimensional feature space of vectors $\Phi(X(t))$ and $\Phi(X(u))$, where the nonlinear mapping $\Phi$ induced by the Gaussian kernel maps the data into the feature space [23].

The inner product of vectors in a kernel feature space $\langle \Phi(X(t)), \Phi(X(u)) \rangle$ is a measure of a similarity in feature space. The inner products $\langle \cdot, \cdot \rangle$ can be computed using a positive definite kernel function that satisfies Mercer’s conditions [21]:

$$G_d(X(t) - X(u)) = \langle \Phi(X(t)), \Phi(X(u)) \rangle$$  \hspace{1cm} (26)
where $F$ is an infinite dimensional reproducing the kernel’s Hilbert space.

This makes it possible to obtain nonlinear versions of any linear algorithm expressed in terms of inner products without knowing the exact mapping function $\Phi$, and to utilize this measure of similarity in signal processing applications.

By taking the expected value $E_t$ of the inner product of (26) just as in the autocorrelation of random processes, the auto-correntropy function $CE(t,u)$ [21] is defined as

$$CE(t,u) = E_t[\langle \Phi(X(t)), \Phi(X(u)) \rangle]$$

(27)

Thus, the resulting auto-correntropy function $CE(t,u)$ becomes

$$CE(t,u) = E_t[G_\sigma(X(t) - X(u))]$$

(28)

Unlike auto-correntropy using a random variable with a time lag, cross-correntropy [24] is a generalized version of a similarity measure between two different random variables $Z$ and $Y$ defined by

$$CE(Z,Y) = E_t[G_\sigma(Z - Y)]$$

(29)

With the Gaussian kernel, (29) can be rewritten as

$$CE(Z,Y) = E_t[G_\sigma(Z - Y)]$$

(30)

In practice, a sample mean operation with $N$ samples can be used instead of statistical expectation the following cross-correntropy function can be obtained.

$$CE(Z,Y) = \frac{1}{N} \sum_{i=1}^{N} G_\sigma(z_i - y_i)$$

(31)

Since (31) is a practical similarity measure, we examine the idea that cross-correntropy can play the role of the Euclidian distance in an RBFN readjustment during the test phase. To be able to apply the cross-correntropy criterion to the readjustment of RBFN weights during the test phase, we need to replace samples $z_i$ and $y_i$ in (31) with the $N$ trained output samples from the training phase $Y_N = \{y_1, y_2, \ldots, y_N\}$, and $N$ current system output data during the test phase $Z_N = \{Z_{k,1}, Z_{k,2}, \ldots, Z_{k,N-1}\}$ where $k$ is the iteration time index. Then we can obtain the cross-correntropy criterion to be maximized for sensor-drift compensation as

$$CE(Z,Y) = \frac{1}{N} \sum_{i=1}^{N} G_\sigma(z_{k,i} - y_i)$$

(32)

With weight vector $W$ and RBFN node output vector $V_k$, the gradient of (32) with respect to weight can be calculated as

$$\frac{\partial CE(Z,Y)}{\partial W} = \frac{1}{N\sigma^2} \sum_{i=1}^{N} (y_i - z_{k,i}) \cdot G_\sigma(z_{k,i} - y_i) \cdot V_{k,i}$$

(33)

One of the differences between cross-correntropy and MED is that the cross-correntropy function $CE(Z,Y)$ must be maximized, and the MED cost function $P(W)$ in (12) must be minimized.

Through using a gradient ascent method to maximize the gradient, we update equation (33) with the convergence coefficient $\mu_{CE}$ to readjust the weights in the test phase, as proposed in [22]:

$$W_{k+1} = W_k + \mu_{CE} \frac{1}{N\sigma^2} \sum_{i=1}^{N} (y_i - z_{k,i}) \cdot G_\sigma(z_{k,i} - y_i) \cdot V_{k,i}$$

(34)

For convenience, this updated equation is referred to in this paper as CE. It is noticeable that the complexity of CE is $O(N)$, whereas that of MED is $O(N^2)$.

3.4.1 Performance of CE for Sensor-Drift Compensation

Two gases, ammonia and CO (labeled as 1 and 2, respectively), were used in the odor sensing/identification system [11]. After training an RBFN consisting of seven hidden nodes with 200 input patterns, the trained output is accurately produced, as shown in Fig. 11 (gray dots). RBFN_NSG is used with the same convergence coefficients $\mu_s$ and $\sigma$, as shown in Fig. 2. However, for the test-phase input patterns (0-303: Ammonia, 304-614: CO2) obtained from an odor system with severely aged/contaminated sensors, the test-phase output with the optimized RBFN is greatly dispersed. This is depicted as white round dots in Fig. 11.

The weight readjustment process in the test phase by CE with $\mu_{CE} = 0.0001$, kernel size $\sigma = 10$, and $N$ trained output samples $\{y_1, y_2, \ldots, y_N\}$ yields significantly enhanced compensation performance (indicated by black dots in Fig. 11). The drifted output produces eight errors in this experiment, whereas the compensated system equipped with CE records four errors. This
is a decrease in the error rate of approximately 50%. For ammonia (target level = 1), the compensated output samples are more closely gathered from 1.1 to 1.9, while the drifted output samples are widely dispersed from 0.6 to 2.2. For CO$_2$ (target level = 2), more concentrated samples are observed.

### 4. CONCLUSIONS

The RBFN has been successfully applied to classification and identification in odor-sensing systems. In order to determine the initial centers for hidden layer nodes with given input patterns, a clustering method such as the fuzzy c-means algorithm is used. After the initial centers are fixed, the initial widths are determined in a way that reflects the distribution of the centers and input patterns. Once the initial centers and widths are fixed, the weights between the hidden and output layers can be trained by a single-shot process using SVD.

This RBFN-SVD method provides some useful solutions for training an RBFN for pattern classification problems, but this method often results in unacceptable classification performance when input patterns are not particularly clustered. Because of random initial conditions, it is still difficult to select optimum centers through the fuzzy c-means algorithm. These problems of determining optimum RBFN parameters have led to the employment of a second stage for finely tuning the widths, centers, and weights. In the fine-tuning stage, the centers, widths, and even weights are repeatedly updated by the Stochastic Gradient method.

Once the training performance has been improved by the additional fine-tuning stage based on the SG method, the adaptation coefficients for centers and weights can be optimized by employing normalized and time-varying ones. This normalized SG method is considered one of the most powerful learning algorithms for RBFN training for odor classification.

First, we introduced an unsupervised readjustment method to minimize the Euclidian distance between the PDF of trained and stored output samples and the distance between current output samples that drifted significantly during the test phase. This minimization of ED yields an iterative readjustment equation for RBFN weights during the test phase. The idea of using the PDF information of trained and stored output samples that are highly concentrated on each target level has evolved to make the PDF closer to target distributions by employing a set of Dirac delta functions.

This second version of an unsupervised readjustment method that produces greatly enhanced compensation performance gave birth to a readjustment method for the centers of an RBFN. Now the weights and even centers of an RBFN can be readjusted during the test phase to compensate for sensor drift. This additional center readjustment yields much decreased dispersion for most gases in the experiment. The variance of the worst output samples of gas 8 was reduced by half compared with the case without compensation.

However, the RBFN readjustment based on ED minimization for severely drifted input patterns from highly ill-conditioned volatile gases such as ammonia and CO$_2$ did not operate well. This means a requirement for totally different readjustment methods and a new correntropy concept has been adopted to achieve acceptable sensor-drift compensation even in severely drifted patterns from sensor aging/contamination. The approach of maximizing correntropy for two outputs can effectively readjust the RBFN weights during the test phase and has yielded a well-concentrated distribution. Meanwhile, RBFN readjustment methods based on ED minimization were unable to successfully compensate for the sensor drift.

From our results, observations, and anticipation that the correntropy method can be developed even for RBFN center readjustment in the near future, we conclude that this method will make it possible to create more reliable odor-sensing systems equipped with sensor-drift-compensation capability.

### ACKNOWLEDGMENT

This study is supported by a 2015 Research Grant from Kangwon National University (No. 201510030).

### REFERENCES

[1] J. Moody and C. Darken, “Fast learning in networks of locally-tuned processing units”, *Neural Computation*, Vol. 1, pp. 281-294, 1989.

[2] M. Musavi, W. Ahmed, K. Chan, K. Faris, and D. Hummels, “On the training of radial basis function classifiers”, *Neural Networks*, Vol. 5, pp. 595-603, 1992.

[3] T. Kohonen, “Self-Organization and Associative Memory”, *Springer-Verlag*, 3rd ed., pp. 199-202, 1989.

[4] K. C. Persaud and H-G Byun, “Classification of complex odours using conducting polymer arrays and neural networks”, *Industrial applications of neural networks, Eds. Fogelman Soulé, World Scientific, Singapore, New Jersey*, pp. 85-90, 1998.

[5] D-H. Lee, J. S. Payne, H-G. Byun, and K. C. Persaud,
“Application of radial basis neural networks to odour sensing using a broad specificity array of conducting polymers”, Lecture Notes in Computer Science (Eds. C. von der Malsburg, W. von Seelen, J. C. Vorbruggen, B. Sendhoff), Vol. 1112, pp. 299-304, 1996.

[6] H. Byun, N. Kim, K. Persaud, J. Huh, and D. Lee, “Application of adaptive RBF networks to odor classification using conducting polymer sensor array”, Proceedings of ISOEN’2000, Washington D.C., U.S.A., Brighton, pp. 121-126, 2000.

[7] H. Byun, N. Kim, K. Persaud, J. Huh, and D. Lee, “Application of adaptive RBF network for odor classification under drift effect using conducting polymer sensor array”, Proceedings of Electrochemical Society, pp. 1309-1312, 2006.

[8] I. Cha, and S. Kassam, “Interference cancellation using radial basis function networks”, Signal Processing, Vol. 47, pp. 247-268, 1995.

[9] K. Kwon, N. Kim, H. Byun, and K. Persaud, “On training neural network algorithms for odor identification for future multimedia communication systems”, Proceedings of 2006 IEEE International Conference on Multimedia, pp. 1108-1113, 2006.

[10] N. Kim, H. Byun, and K. Persaud, “Normalization approach to the stochastic gradient radial basis function network algorithm for odor sensing systems”, Sensors and Actuators B, pp. 407-412, 2007.

[11] P Pelosi and K. Persaud, “Toward an artificial nose”, Sensors and Sensory Systems for Advanced Robots, NATO ASI Series, Vol. F43, pp. 49-70, 1998.

[12] J. W. Gardner et al., Electronic Noses Principles and Applications, Oxford University Press, 1999.

[13] M. Zuppa et al., “Drift counteraction with multiple self-organising maps for an electronic nose”, Sensors and Actuators B, pp. 305-317, 2004.

[14] N. Kim, H. Byun, K. Persaud, and J. Huh, “Sensor drift compensation algorithm based on PDF distance minimization”, Proceedings of ISOEN 2009, Vol. 1137, pp. 554-557, 2009.

[15] J. Principe, D. Xu, and J. Fisher, “Information Theoretic Learning” in: S. Haykin, Unsupervised Adaptive Filtration, New York, Wiley, pp. 265-319, 2000.

[16] D. Erdogmus and J. Principe, “An Entropy minimization algorithm for supervised training of nonlinear systems”, IEEE Trans. Signal Processing, Vol. 50, pp. 1780-1786, 2002.

[17] D. Erdogmus, Y. Rao, and J. Principe, “Supervised training of adaptive systems with partially labeled data”, Proceedings of the International Conference on ASSP, pp. v321-v324, 2005.

[18] K. Jeong, J. Xu, D. Erdogmus, and J. Principe, “A new classifier based on information theoretic learning with unlabeled data”, Neural Networks, Vol. 18, pp. 719-726, 2005.

[19] N. Kim, H. Byun, and K. Persaud, “Novel signal processing techniques based on PDF information for sensor-drift compensation”, Sensor Letters, Vol. 9, No. 1, pp. 439-443, 2011.

[20] N. Kim, H. Byun, K. Kwon, K. Persaud, and O.J. Lim, “Unsupervised adjustment of centers in RBF networks for sensor drift compensation”, Proceedings of IMCS 2012, pp. 1802-1804, 2012.

[21] I. Santamaria, P. Pokharel, and J. Principe, “Generalized correlation function: Definition, properties, and application to blind equalization”, IEEE Trans. Signal Processing, Vol. 54, pp. 2187-2197, 2006.

[22] N. Kim, H. Byun, and K. Persaud, “A new compensation method for sensor-drift effect based on the cross-correntropy concept”, Sensor Letters, Vol. 9, No. 2, pp. 710-713, 2011.

[23] V. Vapnik, The Nature of Statistical Learning Theory, New York, Springer Verlag, 1995.

[24] W. Liu, P. Pokharel, and J. Principe, “Correntropy: Properties and applications in non-Gaussian signal processing”, IEEE Trans. on Signal Processing, Vol. 55, No. 6, pp. 2187-2198, 2007.