sensors

MDPI

Article

Self-Taught Learning Based on Sparse Autoencoder for E-Nose in Wound Infection Detection

Peilin He, Pengfei Jia *, Siqi Qiao and Shukai Duan

College of Electronic and Information Engineering, Southwest University, Chongqing 400715, China; qaz321123@email.swu.edu.cn (P.H.); i47sir@126.com (S.Q.); duansk@swu.edu.cn (S.D.)

* Correspondence: jiapengfei200609@126.com; Tel.: +86-187-1664-9336

Received: 18 August 2017; Accepted: 30 September 2017; Published: 7 October 2017

Abstract: For an electronic nose (E-nose) in wound infection distinguishing, traditional learning methods have always needed large quantities of labeled wound infection samples, which are both limited and expensive; thus, we introduce self-taught learning combined with sparse autoencoder and radial basis function (RBF) into the field. Self-taught learning is a kind of transfer learning that can transfer knowledge from other fields to target fields, can solve such problems that labeled data (target fields) and unlabeled data (other fields) do not share the same class labels, even if they are from entirely different distribution. In our paper, we obtain numerous cheap unlabeled pollutant gas samples (benzene, formaldehyde, acetone and ethylalcohol); however, labeled wound infection samples are hard to gain. Thus, we pose self-taught learning to utilize these gas samples, obtaining a basis vector θ. Then, using the basis vector θ, we reconstruct the new representation of wound infection samples under sparsity constraint, which is the input of classifiers. We compare RBF with partial least squares discriminant analysis (PLSDA), and reach a conclusion that the performance of RBF is superior to others. We also change the dimension of our data set and the quantity of unlabeled data to search the input matrix that produces the highest accuracy.

Keywords: electronic nose; self-taught learning; sparse autoencoder; wound infection

1. Introduction

Electronic nose (E-nose), a device composed of a sensor array and an artificial intelligence algorithm, has been successfully used in many fields. It is able to deal with a multitude of problems efficiently, such as food analysis [1–4], disease diagnosis [5–8], environment control [9,10], etc.

Traditional methods for a doctor to diagnose the type of wound infection usually require observing features of the plaie and take a long time to analyse the patient's blood, urine and other aspects, delaying the best time for treatment. In particular, with the development of medical technology as well as higher requirements on disease detection speed and accuracy, the E-nose has great prospects in disease diagnosis. Our previous work has proved that the E-nose can be used to distinguish the classes of wound infections through their special odor [11–14].

In practice, however, if we want to get an E-nose that can distinguish wound infections efficiently and accurately, quantities of wound infection samples are needed to train the classifier, which would cost a lot of money. Such experimental infection samples are not that easy to obtain, let alone labeled wound infection samples. While our wound infection samples are limited, there are some other unlabeled pollutant gas samples which are numerous and obviously easier to obtain, for a lower cost. If we ignore the usage of these samples in other fields, it can lead to waste. To take advantage of them, we introduce transfer learning in our paper.

Transfer learning is the ability to transfer knowledge from one field to other fields, and these fields can share different labels, which distinguish it from traditional machine learning techniques.

That is to say, through transfer learning, we can use some samples from other fields to make up the lack of wound infection samples in our field. Thus, in this paper, we purchase inexpensive chemical solutions to obtain four types of pollutant gases, and can, as a result, get thousands of unlabeled samples through this approach, at low cost.

These unlabeled gas samples are introduced to cope with the lack of labeled wound infection samples, and an enhanced quantum-behaved particle swarm optimization (EQPSO) [15,16] is proposed to improve the performance of classifiers. In the machine learning field, there are some classical algorithms that can roughly be placed in two categories [17]: supervised learning, which concerns obtaining its classifier based on labeled data; and unsupervised learning, which is concerned with obtaining its classifier from unlabeled data. Unavoidably, however, both methods have significant shortcomings: supervised learning needs a large amount of labeled data, while unsupervised learning classifies samples by their different distribution, which makes the accuracy far lower than that of supervised learning. Therefore, semi-supervised learning, a combination of these two types learning framework, is widely adopted in practical application and improves the generalization ability of model. It broadens the range of data set, but because semi-supervised learning is typically based on the assumption that labeled data and unlabeled data can be tagged with the same labels, a new group of machine learning is put forward, which is called "self-taught learning".

Self-taught learning [18,19] is a new machine learning framework and also a type of transfer learning, corresponding to human learning, using unlabeled data in supervised classification tasks. What distinguishes self-taught learning from other learning methods is that self-taught learning can solve such problems as the fact that labeled data and unlabeled data do not share the same class labels, that they may be from entirely different distributions, or that the labeled data might be far less than unlabeled data.

In recent years, self-taught learning has undergone considerable development in many fields [20–22]. In self-taught learning, we construct basis vectors from the unlabeled data. In turn, these basis vectors are used to rebuild input representation, converting training data into representations related to unlabeled data. These new representations are programmed into the classification task and significantly improve the performance of the E-nose. In the algorithm, the most significant step is to contrast basis vectors from the unlabeled data and to rebuild new representations. To rebuild new representations, we take advantage of the neural network and apply the sparsity constraint, which makes the representation of each layer sparse (most of nodes become zero).

However, this is yet to be applied in the field of E-nose for the purpose of distinguishing the label information of wound infection data. In this paper, self-taught learning is proposed to perfect the accuracy of classification. In the rest of this paper, we first describe details of the material and odor sampling experiments in Section 2, then the self-taught learning framework is elaborated on in Section 3. In Section 4, we apply some classical classification algorithms and compare their results, such as partial least squares discriminant analysis (PLSDA) and radial basis function (RBF) [23,24].

2. Experiments and Data Preprocessing

2.1. E-Nose System and Experimental Setup

The labeled wound infection data set and unlabeled gas data set are needed in our project. An E-nose system is used to prepare the data set. In constructing the system, we employ an E-nose, a data acquisition system (DAS), a pump, a rotor flow meter, a three way value, a filter, glass bottles and a computer. The schematic diagram of the experimental system is shown in Figure 1 and the experimental setup is shown in Figure 2.

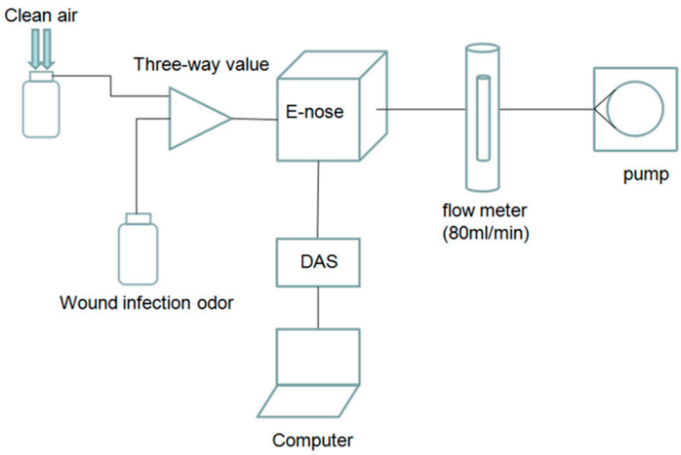

Figure 1. Schematic diagram of the experimental system.

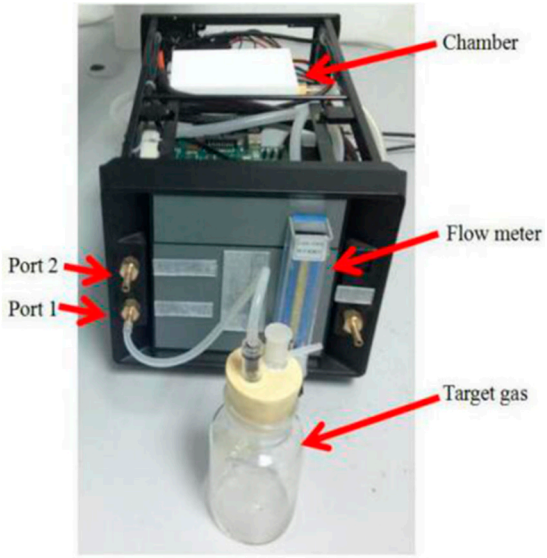

Figure 2. Experimental setup.

When air passes the filter, the air is purified. The flow meter controls the rate of the gas, making it keep at 80 mL/min, and at the same time the pump provides energy for the gas flow. The response signals processed by E-nose will be sampled and saved in a computer via the DAS, which is 32-channel and 14-bit high precision. The sampling frequency is set at 1 Hz. According to the metabolites of pathogens and the response characteristics of gas sensors, a sensor array composed of six sensors is employed to collect the response curve of wound infections and pollutant gases, and response characteristics of gas sensors are shown in Table 1.

Table 1. Sensitive characteristics of gas sensors.

Sensors	Sensitive Characteristics
TGS813	Methane, Propane, Ethanol, Isobutane, Hydrogen, Carbon monoxide
TGS816	Combustible gases, Methane, Propane, Butane, Carbon monoxide, Hydrogen, Ethanol, Isobutane
TGS822	Organic solvent vapors, Methane, Carbon monoxide, Isobutane, n-Hexane, Benzene, Ethanol, Acetone
TGS2600	Gaseous air contaminants, Methane, Carbon monoxide, Isobutane, Ethanol, Hydrogen
MQ135	Ammonia, Benzene series material, Acetone, Carbon monoxide, Ethanol, Smoke

Note: The response of these three sensors is non-specific. Table 1 just lists their main sensitive gases, and they are also sensitive to other gas.

Each sampling experiment is composed of the following three steps:

Step 1: the sensors are exposed to clean air for 3 min;
Step 2: the gas stream containing VOCs of the wound passes over the sensor array for 5 min;
Step 3: the sensors are exposed to clean air again for 15 min.

The sample interval between two experiments is 5 min.

2.2. Experiments and Sampling

Twenty Sparague-Dawley (SD) male rats are used in the experiment in this paper to prepare the labeled data set. These rats are divided into four groups averagely:

1. Wounded but uninfected (the control group);
2. Infected with *P. aeruginosa*;
3. Infected with *E. coli*;
4. Infected with *S. aureus*.

All rats are healthy and in similar condition, and each type has five rats, respectively. Every rat has a 1 cm long wound in the right hind leg, and pathogens are injected into the wound in accordance with their group. The metabolites of three pathogens are shown in Table 2. A total of 20 sampling data are collected for each kind of rat, that is to say, there exist 80 labeled wound infection samples in our project.

Table 2. Pathogens in wound infection and their metabolites.

Pathogens	Metabolites
S. aureus	Acetic acid, Aminoacetophenone, Ammonia, Ethanol, Formaldehyde, Isobutanol, Isopentyl acetate, Isopentanol, Methyl ketones, Trimethylamine, 1-Undecene, 2,5-Dimethylpyrazine isoamylamine, 2-Methylamine
E. coli	Acetaldehyde, Acetic acid, Aminoacetophenone, Butanediol, Decanol, Dimethyldisulfide, Dimethyltrisulfide, Dodecanol, Ethanol, Formaldehyde, Formic acid, Hydrogen sulfide, Indole, Lactic acid, Methanethiol, Methyl ketones, Octanol, Pentanols, Succinic acid, 1-Propanol
P. aeruginosa	Butanol, Dimethyldisulfide, Dimethyltrisulfide, Esters, Methyl ketones, Isobutanol, Isopentanol, Isopentyl acetate, Pyruvate, Sulphur compounds, Toluene, 1-Undecene, 2-Aminoacetophenone, 2-Butanone, 2-Heptanone, 2-Nonanone, 2-Undecanone

And four kinds of pollutant gases including benzene (C_6H_6), formaldehyde (CH_2O), acetone (C_3H_6O), ethylalcohol (C_2H_5) are sampled as the unlabeled data set.

Before the sampling experiments, we firstly set the temperature and humidity of the chamber as 25 °C and 40%. Then we began the gas sampling experiments. Because some of the gases are liquid, a decompression device is used to convert the liquid to the gas phase. We took advantage of the same setup and sensor array used for wound infections to get the pollutant gases' data set and each sampling experiment was strictly executed according to the 3 steps in Section 2.1. In total, we collected 2664 samples of pollutant gases.

To get the real concentration of each gas in the chamber, we extract each gas from the chamber and import it into the gas bag. Then spectrophotometric method is employed to get the concentration of formaldehyde, and the concentration of benzene, acetone and ethylalcohol is determined by gas chromatography (GC). The real concentration of three gas is shown in Table 3. For the four gas, there are 12, 11, 12 and 21 concentration points, respectively, and 12 sampling experiments are made on each concentration point.

Table 3. Concentration of the target gases.

Gases	Concentration Range (ppm)	Number of Samples
benzene	[0.1721, 0.7056]	480 (12 × 12)
formaldehyde	[0.0668, 0.1425]	491 (12 × 11)
acetone	[0.0565, 1.2856]	549 (12 × 12)
ethylalcohol	[0.0832, 0.6732]	1144 (12 × 21)

In most experiments, we only applied 652 samples and detailed information is shown at Table 4.

Table 4. Amount of samples.

Pollutant Gas	Amount of Samples	Wound Infection	Amount of Samples
benzene	132	*S. aureus*	20
formaldehyde	203	*E. coli*	20
acetone	153	*P. aeruginosa*	20
ethylalcohol	164	uninfected	20

Figure 3 illustrates the sensor response process when the sensor array is exposed to four types wound infection odor. It is clear that each curve has a rise when the target gas passes over the sensor array.

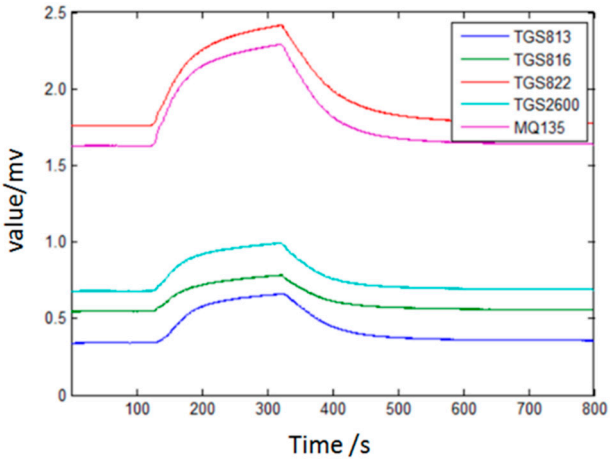

Figure 3. Response curve of the sensor array of *S. aureus* (one of the wound infections).

2.3. Data Preprocessing

To find which feature matrix can obtain the best performance, we extract five different features to construct our data set. These features include:

(1) Maximum value of steady-state response;
(2) Maximum slope of rising edge;
(3) Maximum slope of falling edge;
(4) Integral;
(5) Wavelet transform.

Except for wavelet transform, all other features are easy to understand. Wavelet transform is a kind of local transformation of time and frequency domain, which can efficiently fetch information from the signal, and it also has been used in the E-nose before [25]. It inherited and developed the localization of the short time Fourier transform (STFT), at the same time overcoming the shortcomings,

such as window size, which does not vary with frequency change. Wavelet transform can provide a time–frequency window that changes with the frequency, and is an ideal signal time–frequency analysis and processing tool.

After feature extraction, the labeled data set and unlabeled data set share the same dimension. We use the five features to construct original feature matrices, each row is a feature, and each column is a sensor selected from the 5 sensors. Then, we transform this $n \times n$ matrix into $1 \times n^2$ matrix, building the data set $x \in R^{n^2}$, and n^2 represents the dimension of x. To get training sample x_l, we randomly pick 15 samples from each wound infection sample (in total, 20 samples in each wound infection type), then we have 60 training samples x_l and 20 test samples x_t. We also construct a data set comprised of all unlabeled samples x_u to train a basis θ, and x_u contains four types of gas. In this paper, n is set as 3, 4, 5 to find which n is most efficient to improve the performance of the E-nose.

3. Self-Taught Learning

In this paper, we apply the self-taught learning paradigm with sparse autoencoder and classification algorithms (like RBF) to build a classifier for distinguishing different types of wound infection.

Suppose there is a labeled data set of m samples $\{(x_l^{(1)}, y^{(1)}), (x_l^{(2)}, y^{(2)}), \ldots, (x_l^{(m)}, y^{(m)})\}$. Each $x_l^{(i)} \in R^{n^2}$ denotes an original input feature vector. Each $y^{(i)} \in \{1, 2, \ldots, C\}$ denotes corresponding class label. Additionally, we assume there are k unlabeled samples $x_u^{(1)}, x_u^{(2)}, \ldots, x_u^{(k)} \in R^{n^2}$.

We propose a bold hypothesis, the class labels of unlabeled data set x_u and the class labels of labeled data set have no intersection. Then, we apply the sparse autoencoder algorithm to study a sparse autoencoder from $x_u^{(1)}, x_u^{(2)}, \ldots, x_u^{(k)} \in R^{n^2}$. It can be used to rebuild the representation of input training data set $x_l^{(1)}, x_l^{(2)}, \ldots, x_l^{(m)} \in R^{n^2}$, converting it into a new labeled training set $\{\hat{x}_l^{(1)}, \hat{x}_l^{(2)}, \ldots, \hat{x}_l^{(m)}\}$. These activations are put into the classifier as the new input feature vector, and PLSDA and RBF are employed as classifiers in this paper.

3.1. Sparse Autoencoder

3.1.1. Neural Network

The sparse autoencoder algorithm is an unsupervised learning algorithm that applies back-propagation, which is widely applied to image identification [20,26,27].

A single-layer autoencoder [28,29] is a kind of neural network [30–32] that only has one hidden layer. By hooking together many simple "neurons", a neural network is created. In this paper, each x_u is a neuron. For example, here is a small neural network (Figure 4).

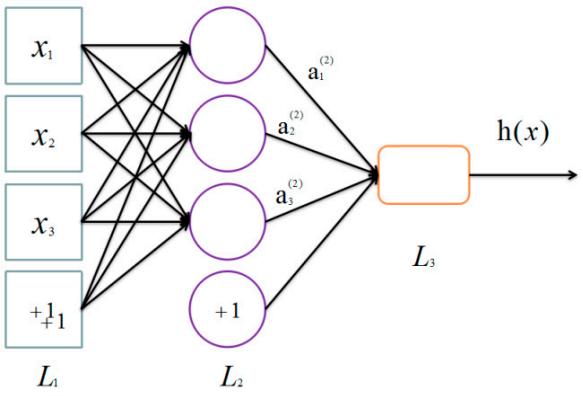

Figure 4. Basic neural network.

Then, there is a way of defining a complex non-linear form of hypotheses $h(x)$, with parameters W, b that can fit our data. Formally, corresponded to an input $x_u \in R^{n^2}$, the activations of x_u are

$$\begin{aligned} a_1^{(2)} &= f(W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + b_1^{(1)}), \\ a_2^{(2)} &= f(W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + b_2^{(1)}), \\ a_3^{(2)} &= f(W_{31}^{(1)}x_1 + W_{32}^{(1)}x_2 + W_{33}^{(1)}x_3 + b_3^{(1)}), \\ h(x) = a_1^{(3)} &= f(W_{11}^{(2)}a_1^{(2)} + W_{12}^{(2)}a_2^{(2)} + W_{13}^{(2)}a_3^{(2)} + b_1^{(2)}), \end{aligned} \tag{1}$$

where we define $f(\cdot)$ to be the sigmoid function $f(z) = \frac{1}{1+\exp(-z)}$, and $a_i^{(l)}$ denotes the activation (output value) of unit i in layer l, $b_i^{(l)}$ represents the bias associated with unit i in layer l , $W_{ij}^{(l)}$ is the parameter (or weight) associated with the connection between unit j in layer l, and unit i in layer $l+1$.

We can write these equations more compactly as

$$a^{(l)} = f(W^{(l-1)}x + b^{(l-1)}). \tag{2}$$

In this sequel, $a^{(l)}$ is the vector of activations of layer l, $W^{(l-1)}$ is the weight matrix of $a^{(l)}$, and similarly $b^{(l-1)}$ is the bias vector that computes $a^{(l)}$.

More generally, we define a equation that

$$z_i^{(l)} = \sum_{j=1}^{n} W_{ij}^{(l-1)} a_j^{(l-1)} + b_i^{(l-1)}. \tag{3}$$

Therefore we have $a_i^{(l)} = f(z_i^{(l)})$. For a neural network which owes p layers, we have output $h(x) = a^{(p)}$, and we call the process to compute $a^{(l)}$ from $a^{(1)}$ to $a^{(p)}$ $(h(x))$ as "the feedforward pass".

After performing a feedforward pass, we initialize $W_{ij}^{(l)}$ and $b_i^{(l)}$ to the value near 0 nearly, then the gradient descent algorithm incorporated with BP (backpropagation) is employed as an optimization algorithm.

The cost function is defined as follows:

There is $J(W,b;x,y) = \frac{1}{2}||h(x)-y||^2$ for each sample, and because we have k unlabeled samples, thus the overall cost function can be written as:

$$\begin{aligned} J(W,b) &= \left[\tfrac{1}{k}\textstyle\sum_{i=1}^{k} J(W,b;x^{(i)},y^{(i)})\right] + \tfrac{\mu}{2}\textstyle\sum_{l=1}^{p-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_l+1}\left(W_{ji}^{(l)}\right)^2, \\ &= \left[\frac{1}{k}\sum_{i=1}^{k}\left(\frac{1}{2}||h(x)-y||^2\right)\right] + \frac{\mu}{2}\sum_{l=1}^{p-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_l+1}\left(W_{ji}^{(l)}\right)^2 \end{aligned} \tag{4}$$

where μ controls the relative importance of the two terms. In our project, we set μ as 3×10^{-3}. Our target is to minimize $J(W,b)$, here we repeatedly implement the batch gradient descent to reduce our cost function $J(W,b)$. One iteration of batch gradient descent is shown as follows:

1. Make $\Delta W^{(l)} := 0$, $\Delta b^{(l)} := 0$ (matrix/vector of zeros) for all l
2. For i = 1 to k, we use BP to compute $\nabla_{W^{(l)}} J(W,b;x,y)$ and $\nabla_{b^{(l)}} J(W,b;x,y)$ firstly, then set

$$\begin{aligned} \Delta W^{(l)} &:= \Delta W^{(l)} + \nabla_{W^{(l)}} J(W,b;x,y), \\ \Delta b^{(l)} &:= \Delta b^{(l)} + \nabla_{b^{(l)}} J(W,b;x,y). \end{aligned} \tag{5}$$

3. Finally, we transform our parameters as:

$$\begin{aligned} W^{(l)} &:= W^{(l)} - \alpha\left[\tfrac{1}{k}\Delta W^{(l)} + \mu W^{(l)}\right], \\ b^{(l)} &:= b^{(l)} - \alpha\left[\tfrac{1}{k}\Delta b^{(l)}\right], \end{aligned} \tag{6}$$

where α is the learning rate. In step 2, it is crucial to compute the partial derivatives via BP, which is detailed described as following:

I For each output unit i in layer p (the output layer), set

$$\delta^{(p)} = -(y - a^{(p)}) \cdot f'(z^{(p)}), \tag{7}$$

where $\delta^{(p)}$ is defined as the difference between the network's activation and the true target value.

II For $l = p-1, p-2, \ldots, 2$, set

$$\begin{aligned} \delta^{(l)} &= \left((W^{(l)})^T \delta^{(l+1)}\right) \cdot f'(z^{(l)}), \\ f'\left(z^{(l)}\right) &= a^{(l)}(1 - a^{(l)}). \end{aligned} \tag{8}$$

III Compute the partial derivatives

$$\begin{aligned} \nabla_{W^{(l)}} J(W, b; x, y) &= \delta^{(l+1)} (a^{(l)})^T, \\ \nabla_{b^{(l)}} J(W, b; x, y) &= \delta^{(l+1)}. \end{aligned} \tag{9}$$

Notably, BP is not that easy to debug and get right. In the following section, we provide a derivative checking procedure to check the correctness of the code and make sure our implementing of gradient descent is correct. In a correct code we have:

$$\begin{aligned} \nabla_{W^{(l)}} J(W, b) &= \tfrac{1}{k} \Delta W^{(l)} + \mu W^{(l)}, \\ \nabla_{b^{(l)}} J(W, b) &= \tfrac{1}{k} \Delta b^{(l)}. \end{aligned} \tag{10}$$

If the equation is satisfied, it proves that we indeed get the correct derivations. In practice, we define θ as a vector unrolling the parameters W, b. Thus, when a function $g(\theta) = \frac{dJ(\theta)}{d\theta}$ is given, we can verify its correctness by checking that whether the following formula is satisfied.

$$g(\theta) \approx \frac{J(\theta + \epsilon) - J(\theta - \epsilon)}{2\epsilon}. \tag{11}$$

In practice, we set ϵ which is always around 10^{-4} to a small constant.

3.1.2. Autoencoders and Sparsity

Thus far, we have described the application of neural network to supervised learning, but we have only used the unlabeled training data set; an autoencoder neural network that combines BP is introduced to deal with such a situation.

The auto encoder tries to learn an identity function that enforces $h(x) \approx x$, which means the target value is $y_u^{(i)} = x_u^{(i)}$.

As Figure 5 shows, this is a simple auto encoder. Our goal is to enforce output $\hat{x}$ to be similar to input x. To achieve this, we set constraints on the ordinary neural network.

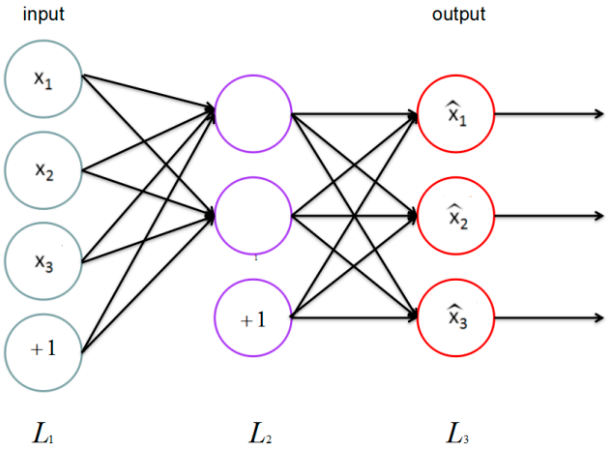

Figure 5. Sparse autoencoder.

We let $a_j^{(2)}(x)$ denote the activation of this hidden unit when the network is given a specific input x. And next step we compute the average activation of hidden unit j.

$$\hat{\rho_J} = \frac{1}{k}\sum_{i=1}^{k}[a_j^{(2)}(x^{(i)})]. \tag{12}$$

We impose a constraint to $\hat{\rho_J}$ that $\hat{\rho_J} = \rho$, where ρ is a sparsity parameter whose value is close to zero. In our paper, we set it as 0.01. j is the sum of whole hidden units in the network.

Performing the above two equations, we are ready to complete the overall cost function on the basis of Equation (4) to be

$$J_{sparse}(W,b) = J(W,b) + \beta\sum_{j=1}^{s_2} KL(\rho||\hat{\rho_J}), \tag{13}$$

where $J(W,b)$ is the same as the original one, β is set as 3 in our project, controlling the weight of the sparsity penalty term. s_2 denotes the number of nodes in the hidden layer; and as Equation (13) shows, we apply KL-divergence as the penalty term, which can be expressed as $KL(\rho||\hat{\rho_J}) = \rho log\frac{\rho}{\hat{\rho_J}} + (1-\rho)log\frac{1-\rho}{1-\hat{\rho_J}}$. Until now, the KL-divergence term has been adopted to satisfy the constraints, also, to integrate the KL-divergence into our derivative calculation, we adjust Equation (8):

$$\delta^{(2)} = \left(\left((W^{(2)})^T\delta^{(3)}\right) + \beta(-\frac{\rho}{\hat{\rho_J}} + \frac{1-\rho}{1-\hat{\rho_J}})\right)\cdot f'(z^{(2)}). \tag{14}$$

Other steps are the same as the neural network algorithm, and we perform a gradient descent on the new objective $J_{sparse}(W,b)$. Still we apply the derivative checking method to verify our code. The algorithm, therefore, encourages the activations a_u to be sparse, in other words, for most of its elements to be zero. At the same time we learn the basis vector θ (unrolling parameters W, b into a long vector). It is the basis of both x_u and x_l.

3.2. Construct New Representation

So far, we have trained a basis vector θ (a set of parameters $W^{(1)}$, $W^{(2)}$, $b^{(1)}$, $b^{(2)}$), which would be used to construct the new labeled data set $\{\hat{x}_l^{(1)}, \hat{x}_l^{(2)}, \ldots, \hat{x}_l^{(m)}\}$ based on original labeled data set. We pose the following formulation to solve the problem, for each $x_l^{(i)}$, we have

$$\hat{x}_l^{(i)} = f(W^{(2)}f(W^{(1)}x_l^{(i)} + b^{(1)}) + b^{(2)}), \tag{15}$$

The reconstructing procedure decreases the difference between the labeled data and unlabeled data as well as transfers knowledge from different domains. These new features are put into multiple types of classifications, such as RBF. The whole algorithm of self-taught learning based on sparse autoencoder is illustrated in Algorithm 1 and Figure 6.

Algorithm 1. Self-taught learning algorithm

Step 1: Minimize $J_{sparse}(W, b)$, train the basis vector θ (unrolling parameters W, b into a long vector) from unlabeled data x_u.
Step 2: Take advantage θ to construct new representation $\hat{x}_l$, replacing original data set x_l.
Step 3: Learn a classifier by applying efficient algorithm (we apply RBF and PLSDA optimized by EQPSO here).
Step 4: Calculate the classification accuracy.

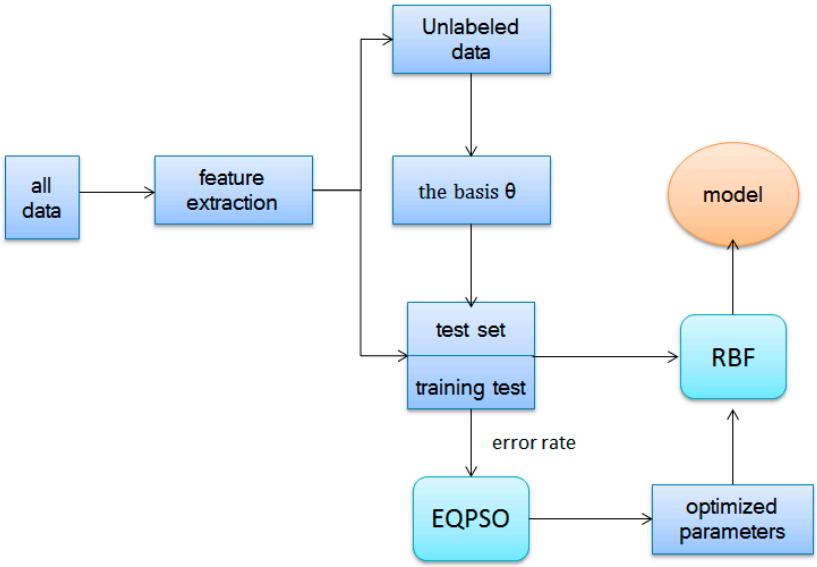

Figure 6. Flow chart of self-taught learning.

4. Results and Discussion

To verify the feasibility of the model we performed some experiments and obtained some results, which will be shown in this section. In order to improve the performance of the classifier, EQPSO is applied to optimize the parameters, both for RBF and PLSDA.

In all experiments, the particle number of optimization algorithm is set to 30, and the algorithm iterates 300 times to find the optimal value. Additionally, we set the hidden layer as 10 at first, the sparsity parameter as 0.01, β as 3 in this project to get a sparse autoencoder. All the data set, which is the input of classifier, has been normalized before self-taught learning.

Before showing the results, we display the input feature matrix and the reconstructed ones in Figure 7. We take one sample of *S. aureus* as an example. In this experiment, the hidden layer is set as 10 and the dimension of input matrix is 3×3. The left picture of Figure 7 shows the feature matrix of input matrix and the right one shows the reconstructed matrix.

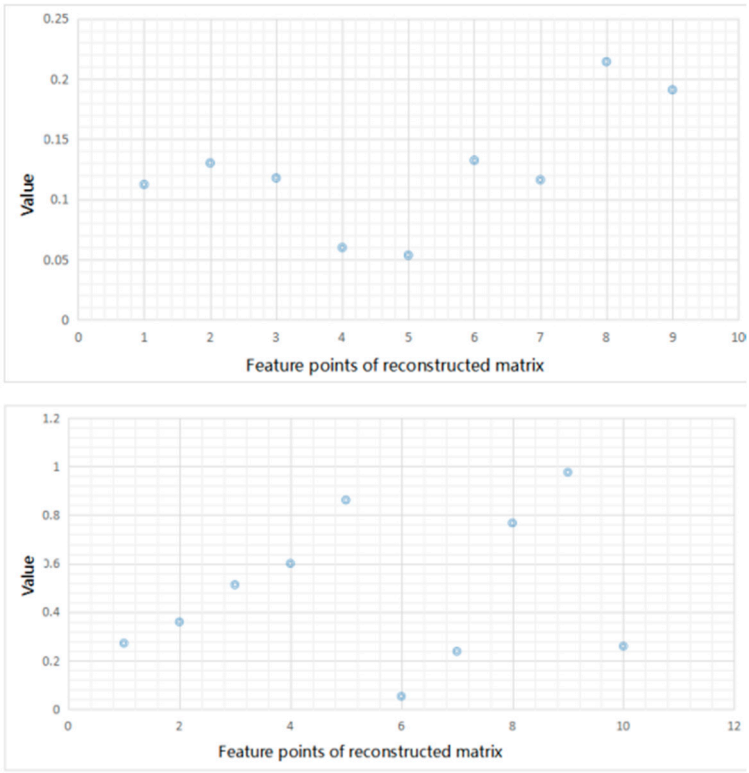

Figure 7. Input feature matrix and reconstructed matrix of one sample.

Firstly, considering that each sensor has a different selectivity pattern, the performances could change without increasing the size of the matrix, but only selecting the best subset of features. So we combine different sensors with different features from dimension 3×3 to 5×5 and finally find that the results are almost the same when dimensions are the same. Then we discuss whether the self-taught learning could improve the performance of the E-nose and how the dimension of feature matrix influences the classification accuracy. We apply RBF as the classifier here and 15 samples of each wound infection serve as the training data, and 5 samples of each wound infections are the test data. The test set is used to verify the performance of the final model. In Table 5, we show the results studied from 652 gas samples.

Table 5. Accuracy of three kinds of feature matrix with radial basis function (RBF) (%).

Dimension	Raw		Spares Autoencoder	
	Training Set	Test Set	Training Set	Test Set
3×3	98.3	60	100	60
4×4	100	70	91.6	75
5×5	88.3	80	80	90

In Table 5, we find that, compared to the raw matrix, from dimension 3×3 to 5×5, the matrix studied from unlabeled samples has higher accuracy overall. However, when the dimension is 3×3, the accuracy of the training set and test set do not change much, which means that if the dimension is small, the accuracy may not improve. At the same time, however, the number of unlabeled samples also contributes to the results of classification. We change the size of unlabeled data set from 652 to 2664 to explore the relationship between them. All of the results are shown in Figure 8.

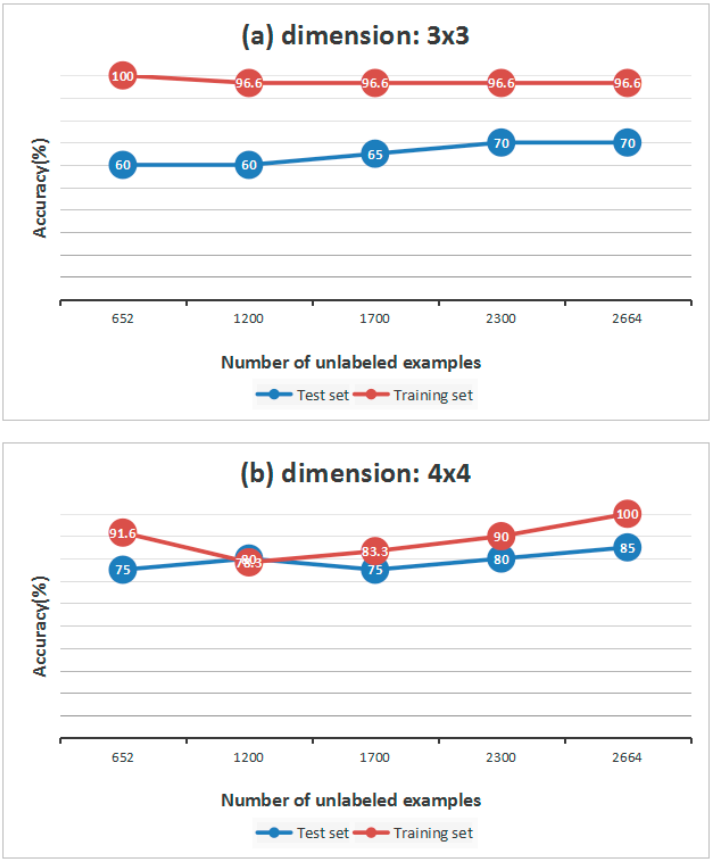

Figure 8. Accuracy with different number of unlabeled samples. (**a**) shows the change of accuracy as the number of examples increases when dimension is 3 × 3; (**b**) shows the change of accuracy as the number of examples increases when dimension is 4 × 4.

In general we can draw a conclusion that, the more unlabeled samples, the higher the accuracy. If the interval of two sizes is small, the accuracy may not change. When the dimension is 3 × 3, the classification accuracy of the test set keeps rising, while in Figure 8b, the curve of the training set goes down when the size of unlabeled data set is 1200, and the curve of the test set goes down when the size of the unlabeled data set is 1700.

It is well established that the classifier is a major part of the E-nose; thus, PLSDA and RBF are introduced to distinguish wound infection samples respectively in our paper. In order to compare the differences in putting the feature matrix into these two classifiers directly and the representations processed by self-taught learning, we calculate the accuracy of both. The accuracy is shown in Table 6.

Table 6. Accuracy with partial least squares discriminant analysis (PLSDA) and RBF (%).

Dimension		RBF		PLSDA	
		Train Set	Test Set	Train Set	Test Set
3 × 3	raw	98.3	60	53.3	45
	sa	100	60	53.3	45
4 × 4	raw	100	70	80	60
	sa	91.6	75	76.6	75
5 × 5	raw	88.3	80	76.6	60
	sa	80	90	78.3	40

Note: "sa" in Table 6 represents "spares autoencoder".

Table 6 shows the results of our experiments and the classification accuracies are clearly presented in it. Comparing Figure 9 with Figure 10, RBF is indeed more suitable for self-taught learning than PLSDA, because all of the accuracies of RBF are higher than accuracies of PLSDA, especially for the test data set. Additionally, comparing (a) with (b) in Figures 9 and 10, the accuracy of self-taught learning rises more quickly than the raw matrix. As for PLSDA, the results are not very good and steady, when the input is raw matrix, the accuracy of test data grows slowly. All results in Figures 9 and 10 prove that RBF is superior to PLSDA in self-taught learning based on the sparse autoencoder. In Figure 10b, when the input is studied from the unlabeled data set, the accuracy of the test set reaches its peak at 4×4, then the curve falls down to 40% at 5×5.

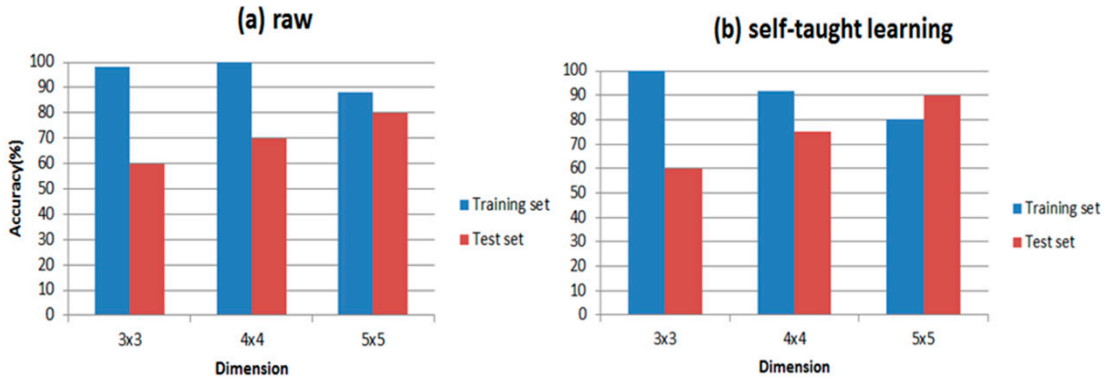

Figure 9. Accuracy with RBF. When dimension changes from 3×3 to 5×5, (**a**) shows the accuracy of unprocessed data set (training set and test set); (**b**) shows the accuracy of data set processed by self taught learning.

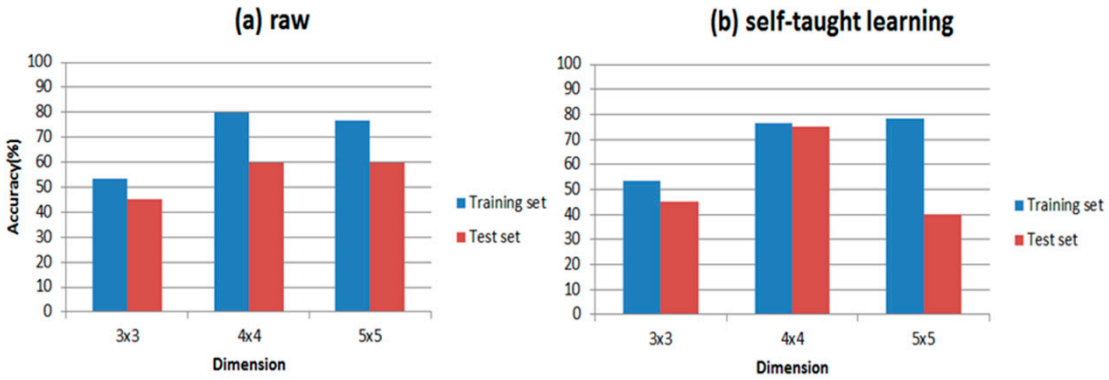

Figure 10. Accuracy with PLSDA. When dimension changes from 3×3 to 5×5, (**a**) shows the accuracy of unprocessed data set (training set and test set); (**b**) shows the accuracy of data set processed by self taught learning.

All experiments above are established under the condition that the hidden layer is ten. In order to study the difference that the hidden layer brings to the accuracy, we set the hidden layer at 5, 10, 20, 40, 100, 700, 2000, 10,000.

Except for hidden layer, all the other parameters remain the same. On account of the instability of the classification accuracy, to make sure the result of the experiment is correct, each program is repeated 5 times.

Tables 7–9, respectively, show the classification accuracy with different hidden layers when the dimension is 5×5, 4×4 and 3×3. From Tables 7–9, we can draw a conclusion that as the hidden layer increases, the classification accuracy always reaches the top at first and falls down later. And when the hidden layer is large, the accuracy always maintains at a certain level; at the same time,

it takes more time to train the model, which means increasing the hidden layer can lead to a waste of time.

Table 7. Accuracy with different hidden layer (%).

Hidden Layer	5	10	20	40	100	700	2000	10,000
Training set	96.6	80	93.3	84.15	91.6	91.6	91.6	91.6
Test set	65	90	90	87.5	80	75	75	75

Note: the dimension is 5 × 5 and the classifier is RBF.

Table 8. Accuracy with different hidden layer (%).

Hidden Layer	5	10	20	40	100	700	2000	10,000
Training set	88.3	91.6	100	89.15	100	91.6	95	100
Test set	80	75	75	80	75	85	80	70

Note: the dimension is 4 × 4 and the classifier is RBF.

Table 9. Accuracy with different hidden layer (%)

Hidden Layer	5	10	20	40	100	700	2000	10,000
Training set	96.6	100	100	98.3	100	96.6	98.3	98.3
Test set	60	60	60	75	65	75	70	65

Note: the dimension is 3 × 3 and the classifier is RBF.

In conclusion, these results prove that the self-taught learning based on sparse autoencoder demonstrates a good performance in improving the accuracy by studying samples from other fields, and that there are a few reasons for why this is possible with the sparse autoencoder. Firstly, as a typical algorithm of traditional training multi-layer network, the BP algorithm is not ideal for only a few layers of network. If all layers are trained at the same time, the time complexity will be too high; If only one layer is trained, the bias will pass by the layer. This will face the opposite problem of the above supervised learning, which will be badly mismatched. Self-taught learning based on sparse autoencoder is a kind of layer-wise-pre-training, which solves the problem effectively. Furthermore, the sparsity constraint makes the representation of each layer sparse (most of nodes become zero). This kind of representation resembles the human brain—when something comes to our mind, only a small number of neurons are stimulated and other neurons are suppressed. This feature is the same with humans when we want to study new things from other fields.

5. Conclusions

The self-taught learning approach is a new model of transfer learning and has not been used in E-nose before; in this paper, we have applied self-taught learning to wound infections classification.

In this paper we introduce a kind of self-taught learning based on a sparse autoencoder to the E-nose in wound infection detection, and we take advantage of PLSDA and RBF which are optimized by EQPSO to classify four kinds of wound infections. Through comparing the results of self-taught learning and the results of the raw data set, we can draw a conclusion that the performance rises when we apply self-taught learning based on sparse autoencoder, especially with the RBF classifier. We also found that the size of unlabeled data set, the type of classifier and the dimensions of data set all have an impact on the accuracy of pattern-recognition.

These results prove that the self-taught learning based on sparse autoencoder has a good performance in transforming knowledge, and indeed, could improve the accuracy by studying samples from other fields. However, it still has its limits. First of all, when the number of unlabeled samples is small, especially when the order of magnitude is smaller than 10, the algorithm barely improves the performance.

Through self-taught learning, we can reduce costs when we train the E-nose for distinguishing wound infections, which is the purpose of our experiments. We all know that the data of such gas is not easy to obtain, thus we study knowledge from unlabeled data set in other fields, which means that we can have a significant amount of data to study, making up for the lack of wound infection samples and transferring it to the area of wound infections.

In future work, we will further study the self-taught learning applied in E-nose, and we believe E-nose will be further improved in the field of medical science.

Acknowledgments: The work is supported by Program for New Century Excellent Talents in University (No. 47), National Science Foundation of China (Nos. 61372139, 61101233, 60972155), China Postdoctoral Science Foundation (No. 2016M602630) and Fundamental Research Funds for the Cetral Universities (No. XDJK2015C073).

Author Contributions: Peilin He was in charge of the project management and proposed the algorithm. Pengfei Jia was responsible for data analysis and the discussion of the results. Siqi Qiao provided valueable advice about the revised manuscript. Shukai Duan was involved in discussions and the experimental analysis.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Natale, C.D.; Macagnano, A.; Mantini, A.; Davide, F.; D'Amico, A.; Paolesse, R.; Boschi, T.; Faccio, M.; Ferri, G. Advances in food analysis by electronic nose. In Proceedings of the IEEE International Symposium on Industrial Electronics, Guimaraes, Portugal, 7–11 July 1997; Volume 1, pp. SS122–SS127.
2. Trihaas, J.; Nielsen, P.V. Electronic nose technology in quality assessment: Monitoring the ripening process of Danish blue cheese. *J. Food Sci.* **2005**, *70*, 44–49.
3. Peris, M.; Gilabert, L.E. A 21st century technique for food control: Electronic noses. *Anal. Chim. Acta* **2009**, *638*, 1–15. [CrossRef]
4. Hui, G.; Wang, L.; Mo, Y.; Zhang, L. Study of grass carp (Ctenopharyngodon idellus) quality predictive model based on electronic nose. *Sens. Actuators B Chem.* **2012**, *35*, 301–308.
5. Green, G.C.; Chan, A.D.C.; Dan, H.; Lin, M. Using a metal oxide sensor (MOS)-based electronic nose for discrimination of bacteria based on individual colonies in suspension. *Sens. Actuators B Chem.* **2011**, *152*, 21–28. [CrossRef]
6. Chapman, E.A.; Thomas, P.S.; Stone, E.; Lewis, C.; Yates, D.H. A breath test for malignant mesothelioma using an electronic nose. *Eur. Respir. J.* **2012**, *40*, 448–454. [CrossRef] [PubMed]
7. Jia, P.; Tian, F.; He, Q.; Fan, S.; Liu, J.; Yang, S.X. Feature extraction of wound infection data for electronic nose based on a novel weighted KPCA. *Sens. Actuators B Chem.* **2014**, *201*, 555–566. [CrossRef]
8. D'Amico, A.; Di Natale, C.; Falconi, C.; Martinelli, E.; Paolesse, R.; Pennazza, G.; Santonico, M.; Sterk, P.J. Detection and identification of cancers by the electronic nose. *Expert Opin. Med. Diagn.* **2012**, *6*, 175–185. [CrossRef]
9. Ameer, Q.; Adeloju, S.B. Polypyrrole-Based Electronic Noses for Environmental and Industrial Analysis. *Sens. Actuators B Chem.* **2005**, *106*, 541–552.
10. Lamagna, A.; Reich, S.; Rodríguez, D.; Boselli, A.; Cicerone, D. The use of an electronic nose to characterize emissions from a highly polluted river. *Sens. Actuators B Chem.* **2008**, *131*, 121–124. [CrossRef]
11. Byun, H.; Persaud, K.C.; Pisanelli, A.M. Wound-state monitoring for burn patients using E-nose/SPME system. *ETRI J.* **2010**, *32*, 440–446. [CrossRef]
12. Tian, F.; Xu, X.; Shen, Y.; Yan, J.; Ma, J.; He, Q.; Liu, T. Detection of wound pathogen by an intelligent electronic nose. *Sens. Mater.* **2009**, *21*, 155–166.
13. Thomas, A.N.; Riazanskaia, S.; Cheung, W.; Xu, Y.; Goodacre, R.; Thomas, C.L.P.; Baguneid, M.S.; Bayat, A. Novel noninvasive identification of biomarkers by analytical profiling of chronic woundsusing volatile organic compounds. *Wound Repair Regen.* **2010**, *18*, 391–400. [CrossRef] [PubMed]
14. Tian, F.; Yan, J.; Xu, S.; Feng, J.; He, Q.; Shen, Y.; Jia, P.; Kadri, C. Classification of electronic nose data on wound infection detection using support vector machine combined GA. *J. Comput. Inf. Syst.* **2012**, *8*, 3340–3357.
15. Li, S.; Wang, R.; Hu, W.; Sun, J. A New QPSO Based BP Neural Network for Face Detection. In Proceedings of the Second International Conference of Fuzzy Information and Engineering, Guangzhou, China, 13–16 May 2007; pp. 355–363.

16. Yang, W.; Wu, W.; Fan, Y.; Li, Z. Particle Swarm Optimization Based on Local Attractors of Ordinary Differential Equation System. *Discret. Dyn. Nat. Soc.* **2014**, *2014*, 1–10. [CrossRef]
17. Pan, S.J.; Yang, Q. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1345–1359. [CrossRef]
18. Raina, R.; Battle, A.; Lee, H.; Packer, B.; Ng, A.Y. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the International Conference on Machine Learning*; ACM: New York, NY, USA, 2007; pp. 759–766.
19. Raina, R. Self-Taught Learning. Ph.D. Thesis, Stanford University, Stanford, CA, USA, 2009.
20. Deng, J.; Zhang, Z.; Marchi, E.; Schuller, B. Sparse Autoencoder-Based Feature Transfer Learning for Speech Emotion Recognition. In *Proceedings of the Humaine Association Conference on Affective Computing and Intelligent Interaction*; IEEE Computer Society: Washington, DC, USA, 2013; pp. 511–516.
21. Amft, O. Self-Taught Learning for Activity Spotting in On-body Motion Sensor Data. In Proceedings of the Annual International Symposium on Wearable Computers, San Francisco, CA, USA, 12–15 June 2011; Volume 1416, pp. 83–86.
22. Lee, H.; Raina, R.; Teichman, A.; Ng, A.Y. Exponential family sparse coding with applications to self-taught learning. In *Proceedings of the International Jont Conference on Artifical Intelligence*; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 2009; pp. 1113–1119.
23. Park, J.; Sandberg, I. Universal Approximation Using Radial-Basis-Function Networks. *Neural Comput.* **1991**, *3*, 246–257. [CrossRef]
24. Schwenker, F.; Kestler, H.A.; Palm, G. Three learning phases for radial-basis-function networks. *Neural Netw.* **2001**, *14*, 439–458. [CrossRef]
25. He, Q.; Yan, J.; Shen, Y.; Bi, Y.; Ye, G.; Tian, F.; Wang, Z. Classification of Electronic Nose Data in Wound Infection Detection Based on PSO-SVM Combined with Wavelet Transform. *Intell. Autom. Soft Comput.* **2012**, *18*, 967–979. [CrossRef]
26. Plaut, D.C.; Nowlan, S.J.; Hinton, G.E. Experiments on Learning by Back Propagation. *Artif. Intell.* **1986**, *16*, 1–54.
27. Liu, H.; Taniguchi, T.; Takano, T.; Tanaka, Y.; Takenaka, K.; Bando, T. Visualization of driving behavior using deep sparse autoencoder. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*; IEEE: New York, NY, USA, 2014; pp. 1427–1434.
28. Leung, H.; Haykin, S. The complex backpropagation algorithm. *IEEE Trans. Signal Process.* **1991**, *39*, 2101–2104. [CrossRef]
29. Liu, H.; Taniguchi, T.; Tanaka, Y.; Takenaka, K. Essential feature extraction of driving behavior using a deep learning method. In Proceedings of the 2015 IEEE Intelligent Vehicles Symposium (IV), Seoul, Korea, 28 June–1 July 2015; pp. 1054–1060.
30. Hagan, M.T.; Demuth, H.B.; Beale, M. *Neural Network Design*; China Machine Press: Beijing, China, 2002.
31. Rowley, H.A. Neural Network-Based Face Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1998**, *20*, 23–38. [CrossRef]
32. Specht, D.F. A general regression neural network. *IEEE Trans. Neural Netw.* **1991**, *2*, 568–576. [CrossRef] [PubMed]