JMLR: Workshop and Conference Proceedings 39:96–111, 2014 ACML 2014

Structured Denoising Autoencoder for Fault Detection and Analysis

Takaaki Tagawa TAGAWA@MOSK.TYTLABS.CO.JP
Yukihiro Tadokoro TADOKORO@MOSK.TYTLABS.CO.JP
TOYOTA CENTRAL R&D LABS., INC., Japan

Takehisa Yairi YAIRI@SPACE.RCAST.U-TOKYO.AC.JP
RCAST, University of Tokyo, Japan

Editor: Dinh Phung and Hang Li

Abstract

This paper proposes a new fault detection and analysis approach which can leverage incomplete prior information. Conventional data-driven approaches suffer from the problem of overfitting and result in high rates of false positives, and model-driven approaches suffer from a lack of specific information about complex systems. We overcome these problems by modifying the denoising autoencoder (DA), a data-driven method, to form a new approach, called the structured denoising autoencoder (StrDA), which can utilize *incomplete* prior information. The StrDA does not require specific information and can perform well without overfitting. In particular, an empirical analysis with synthetic data revealed that the StrDA performs better than the DA even when there is partially incorrect or abstract information. An evaluation using real data from moving cars also showed that the StrDA with incomplete knowledge outperformed conventional methods. Surprisingly, the StrDA results were better even though the parameters of the conventional methods were tuned using faulty data, which are normally unknown. In addition, the StrDA fault analysis was able to extract the true causes of the faulty data; the other methods were unable to do this. Thus, only our proposed method can explain why the faults occurred.

Keywords: Fault Detection and Analysis, Denoising Autoencoder, Semi-supervised Learning.

1. Introduction

To satisfy the need for highly reliable systems, it is necessary to test them in many intensive situations. If any faults are detected, they are thoroughly analyzed by the experts to identify the true cause, and then corrected. This procedure is important in wide variety of fields; for example, aviation, plant and vehicle systems. However, due to the increasing complexity of such systems, this analysis is becoming increasingly costly. One solution is to utilize the data obtained from the systems to apply machine learning technologies. In this settings, we are only given normal data to learn a model. Test data are then evaluated by the model to detect faults, and these are then analyzed to identify their causes. We call this the fault detection and analysis problem. The technologies for fault detection and analysis are categorized and applicable to a *one-class classification* problem which includes, e.g., novelty detection, event detection and change detection (Ding, 2008).

To deal with fault detection and analysis problems, several data-driven methods have been proposed, including principal component analysis, the one-class support vector machine, the local outlier factor, the artificial neural network, and others (Chandola et al., 2009). However, these methods suffer from overfitting, which causes high rates of false positives, and are thus unreliable. Prior knowledge can help us avoid such problems by allowing us to construct specific models of the system using physical equations (Ding, 2008). Unfortunately, as systems become more complex and extensive, building a model becomes increasingly difficult.

Our motivation is to provide a simple way to incorporate prior knowledge without building physical equations, considering that we still have some knowledge about the devices to be monitored and faults to be detected even if it is not precise. For example, we often have information about the interdependence of various attributes, e.g., between the acceleration and the engine rotation. We also can approximate the characteristics of the faults which we want to detect, based on the results of a sensory analysis. Our intuition is that using these types of additional information can makes a model to focus on some important relations in the data, which avoids fitting to trivial relations; data-driven approaches are difficult to do it. Actually, faults are originated from the corruption of the physical relations in the system and it is better to monitor only such the relations to detect and identify the faults well. However, the problem is that these kinds of knowledge are often abstract or partially incorrect. We do not have the detailed information about the model, which is used in the model-based approaches.

The proposed method, the structured denoising autoencoder (StrDA), exploits incomplete prior knowledge to focus on some relations between variables and ignore others; we simply set 1 to focus on and 0 to ignore the relations, and then impose this information on the objective function of the denoising autoencoder (DA). The advantages of the proposed function include controlling the denoising and forming a low-dimensional manifold in which prior knowledge is included. An empirical analysis with synthetic data revealed that the StrDA can obtain correct results even if only incomplete information is available. An evaluation with real data showed that incomplete knowledge can improve the performance of fault detection and analysis; the StrDA outperformed conventional methods which are tuned with faulty data (for data which are normally unknown). We also note that the StrDA fault analysis can determine the causes of the fault; the other methods cannot do this.

2. Related Works

2.1. Contribution Analysis (CA)

CA is a common approach to fault analysis (Ge and Song, 2013). Given a faulty sample of M observed variables $\mathbf{x} = \{x_1, \ldots, x_M\}$ and the corresponding variables $\mathbf{x}_r$ predicted by a learned normal model, CA considers the errors $\mathbf{e} = \{e_1, \ldots, e_M\}$ between them:

$$\mathbf{e} = \mathbf{x} - \mathbf{x}_r. \tag{1}$$

The amount of error e_i represents the contribution of the variable $i \in 1 \ldots M$ to the fault and the variables are then ranked in the order of their contributions. CA determines the

variable with higher contribution as a candidate for the cause (Figure 1). CA has been shown to be successful in studies in multivariate statistical process control (MSPC; Ge and Song (2013)). CA is generally applied with linear approaches since, unlike non-linear models, these methods allow the computation of reconstruction errors.

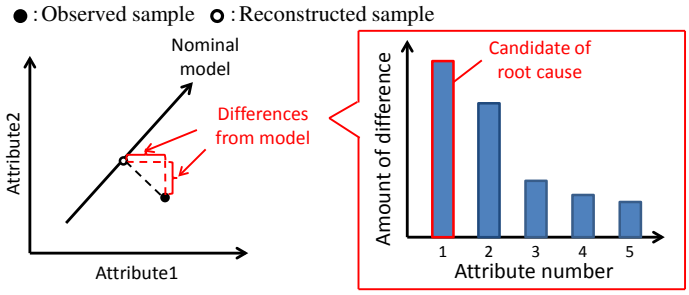

Figure 1: Contribution analysis.

2.2. Principal Component Analysis (PCA)

PCA is commonly used in connection with CA (Ge and Song, 2013). Given N i.i.d. samples of normal observations $\{\mathbf{x}^1, \ldots, \mathbf{x}^{(N)}\}^T$ of the observed variables $\mathbf{x}$, which are scaled to zero mean and unit variance, PCA learns a linear transformation matrix $\mathbf{W} = \{\mathbf{w}_1, \ldots, \mathbf{w}_d\}^T$ to form $d(\leq M)$ principal components $\mathbf{y} = \mathbf{W}\mathbf{x}$, where $\mathbf{y}$ is a low-dimensional linear subspace which represents the given training data. PCA projects a sample $\mathbf{x}$ to the subspace and reconstructs it back to the observation space, as follows:

$$\mathbf{x}_r = \mathbf{W}^T\mathbf{W}\mathbf{x} = \mathbf{W}^T\mathbf{y}, \tag{2}$$

which is viewed as a prediction of the model with a reconstruction error vector $\mathbf{e}$, which can be written as

$$\mathbf{e} = \mathbf{x} - \mathbf{W}^T\mathbf{W}\mathbf{x} = \mathbf{x} - \mathbf{x}_r. \tag{3}$$

Here we can utilize $\mathbf{e} = \{e_1, \ldots, e_M\}$ to conduct CA. We also use $\mathbf{e}$ to calculate the root-mean-square error (RMS)

$$q = \sqrt{\frac{1}{M}\sum_{i=1}^{M} e_i}, \tag{4}$$

which we use as an anomaly score for detecting faults. Although PCA is simple and easy to implement, it is a linear model, and thus when the data structure is complex, it does not usually achieve high-performance fault detection and analysis.

2.3. Denoising Autoencoder (DA)

The DA is a regularized autoencoder for learning generalized features which are useful for tasks such as *classification* (Bengio, 2009). Some researchers have also applied the autoencoder to intrusion detection problems, as a nonlinear extension of PCA to learn the non-linear data manifolds (Modi et al., 2013), which is closed to fault detection problems.

In DA, noise is added to the input vector $\mathbf{x}$. There are various ways to add noise (Vincent et al., 2010), but in this paper, we use Gaussian noise: $\mathbf{x}_n = \mathbf{x} + \mathcal{N}(0, \sigma I)$, where $\mathcal{N}(0, \sigma I)$ is an isotropic Gaussian noise with a shared variance σ. We then minimize the reconstruction errors between the true input and the output of the DA. The objective function $J(\boldsymbol{\theta})$ is given as

$$J(\boldsymbol{\theta}) = \sum_{p=1}^{N} \frac{1}{2} ||\mathbf{x}^{(p)} - \mathbf{x}_r^{(p)}||_2^2, \tag{5}$$

$$\mathbf{x}_r^{(p)} = \mathbf{W}_2 \mathbf{h}_1^{(p)} + \mathbf{b}_2, \tag{6}$$

$$\mathbf{h}_1^{(p)} = g(\mathbf{W}_1 \mathbf{x}_n^{(p)} + \mathbf{b}_1), \tag{7}$$

where p indicates the pth training sample, $\mathbf{x}_r$ is the reconstructed value, $\boldsymbol{\theta} = \{\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}_1, \mathbf{b}_2\}$ are the parameters of DA, and $g(\cdot)$ denotes the sigmoid function

$$g(x) = \frac{1}{1 + \exp(-x)}. \tag{8}$$

We can learn the parameters $\boldsymbol{\theta}$ by using a back-propagation algorithm. The standard autoencoder learns only a model in which the output takes the same value as the input data. In contrast, the DA uses noisy data to learn the model in which the output should be the same as the original normal data. This process requires that the model learns the regularized features $\mathbf{h}_1$ and that it can output data which efficiently ignore trivial noise or unknown faulty behavior. For fault detection and analysis, we compute $\mathbf{x}_r$ by equation (6) and (7) with the input $\mathbf{x}$ instead of $\mathbf{x}_n$. Then the reconstruction error $\mathbf{e} = \mathbf{x} - \mathbf{x}_r$ is used to conduct CA and calculate the RMS (4) as an anomaly score.

3. Structured Denoising Autoencoder (StrDA)

This section describes our proposed method. The StrDA utilizes incomplete prior information about the structure of the given data and/or knowledge of the possible faults.

3.1. Setting the Prior Knowledge

To determine which relations of the variables to be focused on, we express the prior knowledge by using an $M \times M$ matrix $\boldsymbol{\alpha}$, defined as follows:

$$\boldsymbol{\alpha} = \{\alpha_{ij} \in \{0, 1\} | i, j \in 1, \ldots, M\}. \tag{9}$$

Note that $\boldsymbol{\alpha}$ is a symmetric matrix thus $\alpha_{ij} = \alpha_{ji}$. When $\alpha_{ij} = 1$, we focus on the relation between ith and jth variables. On the other hand, when $\alpha_{ij} = 0$, the relation between the ith and jth variables is ignored. In this way, we can use $\boldsymbol{\alpha}$ to select which relations to be monitored (for example, see Figure 2). Note that we do not require precise information in order to improve the fault detection and analysis performance (see Section 4.1 for further discussion). The operator can use incomplete information as long as it is reasonably consistent. The following are examples of situations in which we have prior knowledge of the importance of relations between variables:

- The input/output structure of the variables can be assumed, such as from the accelerator position and engine rotation speed.
- We wish to focus on the relations between certain variables.
- Based on a sensory evaluation, we suspect certain variables.

If we know which relations are important, we can focus on and efficiently monitor only the essential parts of the data. When no such information is available, we must use an unsupervised model, which tends to cause overfitting of the training data. As we discuss in Section 4.1, this prior information can be used to ignore unimportant parts of the data distribution.

3.2. Structured Denoising Objective Function

We extend the DA to include the prior information $\boldsymbol{\alpha}$ by modifying the objective function (5). The proposed objective function is given by

$$\tilde{J}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{p=1}^{N}\sum_{i<j}\left\{\left(e_{ij}^{(p)}\right)^2 + \left(e_{ji}^{(p)}\right)^2\right\}, \tag{10}$$

$$e_{ij}^{(p)} = x_{r,i}^{(p)} - \left\{x_i^{(p)} + (1-\alpha_{ij})(x_{n,i}^{(p)} - x_i^{(p)})\right\}, \tag{11}$$

where $x_{n,i}$ and $x_{r,i}$ are the ith variable of $\mathbf{x}_n$ and $\mathbf{x}_r$. We can use the back-propagation algorithm to optimize the parameters $\boldsymbol{\theta}$. We call (10) the *structured denoising* (SD) objective function, and it reflects our prior information $\boldsymbol{\alpha}$. Recall that the DA uses reconstruction errors for CA, and the RMS is computed as an anomaly score. Therefore, we control these reconstruction errors in order to incorporate the given prior knowledge, and then proceed as in the DA. Actually, the SD objective function controls the denoising with respect to each combination of variables. For example, $\alpha_{ij} = 1$ follows $e_{ij} = x_{r,i} - x_i$, which means that the model should denoise correctly with respect to the i, jth variables. In the case of $\alpha_{ij} = 0$, $e_{ij} = x_{r,i} - x_{n,i}$ indicates that the model does NOT denoise and reconstruct the same value to the noisy input. Finally, in equation (10), the square distances $(e_{ij}^{(p)})^2 + (e_{ji}^{(p)})^2$ are summed up with respect to all combinations of two variables. Consequently, the resulting model can calculate just the reconstruction errors $\mathbf{e} = \mathbf{x} - \mathbf{x}_r$ which exhibit differences with respect to the important relations.

The geometrical interpretation of SD is helpful for understanding our proposed method. We thus introduce PCA denoising and compare it with the proposed method. Figure 3(A) shows PCA denoising. A subspace formed by the principal components learned from normal data is used, and a sample is shown on the subspace. The noise orthogonal to the subspace is removed (denoised) by the projection, and the result has the same value as the training sample. In contrast, the noise is preserved if it was added in parallel to the subspace. Thus, the PCA subspace determines the denoising direction. The relation between denoising and the learned subspace suggests that $\boldsymbol{\alpha}$ constrains the shape of the manifold learned from StrDA by controlling the denoising structure. In Figure 3(B), we see that the sample is correctly denoised with the i, jth variables because $\alpha_{ij} = 1$, but it is not denoised with the

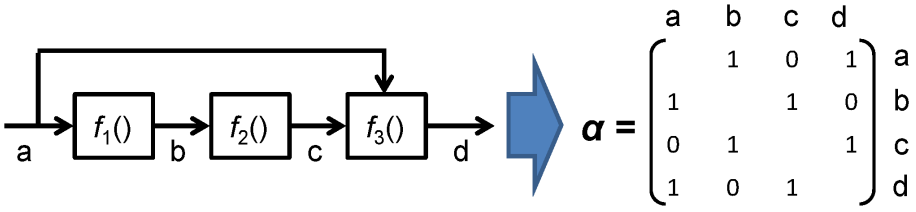

Figure 2: Example of setting prior knowledge by using the matrix $\boldsymbol{\alpha}$. Given the input-output relation of four variables $\{a, b, c, d\}$, we set the relation as important ($\alpha_{ij} = 1$) if two variables are directly connected. Otherwise, we set it as unimportant ($\alpha_{ij} = 0$). This encourages the model to monitor only the direct input-output relations. Actually, the faults should be originated from the corruption of direct physical relations and it is better to ignore relations within $\{a, c\}$ and $\{b, d\}$ where no direct connections exist.

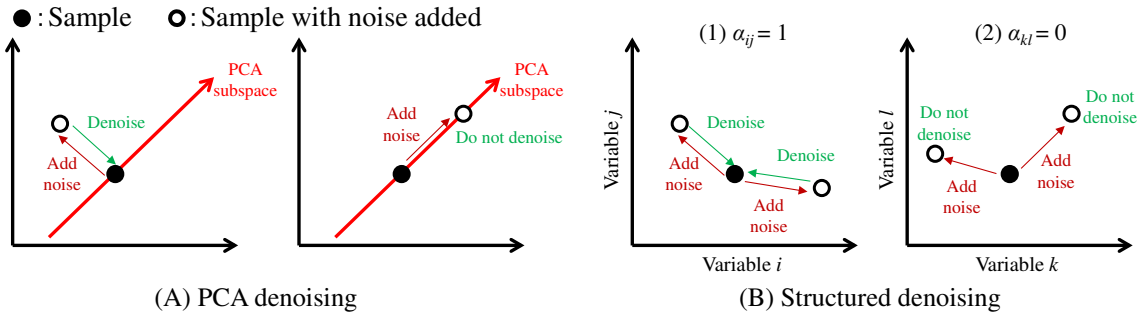

Figure 3: Comparison between PCA denoising and structured denoising.

k, lth variables because $\alpha_{kl} = 0$. These denoising criteria implicitly control the shape of the resulting model.

Our approach based on the denoising autoencoder is different from Bayesian methods in that our prior implicitly applies a prior distribution to the reconstruction errors, whereas Bayesian approaches apply a prior distribution to the model parameters. However, the power of regularization in the denoising is equal between variables because the objective function adopts l_2 norm of the errors that implies the isotropic Gaussian distribution. SD can control the power of regularization and build a complex structured distribution to the errors. Our model only requires setting 0 or 1 to obtain a structured prior distribution, whereas Bayesian approaches often require expert knowledge. This is the advantage of our model.

3.3. Fault Detection and Analysis using StrDA

The StrDA uses the same fault detection and analysis processes as are used in the DA. we compute $\mathbf{x}_r$ by equation (6) and (7) with the input $\mathbf{x}$ instead of $\mathbf{x}_n$. Then the reconstruction error $\mathbf{e} = \mathbf{x} - \mathbf{x}_r$ is used to conduct CA and calculate the RMS (4) as an anomaly score.

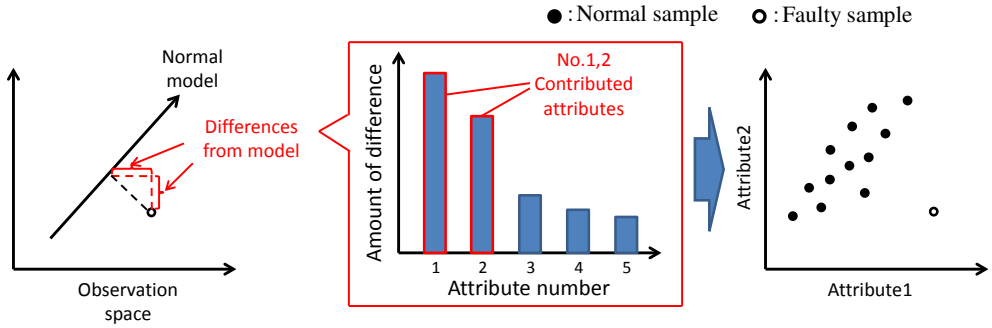

Figure 4: Fault representation using a scatter plot.

The advantage of the StrDA over the DA is that the StrDA reflects prior knowledge in its reconstruction errors, but the DA does not.

Fault representation is an important factor in fault analysis. The aim of our proposal is to develop a method with high usability in a real situation. Even operators without specific technical knowledge want to know what kinds of normal relations are corrupted in a test sample. Thus, we should ensure that our results can be easily understood. One option is to employ a scatter plot; we conduct CA on a given test sample, and the variables are then ranked in the order of their contributions. We would then use the first and the second ranked variables to plot the test sample and the normal data samples that were used to learn the StrDA model (Figure 4). This plot shows how the test sample deviates from normal data samples. This representation offers an intuitive understanding of how the test sample is different from the normal relations with respect to the two selected variables.

It is often difficult to observe faults by analyzing individual samples. Changes occur slowly due to system dynamics, the environment, individual drivers, and deterioration effects. We thus introduce another metric, the average contribution, which is calculated over N' samples, as follows:

$$C_{ij} = \frac{1}{N'} \sum_{p=1}^{N'} \sqrt{\left(e_i^{(p)}\right)^2 + \left(e_j^{(p)}\right)^2} \tag{12}$$

where $e_i^{(p)}$ denotes the reconstruction error of ith variable with pth sample. C_{ij} computes the average l_2 norm with respect to i, jth variable. We apply the same CA approach to C_{ij}, such that combinations of two variables which have a larger C_{ij} will have higher contributions to the faults that occur entirely within the test samples.

4. Experiments

This section shows the results of experiments using our proposed method. First, we produced synthetic data to evaluate the effect of using prior information. The evaluation revealed that even only approximate prior information can improve the performance of fault detection. Second, we used real driving data to evaluate our method in a realistic and changing environment. The results show that our proposed method outperforms the related approaches, both in fault detection and in analysis. Note that our method is gen-

eral and applicable to any kind of *one class classification* settings where prior information is available.

For the computational environment, we used Windows 7 (64-bit) with an Intel$^{(R)}$ Core$^{(TM)}$ i7-3970X CPU @ 3.50 [GHz] and a 64.0 [GB] memory. All implementations were performed by MATLAB R2012b.

4.1. Evaluation with Synthetic Data

The proposed StrDA was compared with the DA using synthetic data, in order to evaluate the effect of utilizing prior information $\boldsymbol{\alpha}$. For both models, we set 200 as the number of hidden variables, and for the input, we added $\sigma = 0.5$ isotropic Gaussian noise $\mathcal{N}(0, \sigma I)$ to the training dataset. The optimization was conducted by using the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm.

The dataset was obtained in the following manner. We considered three multivariate Gaussian distributions. Each distribution has zero mean and a $d_i \times d_i$ covariance matrix Σ_i given by $\Sigma_i = X_i^T X_i$, where X_i is a $d_i \times d_i$ semidefinite matrix in which each element was randomly sampled by a $[0, 1]$ uniform distribution. Σ_i was then adjusted to unit variance. In our experiments, $d_1 = 10$, $d_2 = 6$, and $d_3 = 12$. After sampling 1000 points from each Gaussian distribution, we combined the points to make a 28-dimensional training dataset ($d = d_1 + d_2 + d_3 = 28$). This was used as the normal dataset. Another 1000 samples were calculated from the distribution with a new covariance matrix Σ_3, and this was used as the dataset with pseudo faults. In this situation, the true difference between these two datasets was the change in the covariance matrix Σ_3.

To evaluate the effect of using prior information, several forms of $\boldsymbol{\alpha}$ were considered. Figure 5 shows the true dependency between each set of variables; black elements indicate they are related, and otherwise, they are not. From lower left to upper right, the three black segments denote the dependency of the respective Gaussian distributions with Σ_1, Σ_2, and Σ_3. We also set six squares $S = \{S_1, S_2, S_3, S_4, S_5, S_6\}$ as

$$\begin{aligned} S_1 &= \{(i,j)\,|i,j \in (1,\ldots,10)\}, \quad S_2 = \{(i,j)\,|i,j \in (11,\ldots,16)\}, \\ S_3 &= \{(i,j)\,|i,j \in (17,\ldots,28)\}, \quad S_4 = \{(i,j)\,|i,j \in (11,\ldots,28)\}, \\ S_5 &= \{(i,j)\,|i,j \in (11,\ldots,22)\}, \quad S_6 = \{(i,j)\,|i,j \in (1,\ldots,28)\}. \end{aligned} \tag{13}$$

These squares are indicated by a dashed line in Figure 5. Note that S_3 is the change to be detected. We selected one or more of these segments as the matrix $\boldsymbol{\alpha}$, e.g., $S_\alpha = S_1$, and set these combinations as important, i.e., $\{\alpha_{ij} = 1 |\, (i,j) \in S_\alpha\}$. We used the area under the curve (AUC) of the receiver operating characteristics (ROC) to compare the accuracy of fault detection for several different values of $\boldsymbol{\alpha}$. We randomly sampled the training and the test data ten times each, and we then computed the average of the AUC.

The results are shown in Table 1, where $|S_\alpha \cap S_3|/|S_\alpha|$ denotes the ratio of the true segment $S_\alpha \cap S_3$ to the selected segment S_α. The relation between $|S_\alpha \cap S_3|/|S_\alpha|$ and the AUC is helpful for understanding the effect of different values of $\boldsymbol{\alpha}$. The StrDA outperformed the DA except when $S_\alpha = S_1$ and S_2, and the best performance was obtained when $S_\alpha = S_3$, i.e., when the prior information $\boldsymbol{\alpha}$ was set to the true cause of the changes. The setting of $S_\alpha = S_1 \cup S_2 \cup S_3$ also improved the AUC, which means that information about the normal structure of the data contributes to the fault detection. Surprisingly, such an improvement

was found even when S_α contained an incorrect segment ($S_\alpha = S_4$) or only part of the true segment ($S_\alpha = S_5$). Finally, note that when all combinations were set to be important ($S_\alpha = S_6$), the StrDA learned the data uniformly like an unsupervised approach, so that the AUC is almost the same as it was with the DA. Overall, the StrDA improves the AUC almost monotonically as the ratio $|S_\alpha \cap S_3|/|S_\alpha|$ increases.

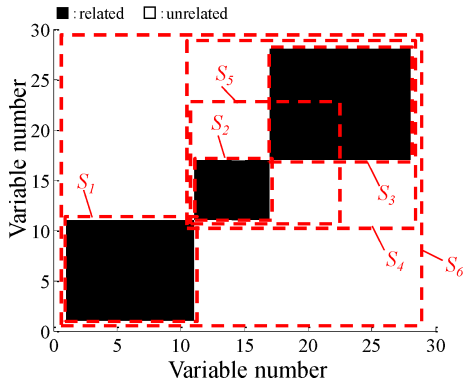

Figure 5: Relations between variables.

Table 1: Average AUC (synthetic data).

	StrDA							DA
S_α	S_1	S_2	S_3	$S_1 \cup S_2 \cup S_3$	S_4	S_5	S_6	
AUC	0.5108	0.4598	0.8825	0.8369	0.8400	0.7935	0.7848	0.7805
$\frac{\lvert S_\alpha \cap S_3 \rvert}{\lvert S_\alpha \rvert}$	0.0000	0.0000	1.0000	0.5143	0.4444	0.2500	0.1837	-

4.2. Evaluation with Real Driving Data

4.2.1. Evaluation Settings

We used driving data to evaluate the performance of both fault detection and analysis. The problem considered here is to detect and analyze the changes in the driving environment. Fault diagnosis equipment recorded information for the vehicle, and the data contained 43 attributes. The data were recorded at intervals of 0.5–1.0 [s]. The attributes are listed in Table 2. We considered four driving conditions, as shown in Table 3. The Drive and the Neutral are the transmission modes of the vehicle. In the Flat Road condition, the vehicle ran on the same flat road with constant acceleration and deceleration. The vehicle goes down slopes with the same acceleration and deceleration under the Downslope condition. In the DtoN data, the transmission mode was changed to the Neutral when the vehicle reached a specific speed with the Drive mode. In the Slow data, the vehicle speed was slowly increased and decreased by the driver. The Normal data were considered the normal data, and the data in the other three conditions were used as the faulty data to be detected. We eliminated any discrete attributes which take only one or two values, and we then adjusted the sampling rate to intervals of 0.5 [s] by using linear interpolation.

The proposed method was compared with several conventional approaches: the mixture of probabilistic principal component analysis (MPPCA, Chandola et al. (2009)), the one-class support vector machine (OCSVM, Schölkopf et al. (2001)), the local outlier factor (LOF, Breunig et al. (2000)), and the DA; see Table 4. The parameter settings for the StrDA and DA were the same as those used in Section 4.1, except for $\boldsymbol{\alpha}$. To provide a fair comparison, we used faulty datasets to tune the parameters of the MPPCA, OCSVM, and LOF. In this case, the AUC of these conventional methods was maximized with respect to the parameter sets shown in Table 4. For the MPPCA, we set the number of probabilistic PCA to c and their dimension to d. The OCSVM was based on a νSVM with a Gaussian kernel, given as

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp\left(-\frac{||\mathbf{x}^{(i)} - \mathbf{x}^{(j)}||^2}{\gamma^2}\right), \tag{14}$$

where ν and γ are parameters. In the LOF, the parameter k determines the number of nearest neighbors.

The value of $\boldsymbol{\alpha}$ was determined by the experimental conditions listed in Table 3. For the Slow data, it was expected that the driver changed the acceleration and deceleration patterns as well as the vehicle speed; we therefore set as important the relations between attributes

Table 2: List of attributes.

No.	Attributes	No.	Attributes
1	Shift Position	24	Objective Air/Fuel Ratio
2	Parking Brake	25	Air/Fuel Ratio
3	Sports Mode Switch	26	Purge Rate
4	Engine Stop Request	27	O_2 Sensor Voltage
5	Idle Control	28	Ignition Timing
6	P Range Racing	29	Objective Exhaust Gas-
7	Warming Request		Recirculation Valve Position
8	Electric W/P Motor Rotation	30	Stroke Sensor 1
9	Vehicular Speed Sensor 1	31	Stroke Sensor 2
10	Vehicular Speed Sensor 2	32	Accumulator Pressure
11	Accelerator Position	33	Front Rear G Sensor
12	Intake Air Volume	34	Regenerative Cooperation Brake
13	Required Throttle Position	35	Executed Regenerative Torque
14	Throttle Position (Sensor Value)	36	Required Regenerative Torque
15	Throttle Position (Directed Voltage)	37	Yaw Rate Sensor 1
16	Throttle Position	38	Yaw Rate Sensor 2
17	Engine Speed	39	Steering Angle Sensor
18	Required Engine Output	40	Lateral G Sensor
19	Objective Engine Speed	41	Yaw Rate Value
20	Real Engine Torque	42	Steering Angle Value
21	Idle Speed Control Flow	43	Zero-Point Corrected-
22	Idle Speed Control Position		Steering Angle Sensor
23	Idle Speed Control Flow (Learned Value)		

Table 3: Driving datasets.

Name	Condition	Mode	# of Samples
Normal	Flat Road	Drive	1450
Slow	Flat Road	Drive	737
DtoN	Flat Road	Neutral	418
Down	Downslope	Drive	92

Table 4: Conventional approaches and the parameter settings.

Approaches	Parameters
StrDA	$d = 200$, $\boldsymbol{\alpha}$ shown in Figure 6
DA	$d = 200$
MPPCA	$c = \{2, 3, 4\}, d = \{2, \ldots, 9\}$
OCSVM	$\nu = \{0.001, 0.005, 0.01, 0.05, \ldots, 1\}$, $\gamma = \{0.001, 0.005, 0.01, 0.05, \ldots, 1000\}$
LOF	$k = \{2, \ldots, 9\}$

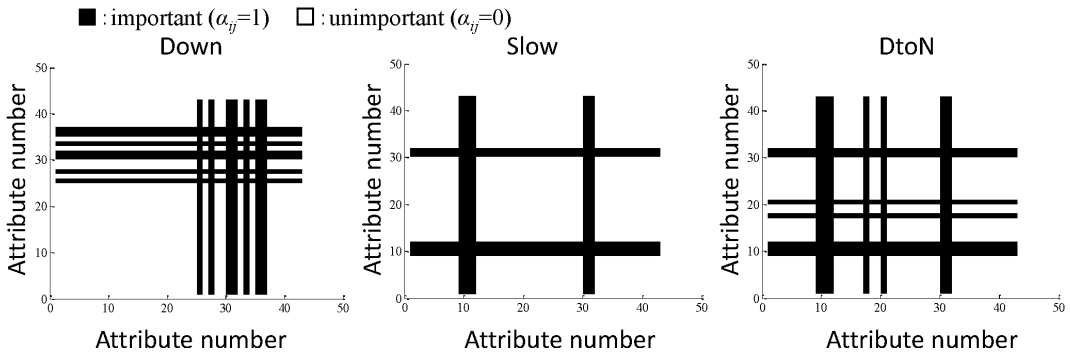

Figure 6: Representation of the prior information $\boldsymbol{\alpha}$.

Nos. 9, 10, 11, 30, and 31, and all the other attributes; i.e., $\alpha_{ij} = 1$ for $i = 9, 10, 11, 30,$ and 31. Note that the Stroke Sensor represents the braking input. For the DtoN data, we set as important the relations between Nos. 9, 10, 11, 17, 20, 30, and 31, and all the other attributes; i.e., $\alpha_{ij} = 1$ for $i = 9, 10, 11, 17, 20, 30,$ and 31; we did this because the responses related to the power trains, i.e., the driver inputs, the engine rotation, the vehicular speed, and the regenerative control, were expected to change. Finally, for the Down data, the influence of the steep slope was expected to cause changes in the air/fuel ratio, braking, and regeneration control. We thus set as important the relations between Nos. 25, 27, 30, 31, 33, 35, and 36, and all the other attributes; i.e., $\alpha_{ij} = 1$ for $i = 25, 27, 30, 31, 33, 35,$ and 36. Finally, we set $\alpha_{ij} = \alpha_{ji}$ for each dataset because $\boldsymbol{\alpha}$ should be symmetric. Figure 6 shows the various representations of $\boldsymbol{\alpha}$, where the black segments denote $\alpha_{ij} = 1$.

4.2.2. Fault Detection Performance

We compared the performance for fault detection by using the AUC. The Normal data were divided into two parts: 1000 samples were used for training the model, and the rest were

Table 5: Average AUC (driving data).

	StrDA	DA	MPPCA	OCSVM	LOF
Slow	**0.8393**	0.6887	0.7575	0.8193	0.8213
DtoN	**0.8905**	0.8289	0.8412	0.8527	0.8562
Down	**0.9331**	0.8632	0.9134	0.9268	0.8715

combined with each of the faulty datasets to form the test datasets. Note that the data were randomly assigned to each part. The parameters for the MPPCA, OCSVM, and LOF were tuned maximizing the AUC for the test data. The AUC was averaged over ten trials.

The results are shown in Table 5. The StrDA outperformed the other approaches with each of the test datasets. These results are interesting because the StrDA used only the training data with some incomplete prior information, while the other methods used the true faulty data samples directly in order to obtain the best results. We note that structural information is richer than label information. In practical situations, we rarely know *a priori* that we have faulty samples, but we often know some (probably incomplete) information about the structure of the normal data and/or the faulty behavior. Thus, the proposed StrDA will be effective in practical use.

4.2.3. Fault Analysis Performance

The performance of the fault analyses were evaluated by using the Slow data, in which the causes of the change is the driver input, i.e., the Accelerator Position and the Stroke Sensor. We evaluated the ability of each method to 1) extract the true causes, and 2) represent the changes of relationships between attributes which include the true cause. 2) is the key to understand why they are the true causes. The evaluation was based on CA. Note that we excluded the OCSVM here, because it is difficult to conduct CA on it. For the LOF, we used the kth nearest neighbor as the reconstruction data. We applied CA to all samples with the Slow data, using Equation (12). The results are shown in Figures 7–10. Each result shows the best combination of eight attributes, and these are plotted for both the training data (Normal) and the test data (Slow).

Figure 7 shows the results for the proposed StrDA. In the first three contributed combinations, we found the apparent differences between the datasets even though the true cause was not extracted. The other combinations extracted the driver inputs, i.e., the Stroke Sensor, which represents braking and the Accelerator Position. These attributions were the true causes of the changes. The scatter plots of these combinations clearly shows the changes in the relations between the training and the test datasets. Based on these figures, operators will be able to understand when there is a change of the relationship between attributes. In this sense, the proposed StrDA contributes to the representation of such changes. Among the conventional methods, the conventional DA, whose result is shown in Figure 8, extracted the changes in the Purge Rate, which was not the true cause. The combinations did not show the strong relations, and thus the DA failed to identify the changes. In this sense, the proposed StrDA, which is the DA with added prior information, improves not only the detection of faults but also the analysis of the faults. With the MPPCA, as shown in Figure 9, the changes were found in the No. 3 and No. 7 contributions, neither

of which is a true cause; the results show a secondary effect, i.e., the true causes, the acceleration and deceleration, changed the behavior of the front/rear G and the vehicle speed, which is shown in Figure 9. These kinds of effects often make fault analysis more difficult because the secondary and/or tertiary effects can result in larger changes than those caused by the true (primary) causes; data-driven unsupervised approaches often extract these kinds of effects. The proposed StrDA can avoid this problem by using the prior information. The LOF was unable to extract meaningful changes other than the Purge Rate, as shown in Figure 10.

4.3. Evaluation of Dimensionality

The StrDA controls denoising by using the prior information $\boldsymbol{\alpha}$ to determine the dimensionality of the resulting model. This section shows empirically that the dimensionality of the resulting model decreases as the number of important relations ($\alpha_{ij} = 1$) decreases.

The Normal data in Table 3 were used for this evaluation. In this setting, the StrDA learned models with eleven kinds of $\boldsymbol{\alpha}$, where $\{0, 10, \ldots, 100\}$ [%], respectively, were randomly set to important ($\alpha_{ij} = 1$). The dimensionality is measured by the saturation ratio, which is the average fraction of saturated hidden units $\mathbf{h}_1$ per example (Rifai et al., 2011). A smaller saturation ratio suggests a larger dimensionality. We regarded values below 0.05 or above 0.95 as being saturated. Figure 11 shows the relation between the ratio of important factors ($\alpha_{ij} = 1$) and the saturation ratio. A decrease in the ratio of important relations caused a reduction in the saturation ratio; this is equivalent to an increase in the dimensionality. A lower ratio of important factors requires the StrDA to denoise in fewer directions, resulting in a model with less reduction of its dimensionality (details are in Section 3.2). Finally, it almost converges to the identity mapping if there are no important relations.

5. Conclusion

This paper proposed a new approach, StrDA. The StrDA is an extension of the conventional DA, and it utilizes prior information about the data structure. The objective function is modified to include this information. Experiments using synthetic data suggest that using the prior information can improve the detection of faults. We considered the problem of detecting changes, and an evaluation with real driving data showed that the StrDA outperformed the conventional approaches that had been tuned with faulty data. In the fault analysis, the StrDA extracted and represented the true causes of the changes, which the conventional approaches were unable to do. Thus, the proposed StrDA is effective for detecting and analyzing faults. Such advantages are valid even when the prior information is incomplete. We believe that our proposed method and presentation in a scatter plot will simplify the detection and attribution of faults.

References

Yoshua Bengio. Learning deep architectures for AI. *Foundations and trends in Machine Learning*, 2(1):1–127, 2009.

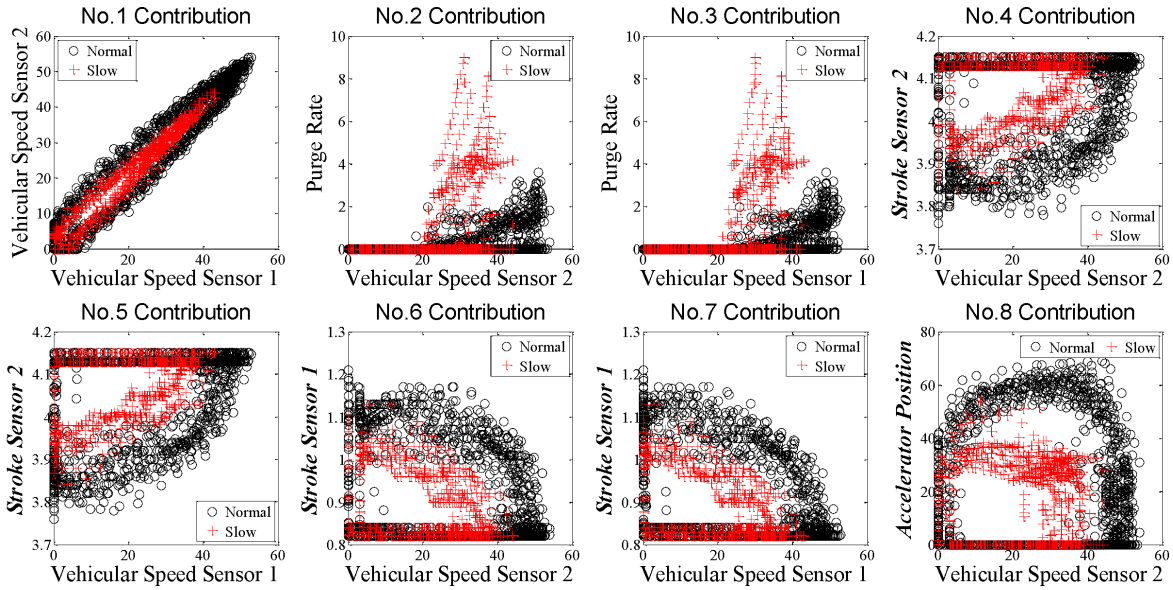

Figure 7: Top eight contributed combinations determined by the StrDA, which successfully extracted the true causes, i.e., Stroke Sensor and Accelerator Position, and the changes in their relationships with the Vehicular Speed Sensor.

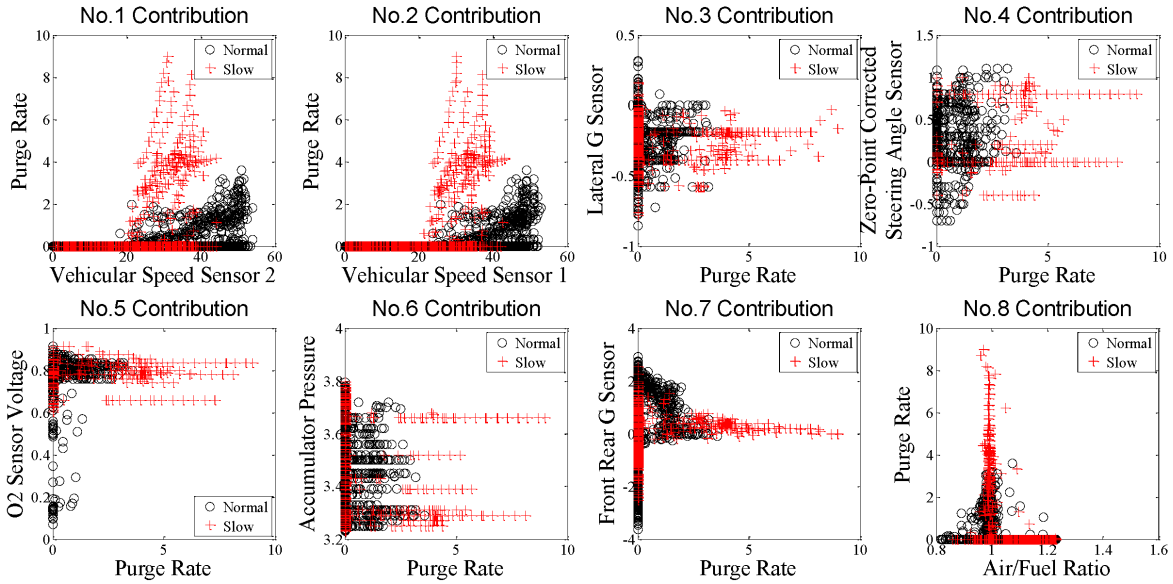

Figure 8: Top eight contributed combinations determined by the DA, which extracted only the differences in the Purge Rate and failed to find true causes.

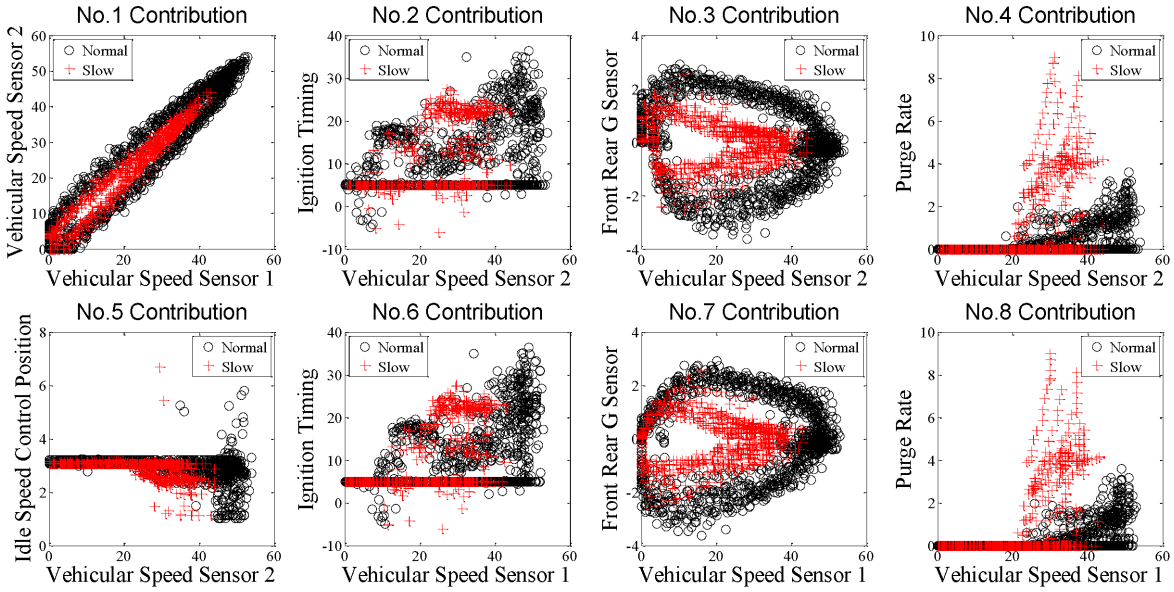

Figure 9: Top eight contributed combinations determined by the MPPCA, which extracted the effects of the true causes, i.e., Front Rear G Sensor and Vehicular Speed Sensor. However, this is not practical information and could obscure the true causes.

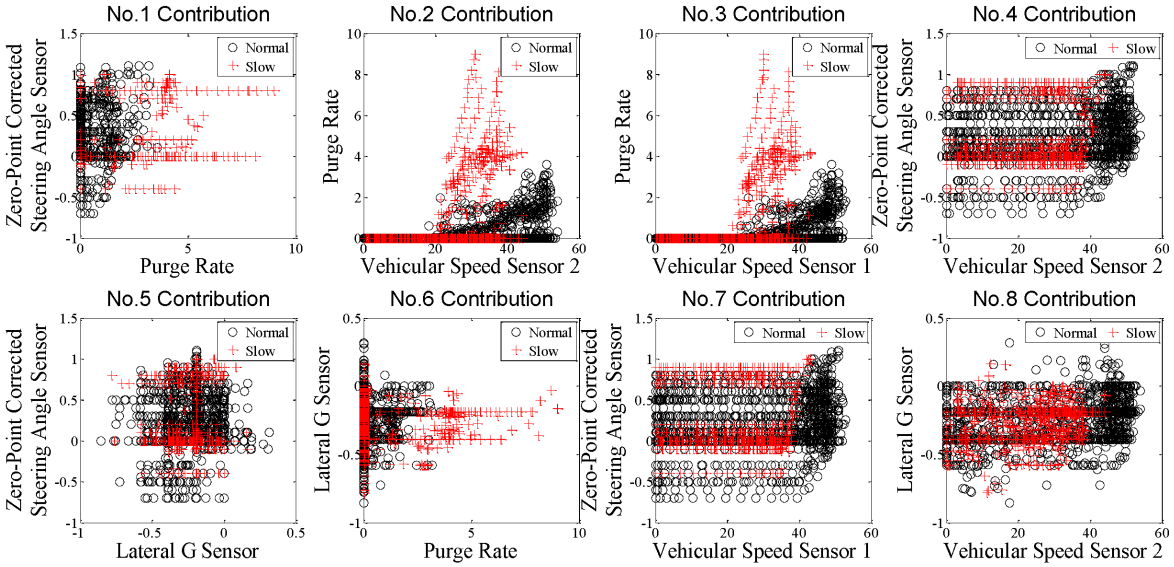

Figure 10: Top eight contributed combinations determined by the LOF. Only the differences in Purge Rate were extracted, and it failed to find the true causes.

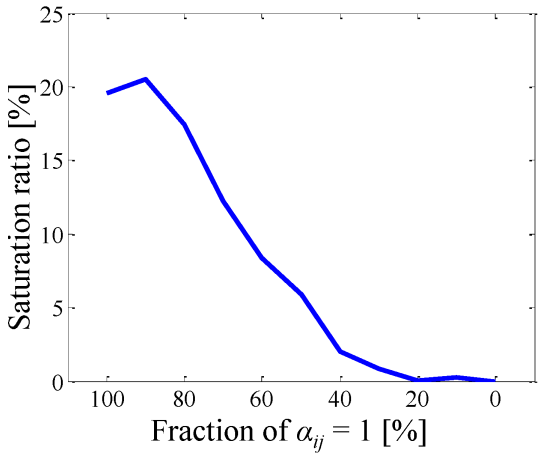

Figure 11: Relation between saturation ratio [%] and $\boldsymbol{\alpha}$.

Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. LOF: Identifying density-based local outliers. In *ACM Sigmod Record*, volume 29, pages 93–104. ACM, 2000.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3):15, 2009.

Steven X. Ding. *Model-based fault diagnosis techniques*. Springer, 2008.

Zhiqiang Ge and Zhihuan Song. *Multivariate Statistical Process Control: Process Monitoring Methods and Applications*. Springer, 2013.

Chirag Modi, Dhiren Patel, Bhavesh Borisaniya, Hiren Patel, Avi Patel, and Muttukrishnan Rajarajan. A survey of intrusion detection techniques in cloud. *Journal of Network and Computer Applications*, 36(1):42–57, 2013.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on Machine Learning*, pages 833–840, 2011.

Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408, 2010.