Research Article

A Label Noise Robust Stacked Auto-Encoder Algorithm for Inaccurate Supervised Classification Problems

Zi-yang Wang , Xiao-yi Luo, and Jun Liang

College of Control Science and Engineering, Zhejiang University, Zheda Road 38, Hangzhou 310027, China

Correspondence should be addressed to Jun Liang; jliang@zju.edu.cn

Received 21 October 2018; Revised 12 March 2019; Accepted 2 April 2019; Published 14 May 2019

Academic Editor: Ricardo Aguilar-Lopez

Copyright © 2019 Zi-yang Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In real applications, label noise and feature noise are two main noise sources. Similar to feature noise, label noise imposes great detriment on training classification models. Motivated by successful application of deep learning method in normal classification problems, this paper proposes a new framework called LNC-SDAE to handle those datasets corrupted with label noise, or so-called inaccurate supervision problems. The LNC-SDAE framework contains a preliminary label noise cleansing part and a stacked denoising auto-encoder. In preliminary label noise cleansing part, the K-fold cross-validation thought is applied for detecting and relabeling those mislabeled samples. After being preprocessed by label noise cleansing part, the cleansed training dataset is then input into the stacked denoising auto-encoder to learn robust representation for classification. A corrupted UCI standard dataset and a corrupted real industrial dataset are used for test, both of which contain a certain proportion of label noise (the ratio changes from 0% to 30%). The experiment results prove the effectiveness of LNC-SDAE, the representation learnt by which is shown robust.

1. Introduction

In real applications, almost all supervised learning suffers from two types of noise, noise among feature variables (process variables) and noise in label variables. Taking industrial processes for example, the noise among feature variables mainly resulted from systemic error in sensor measurement or external disturbances, while noise among label variables is generated due to manual mislabeling. The manual labeling heavily depends on human subjectivity and empirical knowledge. Compared to samples with feature noise, those mislabeled samples impose even more negative effects on trained classifiers [1].

As a research hotspot, many denoising deep learning methods [2] are designed to handle feature noise. For example, denoising auto-encoder (DAE) [3] applies the so-called dropout into the training process, which could randomly map the input variables into zero. By applying dropout module, the feature representations extracted by DAE are proved to be more robust and help raise classification accuracy in recognition, speech recognition, and other fields. Different from DAE, contractive auto-encoder (CAE) [4] obtains robust hidden representations by adding the Jacobean term

of hidden representations into the loss function. Both DAE and CAE perform well in dealing with datasets with feature noise.

It is noted that all these above methods are carried out based on the premise that the labels of the training datasets are totally correct. However, in real applications, part of the samples in dataset are often mislabeled because of manual mistake, especially those samples collected during mode transition procedures. According to reference [5], the label noise is more detrimental than feature noises to the generalization performance of learned classifiers. Most actual datasets include a small part of mislabeled samples. Those severely corrupted dataset is not discussed here. This paper places the emphasis on these datasets containing a minority of mislabeled samples (less than 30%).

In confronting of the dataset containing mislabeled samples, neither DAE nor CAE could provide satisfying results, especially when the noisy sample ratio cannot be neglected. Some traditional classification algorithms, such as SVM [6] and logistic regression, as well as some common used ensemble learning algorithms such as bagging [7] and adaBoost [8] method also partially rely on the correctness of labels. Taking adaBoost algorithm, for example, label

noisy samples are easily to be assigned with improperly large weights [9]. Compared with adaBoost, bagging is relatively insensitive to mislabeled noisy samples; its performance is more robust against noisy samples [10].

Researchers have put forward several effective frameworks to solve label noise problems.

The first type is to add a label noise filter module beforehand to detect those most probable noisy samples, for example, the nearest neighbor criterion [11, 12] and cumulative information criterion [13]. Those data points identified as mislabeled samples or so-called outliers will be removed before feeding to train the latter classifier model. This kind of framework may bring about two problems, one is that data points with clean labels may be removed or falsely corrected; the removal will decrease the size of training datasets and may lead to overfitting problem if the number of residual samples in training dataset is insufficient for model training.

The second type is to add a module to estimate each sample's probability of belonging to specific class, and then samples will be assigned with different weights to distinguish their influence upon model training. The key point is how to determine the confidence of each point and find out suspicious mislabeled ones to assign smaller weights. For example, ensemble learning methods like boosting [14] and adaboost [15] are combined with decision tree classifier to detect the incorrect labels and assign smaller weights upon them. References [16, 17] adopt a probabilistic model of kernel Fisher Discriminate, where the probability of the incorrectly labeled data point will be updated. Other methods [18–20] also adopt a probabilistic framework, treating each sample's label as a latent variable. The current label is taken as the prior to estimate the posterior label possibility, based on which weights of samples are updated.

The third type is to directly improve the classifier itself, making it more robust against label noise. The common strategies include adding penalty term into loss functions or introducing some training tips. Paper [21] combines truncated hinge loss into SVM objective function for better tolerating the label noise. Paper [22] proposes a strategy to make use of a kernel matrix correction to improve the robustness of SVMs.

Currently, deep learning is a new research spot in machine learning area, which proves powerful in fault classification area [23–25]. If given training dataset with accurate labels, deep learning methods are proved to achieve better classification performance than other supervised learning methods, such as SVM, decision tree, and random forest (RF) [26] algorithm. Stacked AE also shows better performance than single AE. It is its deep architecture that captures more autocorrelation features among samples in more complex dynamic, large-scale, multiscale industrial processes.

In ideal supervised classification cases (without label noise), deep learning [27] has achieved promising performance. However, to the best of our knowledge, quite few deep learning related works are proposed in inaccurate supervision area. Besides, paper [28] proves that an effective preprocessing of samples with corrupted labels will effectively improve the performance of traditional supervised algorithms in inaccurate supervision problems. Inspired by these two aspects,

this paper puts forward a framework combining a label noise cleansing part and a deep learning algorithm to solve inaccurate supervision problems. With help of the label noise cleansing part, the ratio of mislabeled samples is reduced as much as possible, then the cleansed training dataset will be provided to a stacked denoising autoencoder (SDAE) [29] for extracting robust representations for classification or fault classification.

The proposed LNC-SDAE in this paper applies the second and third type of thought to solve label noise problems. The main contributions of this paper are listed as follows: (1) first introduce K-fold cross-validation thought into the label noise cleansing (LNC) algorithm, (2) propose a practical strategy to determine the threshold parameter for deciding label revision, (3) put forward an improved Stacked DAE named LNC-SDAE and prove the Stacked DAE algorithm's robustness against label noise to some extent. LNC-SDAE is expected to achieve a satisfying performance in handling training dataset with a small portion of mislabeled samples.

This paper is organized as follows. In Section 2, brief introductions of autoencoder (AE), DAE, CAE, and their stacked forms are given. In Section 3, a new LNC algorithm named KCV LNC is proposed, a label noise robust deep learning method named LNC-SDAE is also proposed for handling inaccurate supervision problems. In Section 4, experiments on a label noisy UCI dataset and a label noisy real industrial process dataset are used for verifying the effectiveness of the proposed LNC-SDAE method in classification and fault classification. In Section 5, some conclusions are drawn.

2. Background

2.1. Autoencoder. The basic autoencoder is a typical unsupervised learning model including an encoder and a decoder part. As shown in Figure 1, the encoder transforms the input x into a feature representation h through a nonlinear function $f(x)$. The decoder transforms the feature representation h back into a reconstructed input $\hat{x}$ through a nonlinear function $g(x)$. The optimal representation is achieved by minimizing the reconstruction loss $J_{AE}(\theta)$, which is shown in (1).

$$J_{AE}(\theta) = \sum_{n=1}^N G(x(n), \hat{x}(n)) = \sum_{n=1}^N \|x(n) - g(f(x(n)))\|^2. \quad (1)$$

where N is the sample number of training dataset and $x(n)$ and $\hat{x}(n)$ are the n^{th} input and reconstructed input.

2.2. Denoising Autoencoder and Contractive Autoencoder. Most extensions of AE are proposed to enhance traditional AE's robustness upon feature noise. Among which, DAE and CAE are two representative methods, both of which outperform traditional AEs when trained with datasets with feature noise.

The main thought of DAE is void repeating extraction of the same feature. DAE increases the redundancy of

FIGURE 1: The structure of autoencoder.

inputs, making the extracted intermediate representation more robust to feature noise of the input. There are two common ways of increasing the feature redundancy [30]. One is to add a Gaussian white noise into the input data, the other is to carry out a stochastic mapping called dropout, both of which are carried out during the training process. When the size of training dataset is small, the trained model may be overfitting and show poor performance on the test data. Both dropout and directly adding noise could partially overcome the overfitting problem; the only difference between them is that the dropout will be turned off during testing phase. It has to be noted that there are two main kinds of dropout strategies. One is directly setting the value of some neurons into zero; the other is randomly cutting the connection between the inputs and activation function or so-called dropout connection. The paper adopts the former one, a keep probability parameter which controls the proportion of neurons setting into zero.

Different from DAE, CAE strengthens the robustness of hidden representations by adding the Jacobian term of hidden representations $\|f(x)\|_F^2$ into the loss function, which is shown in (2).

$$J_{CAE}(\theta) = \sum_{n=1}^N G(x(n), \hat{x}(n)) + \lambda \|f(x(n))\|_F^2 \quad (2)$$

$$= \sum_{n=1}^N (\|x(n) - g(f(x(n)))\|^2 + \lambda \|f(x(n))\|_F^2).$$

where λ is the penalty parameter of Jacobian penalty term.

Similar to DAE, CAE's main improvement is increasing robustness upon perturbations in process variables. The regulation term is the Jacobian norm of hidden representations. It will force the first derivatives of hidden representations to approximate zero or one, making hidden representations more contractive.

Both DAE and CAE's robustness upon feature noise is on the basis of the premise that the samples from training dataset must be correctly labeled. If the training dataset contains many label noisy samples, neither DAE nor CAE will show satisfying result. The degree of negative influence is related to the proportion of mislabeled samples in the training dataset. In the following case study section, the experiment results

will show the defect of DAE in handling training dataset containing more than 10% mislabeled samples.

2.3. Stacked Autoencoder and Stacked Denoising Autoencoder.

The basic framework of SAE and SDAE are quite similar, they are the stacking form of AE or DAE with only one hidden layer. The SAE and SDAE are proved to show more promising performance than single AE and DAE with multiple hidden layers in terms of extracting feature representation and approximating multivariable nonlinear and complex functions [31]. The major difference between SAE and SDAE is that DAE containing a dropout module. A typical SAE consisting of three AEs is shown in Figure 2. In the training process of SAE and SDAE [32], the hidden representation of the i^{th} AE is used as the input for the $(i+1)^{th}$ AE. The representation of the last AE is the output of the SAE, which is input into softmax part for predicting the final label variable. For multiclassification and fault classification problems, the final output is the predicted label, which is usually transformed into a one-hot form. Thus, the number of last softmax layer's output is the total number of classes. Softmax regression is an extension of logistic regression algorithm when dealing with multiclassification problem. The detail description of softmax algorithm for the K class classification problem is as follows.

Here, each input ξ_i for softmax classifier is the last hidden layer generated by SAE or SDAE, the predicted label of i^{th} input sample is represented in a dummy coding style $\mathbf{t}_i = [t_{i,1}, t_{i,2}, \dots, t_{i,K}]$. When predicted $\hat{y} = q$, then $t_{i,q} = 1$; others are all equal to 0. The probability that the i^{th} sample belongs to the q^{th} class is shown in the following. β, b are weight and bias parameters.

$$P(y_i = q | \xi_i) = g(\xi_i, \beta_q, b_q) = \frac{e^{\beta_q^T \xi_i + b_q}}{\sum_{q=1}^K e^{\beta_q^T \xi_i + b_q}} \quad (3)$$

The prediction label $\hat{y}_i = \arg \max_q g(\xi_i, \beta_q, b_q)$. The model parameter is optimized by maximizing the log likelihood in (4).

$$L(\beta, b) = \sum_{i=1}^L \sum_{q=1}^K t_{i,q} \log P(y_i = q | \xi_i) \quad (4)$$

FIGURE 2: The structure of stacked autoencoder.

2.4. Disjoint and Joint-Train. In this paper, our target is to solve inaccurate supervision problems. All training samples are all labeled ones, but some of which may be mislabeled. Therefore, the paper only discusses the supervised form of SAE and SDAE, semisupervised forms are not our focus. There exists two different ways to train supervised SAE or SDAE. One is the disjoint optimization strategy [27], the other is the joint optimization strategy [33]. The disjoint strategy is a typical optimization method in semisupervised learning. It can be separated into two parts, the unsupervised and the supervised part. In these two parts, the structure of SAE and SDAE are kept unchanged. The unsupervised training process is regarded as a pretraining procedure, where the loss function only contains the average reconstruction error of AEs in each layer. Stochastic gradient descent and Adam algorithm could be used to iteratively minimize the loss function so as to get the optimal immediate parameters, such as weight parameters in each AE. The optimal weights obtained in the unsupervised part are taken as the initial value of the corresponding parameters in the supervised part. In the supervised part, the loss function includes the average reconstruction error of each AE and the cross entropy loss to estimate the predicted label and the actual label. The supervised part is more like a fine-tuning process; the weights continue to be updated. The major advantage of disjoint strategy is able to make full use of those unlabeled samples in training dataset. The loss functions of unsupervised and semisupervised part are shown in (5), (6) respectively.

$$J_{unsup}(\theta) = \sum_{l=0}^L \frac{\lambda_l}{Nm_l} \sum_{n=1}^N G(h^l(n), g(f(h^l(n)))) \quad (5)$$

where λ_l and m_l are moderating coefficient and the node number of l^{th} layer, $h^0(n)$ is the n^{th} input, and the $h^l(n)$ is the

l^{th} representation corresponding to the n^{th} input. L denotes the number of layers.

$$J_{semisup}(\theta) = \sum_{l=0}^L \frac{\lambda_l}{Nm_l} \sum_{n=1}^N G(h^l(n), g(f(h^l(n)))) - \frac{1}{N} \sum_{n=1}^N y(n) \log P(\hat{y}(n) = y(n) | x(n)) \quad (6)$$

where $y(n)$ and $\hat{y}(n)$ are actual label and predicted label corresponding to the input.

Since all samples of the training dataset are labeled ones, two strengths of the disjoint optimization could not be fully realized. Rather than disjoint optimization, the joint optimization seems more suitable for the supervised learning task. Using joint optimization means the SAE and SDAE will directly combine the average reconstruction error of each AE and the cross entropy loss of softmax layer as their loss function, like (6).

3. The Proposed Algorithm

3.1. Preprocessing of Label Noise. According to paper [34], a qualified noise cleansing technique should meet three requirements: (1) keep a high noise detection rate at any noise level, (2) can be used without detailed knowledge about model parameters, data distribution, and (3) yield good results with a certain generalization ability, being effective and robust for a variety of datasets. Here, a K-fold cross-validation based LNC part (KCV LNC) proposed in this paper shall also meet the above three requirements.

Different from other applications of cross-validation for deciding optimal parameters, the proposed KCV LNC applies the K-fold cross-validation thought into the label cleansing area, which is motivated by the successful application of C-Support Vector Classification Filter (C-SVCF) algorithm [35]

Input: A Dataset D corrupted with a unknown percent of mislabeled samples ($\leq 30\%$), $D = \{(x_i, \tilde{y}_i) \mid i = 1, 2, \dots, N\}$; The corresponding parameters of classification model C adopted in K -fold cross-validation; The number of folds of cross-validation K ; Cleanse threshold τ .

Output: The cleansed Dataset $\widehat{D} = \{(x_i, \widehat{y}_i) \mid i = 1, 2, \dots, N\}$;

```

1: Partition the  $D$  into  $K$  equal-sized subsets  $D_1, D_2, \dots, D_K$ ;
2: for  $i = 1, 2, \dots, K$  do:
3:   Form the training dataset  $D_{train} = \cup_{j \neq i} D_j$ ;
4:   Form the validation dataset  $D_{validation} = D_i$ ;
5:   Train model  $C$  with  $D_{train}$ ;
6:   Use model  $C$  to predict label of samples in  $D_{validation}$ ;
7:   for  $j = 1, 2, \dots, n$  do:
8:     if  $\arg \max_{k=1,2,\dots,K_s} P(y_j = k \mid x_j) > \tau$ :
9:       revised its label into  $k$ ,  $\widehat{y}_j = k$ ;
10:    else:
11:      retain the original label,  $\widehat{y}_j = \tilde{y}_j$ ;
12:   end
13: Get the cleansed validation dataset  $\widehat{D}_i = \{(x_j, \widehat{y}_j) \mid j = 1, 2, \dots, N\}$ ;
14: end
15: Gather the cleansed subsets into  $\widehat{D} = \{\widehat{D}_1, \widehat{D}_2, \dots, \widehat{D}_K\}$ ;

```

ALGORITHM 1: Label Noise Cleansing Algorithm based on K -fold cross-validation (standard).

in identifying outliers in breast cancer data sets. Besides, KCV LNC part is compatible with any stable classification algorithm.

The framework of the C-SVCF is renamed as CV LNC here. CV LNC and KCV LNC method mainly differ in terms of this K -fold validation step and the threshold for deciding label revision of training samples. CV LNC directly uses the whole training dataset to train the SVC model, and the trained SVC model is trusted unconditionally. Once the predicted label of a sample in validation dataset is inconsistent with the former label, the label of this sample will be revised. As for the proposed KCV LNC method, first, it adds a K -fold cross-validation step beforehand, dividing the whole training dataset into K equal-sized folds. In each iteration, one fold is isolated as the validation dataset; the rest $K-1$ folds are gathered as training dataset for modeling. Second, in each iteration, samples in validation dataset will be checked; those samples whose max posterior probability are over the threshold τ will be revised. The framework of the proposed KCV LNC method is supposed to display a higher revision accuracy rather than CV LNC. It is fulfilled at the price of training more models. Both the CV LNC and KCV LNC are tested and compared in the case study section.

The detailed procedures of the proposed KCV LNC method are listed in Algorithm 1. It has to be noted that K_s is the class number of all samples, N is the sample number, K is the fold number in K -fold algorithm, and τ is the threshold parameter.

As shown in Algorithm 1, we partition the original dataset into K equal-sized subsets. In i^{th} iteration ($i = 1, 2, \dots, K$), the i^{th} subset is the validation dataset, while the rest subsets are gathered into the training dataset. The trained classification model C will be applied on the validation dataset. For each sample in the validation dataset, Model C 's output is its probability of belonging to each class; the total sum of probabilities

is 1. If the largest posterior probability $\arg \max P(y_j = k \mid x_j)$ is less than the threshold τ , ($j = 1, 2, \dots, N/K$, $k = 1, 2, \dots, K_s$), the corresponding sample's label is not allowed to be revised. When a sample's largest posterior probability is over τ , and its predicted label k is inconsistent with the original label, its original label will be seen as a label noise and revised into k . If not, the original label will be retained. After completing K folds, all labels of the original training dataset will be updated, some of the mislabeled samples will be cleansed. In the case study section, the KCV LNC method is tested with some most common used supervised classifiers such as Gradient Boosting Decision Tree (GBDT), random forest (RF), SVM, and softmax classifier.

Except for parameters of selected classifier, the performance of LNC part is also firmly related to two key parameters, K and τ . K is the number of folds in CV algorithm. Researchers proposed many strategies to determine K in CV algorithm, for example, leave-one-out cross-validation (LOOCV) by [36]. In each iteration of LOOCV, only one sample is isolated for validation, and K is equal to the number of samples in the dataset. According to [37], except for K , the final classification or regression performance of CV algorithm also depends on the selected model. If applied with stable models, such as logistic regression, 5-fold CV or 10-fold CV will be an ideal choice. It is because that they show comparable performance with that of LOOCV while costing much less computational resource than LOOCV. If applied with instable models for regression [38], LOOCV must be the first choice, since it presents the smallest variability and significantly smaller MSE, while 5-fold CV or 10-fold CV generates larger MSE. Paper [39] proposes and verifies an empirical method to determine the number of K folds. The value of K meeting the following two requirements will suffice, (1) $K = c \log(N)$ with $1 \leq c \leq 2$, (2) $K \leq N / (3d)$. N is the sample number of the whole training dataset, d is

Input: A Dataset D corrupted with a unknown percent of mislabeled samples($leq 30\%$), $D = \{(x_i, \bar{y}_i) \mid i = 1, 2, \dots, N\}$; The estimate ratio of the correct labeled sample $L\%$; The corresponding parameters of classification model C adopted in K -fold cross-validation; The number of folds of cross-validation K ;

Output: The cleansed Dataset $\widehat{D} \{(x_i, \widehat{y}_i) \mid i = 1, 2, \dots, N\}$;

- 1:
- 2: Partition the D into K equal-sized subsets $D_1, D_2, \dots, D_K$;
- 3: for $i = 1, 2, \dots, K$ do:
- 4: Form the training dataset $D_{train} = \cup_{j \neq i} D_j$;
- 5: Form the validation dataset $D_{validation} = D_i$;
- 6: Train model C with D_{train} ;
- 7: Use model C to re-predict label of samples in D_{train} ;
- 8: for $l = 1, 2, \dots, N - n$ do:
- 9: Store $\max_{k=1,2,\dots,K_s} P(y_l = k \mid x_l)$ as $P(l)$;
- 10: end
- 11: Sort $P(l)$ into a descending order;
- 12: Set the threshold τ as the $L\%$ percentile value of the $P(l)$ list;
- 13: Use model C to get predicted label of samples in $D_{validation}$;
- 14: for $j = 1, 2, \dots, n$ do:
- 15: if $\arg \max_{k=1,2,\dots,K_s} P(y_j = k \mid x_j) > \tau$:
- 16: revised its label into k , $\widehat{y}_j = k$;
- 17: else:
- 18: retain the original label, $\widehat{y}_j = \bar{y}_j$;
- 19: end
- 20: Get the cleansed validation dataset $\widehat{D}_i = \{(x_j, \widehat{y}_j) \mid j = 1, 2, \dots, n\}$.
- 21: end
- 22: Gather the cleansed subsets into $\widehat{D} = \{\widehat{D}_1, \widehat{D}_2, \dots, \widehat{D}_K\}$;

ALGORITHM 2: Label Noise Cleansing Algorithm based on K -fold cross-validation (revised).

the dimension of each sample in the dataset. In the case study section of this paper, the selected K value is strictly conforming to the Yoonsuh's empirical rule.

As for another parameter τ , to predetermine τ as a fixed value is not a rational choice. An ideal label noise cleansing algorithm shall either automatically obtain the optimal value τ or be robust to the change of τ . If adopting Algorithm 1, we have to employ a grid search strategy to search a relatively optimal τ through the range $(1/K_s, 1)$. For example, when handling triple classification problems, the threshold τ is expected to be over $1/3$. In actual applications, the only input of LNC part is a corrupted training dataset. Detailed information about which samples are correctly labeled or mislabeled is unreachable. Thus, grid search cannot be carried out by directly comparing the mislabeled sample ratio before and after LNC step. We have to compare the final classification accuracy of LNC-SDAEs with different τ settings. The final classification accuracy is tested upon test datasets. Larger τ allows less samples to be revised except for searching optimal τ . The paper recommends LNC part to be carried out twice or three times, which could help offset the defects of some classifiers with inferior performance. Some of easily distinguished mislabeled samples are corrected after the first LNC iteration; then, the cleansed training dataset is input back into LNC part.

It is still not practical enough to carry out grid search process. Therefore, the paper provides a recommended value

of τ . The calculation steps are shown in Algorithm 2. Different from Algorithm 1, τ is not an external input parameter in the revised LNC algorithm, but a calculated number. In each iteration, τ is set as $L\%$ percentile of the resorted max posterior probability of each sample in $K-1$ folds. In case study section, the recommended value of τ is tested and compared to the optimal τ obtained by grid search method in terms of the cleansing effect.

3.2. Framework of LNC-SDAE. During the training of deep learning model, mislabeled samples in the training dataset are likely leading to the wrong activation of neurons, harming the final classification accuracy. The proposed LNC-SDAE method adopts the KCV LNC as a preprocessing procedure to detect and revise mislabeled samples, preventing the latter SDAE model from misleading by these mislabeled samples. The idea of setting a LNC part in front of the SDAE model is inspired by some successful applications of different LNC algorithms in raising the robustness of traditional supervised classifiers upon label noise.

The procedures of LNC-SDAE are summarized in Algorithm 3. In LNC-SDAE model, the proposed KCV LNC method is expected to detect and revise the hidden mislabeled samples, ensuring the correctly labeled samples to be the absolute majority. LNC part is often carried out for several times for better cleansing performance. Then the cleansed training dataset is input into the SDAE model to carry out

- 1: Preparation of training dataset and test dataset;
- 2: Max-min normalize the feature variables of original training dataset and test dataset;
- 3: Carry out a LNC algorithm to detect and revise the mislabeled samples in training dataset;
- 4: Use the cleansed training dataset to train SDAE model;
- 5: Input the normalized test dataset into the trained SDAE network and obtain the classification results of test dataset and calculate the classification accuracy.

ALGORITHM 3: Procedures of LNC-SDAE algorithm.

the fault classification, where the dropout strategy is adopted. The dropout strategy applied in SDAE model has been proved to be effective for raising SDAE's robustness upon feature noise by other researchers. Whether it is useful in improving SDAE's robustness upon label noise will be checked out in the case study part.

SDAE's key parameters include the node number of hidden layer in each DAE and the penalty coefficient λ to moderate the reconstruction error of each layer. The node number is roughly determined based on the dimension and scale of training samples, while the penalty coefficient λ of each layer is set to ensure that all layers enjoy comparable influence weights.

The paper adopts a SDAE model containing three single hidden layer DAEs. All encoder and decoder activation functions in SDAE model are sigmoid function. The cleansing effects of the proposed KCV LNC and CV LN [35] are compared in case study section to verify the effectiveness of KCV LNC in revising mislabeled samples. Then the classification performance of LNC-SDAE trained with corrupted training dataset is compared with that of SDAE trained with standard training dataset, to verify LNC-SDAE's robustness upon label noise. For the sake of contrast, common parameters of LNC-SDAE and SDAE model are kept the same. It is noted that some of these common parameters may not be assigned with optimal values, but able to ensure SDAE's performance in learning useful representations.

4. Case Study

In case study section, a corrupted breast cancer (Wisconsin) dataset and a corrupted TE process are used for verifying different methods' cleansing and classification performance. Both of two corrupted datasets are generated by randomly adding label noise into original breast cancer dataset and TE dataset. Take the corrupted breast cancer dataset for example. 80% of the samples are randomly selected from the standard breast cancer dataset and gathered as the training dataset, where labels of some samples are revised into wrong labels. The rest samples are kept unchanged and gathered as the test dataset. The corrupted TE dataset is obtained in the same way.

Since the label information of original datasets is known in the backstage, different LNC methods' cleansing performance could be estimated by comparing the number of

residual mislabeled samples before and after carrying out them. But it has to be emphasized that the only input of LNC algorithm is the corrupted training dataset.

When the computational resource is sufficient, the grid search method is preferred for getting the threshold τ of KCV LNC algorithm. When the computational resource is limited, a practical alternative is to carry out KCV LNC with a recommended τ (explained in KCV LNC (Algorithm 2)). Compared with the optimal τ got by grid search, a recommended τ enables KCV LNC to show a slightly inferior performance. This gap will also be partially offset by reusing KCV LNC part, in other words, by inputting the cleansed training dataset back into KCV LNC part again. After several repetitions, regardless of recommended τ or optimal τ , KCV LNC will gradually reach a bottleneck that the residual mislabeled samples stop decreasing. Within first several repetitions, every repetition of KCV LNC could help further revise some of the residual mislabeled samples.

4.1. Case Study of the Breast Cancer Wisconsin (Diagnostic) Dataset. Breast cancer (Wisconsin) dataset is a public dataset from UCI machine learning repository <https://www.ics.uci.edu>. It is widely accepted as a standard dataset for estimating the performance of classification algorithm. The original breast cancer dataset contains a total of 569 samples, including 212 Malignant-class samples and 357 Benign-class samples. Each sample has 30 feature variables, the detailed list of which is given in Table 1. In this paper, we manually add a proportion (10%, 20%, 30%) of label noise into the original breast cancer (Wisconsin) dataset to generate three corrupted breast cancer datasets.

Except for the KCV LNC framework proposed in this paper, CV LNC framework is also applied with four kinds of basic classifiers to handle label noise. Four coordinated classifiers are SVM, Gradient Boost Decision Tree (GBDT), Random Forest (RF), and Softmax.

Here, the cleansing performance is estimated according to the number of residual mislabeled samples. After conducting CV LNC or KCV LNC twice or three times, the number of residual mislabeled samples stops decreasing. The best cleansing performance achieved by CV LNC and KCV LNC with four classifiers is presented in Table 2. The ' $Ratio_{Noise} / N_{Noise}$ ' column stands for the initial ratio/number of the mislabeled samples. The 'change rate' column stands for difference between cleansing results of CV LNC and KCV LNC. The

TABLE 1: The feature variables of breast cancer dataset.

No	Variable Name	No	Variable Name
1	mean radius	16	compactness error
2	mean texture	17	concavity error
3	mean perimeter	18	concave points error
4	mean area	19	symmetry error
5	mean smoothness	20	fractal dimension error
6	mean compactness	21	worst radius
7	mean concavity	22	worst texture
8	mean concave points	23	worst perimeter
9	mean symmetry	24	worst area
10	mean fractal dimension	25	worst smoothness
11	radius error	26	worst compactness
12	texture	27	worst concavity
13	perimeter error	28	worst concave points
14	area error	29	worst symmetry
15	smoothness error	30	worst fractal dimension

TABLE 2: The cleansing performance when applying four classifiers with CV LNC and KCV LNC (Algorithm 1) structure upon breast cancer dataset.

Classifier type	$Ratio_{Noise}/N_{Noise}$	CV LNC	KCV LNC (A1)	Change rate
Softmax	10% / 46	10	10	0.00%
Softmax	20% / 91	17	19	11.76%
Softmax	30% / 136	32	27	-15.63%
<i>Res Ratio_{Noise}</i>		4.32%	4.10%	
Classifier type	$Ratio_{Noise}/N_{Noise}$	CV LNC	KCV LNC (A1)	Change rate
SVM	10% / 46	9	9	0.00%
SVM	20% / 91	15	14	-6.66%
SVM	30% / 136	21	22	4.76%
<i>Res Ratio_{Noise}</i>		3.30%	3.30%	
Classifier type	$Ratio_{Noise}/N_{Noise}$	CV LNC	KCV LNC (A1)	Change rate
GBDT	10% / 46	44	13	-70.45%
GBDT	20% / 91	83	10	-87.95%
GBDT	30% / 136	126	28	-77.78%
<i>Res Ratio_{Noise}</i>		18.53%	3.74%	
Classifier type	$Ratio_{Noise}/N_{Noise}$	CV LNC	KCV LNC (A1)	Change rate
RF	10% / 46	34	10	-70.59%
RF	20% / 91	81	16	-80.25%
RF	30% / 136	129	31	-75.97%
<i>Res Ratio_{Noise}</i>		17.87%	4.17%	

‘*Res Ratio_{Noise}*’ row presents the average ratio of the residual mislabeled samples after carrying out LNC.

For breast cancer dataset, the dimension $d = 30$, the whole number of the dataset $N = 455$; thus, $\log(455) \leq K \leq 2 \log(455)$, $2.65 \leq K \leq 5.3$, and $K \leq N/(3d) = 455/90 \approx 5.06$, and thus we select $K = 5$ in KCV LNC part.

After comparing the CV LNC column and the KCV LNC column in Table 2, it is found that the proposed KCV LNC structure presents a better performance on revising mislabeled samples than CV LNC structure. All four classifiers’ cleansing performance gets promotion if CV LNC is replaced with the proposed KCV LNC. When the coordinated

classifiers are GBDT and RF classifiers, the difference is quite big, with the average change rate approaching 80%.

Compared with CV LNC, KCV LNC shows more stable cleansing performance. When applied with GBDT and RF classifier, CV LNC structure could only achieve rather disappointing cleansing performance, with 18.53% and 17.87% of mislabeled samples unrevised. When the coordinated classifiers are softmax and SVM classifiers, CV LNC structure could achieve much better cleansing performance, with only 4.32% and 3.30% of the mislabeled samples remaining. Differently, when applied with KCV LNC structure, all four classifiers could achieve good cleansing performance. Their

TABLE 3: The cleansing performance and thresholds when applying SVM classifier with KCV LNC (Algorithm 1) and KCV LNC (Algorithm 2) upon breast cancer dataset.

$Ratio_{Noise}$ N_{Noise}	KCV LNC (A1)	threshold grid search	KCV LNC (A2)	threshold L% percentile	L%
10% / 46	9	0.74	11	0.72	80
20% / 91	14	0.82	20	0.65	80
30% /136	22	0.84	29	0.58	80

TABLE 4: The classification results of SDAE and LNC-SDAE upon breast cancer dataset.

$Ratio_{Noise}$ N_{Noise}	keep_prob	SDAE trained with corrupted dataset	LNC-SDAE trained with corrupted dataset	SDAE trained with original dataset
0.1/46	1.0	0.939	0.939	0.974
0.1/46	0.9	0.947	0.956	0.939
0.1/46	0.8	0.939	0.956	0.930
average		0.942	0.950	0.947
$Ratio_{Noise}$ N_{Noise}	keep_prob	SDAE trained with corrupted dataset	LNC-SDAE trained with corrupted dataset	SDAE trained with original dataset
0.2/91	1.0	0.868	0.930	0.939
0.2/91	0.9	0.930	0.947	0.939
0.2/91	0.8	0.904	0.939	0.965
average		0.901	0.939	0.947
$Ratio_{Noise}$ N_{Noise}	keep_prob	SDAE trained with corrupted dataset	iLNC-SDAE trained with corrupted dataset	SDAE trained with original dataset
0.3/136	1.0	0.737	0.912	0.974
0.3/136	0.9	0.877	0.965	0.965
0.3/136	0.8	0.807	0.947	0.982
average		0.807	0.942	0.974
average gap			0.060	0.073

average ratios of residual mislabeled samples are all around 4%. Thus, we recommend that when the coordinated classifier is decision tree related classifier, it is better to apply them with KCV LNC structure rather than CV LNC structure. KCV LNC could offset a major defect of extensions of decision tree method, the sensitivity to feature noise and label noise.

From Table 2, we can find that applying KCV LNC with SVM achieves the best cleansing performance. Regardless of the initial ratio of mislabeled samples, the average ratio of residual mislabeled samples is only 3.30%. Thus, we apply KCV LNC with SVM as the optimal LNC combination upon corrupted breast cancer dataset.

After fixing the SVM as the classifier, a comparison is made between KCV LNC with τ obtained by grid search and KCV LNC with a recommended τ , which is shown in Table 3. The recommended τ could be obtained according to KCV LNC (Algorithm 2). The threshold τ could be set as the L% percentile of the resorted max posterior probability of each sample in K-1 folds, shown in 'threshold L% percentile' column. L is an estimated percent of correctly labeled samples in training dataset, shown in L% column. In normal conditions, mislabeled samples are supposed to be in the minority, less than 30%. L is recommended to be set as 70, 80 or 90 here. Larger L means smaller ratios of samples are suspected as

mislabeled ones. After comparison, we find that when $L = 80$, softmax classifier with KCV LNC shows the best cleansing performance.

Cleansing performance is also estimated based on the number of residual mislabeled samples after conducting KCV LNC (A1, A2), shown in the KCV LNC (A1) column and KCV LNC (A2) column. The recommended τ is inferior to the τ obtained by grid search methods. However, considering the gap is not that big; this recommended threshold τ will suffice especially when computational resources is limited.

After being processed by the LNC part, the cleansed training dataset is used for training the SDAE which also contains three single-layer DAEs. m_1 , m_2 , and m_3 are the hidden layer size of three single-layer DAEs, $m_1 = 300$, $m_2 = 150$, $m_3 = 50$. Thus the node numbers of each layer of DAEs are [30, 300, 30], [300, 150, 300], and [150, 50, 150]. λ_1 , λ_2 , and λ_3 are moderate coefficients used for balancing the reconstruction error of three DAEs' hidden layers, allowing each layer's reconstruction error to hold a comparative weight upon loss function. Here, $\lambda_1 = 1000.0$, $\lambda_2 = 10.0$, and $\lambda_3 = 0.1$.

The classification results of SDAE and LNC-SDAE upon the breast cancer dataset are listed in Table 4. In Table 4, three columns stand for the final fault classification results of SDAE trained with corrupted dataset, LNC-SDAE trained with

FIGURE 3: The classification results upon breast cancer dataset by SDAE trained with corrupted dataset, LNC-SDAE trained with corrupted dataset, and SDAE trained with original dataset.

corrupted dataset, and SDAE trained with original dataset, respectively. Original dataset denotes the original training dataset without adding any label noise. Corrupted dataset denotes the training dataset corrupted with fixed ratio label noise. The 'keep_prob' column denotes the keep probability in the dropout module of SDAE. For example, if the keep_prob is 0.8, meaning 20% of the elements are set as zero, while the other 80% of the elements of this layer is scaled up by $(1/0.8)$; the scaling is set for keeping the expected sum of this layer unchanged. The above classification results are also displayed in the form of graph, shown in Figure 3.

From 'SDAE trained with corrupted dataset' column in Table 4, we can find when the initial noise ratio increasing from 10% to 20% and 30%, and the average classification drops from 0.942 to 0.901 and 0.807. In comparison, SDAE trained with original dataset provides a 0.947 classification accuracy. When the residual mislabeled samples are in the absolute minority ($\leq 10\%$), the final classification of LNC-SDAE are quite close to that of SDAE trained with original dataset (gap is only 0.5%). We infer from the above result that the negative effect of mislabeled samples could be partially

offset by other correctly labeled samples in dataset and the epoch training mechanism of SDAE.

However, when mislabeled samples accumulate to a specific amount, for example, 30%, SDAE's own robustness is no longer reliable. Thus, it is better to keep the ratio of mislabeled samples in the training dataset at a low level ($\leq 10\%$), ensuring SDAE to generate a reliable classification accuracy.

From Table 2, it is found that SVM classifier with KCV LNC is the best LNC combination for corrupted breast cancer dataset. The final ratio of residual mislabeled samples is 3.30%.

The final classification results shown in Table 4 prove the effectiveness of KCV LNC structure, since the final classifications of LNC-SDAE trained with corrupted dataset is quite similar to that of SDAE trained with original dataset. In Table 4, the average classification accuracy gap between the LNC-SDAE trained with corrupted dataset and the SDAE trained with original dataset is only 1.3%. It is also illustrated in Figure 3; the orange line and green line keep close. Based on results in Table 4 and Figure 3, a conclusion could be

FIGURE 4: The diagram of benchmark Tennessee Eastman process.

drawn that LNC-SDAE is robust upon label noise in breast cancer dataset.

4.2. Case Study of the Tennessee Eastman Process. The well-known Tennessee Eastman chemical process [40] is also applied for real case study in this paper. The main flow diagram of the TE process is shown in Figure 4. The simulator for TE process is downloaded from the website <http://depts.washington.edu/control/LARRY/TE/download.html>. TE process contains 41 measured variables and 11 manipulated variables, among which 22 manipulated variables and 11 process variables are collected at the frequency of 3 min. Other variables are collected every 6 or 15 min. In this section, 22 manipulated variables and 11 process variables are selected as feature variables.

TE process introduces 21 programmed process faults called IDVs [41], whose detailed list is presented in Table 5. IDV 0 is a normal case, while IDV 1-21 are all faulty cases. Each IDV case includes a training dataset (480 samples) and a test dataset (800 samples). Here, some reoccurring IDVs are gathered into one dataset to verify one method's fault classification ability as paper [42] does. Three pairs of training dataset and test datasets are listed in Table 6. Datasets 1, 2, and 3 all have 2400 samples for training and 4000 samples for test. In Table 6, N_{tr} denotes the number of samples in training dataset, N_{tst} denotes the number of samples in test dataset, and N_v is the number of feature variables in training and test dataset.

In case study of TE process, the proposed KCV LNC method and the CV LNC method are also applied with four classifiers (SVM, GBDT, RF, and Softmax) to handle label noise problems. The detailed results of their cleansing performance are listed in Tables 7–10, respectively. The dimension $d = 33$, $N_{tr} = 2400$; thus, $\log(2400) \leq K \leq 2\log(2400)$, $3.38 \leq K \leq 6.76$ and $K \leq N_{tr}/(3d) = 2400/99 \approx 24$. Thus, the number of folds K in KCV LNC method is set as $K=5$. And the thresholds of KCV LNC structure upon TE 1,2,3 dataset are listed in Table 11.

Similar to corrupted breast cancer dataset, label noise is manually added to the TE dataset 1,2,3. The cleansing effects of different methods are also estimated by the number of residual mislabeled ones.

The cleansing performance of CV LNC and KCV LNC with four different classifiers upon TE 1,2,3 dataset is listed in Tables 7–10, respectively. In Tables 7–10, the 'CV LNC' column also contains the number of residual mislabeled samples after applying classifiers with CV LNC structure, while the 'KCV LNC' column contains the number of residual mislabeled samples after applying classifiers with KCV LNC. The 'change rate' column shows the difference between the residual mislabeled samples after carrying out CV LNC and KCV LNC (A1). The ' $Res\ Ratio_{Noise}$ ' row shows the average ratio of residual mislabeled samples after conducting these LNC algorithms.

By analyzing the cleansing results shown in Tables 7–10, we also find KCV LNC shows a better performance than CV

TABLE 5: Descriptions of process faults in TE process.

IDV	Process variable	Type
1	A C feed ratio B composition constant (stream 4)	step
2	B composition A C ratio constant (stream 4)	step
3	D feed temperature (stream 2)	step
4	reactor cooling water inlet temperature	step
5	condenser cooling water inlet temperature	step
6	A feed loss (stream 1)	feed
7	C header pressure loss-reduced availability (stream 4)	step
8	A B C feed composition (stream 4)	variation
9	D feed temperature (stream 2)	random
10	C feed temperature (stream 4)	variation
11	reactor cooling water inlet temperature	random
12	condenser cooling water inlet temperature	variation
13	reaction kinetics slow	drift
14	reactor cooling water valve	sticking
15	condenser cooling water valve	sticking
16	unknown	unknown
17	unknown	unknown
18	unknown	unknown
19	unknown	unknown
20	unknown	unknown
21	valve position constant (stream 4)	constant position

TABLE 6: The information of TE 1, TE 2, and TE 3 dataset.

Dataset	N_{tr}	N_{fst}	N_v	IDVs
1	2400	4000	33	IDV 1,2,6,7,8
2	2400	4000	33	IDV 1,2,6,8,13
3	2400	4000	33	IDV 0,1,2,4,7

TABLE 7: The cleansing performance when applying Softmax classifier with CV LNC and KCV LNC (Algorithm 1) structure upon TE 1,2,3 dataset.

Dataset	Classifier type	$Ratio_{Noise}/N_{Noise}$	CV LNC	KCV LNC (A1)	Change rate
TE1	Softmax	10% / 240	38	52	36.84%
TE1	Softmax	20% / 480	63	61	-3.17%
TE1	Softmax	30% / 720	84	67	-20.24%
TE2	Softmax	10% / 240	72	83	15.28%
TE2	Softmax	20% / 480	134	117	-12.69%
TE2	Softmax	30% / 720	193	144	-25.39%
TE3	Softmax	10% / 240	105	113	7.62%
TE3	Softmax	20% / 480	214	214	0.00%
TE3	Softmax	30% / 720	318	340	6.92%
<i>Res Ratio_{Noise}</i>			5.65%	5.51%	

TABLE 8: The cleansing performance when applying SVM classifier with CV LNC and KCV LNC (Algorithm 1) structure upon TE 1,2,3 dataset.

Dataset	Classifier type	$Ratio_{Noise}/N_{Noise}$	CV LNC	KCV LNC (A1)	Change rate
TE1	SVM	10% / 240	51	48	-5.88%
TE1	SVM	20% / 480	75	65	-13.33%
TE1	SVM	30% / 720	88	87	-1.14%
TE2	SVM	10% / 240	75	83	10.67%
TE2	SVM	20% / 480	150	111	26.00%
TE2	SVM	30% / 720	210	140	-33.33%
TE3	SVM	10% / 240	177	178	0.56%
TE3	SVM	20% / 480	357	325	-8.96%
TE3	SVM	30% / 720	518	361	-30.31%
<i>Res Ratio_{Noise}</i>			7.88%	6.47%	

TABLE 9: The cleansing performance when applying GBDT classifier with CV LNC and KCV LNC (Algorithm 1) structure upon TE 1,2,3 dataset.

Dataset	Classifier type	$Ratio_{Noise}/N_{Noise}$	CV LNC	KCV LNC (A1)	Change rate
TE1	GBDT	10% / 240	82	72	-12.20%
TE1	GBDT	20% / 480	147	96	-34.69%
TE1	GBDT	30% / 720	208	90	-56.73%
TE2	GBDT	10% / 240	78	53	-32.05%
TE2	GBDT	20% / 480	134	80	-40.30%
TE2	GBDT	30% / 720	200	107	-46.50%
TE3	GBDT	10% / 240	93	31	-66.67%
TE3	GBDT	20% / 480	135	34	-74.81%
TE3	GBDT	30% / 720	183	67	-63.39%
<i>Res Ratio_{Noise}</i>			5.83%	2.92%	

TABLE 10: The cleansing performance when applying RF classifier with CV LNC and KCV LNC (Algorithm 1) structure upon TE 1,2,3 dataset.

Dataset	Classifier type	$Ratio_{Noise}/N_{Noise}$	CV LNC	KCV LNC (A1)	Change rate
TE1	RF	10% / 240	197	70	-64.47%
TE1	RF	20% / 480	424	119	-71.93%
TE1	RF	30% / 720	663	175	-73.60%
TE2	RF	10% / 240	204	95	-53.43%
TE2	RF	20% / 480	434	143	-67.05%
TE2	RF	30% / 720	679	215	-68.34%
TE3	RF	10% / 240	209	40	-80.86%
TE3	RF	20% / 480	437	78	-82.15%
TE3	RF	30% / 7200	667	105	-84.26%
<i>Res Ratio_{Noise}</i>			18.12%	4.81%	

TABLE 11: The cleansing performance and thresholds when applying KCV LNC (Algorithm 1) and KCV LNC (Algorithm 2) upon TE 1,2,3 dataset.

Dataset	$Ratio_{Noise}$	KCV LNC	threshold	KCV LNC	threshold	L%
Classifier	N_{Noise}	(A1)	grid search	(A2)	L% percentile	
TE1	10% / 240	52	0.87	53	0.73	80
Softmax	20% / 480	61	0.92	65	0.63	80
	30% / 720	67	0.94	80	0.54	80
TE2	10% / 240	53	0.99	73	0.91	70
GBDT	20% / 480	80	0.98	113	0.78	70
	30% / 720	107	0.99	211	0.64	70
TE3	10% / 240	31	0.98	36	0.9	70
GBDT	20% / 480	34	0.99	63	0.76	70
	30% / 720	67	0.99	126	0.62	70

TABLE 12: The classification results of SDAE and LNC-SDAE upon TE 1 Dataset.

$\frac{Ratio_{Noise}}{N_{Noise}}$	keep_prob	SDAE trained with corrupted dataset	LNC-SDAE trained with corrupted dataset	SDAE trained with original dataset
0.1/240	1.0	0.927	0.932	0.941
0.1/240	0.9	0.905	0.944	0.913
0.1/240	0.8	0.863	0.874	0.907
average		0.899	0.916	0.920
$\frac{Ratio_{Noise}}{N_{Noise}}$	keep_prob	SDAE trained with corrupted dataset	LNC-SDAE trained with corrupted dataset	SDAE trained with original dataset
0.2/480	1.0	0.846	0.924	0.936
0.2/480	0.9	0.871	0.951	0.938
0.2/480	0.8	0.824	0.870	0.905
average		0.847	0.915	0.926
$\frac{Ratio_{Noise}}{N_{Noise}}$	keep_prob	SDAE trained with corrupted dataset	LNC-SDAE trained with corrupted dataset	SDAE trained with original dataset
0.3/720	1.0	0.727	0.927	0.931
0.3/720	0.9	0.848	0.933	0.919
0.3/720	0.8	0.821	0.903	0.898
average		0.799	0.921	0.916
average gap			0.069	0.073

LNC. In Tables 7 and 8, when applied with softmax or SVM classifier, the KCV LNC (A1) method is found to achieve nearly the same performance as CV LNC method. From the Tables 9 and 10, we can also find that when applied with decision tree related classifiers, like GBDT and RF classifier, KCV LNC (A1) method achieves much better performance than CV LNC method. When applying CV LNC with RF, the average ratio of residual mislabeled samples in processed dataset is 18.12%, while that of KCV LNC (A1) with RF is only 4.81%.

We also find the KCV LNC shows a better generalization capability than CV LNC. Applied with different coordinated classifiers, CV LNC shows fluctuant cleansing performance while KCV LNC shows more stable cleansing performance. When applied with CV LNC method, SVM and RF classifiers show inferior cleansing performance than softmax and GBDT classifier. For example, when applying CV LNC, the $Res\ Ratio_{Noise}$ gap between the worst RF and the best softmax is $18.12\% - 5.65\% = 12.47\%$. When adopting KCV LNC structure with these four classifiers, the $Res\ Ratio_{Noise}$ gap is less than $6.47\% - 2.92\% = 3.55\%$.

Based on above results, we select softmax classifier, GBDT, and GBDT classifier as the optimal classifier for handling TE 1, 2, 3 dataset; all of them are applied with KCV LNC structure.

Similar to case study of breast cancer dataset, a performance comparison is made between KCV LNC with τ obtained by grid search and KCV LNC with a recommended τ , shown in Table 11. Cleansing performance is also estimated based on the ratio of residual mislabeled samples after adopting different τ in KCV LNC, shown in the KCV LNC (A1) column and KCV LNC (A2) column. The recommended setting of Percentile L is 70, 80, or 90.

After comparison, we get the optimal L value when applying different coordinated classifiers with KCV LNC structure. After fixing L, the recommended value of τ is obtained. Based on the number of residual mislabeled samples, the recommended τ is inferior to the τ obtained from grid search methods. When handling TE 1 dataset, their performance gap is the least. In normal condition, KCV LNC (A1) is still the first choice for handling label noise cleansing task by default.

After processed by LNC part, the cleansed training datasets are input into SDAE to extract representations and carry out the final fault classification. The final fault classification results of different methods upon TE 1, 2, 3 dataset are shown in Tables 12–14. The results are also displayed in the form of graph, as shown in Figures 5–7. The SDAE adopted here also contains three single-layer DAEs. Important parameters are determined as follows, $\lambda_1 = 1000.0$, $\lambda_2 = 10.0$, $\lambda_3 = 0.1$, $m_1 = 300$, $m_1 = 150$, and $m_1 = 50$. Different settings of dropout probability are also tested here, which are shown in ‘keep_prob’ columns in Tables 12–14.

By analyzing Tables 12–14 and Figures 5–7, our findings are as follows. First, label noise in training dataset impacts a negative influence upon classification accuracy. With the initial label noise ratio increases from 10% to 30%, the average classification accuracy gap between SDAE trained with corrupted dataset and SDAE trained with original dataset becomes larger. Taking TE 1 for example (Table 12), when the initial label noise ratios are 10%, 20%, and 30%, their gaps are 0.038, 0.079, and 0.082, respectively.

Second, KCV LNC is effective in cleansing label noise, so as to help improve the final classification accuracy of LNC-SDAE. By analyzing the ‘average’ rows of Tables 12–14, we could find the mean gaps between SDAE trained with corrupted dataset and LNC-SDAE trained with corrupted

TABLE 13: The classification results of SDAE and LNC-SDAE upon TE 2 Dataset.

$Ratio_{Noise}$ N_{Noise}	keep_prob	SDAE trained with corrupted dataset	LNC-SDAE trained with corrupted dataset	SDAE trained with original dataset
0.1/240	1.0	0.823	0.881	0.908
0.1/240	0.9	0.889	0.924	0.913
0.1/240	0.8	0.917	0.923	0.919
average		0.876	0.909	0.914
$Ratio_{Noise}$ N_{Noise}	keep_prob	SDAE trained with corrupted dataset	LNC-SDAE trained with corrupted dataset	SDAE trained with original dataset
0.2/480	1.0	0.747	0.862	0.911
0.2/480	0.9	0.896	0.921	0.932
0.2/480	0.8	0.896	0.919	0.931
average		0.846	0.901	0.925
$Ratio_{Noise}$ N_{Noise}	keep_prob	SDAE trained with corrupted dataset	LNC-SDAE trained with corrupted dataset	SDAE trained with original dataset
0.3/720	1.0	0.722	0.900	0.918
0.3/720	0.9	0.896	0.920	0.926
0.3/720	0.8	0.895	0.899	0.917
average		0.838	0.907	0.920
total gap			0.052	0.066

FIGURE 5: The classification results upon TE 1 dataset by SDAE and LNC-SDAE.

TABLE 14: The classification results of SDAE and LNC-SDAE upon TE 3 Dataset.

$Ratio_{Noise}$ N_{Noise}	keep_prob	SDAE trained with corrupted dataset	LNC-SDAE trained with corrupted dataset	SDAE trained with original dataset
0.1/240	1.0	0.854	0.904	0.896
0.1/240	0.9	0.715	0.863	0.809
0.1/240	0.8	0.601	0.624	0.659
average		0.723	0.797	0.788
$Ratio_{Noise}$ N_{Noise}	keep_prob	SDAE trained with corrupted dataset	LNC-SDAE trained with corrupted dataset	SDAE trained with original dataset
0.2/480	1.0	0.738	0.891	0.893
0.2/480	0.9	0.723	0.867	0.842
0.2/480	0.8	0.631	0.645	0.694
average		0.697	0.801	0.810
$Ratio_{Noise}$ N_{Noise}	keep_prob	SDAE trained with corrupted dataset	LNC-SDAE trained with corrupted dataset	SDAE trained with original dataset
0.3/720	1.0	0.699	0.899	0.906
0.3/720	0.9	0.609	0.791	0.818
0.3/720	0.8	0.570	0.627	0.633
average		0.626	0.773	0.786
total gap			0.108	0.112

FIGURE 6: The classification results upon TE 2 dataset by SDAE and LNC-SDAE.

FIGURE 7: The classification results upon TE3 dataset by SDAE and LNC-SDAE.

dataset are 6.9% (TE 1), 5.2%(TE 2), and 10.8%(TE 3), respectively.

Third, we can find that the classification accuracies of LNC-SDAE trained with corrupted dataset and SDAE trained with original dataset are very close. The mean gaps between them are 0.4% (TE 1), 1.4%(TE 2), and 0.4%(TE 3), respectively. It supports the effectiveness of LNC-SDAE in handling inaccurate classification problems, and the robustness of LNC-SDAE structure against label noise.

To sum up, we can draw several conclusions. Firstly, when applied with different stable classifiers, the proposed KCV LNC method shows a stable performance in revising the mislabeled training samples. Regardless of the initial noise ratio ($\leq 30\%$) in training dataset, KCV LNC method is able to revise most of mislabeled samples, ensuring the ratio of residual mislabeled samples lower than 10%. Secondly, KCV LNC keeps functioning properly even when there is lack of the prior knowledge about the target dataset. Thirdly, with the help of KCV LNC structure and dropout training strategy, the LNC-SDAE framework shows convincing robustness upon label noise and capability in handle inaccurate supervised classification problems.

5. Conclusion

In this paper we first propose a new label noise cleansing algorithm based on K-fold cross-validation thought named as KCV- LNC. It is compatible with different stable classifiers to fulfill the label noise cleansing task. On the basis of KCV LNC part, a new label noise robust deep learning framework named as LNC-SDAE is put forward. The KCV LNC part is prepositioned to revise mislabeled training samples into correct ones, preventing the latter SDAE model to be misled by mislabeled samples during the training procedure.

Most of the parameters of KCV LNC could be decided according to the scale of the dataset itself. Except for threshold τ , a parameter is used for permitting the label revision. The optimal value of τ is obtained through grid search method by default. The paper also gives another practical approach to determine τ , which also suffices in most cases. The experiment result shows that when the initial noise ratio of the corrupted dataset is less than 30%, the KCV LNC part is able to revise most of mislabeled samples and ensure the ratio of residual mislabeled samples is lower than 10%.

In LNC-SDAE, the training dataset is first processed by KCV LNC part. Then the cleansed training dataset is

provided to SDAE model to learn representation for the final classification. A conventional dropout module is applied to enhance the robustness of LNC-SDAE. The experiment result shows that the classification accuracy of LNC-SDAE trained with the corrupted training dataset is nearly the same with that of SDAE trained with the training dataset without any noise label. LNC-SDAE is proved effective in handling inaccurate supervised classification problems.

Data Availability

(1) *The Breast Cancer Dataset*. The Breast Cancer Dataset supporting this study is downloaded from UCI machine learning repository, [http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+\(Diagnostic\)](http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)). The Breast Cancer Dataset is also from previously published paper [35], which has been cited in this study. (2) *The TE 1, TE 2, TE 3 Dataset*. The Tennessee Eastman (TE) process data are from previously reported studies and datasets, which have been cited. The processed data are described in the following paper [40] published in 1995. The original TE dataset could be downloaded from the website below. <https://github.com/camaramm/tennessee-eastman-profBraatz>. The original TE dataset contains 1 normal condition dataset, (480 normal samples in training dataset and 960 normal samples in test dataset) and 21 faulty datasets (480 faulty samples in training dataset and 160 normal samples and 800 faulty samples in test dataset). In this study, the focus is upon multifault classification problems. Thus, we select several typical faulty datasets and combine them to construct TE1, TE2, TE3 datasets for experiment. TE1, TE2, TE3 dataset are depicted in Tab 2 of the paper [42] published in 2017.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors want to thank the National Natural Science Foundation of China for their support for this work (Projects U1664264, U1509203).

References

- [1] S. Fefilyatov, M. Shreve, and K. Kramer, "Label-noise reduction with support vector machines," in *International Conference on Pattern Recognition*, pp. 3504–3508, 2012.
- [2] Z. Ge, Z. Song, S. X. Ding, and B. Huang, "Data mining and analytics in the process industry: the role of machine learning," *IEEE Access*, vol. 5, pp. 20590–20616, 2017.
- [3] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the the 25th International Conference*, pp. 1096–1103, Helsinki, Finland, July 2008.
- [4] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: explicit invariance during feature extraction," in *Proceedings of the 28th International Conference on Machine Learning (ICML '11)*, pp. 833–840, Bellevue, Wash, USA, July 2011.
- [5] N. Tomašev and K. Buza, "Hubness-aware kNN classification of high-dimensional data in presence of label noise," *Neurocomputing*, vol. 160, pp. 157–172, 2015.
- [6] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, no. 3, pp. 220–273, 1995.
- [7] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.
- [8] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," in *Proceedings of the Annual Conference on Computational Learning Theory*, pp. 80–91, 1998.
- [9] B. Sun, S. Chen, J. Wang, and H. Chen, "A robust multi-class AdaBoost algorithm for mislabeled noisy data," *Knowledge-Based Systems*, vol. 102, pp. 87–102, 2016.
- [10] J. Cao, S. Kwong, and R. Wang, "A noise-detection based AdaBoost algorithm for mislabeled data," *Pattern Recognition*, vol. 45, no. 12, pp. 4451–4465, 2012.
- [11] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-2, no. 3, pp. 408–421, 1972.
- [12] C. E. Brodley and M. A. Friedl, "Identifying and eliminating mislabeled training instances," in *Proceedings of the 13th National Conference on Artificial Intelligence*, pp. 799–805, AAAI Press, Portland, Ore, USA, August 1996.
- [13] I. Guyon, N. Matic, and V. Vapnik, "Discovering informative patterns and data cleaning. discovering informative patterns and data cleaning," in *American Association for Artificial Intelligence*, pp. 181–203, 2013.
- [14] A. Karmaker and S. Kwek, "A boosting approach to remove class label noise," *International Journal of Intelligent Systems*, vol. 3, no. 3, pp. 169–177, 2005.
- [15] T. G. Dietterich, "Experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization," *Machine Learning*, vol. 40, no. 2, pp. 139–157, 2000.
- [16] Y. Li, L. F. A. Wessels, D. D. Ridder, and M. J. T. Reinders, "Classification in the presence of class noise using a probabilistic kernel fisher method," *Pattern Recognition*, vol. 40, no. 12, pp. 3349–3357, 2007.
- [17] N. D. Lawrence, "Estimating a kernel fisher discriminant in the presence of label noise," in *Proceedings of the Eighteenth International Conference on Machine Learning*, pp. 306–313, 2001.
- [18] C. Pal, G. Mann, and R. Minerich, "Putting semantic information extraction on the map: Noisy label models for fact extraction," in *Proceedings of the American Association for Artificial Intelligence*, pp. 80–85, Canada, July 2007.
- [19] J. Bootkrajang and A. Kabán, "Label-noise robust logistic regression and its applications," in *Proceedings of the American Association for Artificial Intelligence*, pp. 143–158, 2012.
- [20] Z. Ge, "Process data analytics via probabilistic latent variable models: a tutorial review," *Industrial & Engineering Chemistry Research*, vol. 57, no. 38, pp. 12646–12661, 2018.
- [21] Y. Wu and Y. Liu, "Robust truncated hinge loss support vector machines," *Journal of the American Statistical Association*, vol. 102, no. 479, pp. 974–983, 2007.
- [22] B. Biggio, B. Nelson, and P. Laskov, "Support vector machines under adversarial label noise," *Journal of Machine Learning Research*, vol. 20, no. 3, pp. 97–112, 2011.
- [23] F. Lv, C. Wen, Z. Bao, and M. Liu, "Fault diagnosis based on deep learning," in *Proceedings of the 2016 American Control Conference (ACC)*, pp. 6851–6856, Boston, Mass, USA, July 2016.

- [24] J. Zhu, Z. Ge, Z. Song, and F. Gao, "Review and big data perspectives on robust data mining approaches for industrial process modeling with outliers and missing data," *Annual Reviews in Control*, vol. 46, pp. 107–133, 2018.
- [25] L. Yao and Z. Ge, "Deep learning of semisupervised process data with hierarchical extreme learning machine and soft sensor application," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 2, pp. 1490–1498, 2017.
- [26] A. Liaw and M. Wiener, "Classification and regression by randomforest," *R News*, vol. 23, no. 23, 2002.
- [27] Y. Lin and W. Yan, "Study of soft sensor modeling based on deep learning," in *Proceedings of the American Control Conference*, pp. 5830–5835, 2015.
- [28] S. Fefilatyev, M. Shreve, K. Kramer et al., "Label-noise reduction with support vector machines," in *Proceedings of the 21st International Conference on Pattern Recognition, ICPR 2012*, pp. 3504–3508, Japan, November 2012.
- [29] P. Vincent, H. Larochelle, I. Lajoie, and P. Manzagol, "Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, no. 12, pp. 3371–3408, 2010.
- [30] W. Yan, P. Guo, L. gong, and Z. Li, "Nonlinear and robust statistical process monitoring based on variant autoencoders," *Chemometrics and Intelligent Laboratory Systems*, vol. 158, pp. 31–40, 2016.
- [31] W. Sun, S. Shao, R. Zhao, R. Yan, X. Zhang, and X. Chen, "A sparse auto-encoder-based deep neural network approach for induction motor faults classification," *Measurement*, vol. 89, pp. 171–178, 2016.
- [32] L. Jiang, Z. Ge, and Z. Song, "Semi-supervised fault classification based on dynamic Sparse Stacked auto-encoders model," *Chemometrics and Intelligent Laboratory Systems*, vol. 168, no. 15, pp. 72–83, 2017.
- [33] W. Haiyan, Y. Haomin, L. Xueming, and R. Haijun, "Semi-supervised autoencoder: a joint approach of representation and classification," in *Proceedings of the 2015 International Conference on Computational Intelligence and Communication Networks (CICN)*, pp. 1424–1430, Jabalpur, India, December 2015.
- [34] Y. Xiao, T. M. Khoshgoftaar, and N. Seliya, "The partitioning and rule-based filter for noise detection," in *Proceedings of the IEEE International Conference on Information Reuse and Integration*, pp. 205–210, 2005.
- [35] J. Thongkam, G. Xu, Y. Zhang, and F. Huang, "Support Vector Machine for Outlier Detection in Breast Cancer Survivability Prediction," in *Advanced Web and Network Technologies, and Applications*, vol. 4977 of *Lecture Notes in Computer Science*, pp. 99–109, Springer, Berlin, Germany, 2008.
- [36] M. Stone, "Deep learning of semi-supervised process data with hierarchical extreme learning machine and soft sensor application," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 36, pp. 111–147, 1974.
- [37] D. Krstajic, L. J. Buturovic, D. E. Leahy, and S. Thomas, "Cross-validation pitfalls when selecting and assessing regression and classification models," *Journal of Cheminformatics*, vol. 6, no. 1, article 10, pp. 1–15, 2014.
- [38] Y. L. Zhang and Y. H. Yang, "Cross-validation for selecting a model selection procedure," *Journal of Econometrics*, vol. 187, no. 1, pp. 95–112, 2015.
- [39] Y. Jung, "Multiple predicting K-fold cross-validation for model selection," *Journal of Nonparametric Statistics*, vol. 30, no. 1, pp. 197–215, 2018.
- [40] P. Lyman and C. Georgakis, "Plant-wide control of the Tennessee Eastman problem," *Computers & Chemical Engineering*, vol. 19, no. 3, pp. 321–331, 1995.
- [41] J. J. Downs and E. F. Vogel, "Plant-wide industrial process control problem," *Computers & Chemical Engineering*, vol. 17, no. 3, pp. 245–255, 1993.
- [42] D.-Y. Zhu, J.-Y. Shen, W.-P. Huang, and J. Liang, "Fault classification based on modified active learning and weighted SVM," *Journal of Zhejiang University (Engineering Science)*, vol. 51, no. 4, pp. 697–705, 2017.

Submit your manuscripts at
www.hindawi.com