Received: 9 April 2021 | Revised: 26 June 2021 | Accepted: 4 July 2021

DOI: 10.1002/int.22582

RESEARCH ARTICLE

WILEY

Improved autoencoder for unsupervised anomaly detection

Zhen Cheng | **Siwei Wang** | **Pei Zhang** | **Siqi Wang** | **Xinwang Liu** | **En Zhu**

School of Computer, National University of Defense Technology, Changsha, China

Correspondence
Zhen Cheng, School of Computer, National University of Defense Technology, No. 109 Deya Road, Kaifu District, Changsha, 410073 Hunan, China.
Email: chengzhen16@nudt.edu.cn

Funding information
National Key Research and Development Program of China, Grant/Award Number: 2018YFB0204301; National Natural Science Foundation of China, Grant/Award Number: 62006236; Hunan Provincial Natural Science Foundation, Grant/Award Number: 2020JJ5673; Graduate Research and Innovation Projects of Jiangsu Province, Grant/Award Number: 18KJB520041; National University of Defense Technology (NUDT) Research Project, Grant/Award Number: ZK20-10

Abstract

Deep autoencoder-based methods are the majority of deep anomaly detection. An autoencoder learning on training data is assumed to produce higher reconstruction error for the anomalous samples than the normal samples and thus can distinguish anomalies from normal data. However, this assumption does not always hold in practice, especially in unsupervised anomaly detection, where the training data is anomaly contaminated. We observe that the autoencoder generalizes so well on the training data that it can reconstruct both the normal data and the anomalous data well, leading to poor anomaly detection performance. Besides, we find that anomaly detection performance is not stable when using reconstruction error as anomaly score, which is unacceptable in the unsupervised scenario. Because there are no labels to guide on selecting a proper model. To mitigate these drawbacks for autoencoder-based anomaly detection methods, we propose an Improved AutoEncoder for unsupervised Anomaly Detection (IAEAD). Specifically, we manipulate feature space to make normal data points closer using anomaly detection-based loss as guidance. Different from previous methods, by integrating the anomaly detection-based loss and autoencoder's reconstruction loss, IAEAD can jointly optimize for anomaly detection tasks and learn

representations that preserve the local data structure to avoid feature distortion. Experiments on five image data sets empirically validate the effectiveness and stability of our method.

KEYWORDS
autoencoder, deep learning, unsupervised anomaly detection

1 | INTRODUCTION

Anomaly detection refers to finding data that are significantly different from the others. These data are often called outliers, anomalies, faults, defects, novelty, or errors in different literature contexts. Anomaly detection has drawn a large amount of attention due to its broad applications in many different domains, including financial fraud detection,[1] cybersecurity intrusion detection,[2,3] and much more. Many methods have been proposed to tackle anomaly detection over the years. A vast majority of the methods rely on the availability of labels. Labels indicate whether a chosen sample is normal or anomalous. Based on the availability of labels, anomaly detection can be classified into three categories.[4] (1) Supervised anomaly detection (SAD) involves training a supervised binary or multiclass classifier, using labels of both normal and anomalous data samples. (2) Semisupervised anomaly detection (SSAD) uses only normal data to detect anomalies. The labels of normal samples are far easier to obtain than anomalies, so solutions in this category are more widely adopted. (3) Unsupervised anomaly detection (UAD) detects anomalies based on intrinsics of the training data. It handles unlabeled data or anomaly-contaminated data, including both normal and anomalous data.

Supervised and SSAD methods require at least a single class of training data labeled as "normal." Such approaches have two major limitations. First, most of the data in real-world settings does not have corresponding labels, which limits the application to a small portion of the data sets. Second, creating the labels itself is a costly process. It is time-consuming, especially when it comes to large data sets. Not to mention some particular fields like medical image recognition which requires experts for image recognition. In this paper, we focus on UAD.

In a univariate data setting or a data set with a known distribution, extensive research has been done to detect anomalies. However, when it comes to high dimensional data like images, traditional methods typically are not able to characterize the multivariate distribution or extract relevant information for anomaly detection. With the advances in deep neural networks, deep learning-based anomaly detection methods have become increasingly popular. They show huge advantages compared with traditional methods such as principal component analysis (PCA),[5] support vector machine (SVM),[6] and isolation forest (IF)[7] in image/video anomaly detection tasks.

Autoencoders are the majority of deep anomaly detection methods.[8] These methods use autoencoders for reconstructing images and assume that normal data and anomalous data could result in significantly different latent embeddings. Thus differences in the corresponding reconstruction errors can be used to distinguish the two types of samples.[9] However, this assumption does not always hold in practice. It is observed that autoencoders are powerful enough to reconstruct both the normal data and anomalous data well, leading to poor performance for anomaly detection. In UAD, this problem becomes more severe because the

training data is contaminated with anomalies. During the training phase, the autoencoder is encouraged to reconstruct both the normal data and anomalous data well, making the reconstruction error of the two types of samples less separable in the testing phase. Overall, the problem of the autoencoder-based anomaly detection methods is that minimizing the reconstruction error does not necessarily mean maximizing anomaly detection performance. An autoencoder with lower reconstruction error often leads to poor anomaly detection performance. In Section 4.4.3, we empirically validate this by monitoring the anomaly detection performance during the training.

To address this problem, many improved autoencoder-based methods have been proposed. Xia et al.[10] propose a discriminative reconstruction-based autoencoder (DRAE), which makes the normal samples and anomalous samples more separable by injecting discriminate information in the training process of the autoencoder. Inspired by Robust PCA,[11] Zhou et al.[12] propose a robust autoencoder to split the training data into two parts, representing the normal data and anomalous data, respectively. Gong et al.[13] add a memory module to the vanilla autoencoder to memorize the normal data during the training phase. Reconstruction is based on the memorized normal data instead of the original inputs. Lai et al.[14] add a robust subspace recovery layer (RSR layer) to the vanilla autoencoder. The RSR layer is supposed to extract the underlying subspace from a latent representation of the training data and remove anomalies far away from this subspace.

Although many attempts have been made, the performance of anomaly detection remains low, especially when the training data is anomaly-contaminated, namely in UAD. All the above methods try to add a module to the vanilla autoencoder structure, and then detect anomalies using the reconstruction loss. As we mentioned before, minimizing reconstruction loss does not go straight with maximizing anomaly detection performance. We argue that reconstruction loss is not a good scoring strategy for anomaly detection. Therefore, we need to find a more reasonable scoring strategy.

Deep support vector data description (Deep SVDD)[15] is another deep anomaly detection method beyond autoencoder-based methods. It can trace back to classical shallow methods One-Class SVM (OC-SVM)[16] and SVDD.[17] Deep SVDD aims to train a neural network while minimizing the volume of a hypersphere that encloses the network representations of the data. In the ideal case, mappings of normal examples fall within, whereas mappings of anomalous examples fall outside the hypersphere. Deep SVDD then detects the anomalies based on the distance to the centroid of the hypersphere. The main drawback of Deep SVDD is the phenomenon of feature collapse. Pretrained autoencoders as network initialization cannot avoid this issue. During the training phase, the deep anomaly detection objective may misguide the feature learning and can not preserve the local structure of training data, leading to corruption of embedded space.

To deal with these issues, in this paper, we assume that both anomaly detection oriented loss guidance and the local structure preservation mechanism are essential for deep anomaly detection. Inspired by Ruff et al.,[15] we use autoencoders to learn embedded features and to preserve the local structure of data generating distribution. We propose to incorporate the SVDD loss into the autoencoder framework to detect anomalies in the feature space similar to Deep SVDD instead of reconstruction loss. In this way, the proposed framework can jointly perform anomaly detection optimization and learn representative features with the local structure preservation. We refer to our method as Improved AutoEncoder for Anomaly Detection (IAEAD). The optimization of IAEAD can directly perform mini-batch stochastic gradient descent and backpropagation. At last, extensive experiments are carefully designed

and conducted. The experimental results validate our assumption and the effectiveness of our IAEAD.

Compared with existing autoencoder-based methods, we summarize the contributions of our work as below:

- We propose a novel autoencoder-based method for anomaly detection, which can manipulate the feature space guided by anomaly detection-related loss for better performance. Our method can jointly perform anomaly detection and representative learning with the local structure preservation.
- Instead of using reconstruction error for anomaly scoring, we propose to detect anomaly in the feature space and to use the distance to the centroid in feature space as a new anomaly scoring strategy, which is more appropriate for anomaly detection.
- We perform extensive experiments on five image data sets to validate the effectiveness and stableness of our method compared to other state-of-the-art autoencoder-based methods.

The rest of this paper is organized as follows. Section 2 is the related work to our paper. Section 3 presents the proposed method. Section 4 shows the experimental results with evaluation. Section 5 concludes the paper.

2 | RELATED WORK

Our proposed method falls into the category of deep anomaly detection. In this section, we first introduce some necessary notations in our paper, then we review the existing deep anomaly detection methods and introduce some preliminaries (Table 1).

2.1 | Deep anomaly detection

Many deep anomaly detection methods have been proposed because of the rising of deep learning. These methods have achieved promising results. Existing deep UAD methods can be roughly divided into three categories: generative model-based methods, self-supervised based methods and deep one-class classification methods. We briefly introduce these three types of methods.

Generative model-based methods can be further divided into autoencoder-based methods and generative adversarial network-based methods according to the backbone network used. Our method is closely related to autoencoder-based methods, so we focus on autoencoder-based methods here. Autoencoders are the fundamental architectures used in UAD and have been extensively studied. These models use the autoencoder for reconstructing images and assume that normal data and anomalous data could lead to significantly different embeddings in feature space. Thus we can utilize differences in the corresponding reconstruction errors to detect anomalies. Sakurada et al.[9] incorporate an autoencoder for anomaly detection. They find that the latent embeddings in the hidden layer of autoencoders are distinguishable between normal data and anomalous data. Xia et al.[10] propose an early autoencoder-based model to tackle UAD problems. By injecting discriminate information in the training process of the autoencoder, the method they proposed could make the normal samples and anomalous samples more separable. Note the original DRAE uses deep features extracted from a seven-layer CNN pretrained on ImageNet instead of raw pixels as the input to the autoencoder.

TABLE 1 Main notations used in the paper

Notation	Meaning	Notation	Meaning
SAD	Supervised anomaly detection	n	The number of samples
SSAD	Semi-supervised anomaly detection	ρ	Anomaly ratio
UAD	Unsupervised anomaly detection	S	Anomaly score function
Deep SVDD	Deep support vector data description	x_i	The ith sample
OC-SVM	One-Class SVM	f_w	The encoder network with parameter w
CAE	Convolutional autoencoder	g_u	The decoder network with parameter u
DRAE	Discriminative reconstruction based autoencoder	C	Total class number
RDAE	Robust deep autoencoder	c	One specific class
DAGMM	Deep autoencoding gaussian mixture model	$\boldsymbol{c}$	Centroid of hypersphere in Deep SVDD
MemAE	Memory-augmented deep autoencoder	R	Radius of hypersphere in Deep SVDD
RSRAE	Robust subspace recovery based autoencoder	ν	Trade-off parameter of Deep SVDD
AUROC	Area under the receiver operating characteristic curve	λ	Trade-off parameter of our method
AUPR	Area under the precision-recall curve		

Inspired by Robust PCA,[11] Zhou et al.[12] propose a decoupled solution that decomposes the inputs into a low-rank part from normal data and a sparse part from anomalous data, which improves the robustness of the vanilla autoencoder by the splitting. Zong et al.[18] combine autoencoder and Gaussian mixture model to tackle UAD problems. Autoencoder generates a low-dimensional representation of the data and the Gaussian mixture model performs density estimation on the compact representation. Instead of using decoupled two-stage training and the standard expectation-maximization (EM) algorithm, they jointly and simultaneously optimize the parameters of the autoencoder and the mixture model. Gong et al.[13] add a memory module to the vanilla autoencoder. Instead of using the encoding of original inputs, They use the encoding to retrieve the most relevant memory items for construction. The memory module is encouraged to represent the prototypical elements of normal data during the training stage. During the testing stage, anomalous data will not be constructed well using the memorized normal data, thus can be separated from the normal data. Lai et al.[14] add a RSR layer to the vanilla autoencoder. The RSR layer is supposed to extract the underlying subspace from a latent representation of the training data and removes anomalies far away from this subspace.

Recently, self-supervised methods are showing promising results in representation learning. Several self-supervised-based methods for anomaly detection have been proposed and have made great achievements for image anomaly detection. Golan et al.[19] are the first to introduce self-supervised learning or transformation-based methods to anomaly detection. They use multiple carefully designed image geometric transformations and create a self-labeled auxiliary data set for

transformation classification pretask. The assumption is that the pretask model could not perform equally well on the classification of transformations of anomalous data. Wang et al.[20] propose a similar self-supervised method to solve UAD. They introduce more self-label methods like patch rearranging and irregular affine transformations to generate the auxiliary data set. Furthermore, they propose several scoring strategies for self-supervised based anomaly detection methods. Although self-supervised based methods show promising results on anomaly detection tasks, their performance is heavily dependent on the interaction between transformations and the benchmark data set. For rotation transformation, the performance would drop significantly when the images of normal data are captured from multiple orientations.[21]

One class classification, which can also be framed as SSAD, is a classical machine learning problem. Deep SVDD[15] is the first introduced deep one-class classification method. It can trace back to classical shallow methods OC-SVM[16] and SVDD.[17] Deep SVDD aims to train a neural network while minimizing the volume of a hypersphere that encloses the network representations of the data. The main drawback of Deep SVDD is the phenomenon of feature collapse.

The proposed method intrinsically is an improved version of vanilla CAE with incorporating the objective of Deep SVDD. It excels Deep SVDD by simplicity without pretraining and outperforms both CAE and Deep SVDD regarding anomaly detection performance by all metrics. Since our method mainly depends on autoencoder and Deep SVDD, we introduce them in more detail in the following sections.

2.2 | Vanilla autoencoder for anomaly detection

An autoencoder is a feed-forward neural network that is trained to reconstruct its input to its output. Internally, an autoencoder has a hidden layer that describes a code, called the feature, a nonlinear representation of the input data. The neural network consists of two parts: an encoder function $f_w(\cdot)$ and a decoder function $g_u(\cdot)$. It learns a code from the input through a pair of encoding and decoding phases.

$$\hat{x} = g_u(f_w(x)), \tag{1}$$

where x is the input data, $\hat{x}$ is the reconstructed version of the input data. The key idea is to train $f_w(\cdot)$ and $g_u(\cdot)$ to minimize the difference between x and $\hat{x}$.

$$\min_{u,w} \|x - g_u(f_w(x))\|, \tag{2}$$

where $\|\cdot\|$ is commonly designed to be the ℓ_2-norm.

We can use an arbitrary structure for both $f_w(\cdot)$ and $g_u(\cdot)$. By allowing many layers of network for encoder and decoder, we can get a deep autoencoder capable of effectively representing complex input data.

Autoencoder-based anomaly detection assumes that anomalies are corresponding to larger reconstruction losses and normal data corresponding to smaller reconstruction losses. Therefore, in the vanilla autoencoder for anomaly detection, or most autoencoder-based anomaly detection, reconstruction loss is often chosen to be the anomaly score. Given an input data x_i, when the training of autoencoder is finished on the training set, the anomaly score of x_i can be denoted as $S(x_i)$

$$S(x_i) = \|x_i - g_u(f_w(x_i))\|, \tag{3}$$

where $\|\cdot\|$ is commonly designed to be the ℓ_2-norm or ℓ_1-norm.

In our methods, the vanilla autoencoder is used as the base network. Later we add the Deep SVDD loss term to the reconstruction loss for optimization.

2.3 | Deep SVDD for anomaly detection

Deep SVDD[15] starts with pretraining an autoencoder for the network initialization. After initialization, the network is trained using the following anomaly detection objective:

$$\min_{R,\mathcal{W}} R^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\max\{0, \|\phi(\boldsymbol{x}_i;\mathcal{W}) - \boldsymbol{c}\|^2 - R^2\} + \frac{\lambda}{2}\sum_{\ell=1}^{L}\|\boldsymbol{W}^{\ell}\|_F^2, \tag{4}$$

where ϕ denotes the neural network with $L \in \mathbb{N}$ hidden layers and set of weights $\mathcal{W} = \{\mathbf{W}^1, \mathbf{W}^2, ..., \mathbf{W}^L\}$, $\mathbf{W}^{\ell}$ denotes the weights of layer $\ell \in \{1, 2, ..., L\}$, x_i is one of the samples in the training set, n denotes the total number of samples in the training set, $R > 0$ denotes the radius of the hypersphere, $\boldsymbol{c}$ is the desired centroid of all training data in the feature space, $\nu \in (0, 1]$ is a hyperparameter.

In the objective above, minimizing R corresponds to minimizing the volume of the hypersphere. The second term is a penalty term for data points mapping outside the hypersphere, that is, if its distance to the centroid $\|\phi(\boldsymbol{x}_i;\mathcal{W}) - \boldsymbol{c}\|$ is greater than radius R. Hyperparameter ν balances the volume of the hypersphere and violations of the boundary.

By minimizing the objective, Deep SVDD aims to find a data-enclosing hypersphere of the smallest size. Most representations of the normal data will lie in the hypersphere, while most of the anomalous data will lie outside the hypersphere. The learned hypersphere is characterized by centroid $\boldsymbol{c}$ and radius R. After the training is finishes, the anomaly score can be calculated by the distance to $\boldsymbol{c}$. The anomaly score of a given input data x_i can be denoted as follows:

$$S(x_i) = \|\phi(\boldsymbol{x}_i;\mathcal{W}) - \boldsymbol{c}\|, \tag{5}$$

where $\|\cdot\|$ is commonly designed to be the Euclidean distance.

The most significant contribution of Deep SVDD is the anomaly detection objective, which traces back to traditional OC-SVM and SVDD. It works by manipulating the feature space, making it suitable for anomaly detection. However, there is no guarantee of manipulating features without causing the feature distortion after the pretraining finished. We deal with this by explicitly preserving the local structure of training data and jointly optimizing the autoencoder using both the anomaly detection loss and reconstruction loss.

3 | IMPROVED AUTOENCODER FOR UAD

We first formulate the problem of UAD. Consider a data space $\mathcal{X}$ (in this context, the space of images), an unlabeled data collection $X = \{x_i \in \mathbb{R}^{C\times H\times W}\}_{i=1}^{n} \subseteq \mathcal{X}$, where N denotes the total number of samples in X, C, H, and W denotes the dimensions of image channels, height, and width. X consists of a normal data set X_{no} and an anomaly set X_{an}, which are sampled from different distributions.[22] The goal of UAD is to build a model $M(\cdot)$ for deciding whether a given sample is from X_{no} or X_{an}, without any labels.

Our method is based on the autoencoder, define encoder as $f_w: x_i \to z_i$ and decoder as $g_u: z_i \to \hat{x}$, where z_i is the embedded point of x_i in the feature space $\mathcal{F}$ and $\hat{x}_i$ is the reconstructed input x_i. Instead of using reconstruction loss as anomaly scores, we propose to

detect anomalies in the feature space. Therefore, we aim to find a good f_w that makes embedded points $\{z_i\}_{i=1}^n$ more suitable for anomaly detection tasks. To this end, the two components introduced in Section 2 are essential: the autoencoder and Deep SVDD loss. Autoencoder is a classical method for learning representations in an unsupervised manner, which can preserve intrinsic local structure in data. The Deep SVDD loss, borrowed from Ruff,[15] is responsible for manipulating the feature space for the following anomaly detection task. We add the Deep SVDD loss as a regularization to the original autoencoder optimization objective. The whole network structure is illustrated in Figure 1. We define the objective of our method as min L, L is the summary of reconstruction error and Deep SVDD loss as below

$$L = L_{rec} + \lambda L_{reg}, \tag{6}$$

where L_{rec} and L_{reg} are reconstruction loss and Deep SVDD loss, respectively, $\lambda > 0$ is the trade-off coefficient that controls the degree of distorting feature space. When $\lambda = 0$, Equation (8) reduces to the objective of CAE.

The aim of Deep SVDD loss L_{reg} is to learn a minimum hypersphere characterized by radius $R > 0$ and centroid $\boldsymbol{c} \in \mathcal{F}$ in the feature space, and to map most of the normal data inside the hypersphere and most of the anomalous data outside the hypersphere at the same time. We define the Deep SVDD loss as below

$$L_{reg} = R^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\max\{0, \|f_w(\boldsymbol{x}_i) - \boldsymbol{c}\|^2 - R^2\}, \tag{7}$$

where ν is hyperparameter balances the volume of the hypersphere and violations of the boundary. Details can be found in Section 2.3 and the original Deep SVDD paper.[15]

The overall objective of our method can be written as follows:

$$\min_{u,w,R} \|x_i - g_u(f_w(x_i))\| + \lambda\left\{R^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\max\{0, \|f_w(x_i) - \boldsymbol{c}\|^2 - R^2\}\right\}, \tag{8}$$

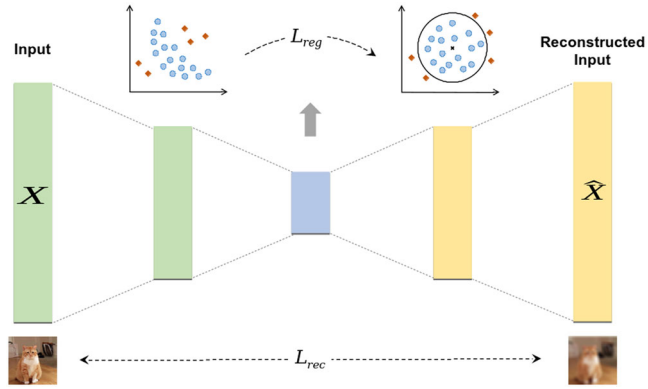

FIGURE 1 The network structure of IAEAD. The network is trained to minimize both the reconstruction loss L_{rec} and the Deep SVDD loss L_{reg}. Reconstruction loss is used to learn representations of original data with local structure preserved. Deep SVDD loss is used to manipulate the feature space for the following anomaly detection. SVDD, support vector data description. IAEAD, improved AutoEncoder for anomaly detection [Color figure can be viewed at wileyonlinelibrary.com]

where w is the parameter of encoder f_w, u is the parameter of decoder g_u, λ is the trade-off hyperparameter to balance the reconstruction loss and Deep SVDD loss.

Unlike most autoencoder-based anomaly detection methods, we do not use the reconstruction loss as anomaly scores. Instead, we propose to calculate the anomaly score in the feature space. Different from Deep SVDD, which uses the distance of embedded points to the centroid $\boldsymbol{c}$, as shown in Equation (5), we empirically find that using the distance of embedded points to the mean center of all embedded points performs slightly better. The anomaly score can be calculated as follows:

$$S(x_i) = \left\| f_w(x_i) - \sum_{i=1}^{n} f_w(x_i) \right\|, \tag{9}$$

where $\|\cdot\|$ is designed to be the Euclidean distance in this paper.

4 | EXPERIMENTS

In this section, we introduce the data sets used in this paper and the experimental setting of UAD tasks. Then, we extensively evaluate the effectiveness of our approach and compare it with other state-of-the-art autoencoder-based methods. We also conduct an ablation study to explore the effect of each part of our framework. Our experiment code is available at https://github.com/wogong/pt-iaead.

4.1 | Compared methods

We compare our method to state-of-the-art autoencoder-based deep learning approaches.

Convolutional AutoEncoder (CAE).[23] The vanilla CAE serves as a baseline for autoencoder-based UAD methods.

Deep SVDD[15] is a deep end-to-end version of the classic SVDD.[17] We use the officially released code of Deep SVDD* to conduct related experiments on all benchmark data sets. Deep SVDD is designed for SSAD, so we update the data set input code based on the published version. Our altered version of Deep SVDD is available at https://github.com/wogong/Deep-SVDD-PyTorch.

DRAE[10] is an early autoencoder-based model to tackle the UAD problem. DRAE makes the normal samples and anomalous samples more separable by injecting discriminate information in the training process of the autoencoder. However, the original DRAE uses deep features extracted from a seven-layer CNN pretrained on ImageNet instead of raw pixels as the input to the autoencoder. We use raw pixels as input for DRAE in our experiments to make the comparison fair.

Robust Deep AutoEncoder (RDAE)[12] is inspired by Robust PCA[11] and splits the input data X into two parts, $X = L_D + S$, where L_D represents the normal data and can be effectively reconstructed by autoencoder, and S contains the anomalies and noise in the original data X. RDAE improves the robustness of the vanilla autoencoder by the splitting.

Deep Autoencoding Gaussian Mixture Model (DAGMM)[18] combines autoencoder and Gaussian mixture model to tackle UAD problems. Autoencoder generates a low-dimensional representation of the data, and the Gaussian mixture model performs density estimation on the compact representation. Instead of using decoupled two-stage training and the standard

Expectation-Maximization (EM) algorithm, DAGMM jointly and simultaneously optimizes the parameters of the autoencoder and the mixture model.

Memory-augmented deep AutoEncoder (MemAE)[13] adds a memory module to the vanilla autoencoder. Instead of using the encoding of original inputs, MemAE uses the encoding to retrieve the most relevant memory items for construction. The memory module is encouraged to represent the prototypical elements of normal data during the training stage. During the testing stage, anomalous data will not be constructed well using the memorized normal data, thus can be separable from the normal data. We use the same autoencoder structure reported in the original paper for conducting experiments. We refer to the officially released code[†] for network structure details.

RSR-based AutoEncoder (RSRAE)[14] adds a RSR layer to vanilla autoencoder. The RSR layer is supposed to extract the underlying subspace from a latent representation of the training data and removes anomalies far away from this subspace.

4.2 | Data sets

We evaluate the proposed approach on five public image data sets, which are widely used in anomaly detection literature. We note that in all our experiments, raw pixels are directly used as inputs with their pixel value scaled to reside in range $[-1, 1]$. Following are brief introductions of the five benchmark data sets.

MNIST[24] is a well-known digit recognition data set consisting of 70,000 handwritten grayscale digit images with each in size of 28×28. MNIST has a training set of 60,000 examples and a test set of 10,000 examples.

Fashion-MNIST[25] is a data set of Zalando's article images, consisting of a training set of 60,000 examples and a test set of 10,000 examples. Fashion-MNIST is a more challenging data set compared to MNIST. Each example is a 28×28 grayscale image associated with a label from 10 classes.

SVHN[26] is a real-world digit image data set obtained from house numbers in Google Street View images, consisting of over 600,000 digit images. SVHN can be seen as similar in flavor to MNIST, but comes from a significantly harder, unsolved, real-world problem (recognizing digits and numbers in natural scene images). In this paper, we only use the training set consisting of 73,257 digits.

CIFAR-10[27] is a natural image data set. The objects in images come from objects in daily life. It consists of 60,000 color images in the size of 32×32, with 6000 images per class.

CIFAR-100[27] is basically the same as the CIFAR-10, except it has 100 classes containing 600 images in each class. There are 500 training images and 100 testing images per class. The 100 classes in the CIFAR-100 are grouped into 20 superclasses. Each image comes with a "fine" label (the class to which it belongs) and a "coarse" label (the superclass to which it belongs).

4.3 | Experimental protocol

To construct an image set with anomaly examples for UAD, we follow the standard procedure one-vs-all evaluation scheme from the previous literature[10,12,20,28,29]: Given a standard image benchmark data set with C classes, from which we create C different experiments. For each $1 \leqslant c \leqslant C$, all images from class c are retrieved as normal examples, while anomalies are

randomly sampled from the rest of the classes by an anomaly ratio ρ. The assigned labels indicating whether an example is normal or anomalous are unknown to anomaly detection methods and only used for final evaluation. We shift ρ from 5% to 25% by a stage of 5% to construct anomaly detection tasks with different anomaly ratios. We use each class of a benchmark data set as normal examples in turn and report the overall UAD performance as the average performance on all classes. As for performance metric, we adopt the commonly used area under the receiver operating characteristic curve (AUROC) and area under the precision-recall curve (AUPR) as threshold-independent metrics.[30] Every experiment is repeated five times to report the average results and standard deviation.

Hyperparameters and Optimization Methods. We use the same CAE architecture from Reference [19] with a four-block encoder and four-block decoder. For the encoder, every block consists of a convolution layer, a batch normalization layer, and an activation layer. For the decoder, every block consists of a deconvolution layer, a batch normalization layer, and an activation layer. We adopt the same architecture for all benchmark data sets.

The stochastic gradient descent (SGD) optimizer with default settings in PyTorch[31] is used for optimizing the autoencoder for all five data sets. We set the initial learning rate to 0.1 and drop the learning rate by half every 50 epochs. The batch size for all methods is set to 256. The number of epochs is set to 250 on all benchmark models, except for the training of Deep SVDD, of which the epoch is set to the same value as the released code by authors.

As for the hyperparameter ν of Deep SVDD loss, the original paper[15] chooses ν from $\nu \in \{0.01, 0.1\}$ and reports the best results. To avoid the introduction of supervised information, we fix $\nu = 1$ across all experiments in this paper. We delay the incorporating of Deep SVDD loss learning by 20 epochs' warm-up. The centroid c is initialized as the mean of the network representation right after the warm-up finished. For radius R, we solve for R via line search every mini-batch, which is the same strategy as the original paper.[15]

For CAE, DRAE, RDAE, DAGMM, and RSRAE, we use the same CAE architecture as our method for a fair comparison. We do not use more complex CAE (e.g., CAE using skip connection or more layers) since they usually lower anomalies' reconstruction error but do not contribute to CAE's UAD performance.[20] For CAE, we use the SGD optimizer, which is the same as our method. For DRAE, RDAE, DAGMM, and RSRAE, we use the code released by Wang et al.,[20,‡] which uses Adam optimizer.

4.4 | Experiment results and analysis

4.4.1 | UAD performance

For each benchmark with C classes, we can construct C UAD tasks. The performance corresponding to the benchmark is calculated as the average performance on the C constructed tasks.

The experiment results of different data sets under anomaly ratio 10% and 20% are shown in Table 2. We report the UAD performance by AUROC and AUPR as mentioned above. In the table, we use AUPR-in and AUPR-out to denote the AUPR calculated when normal samples and anomalous samples are used as positive classes, respectively. It can be seen from the table that our method achieves the best results under all metrics on MNIST, Fashion-MNIST. Actually, it significantly outperforms compared methods by a large margin (10% under AUROC). As for SVHN, CIFAR10, and CIFAR100, our method achieves comparable or slightly better

TABLE 2 AUROC/AUPR-IN/AUPR-OUT (%) for UAD methods

Data set	ρ	CAE[23]	DRAE[10]	RDAE[12]	DAGMM[18]	MemAE[13]	RSRAE[14]	Deep SVDD[15]	Ours
AUROC (%)									
MNIST	10%	56.56 ± 1.72	67.92 ± 1.43	71.76 ± 0.99	63.23 ± 2.15	69.13 ± 0.15	82.08 ± 3.09	78.86 ± 0.65	**92.05 ± 0.31**
	20%	55.82 ± 1.87	64.74 ± 2.62	67.00 ± 0.69	65.88 ± 2.89	65.56 ± 0.26	80.37 ± 2.04	74.03 ± 0.83	**87.57 ± 0.07**
F-MNIST	10%	67.43 ± 0.50	65.96 ± 1.72	75.34 ± 1.19	70.36 ± 3.18	72.10 ± 0.31	75.19 ± 2.78	79.79 ± 0.41	**88.69 ± 0.19**
	20%	64.64 ± 0.75	63.63 ± 1.07	70.94 ± 1.13	66.00 ± 4.96	67.88 ± 0.09	72.15 ± 2.55	74.10 ± 0.59	**84.70 ± 0.11**
SVHN	10%	52.23 ± 0.12	51.10 ± 0.27	51.44 ± 0.27	50.09 ± 0.29	**58.29 ± 0.12**	51.19 ± 0.44	55.05 ± 0.43	54.07 ± 0.23
	20%	51.76 ± 0.02	50.88 ± 0.18	51.67 ± 0.22	49.99 ± 0.24	**56.55 ± 0.03**	51.51 ± 0.40	53.91 ± 0.37	53.76 ± 0.10
CIFAR-10	10%	55.21 ± 0.36	56.09 ± 0.21	54.02 ± 1.63	54.05 ± 0.92	55.69 ± 0.17	54.77 ± 1.84	55.70 ± 0.53	**56.65 ± 0.39**
	20%	54.30 ± 0.05	55.60 ± 0.09	52.46 ± 1.54	54.68 ± 0.70	54.68 ± 0.09	53.62 ± 1.98	54.92 ± 0.26	**55.61 ± 0.05**
CIFAR-100	10%	56.36 ± 0.23	55.66 ± 0.49	53.62 ± 0.40	54.19 ± 1.37	55.03 ± 0.04	53.56 ± 0.48	55.85 ± 0.31	**56.77 ± 0.18**
	20%	55.18 ± 0.56	55.25 ± 0.30	52.86 ± 1.43	53.80 ± 0.61	54.30 ± 0.11	53.54 ± 1.13	54.54 ± 0.21	**55.68 ± 0.17**
AUPR-IN (%)									
MNIST	10%	89.10 ± 0.36	92.81 ± 0.37	89.11 ± 1.28	92.89 ± 0.77	93.13 ± 0.00	96.55 ± 0.75	96.54 ± 0.17	**99.00 ± 0.04**
	20%	79.32 ± 0.60	84.92 ± 1.00	75.56 ± 2.58	86.42 ± 1.04	84.22 ± 0.11	92.15 ± 1.02	90.87 ± 0.44	**95.28 ± 0.07**
F-MNIST	10%	93.72 ± 0.05	93.51 ± 0.47	83.25 ± 1.22	92.70 ± 2.92	95.54 ± 0.08	95.29 ± 0.52	96.74 ± 0.06	**98.39 ± 0.02**
	20%	86.05 ± 0.36	85.87 ± 0.52	72.83 ± 2.64	86.66 ± 2.70	88.64 ± 0.07	89.74 ± 0.93	90.94 ± 0.31	**95.28 ± 0.07**
SVHN	10%	90.72 ± 0.03	90.48 ± 0.11	90.34 ± 0.08	90.00 ± 0.14	**92.48 ± 0.04**	90.37 ± 0.08	91.38 ± 0.12	91.33 ± 0.10
	20%	80.96 ± 0.03	80.43 ± 0.11	80.78 ± 0.17	79.90 ± 0.09	**83.64 ± 0.03**	80.47 ± 0.16	81.95 ± 0.20	82.16 ± 0.05
CIFAR-10	10%	90.58 ± 0.11	90.81 ± 0.11	90.59 ± 0.51	**91.28 ± 0.57**	90.73 ± 0.04	90.89 ± 0.42	90.64 ± 0.17	90.92 ± 0.11
	20%	81.07 ± 0.04	81.68 ± 0.07	80.74 ± 0.82	**81.76 ± 0.37**	81.26 ± 0.03	81.05 ± 0.73	81.13 ± 0.25	81.67 ± 0.08
CIFAR-100	10%	91.16 ± 0.03	90.93 ± 0.15	90.47 ± 0.13	91.12 ± 0.23	90.95 ± 0.01	90.72 ± 0.21	91.00 ± 0.19	**91.33 ± 0.05**
	20%	81.82 ± 0.20	81.70 ± 0.10	80.69 ± 0.78	81.45 ± 0.42	81.60 ± 0.06	81.39 ± 0.71	81.43 ± 0.16	**82.12 ± 0.04**

TABLE 2 (Continued)

Data set	ρ	CAE[23]	DRAE[10]	RDAE[12]	DAGMM[18]	MemAE[13]	RSRAE[14]	Deep SVDD[15]	Ours
AUPR-OUT (%)									
MNIST	10%	21.31 ± 2.08	33.03 ± 0.94	35.81 ± 0.77	20.63 ± 4.42	26.80 ± 0.02	46.84 ± 2.72	36.19 ± 1.53	**59.31 ± 1.48**
	20%	32.64 ± 2.40	39.89 ± 3.64	43.23 ± 0.75	33.48 ± 5.21	38.35 ± 0.29	55.62 ± 3.02	44.48 ± 1.03	**63.33 ± 0.11**
F-MNIST	10%	24.44 ± 1.50	24.16 ± 2.10	31.72 ± 1.39	35.44 ± 2.61	23.13 ± 0.20	33.91 ± 4.09	36.71 ± 1.06	**52.76 ± 0.97**
	20%	35.23 ± 1.06	34.21 ± 0.91	41.40 ± 0.88	42.04 ± 4.45	34.91 ± 0.08	40.41 ± 4.62	43.69 ± 0.60	**59.16 ± 0.20**
SVHN	10%	10.70 ± 0.04	10.54 ± 0.14	10.45 ± 0.14	**19.79 ± 2.16**	12.61 ± 0.08	10.45 ± 0.15	11.96 ± 0.12	11.26 ± 0.08
	20%	21.07 ± 0.00	20.70 ± 0.13	21.09 ± 0.05	**30.74 ± 1.66**	23.41 ± 0.01	21.29 ± 0.26	22.55 ± 0.18	21.98 ± 0.12
CIFAR-10	10%	14.12 ± 0.14	14.61 ± 0.16	13.03 ± 0.57	13.69 ± 0.28	13.98 ± 0.06	13.04 ± 0.92	13.91 ± 0.16	**14.15 ± 0.14**
	20%	25.50 ± 0.15	26.66 ± 0.09	23.38 ± 1.22	25.62 ± 1.02	25.66 ± 0.12	24.53 ± 1.45	25.39 ± 0.28	**25.68 ± 0.09**
CIFAR-100	10%	15.18 ± 0.21	14.75 ± 0.14	13.89 ± 0.43	13.62 ± 0.67	14.15 ± 0.09	12.36 ± 0.16	13.89 ± 0.06	**15.26 ± 0.22**
	20%	26.41 ± 0.64	26.60 ± 0.38	24.33 ± 0.66	24.26 ± 0.51	25.08 ± 0.08	23.70 ± 0.84	24.70 ± 0.01	**26.51 ± 0.16**

Note: The best performance is in bold.

performance on all metrics than compared methods. For SVHN, our method is not as good as MemAE under AUROC, but still outperforms CAE and Deep SVDD, which are the base of our method. We attribute the performance improvement compared to CAE as the introduction of Deep SVDD loss and the new scoring strategy. Because minimizing reconstruction errors only does not necessarily mean maximizing anomaly detection performance. Thus using reconstruction error as the anomaly score is not that reasonable. The performance of Deep SVDD suffers from feature collapse, while our method combines both Deep SVDD loss and reconstruction error. It can preserve the local structure of training data and learning meaningful features and results in better performance compared with Deep SVDD.

To show the comparison between different methods more clearly and to compare UAD performance under different anomaly ratios, we report the UAD performance on all five data sets by AUROC under ρ from 5% to 25% as a line graph in Figure 2. From the figure we can see that all methods perform worse when the anomaly ratio becomes high, which is quite intuitive for a higher anomaly ratio means more noise in training, corresponding to a harder UAD task. We also note that when the performance is low, this trend is not stable for some methods, such as the line of RDAE on CIFAR10. From the figure we conclude that the trend does not hold when the method is just a little better than random guessing (corresponding to 50% AUROC). We can also see that the line of our method is consistently above the lines of other methods,

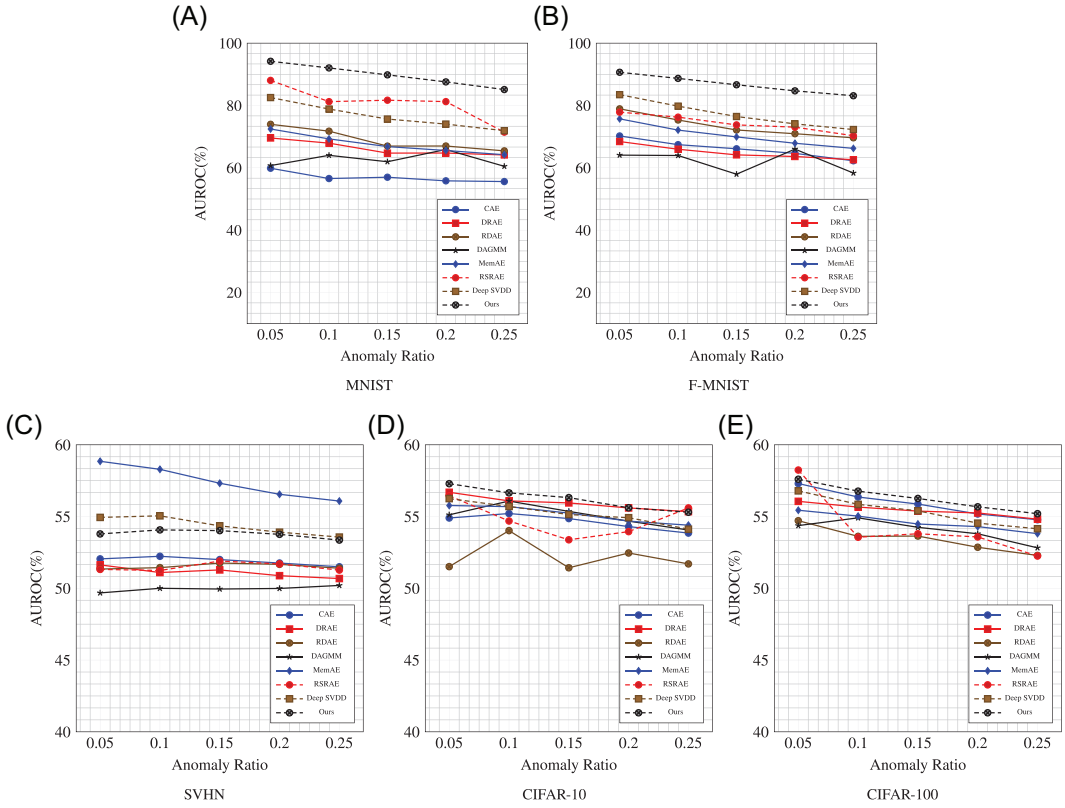

FIGURE 2 Unsupervised anomaly detection performance (AUROC) comparison with varying ρ from 5% to 25%. AUROC, area under the receiver operating characteristic curve [Color figure can be viewed at wileyonlinelibrary.com]

except for SVHN, on which our method performs slightly worse than MemAE, but still better than CAE and Deep SVDD.

To compare the performance for each image class, we also report the AUROC results for each class of the five benchmark data sets in Tables 3 and 4. As the tables show, on each image class of MNIST and Fashion-MNIST, we obtain consistently better performances, showing the effectiveness of our method. On more complex data sets, like CIFAR10 and CIFAR100, we have different performance comparison results on different classes due to the large interclass variance. Overall, our method achieves slightly better or comparable performance on these complex data sets.

4.4.2 | Model stability

The model stability of UAD methods is essential because validation during the training phase is impossible in the unsupervised scenario. There is no way to obtain the best checkpoint for an UAD model without validation. A stable model can ensure the performance of the final model is not unacceptable. The stability of model performance is mainly reflected in three aspects[32]: (1) Whether the model can reach convergence after acceptable training epochs in one training attempt. (2) Whether the model can reach a stable performance level in multiple training attempts using the same training configuration. (3) Whether the model can achieve good performance stably in various data sets and training configurations.

To assess the stability of our method, we monitor the UAD performance by AUROC of our method and the CAE during the training phase. Figure 3 shows the AUROC in different training epochs. In general, the UAD performance of our method improves at the initial stage of training and then stabilizes as the training epochs continue to increase. As a comparison, the AUROC-epoch curve of the CAE fluctuates much more during the training, sometimes even cannot reach convergence, such as the experiment on number 0 of MNIST, where the AUROC keeps decreasing during training. This validates that minimizing the reconstruction error does not necessarily mean maximizing the performance of anomaly detection, we mentioned in Section 1. In fact, lower reconstruction error leads to lower AUROC in Figure 3A.

Thus, through our method, we can achieve a highly reliable model through acceptable training epochs in UAD tasks without validation.

4.4.3 | Hyperparameter discussion

In this section, we discuss the effect of coefficient λ of Deep SVDD loss in Equation (8). We conduct experiments on the MNIST data set by sampling λ in range $[5e-6, 5e-2]$. The range is designed to balance the value of reconstruction loss and Deep SVDD loss, ensuring that the difference between these two values is not too significant in order of magnitude. The optimizer is set to the same as other experiments. As shown in Figure 4, we have the following observations.

Our method achieves the best performance at $\lambda \approx 5e-5$ when $\lambda \in [0, 0.05]$. When $\lambda = 0$, our method reduces to vanilla autoencoder for anomaly detection with the anomaly scoring strategy we proposed. From Figure 4 we can see that simply changing the anomaly scoring strategy will boost CAE's performance. This validates the effectiveness of detecting anomalies in the feature space.

When λ is too large, which means the weight of Deep SVDD loss is high, the performance decreases. Because λ with a too-large value tends to distort latent feature space, which will

TABLE 3 AUROC (%) for UAD methods when $\rho = 0.1$

Data set	Class name	CAE	DRAE	RDAE	DAGMM	MemAE	RSRAE	Deep SVDD	Ours
MNIST	0	35.04 ± 5.78	52.92 ± 8.61	62.51 ± 5.83	42.85 ± 6.39	61.50 ± 0.91	85.18 ± 4.12	81.92 ± 1.68	96.59 ± 0.54
	1	97.21 ± 1.01	96.99 ± 0.75	99.19 ± 0.08	84.89 ± 1.88	97.80 ± 0.24	97.50 ± 1.32	95.29 ± 0.17	99.32 ± 0.07
	2	45.55 ± 2.61	54.70 ± 6.80	61.12 ± 1.81	60.34 ± 2.12	57.23 ± 0.72	74.10 ± 3.60	69.23 ± 3.04	83.27 ± 0.73
	3	50.90 ± 1.98	62.45 ± 5.03	65.91 ± 1.84	61.38 ± 3.38	62.66 ± 0.95	60.84 ± 7.65	75.27 ± 1.31	91.05 ± 0.44
	4	62.53 ± 4.20	75.59 ± 3.36	76.05 ± 1.20	59.13 ± 4.29	71.91 ± 0.01	82.68 ± 3.85	72.69 ± 2.55	91.94 ± 0.59
	5	52.93 ± 3.17	56.48 ± 7.36	65.11 ± 2.43	55.44 ± 1.79	59.12 ± 0.25	78.30 ± 3.44	71.03 ± 1.86	86.88 ± 0.69
	6	52.84 ± 8.83	68.86 ± 6.94	75.03 ± 1.94	64.90 ± 7.18	72.21 ± 0.76	91.31 ± 2.81	84.60 ± 0.58	95.69 ± 0.28
	7	75.41 ± 2.71	84.39 ± 1.86	85.64 ± 1.11	74.55 ± 5.90	80.56 ± 1.33	90.57 ± 2.01	79.41 ± 2.33	94.25 ± 0.44
	8	32.25 ± 7.00	53.08 ± 1.10	51.38 ± 2.30	59.72 ± 6.97	54.12 ± 0.00	74.30 ± 1.82	74.51 ± 2.49	87.04 ± 0.45
	9	60.91 ± 6.87	73.77 ± 4.09	75.71 ± 2.60	69.08 ± 2.79	74.16 ± 0.27	85.99 ± 4.14	84.67 ± 1.24	94.50 ± 0.21
	average	56.56 ± 1.72	67.92 ± 1.43	71.76 ± 0.99	63.23 ± 2.15	69.13 ± 0.15	82.08 ± 3.09	78.86 ± 0.65	**92.05 ± 0.31**
F-MNIST	t-shirt	62.67 ± 3.85	59.47 ± 4.55	70.56 ± 4.18	72.98 ± 8.47	71.30 ± 0.63	81.09 ± 2.64	78.03 ± 0.62	89.67 ± 0.36
	trouser	94.81 ± 0.25	79.04 ± 4.95	96.94 ± 0.36	77.49 ± 8.31	95.21 ± 0.05	81.83 ± 5.37	92.53 ± 0.95	97.53 ± 0.31
	pullover	58.08 ± 1.64	57.75 ± 4.54	68.79 ± 4.73	55.61 ± 2.66	72.71 ± 0.55	70.36 ± 3.11	75.89 ± 0.59	86.34 ± 0.96
	dress	73.56 ± 2.06	72.38 ± 5.68	79.12 ± 3.22	71.48 ± 5.28	75.96 ± 0.40	83.01 ± 3.96	84.70 ± 0.13	93.00 ± 0.41
	coat	62.40 ± 0.29	61.88 ± 3.41	73.23 ± 1.59	50.69 ± 7.61	74.12 ± 1.23	69.03 ± 8.06	79.76 ± 1.57	90.75 ± 0.34
	sandal	73.87 ± 3.57	72.94 ± 2.86	73.64 ± 1.33	90.52 ± 1.37	60.29 ± 1.73	77.20 ± 7.96	73.99 ± 1.93	75.37 ± 0.70
	shirt	47.75 ± 3.67	54.36 ± 2.41	66.34 ± 3.00	54.03 ± 2.18	63.95 ± 0.15	62.81 ± 7.53	71.48 ± 1.00	82.51 ± 0.66
	sneaker	85.41 ± 3.63	86.26 ± 4.30	91.15 ± 1.14	82.36 ± 5.20	82.08 ± 0.58	90.50 ± 2.51	90.98 ± 0.90	97.39 ± 0.31
	bag	44.74 ± 1.80	44.26 ± 1.38	51.25 ± 2.93	60.09 ± 0.64	50.66 ± 0.84	61.60 ± 9.05	68.51 ± 0.89	79.07 ± 0.77
	ankle-boot	71.01 ± 3.33	71.29 ± 3.47	82.35 ± 5.15	88.36 ± 1.93	74.71 ± 0.48	74.42 ± 5.07	82.04 ± 0.85	95.30 ± 0.44
	average	67.43 ± 0.50	65.96 ± 1.72	75.34 ± 1.19	70.36 ± 3.18	72.10 ± 0.31	75.19 ± 2.78	79.79 ± 0.41	**88.69 ± 0.19**

TABLE 3 (Continued)

Data set	Class name	CAE	DRAE	RDAE	DAGMM	MemAE	RSRAE	Deep SVDD	Ours
SVHN	0	50.44 ± 0.23	49.42 ± 1.12	50.07 ± 0.98	49.15 ± 0.69	58.27 ± 0.29	50.99 ± 1.20	56.61 ± 0.28	54.26 ± 1.54
	1	58.95 ± 0.27	56.13 ± 1.29	57.16 ± 2.51	49.54 ± 1.25	63.83 ± 0.12	53.10 ± 0.61	57.79 ± 1.14	55.22 ± 0.69
	2	52.45 ± 0.22	52.13 ± 0.91	51.21 ± 1.22	48.41 ± 0.49	58.09 ± 0.09	50.91 ± 1.23	56.12 ± 1.53	54.75 ± 0.43
	3	50.09 ± 0.25	50.57 ± 0.57	49.73 ± 0.62	50.94 ± 0.30	55.91 ± 0.44	52.12 ± 0.97	54.56 ± 0.39	53.36 ± 0.20
	4	54.40 ± 0.31	53.44 ± 1.59	53.99 ± 1.23	50.61 ± 1.02	60.28 ± 0.33	51.74 ± 1.10	54.49 ± 1.75	55.20 ± 0.43
	5	48.26 ± 0.21	49.87 ± 1.08	49.83 ± 1.64	51.93 ± 0.75	54.89 ± 0.10	49.75 ± 0.73	54.00 ± 0.97	53.73 ± 0.26
	6	50.62 ± 0.07	49.77 ± 0.84	49.80 ± 0.71	49.90 ± 0.89	56.00 ± 0.25	50.06 ± 1.68	53.97 ± 0.19	52.53 ± 0.20
	7	55.82 ± 0.27	50.46 ± 1.03	53.69 ± 0.80	48.36 ± 0.84	60.79 ± 0.37	52.33 ± 1.82	55.30 ± 0.79	53.48 ± 0.71
	8	50.46 ± 0.18	49.33 ± 1.15	49.27 ± 0.47	51.08 ± 0.84	56.55 ± 0.33	50.91 ± 0.83	54.75 ± 0.16	54.42 ± 0.62
	9	50.84 ± 0.13	49.90 ± 1.19	49.67 ± 0.95	50.98 ± 0.80	58.33 ± 0.24	50.00 ± 2.18	52.91 ± 0.63	53.77 ± 0.46
	Average	52.23 ± 0.12	51.10 ± 0.27	51.44 ± 0.27	50.09 ± 0.29	**58.29 ± 0.12**	51.19 ± 0.44	55.05 ± 0.43	54.07 ± 0.23

Note: The best performance is in bold.

TABLE 4 AUROC (%) for UAD methods when $\rho = 0.1$

Data set	Class name	CAE	DRAE	RDAE	DAGMM	MemAE	RSRAE	Deep SVDD	Ours
CIFAR-10	airplane	70.00 ± 0.72	70.77 ± 1.34	68.11 ± 5.28	46.67 ± 0.73	65.87 ± 0.50	63.81 ± 6.62	54.31 ± 1.97	62.43 ± 1.12
	automobile	36.45 ± 0.73	37.88 ± 1.39	41.08 ± 5.31	55.72 ± 5.87	36.31 ± 0.23	42.91 ± 7.25	48.20 ± 1.62	38.64 ± 1.87
	bird	67.52 ± 0.15	65.68 ± 0.75	57.47 ± 5.51	47.46 ± 2.35	68.94 ± 0.29	60.09 ± 3.77	64.67 ± 1.07	65.64 ± 0.92
	cat	60.33 ± 0.21	59.42 ± 0.64	56.05 ± 4.99	51.86 ± 1.43	53.91 ± 0.36	51.28 ± 3.02	49.70 ± 0.62	47.09 ± 1.32
	deer	58.57 ± 0.77	61.72 ± 0.68	69.28 ± 1.70	51.15 ± 3.18	67.29 ± 0.36	65.64 ± 2.12	66.42 ± 1.44	70.88 ± 1.02
	dog	59.53 ± 0.14	60.90 ± 0.33	49.34 ± 3.15	55.55 ± 4.51	53.83 ± 0.36	45.64 ± 1.43	47.63 ± 1.22	50.36 ± 0.73
	frog	40.42 ± 1.91	43.28 ± 1.72	55.96 ± 0.13	61.74 ± 3.99	56.33 ± 0.38	64.29 ± 2.84	69.52 ± 1.25	69.92 ± 1.11
	horse	48.62 ± 0.12	51.05 ± 1.02	49.93 ± 4.14	59.02 ± 2.78	49.83 ± 0.39	44.17 ± 1.36	50.49 ± 0.72	51.72 ± 1.06
	ship	72.18 ± 0.18	72.01 ± 2.21	55.29 ± 8.68	46.10 ± 4.05	68.81 ± 0.59	64.14 ± 5.37	56.80 ± 0.67	65.76 ± 1.08
	truck	38.43 ± 0.82	38.19 ± 1.53	37.65 ± 5.74	65.23 ± 4.25	35.78 ± 0.51	45.71 ± 7.22	49.23 ± 0.36	44.08 ± 1.54
	average	55.21 ± 0.36	56.09 ± 0.21	54.02 ± 1.63	54.05 ± 0.92	55.69 ± 0.17	54.77 ± 1.84	55.70 ± 0.53	**56.65 ± 0.39**
CIFAR-100	aquatic mammals	60.86 ± 1.68	65.81 ± 1.47	60.59 ± 2.65	49.25 ± 2.73	65.39 ± 0.62	56.70 ± 4.05	58.40 ± 1.09	61.28 ± 0.97
	fish	64.49 ± 0.88	64.90 ± 0.36	53.09 ± 3.30	47.98 ± 3.94	64.07 ± 0.28	54.21 ± 6.70	58.00 ± 0.86	53.71 ± 1.89
	flowers	38.02 ± 1.74	34.45 ± 1.13	31.95 ± 3.03	65.44 ± 4.63	34.86 ± 0.64	60.38 ± 5.09	38.96 ± 6.37	52.61 ± 1.80
	food containers	64.17 ± 0.50	62.81 ± 1.03	55.59 ± 4.44	45.40 ± 3.10	62.54 ± 0.50	59.39 ± 1.68	52.57 ± 1.28	50.62 ± 0.86
	fruit and vegetables	54.88 ± 1.59	53.08 ± 1.66	40.60 ± 3.20	62.96 ± 4.99	50.22 ± 0.80	54.99 ± 4.61	49.99 ± 2.87	46.82 ± 0.69
	household electrical devices	57.45 ± 1.94	54.08 ± 2.12	47.22 ± 4.72	46.60 ± 4.81	48.89 ± 0.36	49.15 ± 6.31	46.35 ± 0.64	40.76 ± 1.20
	household furniture	64.29 ± 3.27	62.16 ± 1.27	54.17 ± 1.79	53.65 ± 4.32	57.81 ± 0.60	59.25 ± 3.97	50.61 ± 0.92	48.62 ± 0.92
	insects	47.26 ± 0.35	46.89 ± 1.21	46.46 ± 2.15	51.70 ± 2.73	50.36 ± 0.35	52.91 ± 4.04	60.92 ± 0.79	56.40 ± 1.41
	large carnivores	54.67 ± 4.16	51.59 ± 1.79	60.97 ± 1.84	59.07 ± 5.22	54.71 ± 0.56	53.08 ± 6.18	57.61 ± 1.80	62.05 ± 1.49
		64.00 ± 0.98	65.94 ± 1.84	60.55 ± 5.40	57.65 ± 7.81	62.17 ± 0.15	54.30 ± 8.83	58.02 ± 2.44	67.56 ± 1.52

TABLE 4 (Continued)

Data set	Class name	CAE	DRAE	RDAE	DAGMM	MemAE	RSRAE	Deep SVDD	Ours
	large man-made outdoor things								
	large natural outdoor scenes	79.49 ± 0.47	82.76 ± 1.04	75.45 ± 3.74	53.43 ± 8.55	79.11 ± 0.51	60.22 ± 8.13	75.85 ± 1.44	73.64 ± 1.15
	large omnivores and herbivores	51.87 ± 0.61	55.24 ± 1.45	58.60 ± 1.86	58.95 ± 2.20	52.94 ± 0.72	51.38 ± 4.68	53.67 ± 0.88	60.31 ± 1.35
	medium-sized mammals	60.48 ± 0.08	57.33 ± 1.35	60.10 ± 4.39	61.17 ± 4.60	54.94 ± 0.50	54.96 ± 4.57	55.46 ± 0.96	63.14 ± 1.01
	noninsect invertebrates	50.88 ± 0.34	51.16 ± 1.06	53.54 ± 2.80	46.66 ± 2.16	54.72 ± 0.44	58.46 ± 2.30	63.01 ± 0.84	59.40 ± 0.66
	people	48.62 ± 2.66	47.83 ± 1.94	45.11 ± 1.09	54.28 ± 3.18	42.45 ± 0.31	42.03 ± 0.83	43.79 ± 1.26	39.70 ± 0.66
	reptiles	52.95 ± 0.70	53.90 ± 0.72	57.22 ± 0.98	51.70 ± 2.62	54.16 ± 0.37	56.79 ± 1.69	62.08 ± 1.54	60.79 ± 1.51
	small mammals	58.82 ± 0.85	61.27 ± 1.28	64.12 ± 2.27	53.49 ± 2.02	61.97 ± 1.01	54.24 ± 3.13	61.40 ± 1.00	65.95 ± 1.65
	trees	59.66 ± 3.08	58.21 ± 3.51	56.19 ± 3.88	59.70 ± 5.16	61.46 ± 1.11	44.08 ± 5.98	66.52 ± 2.18	70.22 ± 0.77
	vehicles 1	41.00 ± 0.27	34.60 ± 2.06	41.54 ± 4.05	53.78 ± 4.45	37.61 ± 0.88	43.97 ± 2.40	50.72 ± 1.66	49.02 ± 1.54
	vehicles 2	53.42 ± 1.95	49.14 ± 1.59	49.34 ± 2.49	50.85 ± 4.94	50.31 ± 0.16	50.72 ± 3.23	53.03 ± 1.93	52.77 ± 1.69
	average	56.36 ± 0.23	55.66 ± 0.49	53.62 ± 0.40	54.19 ± 1.37	55.03 ± 0.04	53.56 ± 0.48	55.85 ± 0.31	**56.77 ± 0.18**

Note: The best performance is in bold.

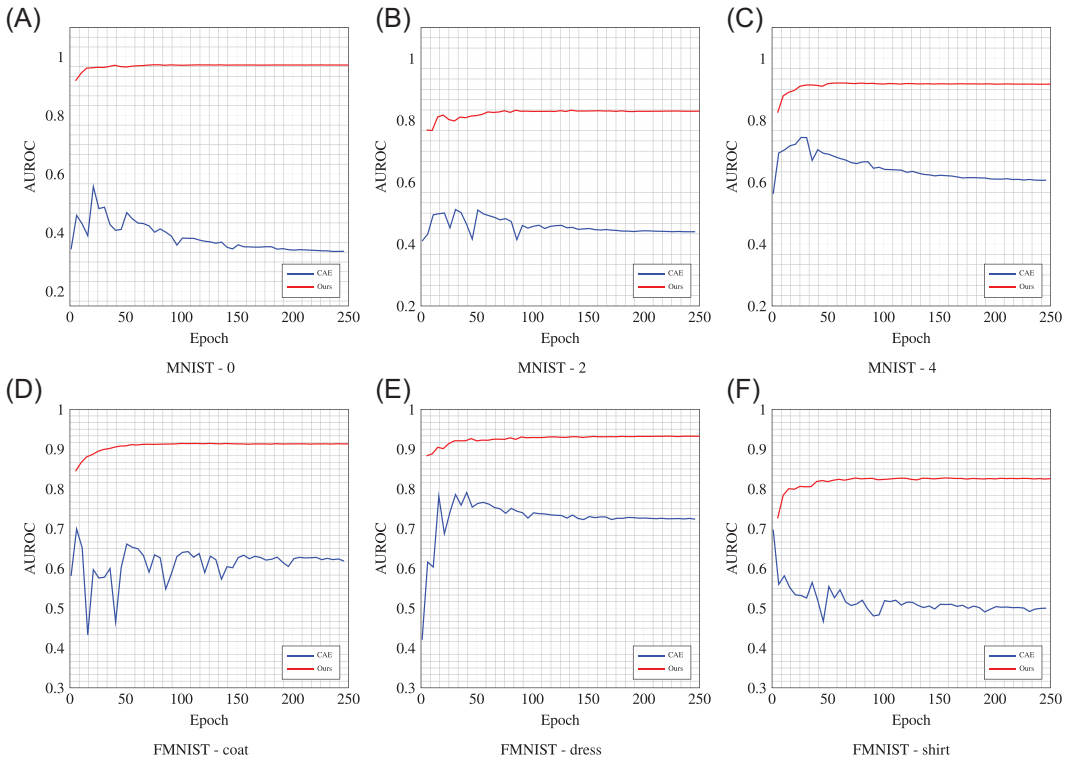

FIGURE 3 UAD performance (AUROC) vs. epoch during the training phase. We plot the results of experiments on Fashion-MNIST and CIFAR10 under $\rho = 10\%$. For MNIST, we show the results of the following classes as denoted in the caption: (A) Number 0, (B) Number 2, (C) Number 4. For Fashion-MNIST, we show the results of the following classes as denoted in the caption: (A) coat, (B) dress, (C) shirt. Our method achieves higher and more stable AUROC on all experiments compared with CAE. AUROC, area under the receiver operating characteristic curve; CAE, Convolutional autoencoder; UAD, unsupervised anomaly detection [Color figure can be viewed at wileyonlinelibrary.com]

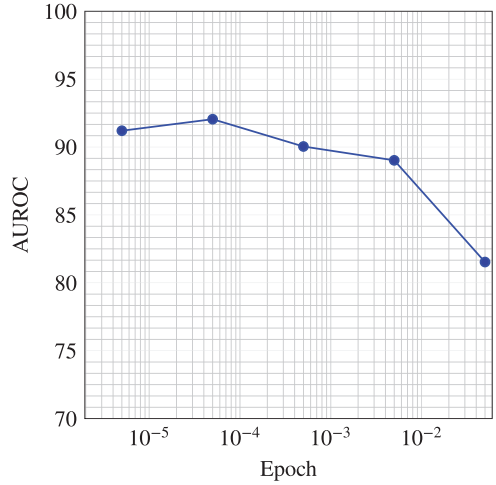

FIGURE 4 The effect of coefficient λ on anomaly detection performance (AUROC) for MNIST data set. The value is averaged on all classes. AUROC, area under the receiver operating characteristic curve [Color figure can be viewed at wileyonlinelibrary.com]

harm the anomaly detection task. A proper λ balances the reconstruction loss and Deep SVDD loss, leading to the highest performance. So we fix $\lambda = 5e - 5$ in all our experiments.

In summary, the above experimental results have well demonstrated the effectiveness of our proposed method compared with other state-of-the-art autoencoder-based methods. We attribute the superiority of the proposed method as two aspects: (i) We propose to detect anomalies in the feature space, using the distance of embedded points to the mean center of all embedded points, instead of the reconstruction loss, as our anomaly score. Experimental results validate the effectiveness and stableness of our proposed scoring strategy. (ii) We incorporate Deep SVDD loss into vanilla autoencoder for anomaly detection. By minimizing Deep SVDD loss, the feature space is manipulated to be more suitable for anomaly detection, which further improves the performance of anomaly detection.

5 | CONCLUSION

In this article, we propose a novel improved autoencoder for unsupervised autoencoder to solve UAD. We incorporate Deep SVDD loss into the autoencoder framework, whereas two processes can negotiate with each other serving for learning tasks. Specifically, our method can learn a better representation suitable for anomaly detection tasks, as well as keep the local data structure. Moreover, we propose a novel scoring strategy for autoencoder-based anomaly detection methods, which is using the distance relationship in feature space instead of reconstruction loss. The new scoring consistently outperforms the traditional scoring strategy and is more stable at the same time, which is demonstrated by our experiments. The drawback of our method is that it does not show a large performance gain on complex data sets. In the future, we will explore updating the Deep SVDD loss with more advanced consideration, hoping to improve its performance on complex data sets further.

ACKNOWLEDGMENTS

This study was supported in part by the National Key Research and Development Program of China under Grant 2018YFB0204301, in part by the National Natural Science Foundation of China under Grant 62006236, in part by the Hunan Provincial Natural Science Foundation under Grant 2020JJ5673, in part by the Natural science research project in universities of Jiangsu Province (18KJB520041) and in part by the National University of Defense Technology (NUDT) Research Project under Grant ZK20-10.

AUTHOR CONTRIBUTIONS

Conceptualization: Zhen Cheng and Siqi Wang. *Methodology*: Zhen Cheng and Siwei Wang. *Validation*: Pei Zhang and Xinwang Liu. *Formal analysis*: Zhen Cheng, En Zhu, Siqi Wang, and Xinwang Liu. *Writing–original draft preparation*: Zhen Cheng. *Writing–review and editing*: Zhen Cheng and Pei Zhang. *Funding acquisition*: En Zhu. All authors reviewed and approved the final manuscript. All authors contributed to the manuscript preparation of the paper.

ENDNOTES

*https://github.com/lukasruff/Deep-SVDD-PyTorch

†https://github.com/donggong1/memae-anomaly-detection

‡https://github.com/demonzyj56/E3Outlier

ORCID

Zhen Cheng https://orcid.org/0000-0002-8217-9600
Siwei Wang https://orcid.org/0000-0001-9517-262X
Pei Zhang https://orcid.org/0000-0002-3018-5080
Siqi Wang https://orcid.org/0000-0002-8134-9508
Xinwang Liu https://orcid.org/0000-0001-9066-1475
En Zhu https://orcid.org/0000-0003-2305-7555

REFERENCES

1. Ahmed M, Mahmood AN, Islam MR. A survey of anomaly detection techniques in financial domain. *Future Generation Comput Syst*. 2016;55:278-288.
2. Panigrahi R, Borah S, Bhoi AK, et al. A consolidated decision tree-based intrusion detection system for binary and multiclass imbalanced datasets. *Mathematics*. 2021;9(7):751.
3. Panigrahi R, Borah S, Bhoi AK, et al. Performance assessment of supervised classifiers for designing intrusion detection systems: a comprehensive review and recommendations for future research. *Mathematics*. 2021;9(6):690.
4. Chandola V, Banerjee A, Kumar V. Outlier detection: a survey. *ACM Comput Surveys*. 2007;14:15.
5. Wold S, Esbensen K, Geladi P. Principal component analysis. *Chemom Intell Lab Syst*. 1987;2(1-3):37-52.
6. Cortes C, Vapnik V. Support-vector networks. *Mach Learn*. 1995;20(3):273-297.
7. Liu FT, Ting KM, Zhou ZH. Isolation forest. In: 2008 *Eighth IEEE International Conference on Data Mining*. IEEE; 2008:413-422.
8. Chalapathy R, Chawla S. Deep learning for anomaly detection: a survey. arXiv:1901.03407 [cs, stat]; 2019.
9. Sakurada M, Yairi T. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: *Proceedings of the MLSDA* 2014 2*nd Workshop on Machine Learning for Sensory Data Analysis*. ACM; 2014:4.
10. Xia Y, Cao X, Wen F, Hua G, Sun J. Learning discriminative reconstructions for unsupervised outlier removal. In: *Proceedings of the IEEE International Conference on Computer Vision*. ACM; 2015:1511-1519.
11. Candès EJ, Li X, Ma Y, Wright J. Robust principal component analysis? *J ACM*. 2011;58(3):11-37. https://doi.org/10.1145/1970392.1970395
12. Zhou C, Paffenroth RC. Anomaly detection with robust deep autoencoders. In: *Proceedings of the* 23*rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM; 2017:665-674.
13. Gong D, Liu L, Le V, et al. Memorizing normality to detect anomaly: memory-augmented deep autoencoder for unsupervised anomaly detection. In: *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019*, Seoul, Korea (South), October 27–November 2, 2019. IEEE; 2019:1705-1714.
14. Lai CH, Zou D, Lerman G. Robust subspace recovery layer for unsupervised anomaly detection. In: 8*th International Conference on Learning Representations, ICLR 2020*, Addis Ababa, Ethiopia, April 26–30, 2020. Openreview.net; 2020.
15. Ruff L, Görnitz N, Deecke L, et al. Deep one-class classification. In: Krause A, Dy JG, eds. *Proceedings of the* 35*th International Conference on ICML* 2018, Stockholmsmässan, Stockholm, Sweden, July 10–15, 2018. Vol. 80 of Proceedings of Research. PMLR; ACM; 2018.
16. Schölkopf B, Platt JC, Shawe-Taylor J, Smola AJ, Williamson RC. Estimating the support of a high-dimensional distribution. *Neural Computat*. 2001;13(7):1443-1471.
17. Tax DM, Duin RP. Support vector data description. *Mach Learn*. 2004;54(1):45-66.
18. Zong B, Song Q, Min MR, et al. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: 6*th International Conference on Learning Representations, ICLR* 2018, Conference Track Proceedings, Vancouver, BC, Canada, April 30–May 3, 2018. OpenReview.net; 2018.
19. Golan I, El-Yaniv R. Deep anomaly detection using geometric transformations. In: *Advances in Neural Information Processing Systems* 31: *Annual Conference on Neural Information Processing Systems 2018*, NeurIPS 2018, December 3–8, 2018, Montréal, Canada MIT Press; 2018:9781-9791.
20. Wang S, Zeng Y, Liu X, et al. Effective end-to-end unsupervised outlier detection via inlier priority of discriminative network. In: *Advances in Neural Information Processing Systems* 32: *Annual Conference on*

Neural Information Processing Systems 2019, NeurIPS 2019, December 8–14, 2019, Vancouver, BC, Canada. MIT Press; 2019:5960-5973.
21. Goyal S, Raghunathan A, Jain M, Simhadri HV, Jain P. DROCC: Deep robust one-class classification. In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, July 13–18, 2020, Virtual Event. Vol. 119 of Proceedings of Machine Learning Research. ACM; 2020:3711-3721.
22. Hawkins DM. *Identification of Outliers*. Monographs on Applied Probability and Statistics. London: Chapman and Hall; 1980.
23. Masci J, Meier U, Ciresan DC, Schmidhuber J. Stacked convolutional auto-encoders for hierarchical feature extraction. In: *Artificial Neural Networks and Machine Learning—ICANN 2011—21st International Conference on Artificial Neural Networks*, Espoo, Finland, June 14–17, 2011, Proceedings, Part I. Springer; 2011: 52-59.
24. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. *Proc IEEE*. 1998;86(11):2278-2324.
25. Xiao H, Rasul K, Vollgraf R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747; 2017.
26. Netzer Y, Wang T, Coates A, Bissacco A, Wu B, Ng AY. Reading digits in natural images with unsupervised feature learning. In: *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*. MIT Press; 2011:5.
27. Krizhevsky A. *Learning Multiple Layers of Features from Tiny Images*. Master's thesis. Department of Computer Science, University of Toronto, Toronto, Canada; 2009.
28. Liu W, Hua G, Smith JR. Unsupervised one-class learning for automatic outlier removal. In: 2014 *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR 2014, Columbus, OH, USA, June 23–28, 2014. IEEE; 2014: 3826-3833.
29. Cheng Z, Zhu E, Wang S, Zhang P, Li W. Unsupervised outlier detection via transformation invariant autoencoder. *IEEE Access*. 2021;9:43991-44002. https://doi.org/10.1109/ACCESS.2021.3065838
30. Davis J, Goadrich M. The relationship between Precision-Recall and ROC curves. In: *Proceedings of the 23rd International Conference on Machine Learning*. ACM; 2006: 233-240.
31. Paszke A, Gross S, Massa F, et al. PyTorch: an imperative style, high-performance deep learning library. In: Wallach H, Larochelle H, Beygelzimer A, dAlché-Buc F, Fox E, Garnett R, eds. *Advances in Neural Information Processing Systems*. Vol 32. MIT Press, Curran Associates, Inc.; 2019.
32. Fei Y, Huang C, Jinkun C, Li M, Zhang Y, Lu C. Attribute restoration framework for anomaly detection. *IEEE Trans Multimedia*. 2020:1. https://doi.org/10.1109/TMM.2020.3046884

How to cite this article: Cheng Z, Wang S, Zhang P, Wang S, Liu X, Zhu E. Improved autoencoder for unsupervised anomaly detection. *Int J Intell Syst*. 2021;36:7103-7125. https://doi.org/10.1002/int.22582