Applied Soft Computing 138 (2023) 110176

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Review article

A comprehensive survey on design and application of autoencoder in deep learning

Pengzhi Li [a], Yan Pei [b,*], Jianqiang Li [c]

[a] Graduate School of Computer Science and Engineering, University of Aizu, Aizu-wakamatsu, Fukushima, 965-8580, Japan
[b] Computer Science Division, University of Aizu, Aizu-wakamatsu, Fukushima, 965-8580, Japan
[c] School of Software Engineering, Beijing University of Technology, Beijing, 100124, China

ARTICLE INFO

Article history:
Received 29 August 2022
Received in revised form 12 February 2023
Accepted 3 March 2023
Available online 8 March 2023

Keywords:
Deep learning
Autoencoder
Unsupervised learning
Feature extraction
Autoencoder application

ABSTRACT

Autoencoder is an unsupervised learning model, which can automatically learn data features from a large number of samples and can act as a dimensionality reduction method. With the development of deep learning technology, autoencoder has attracted the attention of many scholars. Researchers have proposed several improved versions of autoencoder based on different application fields. First, this paper explains the principle of a conventional autoencoder and investigates the primary development process of an autoencoder. Second, We proposed a taxonomy of autoencoders according to their structures and principles. The related autoencoder models are comprehensively analyzed and discussed. This paper introduces the application progress of autoencoders in different fields, such as image classification and natural language processing, etc. Finally, the shortcomings of the current autoencoder algorithm are summarized, and prospected for its future development directions are addressed.

Contents

* Corresponding author.
E-mail address: peiyan@u-aizu.ac.jp (Y. Pei).

https://doi.org/10.1016/j.asoc.2023.110176

1. Introduction

In recent years, artificial intelligence algorithms have developed in leaps and bounds. Among these, machine learning has demonstrated considerable strength, and is used in many research areas, such as image recognition, speech recognition, and autonomous driving. The primary affordance of machine learning is that the major features of the data can be extracted autonomously through training [1]. Machine learning can be divided into supervised, unsupervised, and reinforcement learning in terms of the form of learning method. Supervised learning requires inputting data labels with the training data to the model, such as support vector machines (SVM) and convolutional neural networks (CNN). The optimization algorithm is used to minimize the error between the output result and the actual label value. Therefore, data with a large number of labels is very important for supervised learning. These tags are generally labeled by experts in the relevant fields. For example, medical images are labeled by doctors. When the data is very large, it takes a lot of time to manually mark the data. Consequently, if a model only needs to input training data without training labels, it becomes very convenient. This is what has facilitated the development of unsupervised models.

Unsupervised learning refers to models that require only input training data, such as restricted Boltzmann machines and autoencoders. Machine learning can be structurally divided into single-layer structures and multi-layer structures. Multi-layer structures are also more popularly known as deep learning. Regardless of the structure, its main function is to perform data abstraction. Discoveries in neuroscience about the processing of visual information in the brain have contributed to the development of a research area in artificial intelligence that simulates advanced data abstraction [2]. Deep learning is an effective way to stimulate the achievement of this goal. The multilayer structure and abstraction of learning models establish the fundamental framework of deep learning. The multi-layered structure of deep learning is achieved through multiple linear and non-linear transformations. Each layer, from low-level feature extraction to high-level feature extraction, is a process of feature selection and extraction. Neural networks (NN) are one of the methods commonly used to implement deep learning algorithms. Autoencoders [3], constrained Boltzmann machines, and convolutional neural networks [4] are the main models used in deep learning.

Most deep learning models use convolutional neural networks as their basic implementation structure, e.g., Alexnet, VGG, ResNet, etc. Researchers use these deep convolutional neural network models to solve practical problems. Convolutional neural networks and autoencoders are both representative algorithms based on neural networks [5]. The purpose of autoencoders is to learn representation functions for collections of data, which can generate models of the learned data. The conventional autoencoder was first introduced by Rumelhart et al. [6]. A detailed description of the autoencoder is given by Bourlard et al. [7]. Autoencoder consists of two parts: the encoder and the decoder. The function of the encoder, as the name suggests, is to compress and encode the data. It can also be described as converting the original data into other presentation spaces. This process of transformation is known as the encoding phase. The purpose of the encoder is feature extraction. It uses the meaningful information obtained to represent the data. The purpose of the decoder is to restore the data converted by the encoder to its original representation space. The error between the original data and the data obtained by decoding is minimized. Moreover, autoencoders have a better interpretation for feature selection and extraction. The autoencoder algorithm can be considered a great solution if the error is as small as possible.

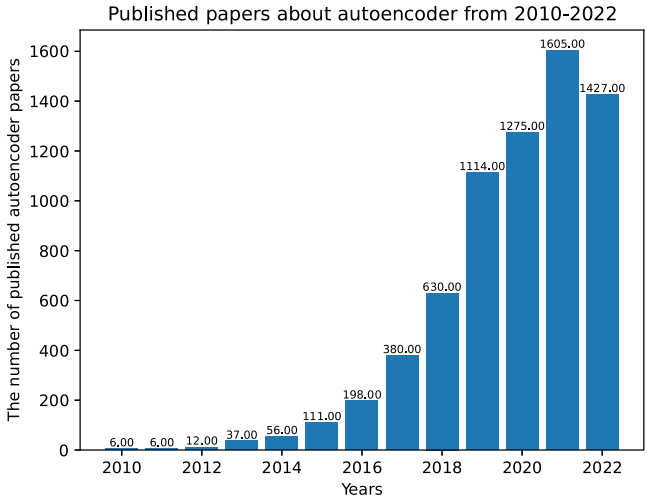

Fig. 1. Papers about autoencoder published in each year from 2010 to 2022. The data was retrieved from the SCOPUS database by searching for papers with the keyword "autoencoder" or "auto-encoder", and used to create statistics by year.

In deep learning, the autoencoder can automatically extract target features, effectively solving the problem of insufficient feature extraction by conventional manual methods, and at the same time effectively avoiding over-fitting. Autoencoder also has some shortcomings, such as the long training time of the deep model due to layer-by-layer training, the poor interpretability of extracted features, and insufficient accuracy. To solve these problems, the researchers conducted in-depth research and put forward some improved autoencoder models.

As shown in Fig. 1, in the SCOPUS database, we use keywords to search autoencoder papers. We can find that the number of automatic encoder papers has been increasing from 2010 to 2020. Autoencoders have evolved rapidly in recent years, with many improved versions being proposed one after another. What is more, they are used in a wide variety of research areas, such as image classification, speech recognition, and data generation, with good results. This indicates that the field of the automatic encoder is a promising research subject. From 2021 to 2022, the number of papers decreased slightly, indicating that the research on autoencoders has slowed down. Through literature investigation, we found two characteristics: (1) In the last two years, many papers merely applied autoencoders to different research fields, with little improvement to the autoencoder itself. (2) The rapid progress of autoencoders focuses on generative models, while not being highly concerned with the improved methods of other types of autoencoders.

Some existing review papers on autoencoders are limited to a certain application field. This paper is not limited in this way, but rather provides a comprehensive classification and method summary of design of autoencoders. Starting from the principle

Table 1
The primary development process of the autoencoder. These improved autoencoder models are arranged in the order in which they are proposed.

Autoencoder name	Name abbreviation	Year
Autoencoder	AE	1986
Denoising Autoencoder	DAE	2008
Sparse Autoencoder	SAE	2011
Contractive Autoencoder	CAE	2011
Convolutional Autoencoder	CoAE	2012
Variational Autoencoder	VAE	2014
Conditional Variational Autoencoder	CVAE	2015
Variational Fair Autoencoder	VFAE	2015
Conditional Variational Autoencoders with GAN	CVAE-GAN	2017
Channel-Recurrent Variational Autoencoder	CRVAE	2017
Wasserstein Autoencoder	WAE	2018
Kernel Method Autoencoder	KAE	2018
Pixel Variational Autoencoder++	PiexlVAE++	2019
Cramer-Wold Autoencoder	CWAE	2020
Nouveau Variational Autoencoder	NVAE	2020
Dizygotic Conditional Variational Autoencoder	DCVAE	2021
Kernelized Linear Autoencoder	KLAE	2021
Dual Contradistinctive Generative Autoencoder	DC-VAE	2021

of the conventional autoencoder, the main improved autoencoders are combed from simple to complex models. According to the principle of improved autoencoder, it is classified. This will help related researchers to better understand the development process of the autoencoder. The application status of autoencoders in different fields, such as image classification and natural language processing, etc., is introduced. The characteristics of important autoencoder models are analyzed and discussed. Finally, the shortcomings of the current autoencoder algorithm are summarized, and projections about prospects for its future development direction are made. The paper summarizes the characteristics of different autoencoders to help researchers to find innovative ideas for autoencoders. In addition, researchers who want to apply autoencoders to their research fields can quickly understand the types and characteristics of different autoencoders through this article. In this way, we hope this paper can contribute to the development of the autoencoder.

The remainder of this paper is organized as follows. In Section 2, we will introduce the fundamentals of autoencoders, and describe their structure etc. In Section 3, we describe some important modifications to and compare different kinds of autoencoders. We summarize each of these by type of autoencoder. In Section 4, we present the practical applications of the autoencoder in various fields. In Section 5, we compare and analyze the shortcomings of an autoencoder and consider its future development trend. Finally, in Section 6 and Section 7 the full text is summarized, and the literature investigation approach used in developing this paper is explained.

2. Conventional autoencoder

After the development of the conventional autoencoder in recent years, there have been many different evolutions. We were able to trace the autoencoder's development process by consulting relevant literature to date. The main process of development is shown in Table 1. At this point, we will introduce and summarize the representative models. Autoencoder is an efficient coding method for learning and acquiring the major features. Autoencoder is implemented in the form of the neural network, which is used to reconstruct its input signal. Because the training process of the autoencoder does not need data labels, it is an unsupervised learning model and method. The structure of an autoencoder usually includes two parts.

(1) The encoder: It learns the major features of the input data (reducing the dimensionality) and transfers the input signal to another space to be presented in another meaningful representation.

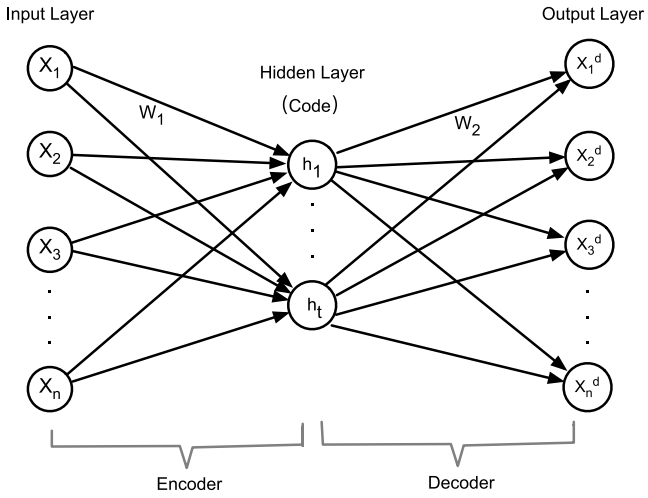

Fig. 2. The structure diagram of an autoencoder. X is the input data of the input layer, h is the hidden layer's data and X^d is the reconstructed output data in the output layer.

(2) The decoder: It turns the converted signal back to the original space and restores the original representation.

As shown in Fig. 2, a complete autoencoder (AE) consists of three different layers, i.e., input layer, hidden layer, and output layer. The neuron numbers in each layer are n, t, and n, respectively. The number of neurons in the input layer and the output layer is the same, and the neuron number in the hidden layer is unlimited. When the number of neurons in the hidden layer is less than the number in the input layer, it is called a sparse structure, as well as a compressed structure. Generally, the neuron number in the hidden layer is less than that in the input layer ($t < n$), which reduces the dimension.

The input layer and the hidden layer form an encoder. The hidden layer and the output layer form a decoder. The original data X is input into the model, and the converted data h is obtained after encoding. This coding process is completed by the formula (1), where W_1 is the weight matrix between the input layer and the hidden layer, and b_1 is the bias vector. $f(x)$ is the activation function of nonlinear transformation.

$$h = f(x) = f(W_1 X + b_1). \tag{1}$$

The decoding process is to map the converted code h to the original space and reconstruct X^d. This process is completed by

the formula (2), where W_2 is the weight matrix between the hidden and output layers and b_2 is the bias vector. $g(x)$ can be a non-linear transformation or an affine transformation function. Some of the more commonly used non-linear activation functions are the sigmoid function and the tanh function.

$$X^d = g(x) = g(W_2 h + b_2). \tag{2}$$

The encoding process is essentially the re-extraction of data into a specific code through a deterministic mapping relationship. The decoding process is the conversion of the specific encoding into the input data. The AE is trained to find the parameters (weight matrix W and bias vector b, represented by θ) that minimize the reconstruction error between X^d and X. In the face of regression and classification problems, there are two general types of reconstruction errors L and loss functions ($J_{AE}(\theta)$) for conventional encoders, as shown in the formulas (3)–(6), where the input data $X = \{x_1, x_2, \ldots, x_n\}$ and the reconstructed output data $X^d = \left\{x_1^d, x_2^d, \ldots, x_n^d\right\}$.

The square error is defined by the formulas (3) and (4).

$$L\left(X, X^d\right) = \|X^d - X\|^2. \tag{3}$$

$$J_{AE}(\theta) = J(X, X^d) = \frac{1}{2}\sum_{i=1}^{n} ||{x_i}^d - x_i||^2. \tag{4}$$

The cross-entropy is defined in formulas (5) and (6).

$$L(X, X^d) = -\sum_{i=1}^{n}(x_i \log x_i^d + (1 - x_i)\log(1 - x_i^d)). \tag{5}$$

$$J_{AE}(\theta) = J(X, X^d) = -\sum_{i=1}^{n}(x_i \log({x_i}^d) + (1 - x_i)\log(1 - {x_i}^d)). \tag{6}$$

As can be seen from the loss function, there are no labels in the loss function. This accounts for the fact that conventional autoencoders are trained by unsupervised methods. The parameters of the decoder are separate from those of the encoder. w_2 has a simple way of taking values to simplify training, $W_2 = W_1^T$. This reduces the training parameters by half and is also known as a tied weight autoencoder (TAE). For the loss function, stochastic gradient descent (SGD) can be used for optimization. Autoencoders and principal components analysis (PCA) have similar effects in terms of dimensionality reduction. Compared to the PCA, the autoencoder is more flexible than the PCA. In general, to control the degree of weight reduction and prevent over-fitting of the autoencoder, a regularization term (also called a weight decay term) will be added to the above loss function, as shown in the formula (7), where the parameter λ controls the strength of the regularization and can range from 0 to 1.

$$J_{ReAE}(\theta) = J(X, X^d) + \lambda \parallel W \parallel_2^2 . \tag{7}$$

The regularization term controls the structure of the network. By constraining the network weights and thus indirectly making the hidden layer neurons sparse, the generalization of the whole autoencoder model is improved. This type of autoencoder is known as the regularization autoencoder (ReAE).

3. Improved version of autoencoder

The goal of autoencoders is to obtain an encoding of the hidden layer by adding linear and non-linear transformations. The encoding is hence decoded so that the output result is as close as possible to the input result. Many improvement methods for autoencoders perform types of processing in the hidden layer, to make the encoded data different from the input data as much as possible. In this case, if the error in the reconstruction between the input and the output is small, the encoding of the hidden layer is a good representation of the input data. In other words, the encoding of the hidden layer is a valid feature learned by the model. Next, in this section, we will introduce several major autoencoders based on our literature investigation.

3.1. Denoising autoencoder

The proposal of the denoising autoencoder (DAE) was inspired by human behavior. Humans can accurately identify a target even when the image is partially obscured. Similarly, if the data reconstructed using data with noise is almost identical to clean data, this encoder has a high generalization ability. Therefore, Vincent et al. proposed the denoising autoencoder [8]. The denoising autoencoder is an extension of the conventional autoencoder. As shown in Fig. 3, a noise layer is added after the input layer, and then the hidden and reconstruction layers are trained with the data that has noise. The denoised autoencoder has the same structure as the conventional autoencoder, except that some type of noise is added to the sample input. Its learning objective is to reconstruct the pure input from the contaminated input. $\tilde{X}$ is the corrupted signal to which noise is added. $\tilde{X}$ is obtained from the clean signal X by a random mapping: $\tilde{X} \sim q_D(\tilde{X}|X)$. The optimization objective of the DAE is to reduce the error between the clean input X and the reconstructed output X^d. Assuming there are S training samples, the formula (8) is the optimization objective function of the DAE.

$$J_{DAE}(\theta) = J(X, X^d) = \sum_{x \in S} L(x, g(f(\tilde{x}))). \tag{8}$$

To obtain a more advanced representation of the features, Vincent et al. used a form of a deep network, stacking the DAEs layer by layer [9]. This results in a network of multiple DAEs connected as the stacked denoising autoencoder (SDAE). The article used this model for image classification and its structure is shown in Fig. 4. It should be noted that SDAE is trained layer by layer. Once f_1 is trained, the output is used as a clean input to f_2 to start training the second layer. This continues, with the coding result of the previous layer serving as input to the next layer until the entire network is trained. The goal of the model is image classification, so the model adds a classifier in Fig. 4. Supervised learning is performed using the true labels of X and the predicted results, and the parameters of the entire network can therefore be tuned using the gradient descent algorithm. Adding noise improves the robustness of the model, but it is necessary to artificially add noise to obtain the damaged signal before each training, which increases the amount of calculation and processing time.

DAE can overcome the noise in the input samples and extract a more robust implicit representation. Therefore, the greatest advantage of DAE is that the reconstructed signal is robust to the noise in the input. But this also introduces an additional process of corrupting the input samples. For SDAE, noise is artificially added to the clean input signal before each network training, which increases the processing time of the model. Moreover, if too much noise is added, it can lead to severe distortion of the input samples, thus reducing the performance of the algorithm.

3.2. Sparse autoencoder

The sparse autoencoder [10] is a modified form of the autoencoder. When the number of neurons in the hidden layer is less than that in the input layer, the autoencoder can learn useful feature structures. Conversely, the autoencoder learns poorly. For this reason, a sparsity constraint is added to make the activation of the neurons in the hidden layer satisfy a certain degree of sparsity. This results in a sparse autoencoder. Assuming that for a

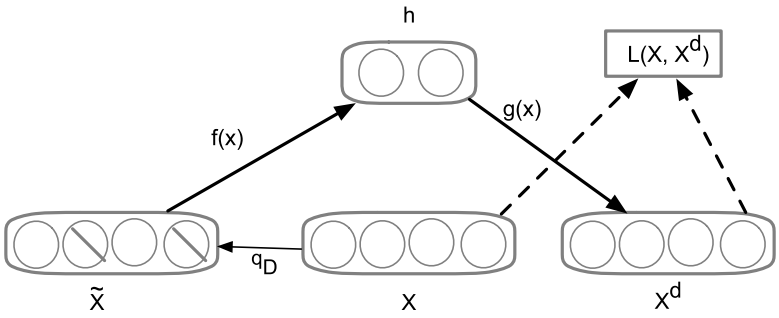

Fig. 3. The structure diagram of denoising autoencoder. X is the input signal, $\tilde{X}$ is the signal after adding noise and X^d is the reconstructed signal.

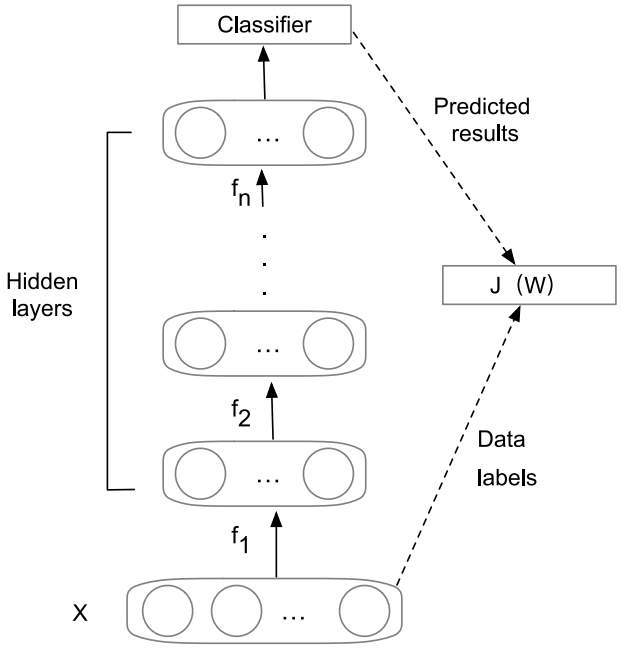

Fig. 4. The structure diagram of stacked denoising autoencoder. Denoising autoencoders are stacked layer by layer, and the output of the previous layer is used as the input of the next layer, so as to form a multi-layer structural model.

given input x, the activation of the jth neuron in the hidden layer is $h_j(x)$, and the average activation of the hidden layer neuron over n training samples is shown in the formula (9).

$$\hat{\rho}_j = \frac{1}{n}\sum_{i=1}^{n} h_j(x_i). \tag{9}$$

For the neurons in the hidden layer to satisfy the sparsity restriction, Kullback–Leibler divergence (KL divergence) is used to make $\hat{\rho}_j$ similar to a sparsity parameter, as shown in the formula (10), where ρ is the sparsity parameter. Generally, ρ is a very small value. The value of the $KL(\rho \parallel \hat{\rho}_j)$ function increases monotonically as the gap between $\hat{\rho}_j$ and ρ increases. When $\hat{\rho}_j = \rho$, the function value reaches a minimum value of 0.

$$KL(\rho \parallel \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho)\log \frac{1-\rho}{1-\hat{\rho}_j}. \tag{10}$$

Consequently, by adding function $KL(\rho \parallel \hat{\rho}_j)$ to the loss function of the autoencoder and optimizing the objective function, it is possible to get $\hat{\rho}_j$ and ρ as close as possible. The resulting loss function for the sparse autoencoder is shown in the formula (11), where β is the weight coefficient that controls the sparse penalty term, which takes values in the range of 0 to 1.

$$J_{SAE}(\theta) = J(X, X^d) + \beta \sum_{j=1}^{t} KL(\rho \parallel \hat{\rho}_j). \tag{11}$$

Based on the sparse encoder, Makhzani et al. proposed a k-sparse encoder model [11]. The k-sparse autoencoders use linear transformations to calculate activation values, and the hidden layer retains only the k-largest activation values. The use of KL divergence is thus avoided by specifying the value of k. Compared to sparse encoders, k-sparse encoders are faster to train and the sparsity of the hidden layer can be obtained accurately. However, the choice of the optimal k remains to be investigated. Different values of k have a significant impact on the results.

3.3. Contractive autoencoder

To further improve the robustness of the representation learning algorithm, Rifai et al. proposed the contractive autoencoder [12,13]. The difference between the contractive autoencoder and the conventional autoencoder is the addition of a penalty term. This penalty term is the Jacobian F norm of hidden layer features, and its expression is shown in the formula (12). In which Jacobian matrix is $J_f(x) = \frac{\partial h_j(x)}{\partial x_i}$ and $h(x)$ is the coding function of hidden layer. The Jacobian norm is the sum of squares of partial derivatives of implicit features to input units. By restricting the value range of the first-order partial derivative, the recessive features are contractible.

$$\| J_f(x) \|_F^2 = \sum_{j=1}^{t}\sum_{i=1}^{n} \left(\frac{\partial h_j(x)}{\partial x_i}\right)^2. \tag{12}$$

The loss function of CAE is shown in the formula (13). λ is the parameter that controls the penalty term (the value range is 0–1). The larger the value, the stronger the robustness of the extracted features, and the more the original information is lost. Although Jacques's term will increase the reconstruction error of the model, it controls the sensitivity of hidden features to the input by minimizing the first partial derivative of hidden features to the input data, and finally keeps the extracted features unchanged to the input within a certain range. It can be seen from the formula (13) that when the encoder is a linear mapping function, the Jacobian term is equivalent to the $L2$ weight attenuation term. At this time, the contraction autoencoder and the regularization autoencoder are the same. When the encoder is a nonlinear mapping function, the contraction autoencoder forces the output of hidden neurons to be in the saturation region of the nonlinear function to achieve contraction. Therefore, the regularization autoencoder can be regarded as a special form of contraction autoencoder. Both CAE and DAE are robust to input noise, but they work on different principles; DAE is robust to signal reconstruction, while CAE is robust to implicit representation.

$$J_{CAE}(\theta) = J(X, X^d) + \lambda \| J_f(x) \|_F^2 . \tag{13}$$

3.4. Marginalized denoising autoencoder

The marginalized denoising autoencoder is an improved autoencoder based on the denoising autoencoder proposed by Chen et al. [14,15]. Compared to DAE, MDAE mainly marginalizes the noise, which also reduces the processing time of the encoder. Marginalization is essentially to use Taylor expansion of the DAE loss function to approximate the expected loss function. After m random additions of noise processing, m corrupted samples can be obtained from the input samples x_i. The loss function of the denoising autoencoder can also be expressed as the formula (14), where $f_\theta() = g(f())$ and x_i^j is the jth corrupted version of input x_i.

$$J_{DAE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{m}\sum_{j=1}^{m}L(x_i, f_\theta(x_i^j)). \tag{14}$$

The noise distribution conforms to the $P(\tilde{x}|x)$ distribution. Consequently, the idea of expectation is introduced, as shown in the formula (15).

$$J_{DAE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}E_{P(\tilde{x}_i|x_i)}(L(x_i, f_\theta(\tilde{x}_i))). \tag{15}$$

The L function in the formula (15) is expanded by the Taylor formula, and the expectation is added to obtain the formula (16), where μ_x is the expected value of $\tilde{x}$ under the noise distribution $P(\tilde{x}|x)$, $\mu_x = E_{P(\tilde{x}|x)}[\tilde{x}]$. $\nabla_{\tilde{x}}L$ and $\nabla^2_{\tilde{x}}L$ are the first-order and second-order derivatives of the loss function L taking values at $\tilde{x}$.

$$\begin{aligned}E[L(x, f_\theta(\tilde{x}))] \approx{}& E[L(x, f_\theta(\mu_x))] + E[(\tilde{x}-\mu_x)^T\nabla_{\tilde{x}}L] \\ &+ E[\frac{1}{2}(\tilde{x}-\mu_x)^T\nabla^2_{\tilde{x}}L(\tilde{x}-\mu_x)].\end{aligned} \tag{16}$$

Let $\sum_x = E[(\tilde{x}-\mu_x)(\tilde{x}-\mu_x)^T]$. Formula (16) can be further simplified to formula (17), where $\sum_x$ is the variance of the noisy samples.

$$E[L(x, f_\theta(\tilde{x}))] \approx L(x, f_\theta(\mu_x)) + \frac{1}{2}tr(\sum_x \nabla^2_{\tilde{x}}L). \tag{17}$$

The processes of adding noise are independent of each other. So $\sum_x$ can be reduced to a diagonal matrix. In this case, only the diagonal terms of the Hessian matrix $\nabla^2_{\tilde{x}}L$ need to be calculated to find the formula (17). The elements of the d-dimensional diagonal terms of $\nabla^2_{\tilde{x}}L$ are calculated as shown in the formula (18). The calculation in literature [16] discards the second term in the formula (18). Since $\frac{\partial^2 L}{\partial Z^2}$ is positive definite, it can be reduced to obtain the formula (19), where Z and D_h represent the hidden layer features and the number of nodes respectively.

$$\frac{\partial^2 L}{\partial \tilde{x}_d^2} = (\frac{\partial Z}{\partial \tilde{x}_d})^T\frac{\partial^2 L}{\partial Z^2}\frac{\partial Z}{\partial \tilde{x}_d} + (\frac{\partial L}{\partial Z})^T\frac{\partial^2 Z}{\partial \tilde{x}_d^2}. \tag{18}$$

$$\frac{\partial^2 L}{\partial \tilde{x}_d^2} \approx \sum_{h=1}^{D_h}\frac{\partial^2 L}{\partial Z_h^2}(\frac{\partial Z_h}{\partial \tilde{x}_d})^2. \tag{19}$$

The final loss function of the MDAE can be obtained as the formula (20). σ^2_{xd} is the dth element of the diagonal term of the matrix $\sum_x = E[(\tilde{x}-\mu_x)(\tilde{x}-\mu_x)^T]$.

$$J_{MDAE}(\theta) = \sum_{x\in S}L(x, f_\theta(\mu_x)) + \frac{1}{2}\sum_{d=1}^{D}\sigma^2_{xd}\sum_{h=1}^{D_h}\frac{\partial^2 L}{\partial Z_h^2}(\frac{\partial Z_h}{\partial \tilde{x}_d})^2. \tag{20}$$

Through the loss function, we can see that the loss function of MDAE can be regarded as a special regularization term added based on the DAE. This regularization term considers the sensitivity of the reconstruction function to the hidden layer and the sensitivity of the hidden layer expression to the damaged signal.

3.5. Convolutional autoencoder

With the rise of convolutional neural networks, convolutional autoencoders [17] have also been developed. Conventional autoencoders generally use fully concatenated layers that do not affect one-dimensional data. However, for images, autoencoders have to turn the input data into one-dimensional vectors before they can be processed, which results in a loss of information about the two-dimensional structure of the image. Based on the autoencoder, convolution and pooling operations in the convolutional neural network are introduced to replace the full connection layer. The convolution and pooling operation can well preserve two-dimensional spatial information.

Convolutional autoencoder is the encoding and decoding of input data using convolution. Local features in the data are extracted by convolution and pooling, and the data is reduced by deconvolution. The structure of a convolutional autoencoder is shown in Fig. 5.

The convolutional autoencoder is similar to the AE. It uses convolution and down-sampling to extract the features of the input signal, and its weights are shared. CoAE combines the advantages of a convolutional neural network and autoencoder. Its loss function can be expressed as the formula (21) just like the regularization autoencoder.

$$J_{CoAE}(\theta) = J(X, X^d) + \lambda \parallel W \parallel_2^2. \tag{21}$$

3.6. Variational autoencoder

The variational autoencoder (VAE) was proposed by Kingma et al. [18]. The main principle of variational autoencoder is to map a set of data into an ideal Gaussian distribution through an encoder. Then the samples sampled by Gaussian distribution are input into the decoder to generate reconstructed data. It is a data generation model based on variational Bayesian inference, and its structure is shown in Fig. 6, where E and D denote the encoder and decoder respectively. Symbols $\otimes$ and $\oplus$ denote the multiplication and addition of vector elements, respectively. Input X into the encoder to obtain μ and σ, and introduce Gaussian distribution ε to get probability coding Z. Finally, the decoder decodes Z to restore X. Suppose there is a set of functions for generating X from Z, each function uniquely determined by θ. The optimization goal of the variational autoencoder is to maximize the probability $p(x)$ generated by X by optimizing θ on the premise of sampling Z. According to the Bayes formula, formula (22) can be obtained.

$$P(x) = \int f(x|z)P(z)dz. \tag{22}$$

The VAE obtains the probability distribution of the hidden variable Z by adding an encoding network that acts as an inference in front of the generative network model. A function $Q(z|x)$ is therefore introduced to perform the function of the coding network. The objective of this function is to obtain the distribution of hidden variable Z that can generate reconstruction X under the condition of X. We want $Q(z|x)$ to be as close as possible to the ideal $P(z|x)$, so the KL divergence (abbreviated as D) is used to evaluate the similarity of the two variables, as shown in the formula (23).

$$D[Q(z|x) \parallel P(z|x)] = E_{Q(z|x)}[\log Q(z|x) - \log P(z|x)]. \tag{23}$$

$$\log P(x) - D[Q(z|x) \parallel P(z|x)] = E_{Q(z|x)}[\log P(x|z)] - D[Q(z|x) \parallel P(z)]. \tag{24}$$

$$J_{VAE} = E_{Q(z|x)}[\log P(x|z)] - D[Q(z|x) \parallel P(z)]. \tag{25}$$

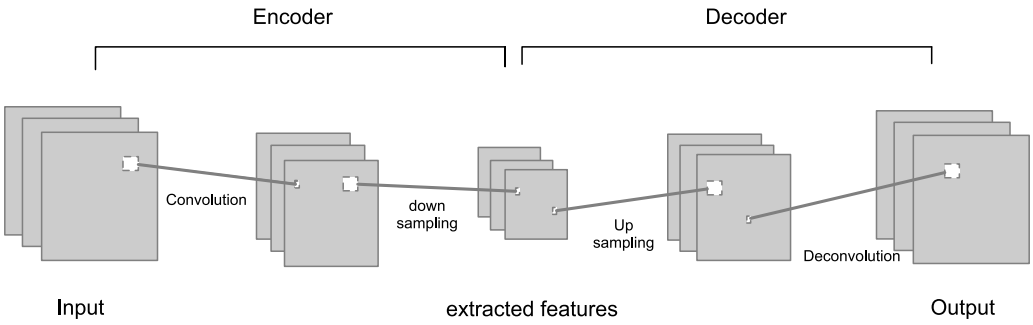

Fig. 5. The structure diagram of convolutional autoencoder. The encoder is convolution and downsampling, and the decoder is upsampling and deconvolution.

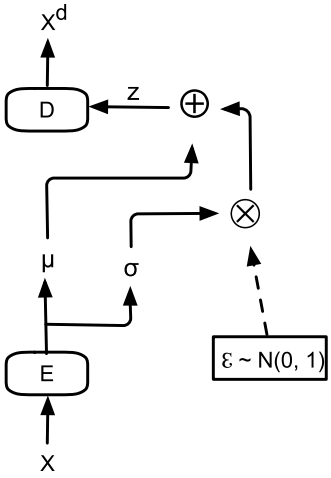

Fig. 6. The structure diagram of variational autoencoder. It inputs X into the encoder to obtain μ and σ, and introduces Gaussian distribution ε to obtain probability coding Z. The decoder decodes z to generate X^d.

Next, $P(x|z)$ is expanded by the Bayesian formula, and the formula (24) can be obtained after simplification. Thus, the loss function of VAE is obtained, as shown in the formula (25). The loss function of VAE can be divided into two parts. One part is that the hidden variable Z is constrained by a standard distribution. The other part is to make the final result as close as possible to the input data.

3.7. Improved models based on variational autoencoder

VAE is a very popular model for data generation, especially image generation, and it has been rapidly developed and widely used. However, conventional VAE has some shortcomings. The images generated for complex conditions are also often blurred. In recent years, many improved models have been obtained through continuous improvement of the VAE structure. These improved models have also achieved good results in several fields. Some of the main improved models based on VAE are presented below.

Conventional VAE can generate input data approximately, but it cannot generate specific types of data directionally. To solve this problem. Makhzani et al. proposed a conditional variational autoencoder (CVAE) [19]. Input data x and partial tags (y) of x into the encoder part of CVAE. This will generate the data of the specified category. The structure of CVAE is similar to that of VAE, so the calculation method and optimization method of CVAE is consistent with that of VAE. Because there are some labels Y in the input, CVAE becomes a semi-supervised learning form.

Louizos et al. proposed a variational fair autoencoder (VFAE) [20]. To separate the noise in the input data from the hidden variable information, and improve the learning accuracy of the hidden variable representation. VFAE introduces the maximum mean difference (MMD) [21] as a regular term to weaken the relationship between input noise and hidden layer variables. VFAE also provides a good foundation for representation learning with invariant feature models.

The quality of image reconstruction or image generation based on VAE needs to be improved. GAN can make use of its antagonism to ensure the authenticity of the generated image, so Bao et al. [22] proposed to add the discriminator of GAN after the decoder of CVAE [23]. In this way, the image generated by CVAE has higher quality, and the CVAE-GAN structure diagram is shown in Fig. 7.

$$L = \lambda_1 L_{KL} + \lambda_2 L_G + \lambda_3 L_{GD} + \lambda_4 L_{GC} + L_D + L_C. \tag{26}$$

The optimization objective function of CVAE+GAN is shown in the formula (26), where L_{KL} refers to the KL divergence between the inferred network distribution and the prior distribution of hidden variables. L_G means adding feature matching loss to the reconstruction loss between the generated sample and the real sample. L_{GD} is the correction loss of the discriminator to the generated samples of the generated network. L_{GC} is the classifier's prediction and correction loss for the category of generated network samples. L_D represents the loss of the discriminator and the generating network. L_C indicates the category classification loss of the classifier. CVAE+GAN shows a good effect in image synthesis, which is attributed to the antagonism of GAN. $\lambda_1 - \lambda_4$ are weight parameters. Creswell et al. proposed a conditional VAE-GAN with an Information Factorization (IFcVAE-GAN) [24]. IFcVAE-GAN adds an auxiliary network based on CVAE-GAN for information separation in hidden variable space. Separate the information containing category labels from the reconstructed information. The purpose of this is to directionally generate the specified category sample data by changing the category information. The advantage of IFcVAE-GAN is that it can control the generation of more kinds of image data. However, because of the complexity of the model, the amount of calculation for building the model has increased.

It is found that to improve the performance of VAE, researchers combine VAE with an autoregressive network model. The auto-regressive network model has a strong generating ability, which makes up for the shortcomings of VAE. The variable loss autoencoder (VLAE) [25] is the combination of VAE and autoregressive network models (such as RNN and Pixel-RNN [26], etc.). Shang et al. proposed channel-recurrent variational autoencoders (CRVAE) [27]. CRVAE model combines three parts: Convolutional Variational Autoencoder (CoVAE), Long–Short Term Memory (LSTM), and Generative Adversarial Network (GAN). CRVAE realizes the network structure of multi-channel circular

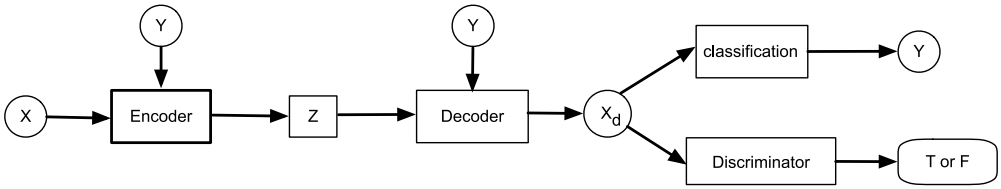

Fig. 7. The structure diagram of CVAE+GAN. X is the input image data, and z is a hidden variable. X_d is the image reconstructed by the generative model. Y is the classification label of the image. X_d obtains binary output (T or F) through discriminative network.

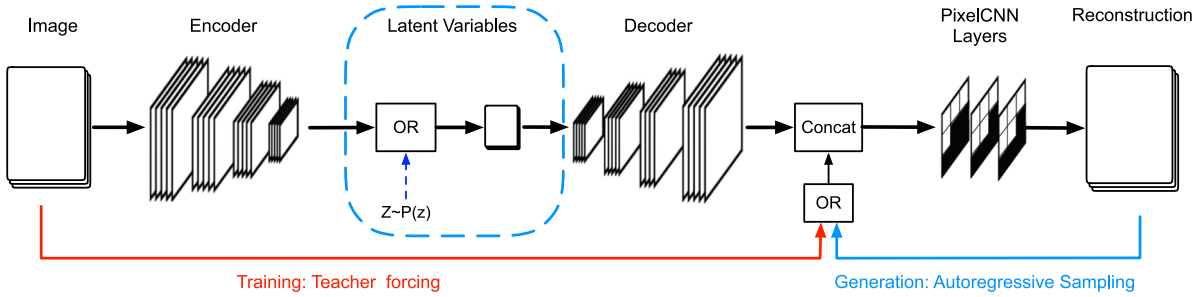

Fig. 8. The structure diagram of pixel variational autoencoder. The input image will be translated to a posterior distribution over the latent variable z, and the model's structure is similar to that of the VAE. Convolution procedures are implemented by the encoder and decoder for both upsampling and downsampling.

interconnection, which overcomes the disadvantages of fuzzy generated images, the poor performance of complex structures, and the unfriendly serialization model. CRVAE uses the structure of a fully convolutional neural network [28] to replace the DNN structure as a hidden variable to represent the inference network of learning feature extraction. This also enhances the ability to extract depth details [29]. LSTM and GAN are added to enhance the learning, expression, and generation ability of time series models. In Ref. [27], two new types of regularization penalties are proposed: KL objective Weighting and Mutual Information Maximization. They are used to replace the regular KL divergence terms, thus improving the accuracy of data generation.

Gulrajani et al. proposed a pixel variational autoencoder (PiexlVAE) [30]. As shown in Fig. 8, PixelVAE uses PixelCNN [31] to simulate an auto-regressive decoder for VAE. It is assumed that VAEs, which are conditionally independent between pixels, are known to suffer from fuzzy samples, while PixelCNN, which models the joint distribution, will produce clear samples. PixelVAE combines the advantages of both, providing meaningful potential performance and producing clear samples at the same time. Based on PixelVAE, to improve its performance. Sadeghi et al. proposed a new PixelVAE++ model [32]. PixelVAE++ is composed of a VAE with three potential variables and a PixelCNN++ for the decoder, and a part of the decoder is reused as an encoder. While maintaining the information on potential variables, the performance is improved.

Vahdat et al. proposed a deep hierarchical variational autoencoder based on depth-wise separable convolutions and batch normalization, which is called NVAE [34]. NVAE uses separable convolution in-depth direction for generating the model and regular convolution for the encoder model. NVAE is equipped with a residual parameterization of normal distributions and its training is stabilized by spectral regularization. By reducing the memory usage of deep VAEs, the training speed is doubled. For a few sample learning, Zhang et al. suggest the dizygotic conditional variational autoencoder (DCVAE) model which achieves multiple feature synthesis and data enhancement purposes [35]. To extract semantic and visual information features, respectively, DCVAE incorporates two conditional variational autoencoders (CVAE). Two separate information characteristics are integrated using an adaptive sharing process to provide comprehensive features. The conditional consistency control module is used to maintain consistency between the original information and the information represented by comprehensive features. Experiments indicate that DCVAE can work well in various data configurations.

Two things affect how well generative autoencoders work: (1) image fidelity (instance fidelity): The reconstructed image matches the input single image as closely as is practical. (2) global set fidelity (set fidelity): The rebuilt outcome must closely match the initial dataset. A dual contradistinctive generative autoencoder (DC-VAE) was created by Parmar et al. in light of this [33]. Fig. 9 is the structure diagram of this new framework. A single variational autoencoder system integrates instance-level differential loss and set-level adversarial loss. Experiments show that these two kinds of losses are very important, and the generation ability of DC-VAE has been significantly improved.

3.8. Wasserstein autoencoder

Tolstikhin et al. proposed a new model called Wasserstein autoencoder (WAE) based on VAE [36]. Because the autoencoder only pays attention to the reconstruction error, it cannot explain the internal structure of the real data itself. Therefore, the distance measure between distributions is introduced to extract the essential characteristics of data from the perspective of data distribution transformation. Commonly used distance measures between distributions are KL divergence, Jenson's Shannon divergence (JS divergence), and Wasserstein divergence. KL divergence will give infinite results when the two distributions have no intersection at all. JS divergence will suddenly jump, and the gradient will disappear, which is a serious problem in the learning algorithm. Wasserstein distance is smooth, which avoids the above two problems. Ref. [36] introduces the Wasserstein distance to measure the difference between different distributions, which is also called the best transmission distance (Optimal Transport, OT). For the two distributions P_X and P_G, we use Wasserstein distance to calculate, as shown in the formula (27), where (X, Y) is the random variable obeying P_X and P_G, respectively. $c(X, Y)$ is an arbitrary metrizable cost function and $p(X \sim P_X, Y \sim P_G)$

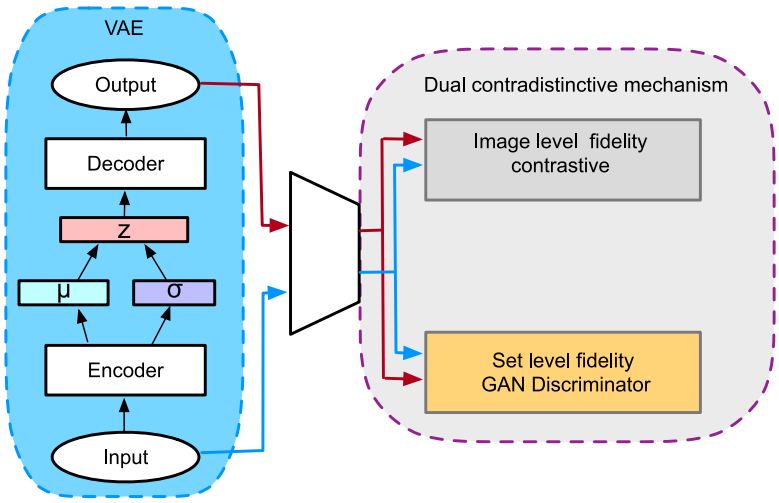

Fig. 9. The structure diagram of dual contradistinctive generative autoencoder model. The model outperforms the variational autoencoder. A single generative model incorporates both adversarial and image-level fidelity.
Source: Adopted from Ref. [33].

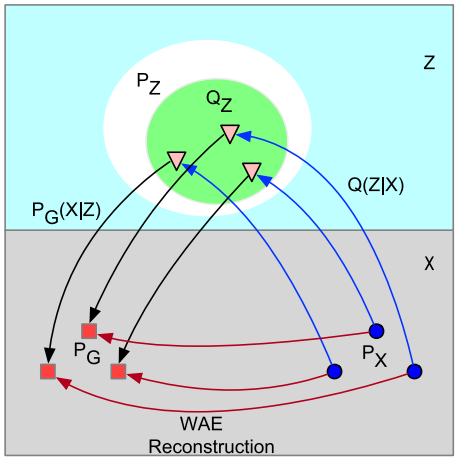

Fig. 10. The structure diagram of the Wasserstein autoencoder. Where X is the input data, Z is the hidden layer coding, and Q_Z (green circle) and P_Z (white circle) are the marginal distribution and prior distribution of the hidden coding Z, respectively. P_X and P_G are the probability distributions of input data and reconstructed data, respectively.
Source: Adopted from Ref. [36].

is the set of all joint distributions with P_X and P_G as marginal distributions. When (χ, d) is a metric space and $c(x, y) = d^p(x, y)$ for $p \geq 1$, the p-th root of W_c is called the p-Wasserstein distance.

$$W_c(P_X, P_G) := \inf_{\Gamma \in p(X \sim P_X, Y \sim P_G)} \mathrm{E}_{(X,Y)\sim\Gamma}[c(X, Y)]. \tag{27}$$

WAE is an algorithm to build a generation model in data distribution. Similar to VAE, WAE has encoder and decoder components. Compared with VAE, WAE has a good invisible structure and is easier to train. The quality of the generated samples is higher, and the reconstructed structure is shown in Fig. 10. In which Q and G are encoder and decoder respectively. P_X and P_G are the probability distributions of input data and reconstructed data, respectively. Q_Z is the continuous mixed distribution of coded Z. P_Z is the prior distribution of coded z. WAE forces Q_Z to match P_Z. The advantage of this is that the hidden coding of different samples has the opportunity to stay away from each other so that the correct results can be reconstructed. To measure the reconstruction loss, WAE uses Wasserstein distance to measure the distance between distribution P_X and P_G.

Assuming that the generated distribution P_G is determined by arbitrary function $G : Z \to \chi$ and decoder $P_G(X|Z)$, which gives formula (28), where $X \sim P_X$, $Z \sim Q(Z|X)$. Q_Z is the marginal distribution of Z, and Q_Z is equal to the prior distribution P_Z. By relaxing the constraint $Q_Z = P_Z$, we can obtain the loss function of WAE, as shown in the formula (29), where O is any nonparametric set of the probabilistic encoder. $G()$ is the mapping function from Z to X. P_G is the distribution generated by G. $c(X, G(Z))$ is an arbitrary measurable loss function for X and $G(Z)$. $D_Z()$ is to calculate the arbitrary divergence between the distributions Q_Z and P_Z. β is a weight parameter ($\beta > 0$).

$$\inf_{\Gamma \in p(X \sim P_X, Y \sim P_G)} \mathrm{E}_{(X,Y)\sim\Gamma}[c(X, Y)] = \inf_{Q: Q_Z = P_Z} \mathrm{E}_{P_X} \mathrm{E}_{Q(Z|X)}[c(X, G(Z))]. \tag{28}$$

$$D_{WAE}(P_X, P_G) = \inf_{Q(Z|X)\in O} \mathrm{E}_{P_X} \mathrm{E}_{Q(Z|X)}[c(X, G(Z))] + \beta \cdot D_Z(Q_Z, P_Z). \tag{29}$$

The optimization goal of WAE consists of reconstruction loss and regularization penalty D_Z, which is used to restrict the difference between two distributions in coding space Z. The author also gives two ways to realize it: one is confrontation training in potential space based on GAN. The other is based on the maximum mean discrepancy (MMD).

Inspired by WAE-MMD's work, Knop et al. proposed a new Cramer-Wold autoencoder (CWAE) [37]. A basic component of CWAE is the Cramer-Wold feature kernel. Its major distinguishing feature is that it has a closed form of the kernel product of radial Gaussians. Consequently, the CWAE model adopts a closed form for the distance between posterior prior and positive prior, thus eliminating the need for sampling to calculate the loss function, which in turn simplifies the optimization process. In the standard benchmark test, the performance of CWAE is usually better than WAE-MMD.

3.9. Kernel method-based autoencoder

Kernel method-based autoencoder (KAE) is one of the autoencoder implementations in the autoencoder family. Different from the other implementations of the autoencoder, its construction is built up on feature map of kernel method, rather

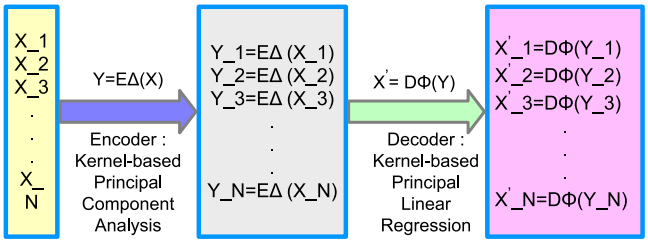

Fig. 11. The structure diagram of kernel method-based autoencoder. $E\Delta$ represents the encoding process of encoder; $D\Phi$ is the decoding process of decoder. $[Y\_1, \ldots, Y\_N]$ refer to the hidden variable obtained by the encoder.

than neural networks. Fig. 11 depicts the structural layout of an autoencoder using kernel principal component analysis (KPCA) that is presented in the literature [38,39]. Kernel-based PCA [40] serves as the foundation for the encoder composition in this. The kernelization of linear regression is the decoder's guiding concept. The kernelized PCA [41] transfers the data into a reproducing kernel Hilbert space (RKHS) to create a new representation of the features, while the kernelized linear regression returns the transferred features to their original data form.

The construction principle of encoder and decoder is as follows.

3.9.1. The encoder of KAE

By effectively carrying out feature selection and extraction in different dimensions spaces, the encoder transforms the data. Refs. [38,39] uses kernel-based principal component analysis to implement this part. The kernel method is one of the classical learning algorithms in the field of machine learning. It attempts to solve problems that cannot be handled by linear models in the original presentation space using the principle of minimizing structural risk in the feature space. It uses kernel functions to establish the relationship between the original space and the inner product of a RKHS. For example, m ($m \in R^d$) and n ($n \in R^d$) are two vectors of real numbers in the inner product space. There is a feature map φ that projects these two vectors into the RKHS, i.e., $\varphi(m) \in R^h$ and $\varphi(n) \in R^h$. The kernel function f implements the relationship : $\langle \varphi(m), \varphi(n) \rangle = f(\langle m, n \rangle)$. One of the in-depth research of principal component analysis is KPCA. It operates by identifying a new coordinate system that maximizes the amount of variance of the data projection in a RKHS.

First, the relationship between the covariance matrix and the kernel matrix (κ) is established, as shown in the formula (30). $\tilde{X}$ is data transformed into a RKHS by $\phi(x)$, where $\tilde{X}^T = [\varphi(x_1), \varphi(x_1), \ldots, \varphi(x_n)]$. Next, the eigenvalue problem on the kernel matrix is solved to find the solution of the eigenvalues of the covariance matrix, as shown in the formulas ((31), (32)), where $\tilde{C} = \tilde{X}^T\tilde{X}$. Assuming that the projected data are central, a kernel matrix is needed to represent a matrix of central data, the Gram matrix, as shown in the formula (33). Formulas (30)–(33) illustrate how to use the kernel function to solve PCA problem in a RKHS.

$$\kappa = \tilde{X}\tilde{X}^T. \tag{30}$$

$$(\kappa)\mu = \lambda\mu. \tag{31}$$

$$(\tilde{C})\tilde{X}^T\mu = \lambda(\tilde{X}^T\mu). \tag{32}$$

$$\tilde{\kappa} = \kappa - 1_N\kappa - \kappa 1_N + 1_N\kappa 1_N. \tag{33}$$

3.9.2. The decoder of KAE

The decoder is used to restore the data to the original representation space. Ref. [38] uses kernel-based linear regression to implement this part. And least squares are used to optimize the regression model. Formula (34) is an optimization problem with ω as the optimization objective. To solve this optimization problem, the normal formula (35) is established to try to obtain a suitable ω (formula (36)).

$$\omega = \arg\min_{\omega} \| Y - X_\omega \|^2. \tag{34}$$

$$X^TX\omega = X^TY. \tag{35}$$

$$\omega = (X^TX)^{-1}X^TY. \tag{36}$$

$$\bar{X} = \begin{bmatrix} 1 & \varphi(x_1)^T \\ \vdots & \vdots \\ 1 & \varphi(x_N)^T \end{bmatrix} \tag{37}$$

$$\omega = (\bar{X}^T\bar{X})^{-1}\bar{X}^TY. \tag{38}$$

$$\omega = \bar{X}^T\alpha. \tag{39}$$

$$\alpha = \bar{\kappa}^{-1}Y. \tag{40}$$

$$Y = \bar{X}\omega = \bar{\kappa}^{-1}\alpha = (1_{N\times N} + \kappa)\alpha. \tag{41}$$

First, the training data X is mapped to the RKHS using the kernel function method to obtain $\bar{X}$, as shown in the formula (37). The next step is to find the linear relationship between the data $\bar{X}$ and the target Y. The normal equation can be rewritten as the formula (38). Assume that $\alpha = \bar{X}(\bar{X}^T\bar{X})^{-2}\bar{X}^TY$ and use the kernel matrix to represent α as shown in the formula (40), where $\bar{\kappa} = 1_{N\times N}+\kappa$. Finally, the final expression for the linear regression of the kernel function is shown in the formula (41). Formulas (34)–(41) describe the process of linear regression solution of high-dimensional data by kernel method.

In summary, combined with Fig. 11, it is known that KAE is a representation learning method. It reconstructs the input interest signal in the presence of a kernel method, which leads to feature extraction. The kernel-based autoencoder is designed to obtain the minimum error between the input and output signals. So the objective function for KAE optimization is the formula (42), where $\Delta : X \to Y$ stands for encoder and $\Phi : Y \to X$ for decoder. KAE is a special coding form. Compared with the neural network, it has better interpretability. KAE is a perspective methodology to implement deep structural machine learning algorithms. Based on KAE, Majumdar et al. modified the decoder, and proposed a kernel linear autoencoder (KLAE) [42]. KLAE is composed of a kernel encoder and kernel decoder. Consequently, the function approximation ability of the stacked autoencoder is simulated. Experiments show that the proposed model performs well in the field of classification.

$$(\Delta, \Phi) = \arg\min_{\Delta,\Phi} \| X - \Phi(\Delta(X)) \| . \tag{42}$$

3.10. Other variation forms of autoencoder

In addition to the improved model introduced earlier, the researchers also present other improved forms. For example, Least Square Variational Bayesian Autoencoder (LSVAE) [43] uses the least squares loss as the regularization term. Xie et al. proposed a discriminative autoencoder (DIAE) [44]. A Fisher discrimination criterion is added to the neurons in the hidden layer to make the hidden features distinguishable. This is very helpful for classifying tasks. By adding a large margin penalty term to the loss function of the autoencoder, a large margin autoencoder (LMAE) is formed [45]. LMAE improves the discrimination ability

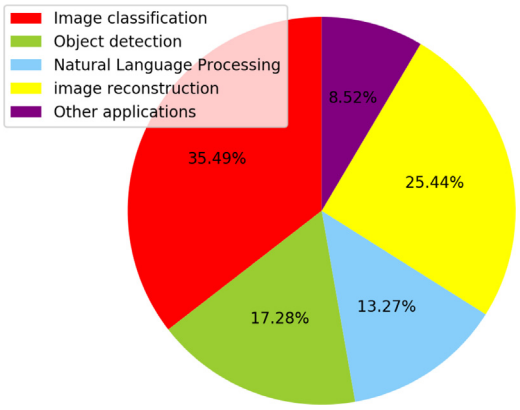

Fig. 12. The proportion of papers on autoencoders in different fields in the recent three years.
Source: The data is retrieved from the SCOPUS database in November 2022.

of the autoencoder. Luo et al. proposed a locally constrained sparse autoencoder (LCSAE) [46]. For classification problems, the locality is more important than sparsity. Therefore, a local constraint penalty term is added to the loss function of the sparse autoencoder.

Besides, there are other ways to apply kernel function in an autoencoder. Some researchers add a kernel function mapping layer in front of an encoder when the encoding processing is carried out. For example, in Ref. [47], the kernel-based denoising autoencoder (KDAE) is suggested. The data is turned into features using kernelization, and the converted features are then fed into a deep network made up of several denoising autoencoders. It is effective in minimizing the loss of data features to extract features in a reproducing kernel Hilbert space using kernel function. It demonstrates the feature representation's robustness is enhanced by the kernel method.

4. Practical applications

Fig. 12 shows the proportion of papers on autoencoders in different fields during that period. As can be seen from Fig. 12, autoencoders are widely used in image classification and image reconstruction. In contrast, there are few papers in the field of natural language processing. This means that the text needs more research than the image. It is a challenge to apply autoencoder to text datasets. Future research on autoencoders is more likely to make a breakthrough in the field of text. Next, we will introduce selected literature from different research fields which deal with autoencoders.

4.1. Image classification

Image classification is one of the hot research directions in the field of computer vision, which deals with areas such as image recognition [48], face recognition [49] and medical image diagnosis [50], etc. Image classification mainly includes feature extraction and classifier design. The feature extraction ability of the autoencoder can be widely used in this field. The following introduces the research of image classification based on an autoencoder.

For the spectral and spatial information contained in hyperspectral images (HSI), as shown in Fig. 13, Zhao et al. proposed a hyperspectral image classification method based on DSAE and 3D Depth Residual Network (3DDRN) [51]. Firstly, the dimension of the original HSI is reduced by DSAE. To alleviate the problem of gradient disappearing with the increase of CNN layers, they added a residual network module to the 3D convolutional neural network (3DCNN) to build 3DDRN [52]. The 3D HSI cube after dimension reduction is input into 3DDRN, and the recognizable joint spectral-spatial features are extracted. Finally, the Softmax classifier is used to classify the depth features identified by 3DDRN. The experimental results show that DSAE can effectively extract low-dimensional features from the original image. Even if the number of samples is limited, this method has good classification performance. Guo et al. proposed a method for crop classification of hyperspectral images by fusing a stacked autoencoder network with a CNN [53]. The method adds a convolutional neural network (CNN) behind a stacked autoencoder (SAE) to perform feature extraction on the reduced-dimensional data. This method can achieve efficient dimensionality reduction and classification of hyperspectral remote sensing data simultaneously with only a simple, unsupervised pre-training, and a single, supervised training. This method achieves a classification accuracy of 98.73% and simplifies the complex process of conventional hyperspectral image classification.

Khozeimeh et al. proposed a CNN-AE-based method to predict the chance of survival in COVID-19 patients [54]. To reduce data imbalance, a new AEs-based data enhancement method was used to process CT images, as shown in Fig. 14. First, set up 10 AEs with the same structure but different parameters. 20 samples are input into 10 trained AEs respectively. After the sample passes through the encoder and decoder of the AE, 20 reconstructed samples will be obtained, which are not the same as the original samples. Because the parameters of each encoder are different, 200 new samples are generated. The enhanced data was used for predictive classification using CNNs. After training with the enhanced data, the accuracy, recall, and specificity of CNN-AE were 96.05%, 98%, and 93.13%, respectively. Vankayalapati et al. proposed a DCNN-based autoencoder image denoising technique to obtain higher accuracy in brain tumor prediction [55]. In image classification tasks, noise in images can have an impact on the progress of classification. Denoising autoencoders have a wide range of applications in denoising. Denoising autoencoders learn input features during image reconstruction to better extract potential representations. Therefore, a combined DAE and DCNN autoencoder is proposed for image denoising. Comparative tests show that this method has good results.

Geng et al. proposed a new deep contraction neural network model (DSCNN) for synthetic aperture radar (SAR) image classification [56]. The gray level co-occurrence matrix, Gabor filter, and histogram of oriented gradient descriptors are used to extract preliminary features from SAR images. A DSCNN network composed of multiple contractive autoencoders (CAE) is trained for supervised optimization classification. The DSCNN network consists of a multi-layer supervised compression automatic encoder (SCAE), as shown in Fig. 15. Each CAE in SCAE is connected to a multiple logistic regression (MLR) module, whose chief responsibility is to add labels and update model parameters while the model is being trained.

$$J^m(\theta^m) = \arg\min(J^m_{pretrain}(\theta^m) + \gamma J^m_{update}(\theta^m)). \tag{43}$$

As a result, pre-training (updating the weight parameters of the autoencoder) and updating the term markers are aspects of the SCAE module training process. The objective function of SCAE is shown in the formula (43), where $J^m_{pretrain}(\theta^m)$ and $J^m_{update}(\theta^m)$ are the pre-training term pre-train update and updating term, respectively. θ^m is the weight of the hidden layer, and $\gamma \in (0, 1)$ is a balancing coefficient. The relationship between implicit characteristics and labels can be efficiently captured by a DSCNN with a supervisory layer. Therefore, DSCNN can effectively represent the

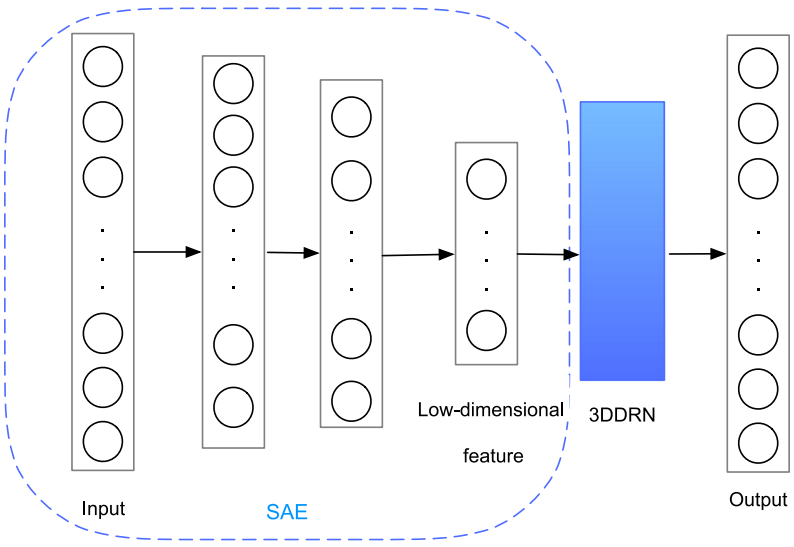

Fig. 13. DSAE-3DDRN model's general structural diagram. After extracting features from the data with the DSAE, classification is carried out using a deep residual neural network (3DDRN).

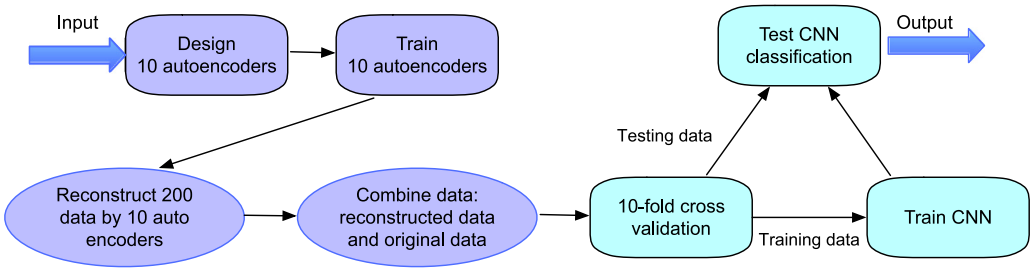

Fig. 14. Flow chart of the primary steps of the CNN-AE method. A convolutional neural network is used to classify the data after it has been improved using autoencoders.

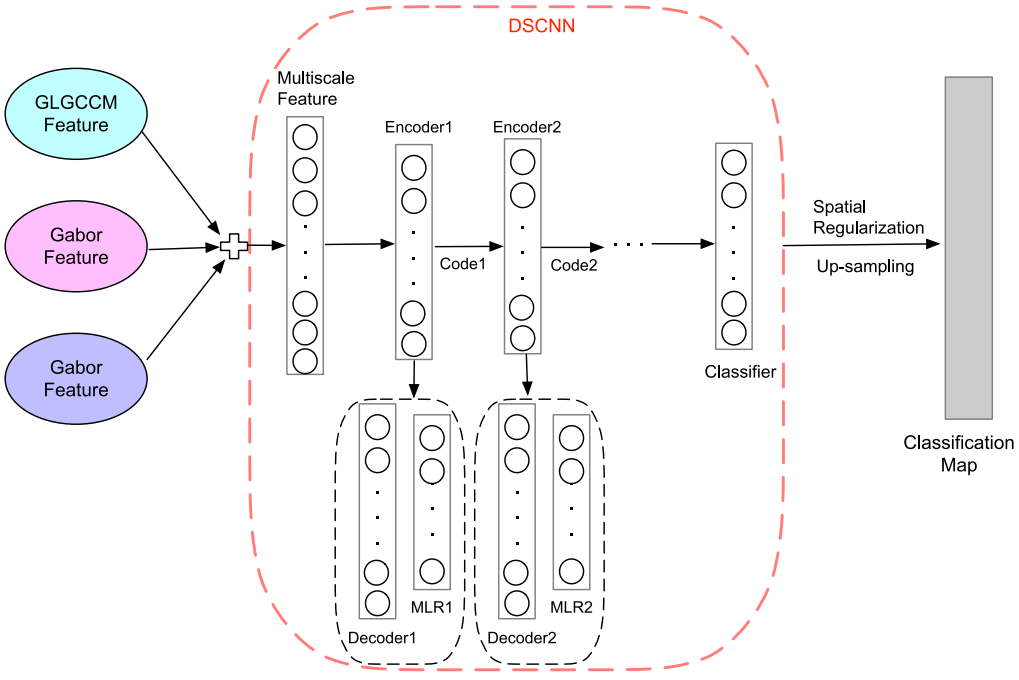

Fig. 15. Framework for the SAR image classification. A neural network constructed by several contractive autoencoders is used to classify three separate features after they have been combined.

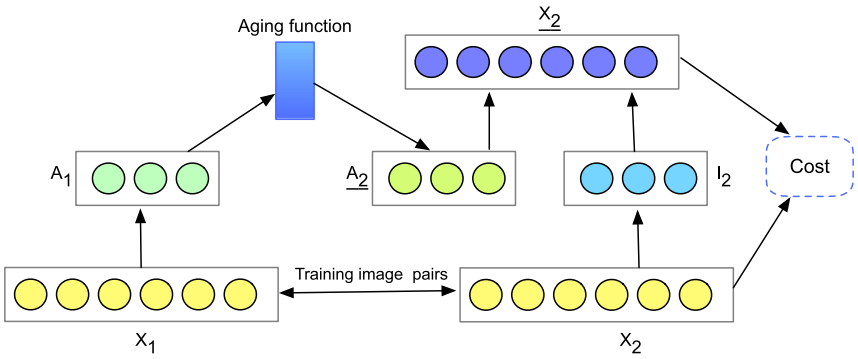

Fig. 16. Aging fitting neural network. X_1 and X_2 are images of the same person at different ages. A_1 and $\underline{A_2}$ are age characteristics, and I_2 is identity characteristics. The image $\underline{X_2}$ is reconstructed by using age features and identity features.

sample features and make excellent predictions on the category labels.

Cheng et al. have put forward a method called double metric learning (DML) for image set classification [57]. This method consists of two parts: feature learning and image classification. First, a metric learning regularization term is added to the hidden neurons of the Discriminant Stacking Autoencoder (DSAE) to obtain a new feature mapping. The purpose of this is to make similar samples closer together in the mapping space. Second, the classifier is trained differently. Fine-tuning DSAE by optimizing the objective function with the regularization term. Finally, based on learning the classifier, two simple voting strategies are designed to classify the image sets. Wang et al. conducted comparative image classification experiments using AE and analyzed the effect of the number of neurons in the hidden layer on the classification accuracy based on the results [58]. The experimental results showed that in the case of simple images, such as MNIST images [59], the classification accuracy could reach over 93% when the number of neurons in the hidden layer was close to the dimensionality of the input data. If the number of neurons is increased further, there is no improvement in the classification accuracy. However, for complex face images, the classification accuracy improves as the number of hidden layer neurons increases. The results of this study provide a basis for exploring the structure of the hidden layer of the autoencoder.

4.2. Object detection

Object detection is one of the burgeoning research fields of artificial intelligence. Target detection has great practical value and application prospects in people's life, such as face detection [60], pedestrian detection [61], video detection [62], medical image detection [63] and fault detection [64], etc. Deep learning technology, represented by an autoencoder, can realize automatic extraction of target features and get rid of the limitation of manual extraction. This also effectively improves the accuracy of target recognition. The following introduces the research of target detection based on an autoencoder.

Xu et al. proposed a coupled autoencoder network (CAN) model for face recognition and retrieval [65]. CAN is a pair of autoencoders connected by two shallow neural networks, which is used to fit complex nonlinear aging and de-aging processes. The conventional autoencoder is used to reconstruct a pair of images of the same person of different ages. The nonlinear factor analysis method is used to decompose the hidden layer expression of the conventional autoencoder, and some characteristic information is obtained. These features include the identity feature (I) which is age-invariant, the age feature (A), and noise. Finally, two shallow neural networks are used to connect two conventional autoencoders to fit the aging and de-aging processes respectively. In this way, the face recognition task with age invariance can be realized. As shown in Fig. 16, Given inputs (X_1, X_2) of one person at different ages. They use aging fitting output combined $\underline{A_2}$ with target identity feature I_2 to reconstruct the older facial image $\underline{X_2}$. The process of de-aging is the opposite. Experiments show that this method is effective for face recognition with changing age. Guo et al. proposed a new compact convolutional automatic encoder (CCAE) for target recognition in SAR images [66]. By imposing compactness constraints on the convolution autoencoder, better feature extraction can be achieved. This method adds the regularization term of sample distance in the class to the loss function. By minimizing the reconstruction error and the distance between samples in the class, more distinctive abstract features can be generated, and the effectiveness of the method is verified by experiments on MSTAR data sets. This method is superior to the existing methods based on deep learning in the case of small-sample training.

Wen et al. combined SAE with transfer learning and proposed a fault diagnosis method based on deep transfer learning (DTL) [67]. As shown in Fig. 17, the method adopts three-layer SAE to extract the features of the original data and adds the maximum mean discrepancy (MMD) penalty term to the network loss function. To obtain the features of the implicit correlation between resource files and target files, they build a multi-layer SAE. Using the labels from the resource files, the parameters of the entire model are adjusted during the training process. The transfer model's goal function for optimization is the formula (44). This function adds the MMD constraint regularization term for the difference between two data characteristics in comparison to the loss function of the autoencoder model.

$$J_{DTL}(\theta) = Loss(Y_s, \hat{Y}_s) + \mu MMD(F^t, F^s). \tag{44}$$

SAE is used to extract abstract features of samples layer by layer, and the difference between abstract features of training and testing samples is minimized. Experiments show that the prediction accuracy of this method can reach 99.82%. It is an advantageous method in the field of fault diagnosis. Chen et al. proposed a hash addressable memory autoencoder with multi-scale attention blocks to detect abnormal CT images [68]. The input image is reconstructed by constructing a convolution autoencoder (CoAE) network. If there are abnormal conditions in the image, there will be a high reconstruction error. On the contrary, it produces a lower error-rate. To alleviate the problem of restricted static convolution operators, the design of a multi-scale attention block is added. This method also introduces a hash storage module for fast retrieval to prove that exceptions

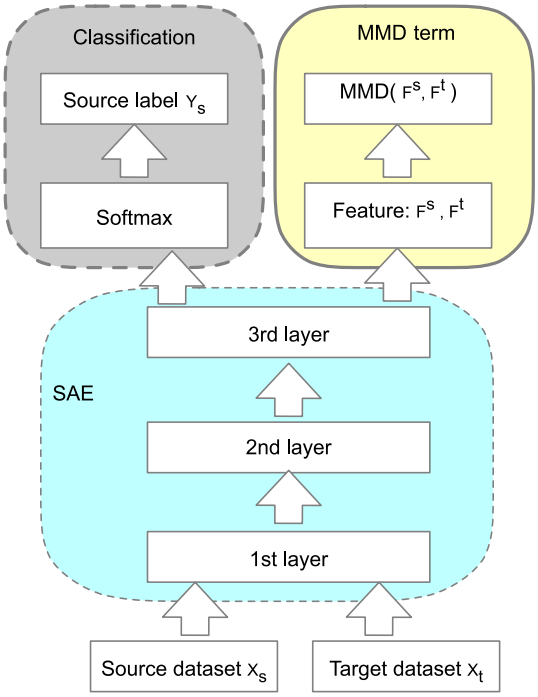

Fig. 17. Flow chart for the deep transfer learning. The multi-layer autoencoder network receives input from both the resource data and the target data in order to extract features and classify the data.

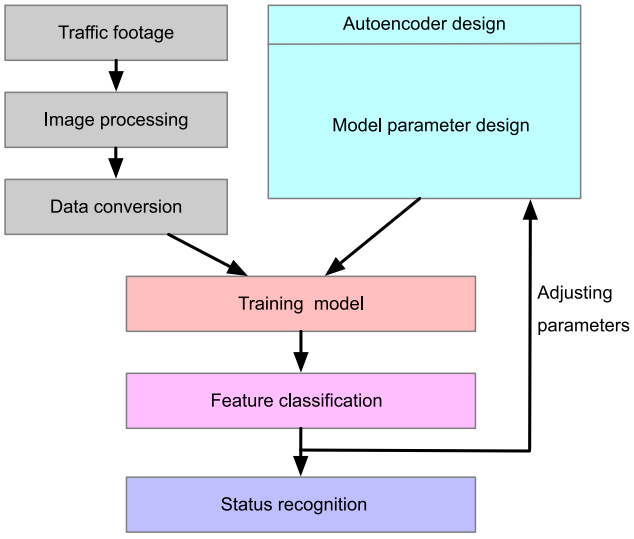

Fig. 18. The steps for traffic state recognition. The video data is processed by image processing, and the image set is input to the designed autoencoder model for training. The autoencoder model classifies images to implement traffic state recognition.

will produce higher classification reconstruction errors. In addition, MSE and Wasserstein loss will be combined to improve the distribution of coded data.

Yang et al. proposed a fault diagnosis method based on sparse autoencoder (SAE) and multi-head deep neural network (DNN) [69]. Multi-head DNN includes an encoder, decoder, and classifier. High-level representation is extracted from input data by using a multi-layer nonlinear transformation of an encoder. The extracted features are input by the decoder and classification module. The purpose of the decoder is to reconstruct the input data, and the classification module is the label for predicting the input data. In addition, the multi-head DNN is directly trained by modifying the linear unit activation function, which reduces the training calculation. The experimental results show that this method has achieved satisfactory results in new things detection and fault diagnosis. Gao et al. proposed a fault diagnosis method for high-voltage switches based on the combination of a semi-supervised stack autoencoder (SSAE) and an integrated extreme learning machine (IELM) [70]. First, the signal is decomposed by fully integrated empirical mode decomposition and adaptive noise. Thus, the time–frequency energy matrix is obtained. Second, the semi-supervised stack autoencoder is used to automatically extract the features of the energy matrix. Finally, the integrated extreme learning machine is used to establish a two-level classifier. The first level is used to identify normal or abnormal states, and the second level is used to identify specific fault types in abnormal states. Experimental results show that the classification accuracy of this method reaches 99.5%.

Extracting traffic data from the original video is a complicated task. To reduce the complexity of the algorithm, As shown in Fig. 18, a technique for intelligent traffic scenario identification is presented by Cai et al. [71]. The approach optimizes the structural parameters to successfully reduce the image's dimensionality. In order to extract features from images and reduce their dimensionality, a deep autoencoder with numerous layers is first built. The five lightweight classifiers were then each integrated with the specified autoencoder model. This approach yields a number of traffic state identification models, including AE+Linear, AE+SVM, AE+DNN, AE+DNN_Linear, and AE+k-means. The lightweight design approach suggested in this paper outperforms more intricate deep convolutional neural network-based models for traffic status recognition, according to experimental results. Additionally, this lessens the complexity of models for intelligent traffic detection. Target tracking is an important problem in computer vision. Generally speaking, the environment of the identified target is complex. Consequently, the recognition results are easily affected by various factors, such as illumination, occlusion, etc. To solve this problem, Li et al. [72] used the mean shift (MS) algorithm of the local probability model (LPM) to combine target detection and tracking. Thereby realizing the integration of detection and tracking in a complex environment. In the aspect of target detection, a deep learning method based on SDAE is used to train and predict LPM. The experimental results show that SDAE has a good extraction effect for features in a complex environment, thus increasing the robustness of the model.

4.3. Natural language processing

The field of natural language processing artificial intelligence mainly focuses on the research direction of text data processing, such as sentiment analysis [73], text classification [74], medical case diagnosis [75], speech recognition [76], and recommendation algorithm [77], etc. A common feature of voice and text data is serialization. Compared with image data, serialized text data is more difficult to process and extract continuous features. With the development of the autoencoder, some complicated serialization models have been proposed by researchers. This effectively solves the problem of text data processing. The following introduces the research of natural language processing based on autoencoder.

Dervishaj et al. proposed a new recommendation method using GAN-based matrix factorization (GANMF) [78]. That is, to learn the potential factors of users and projects in matrix decomposition settings, and use them for general top-N recommendation questions. Given a group of users U, a group of items I, and

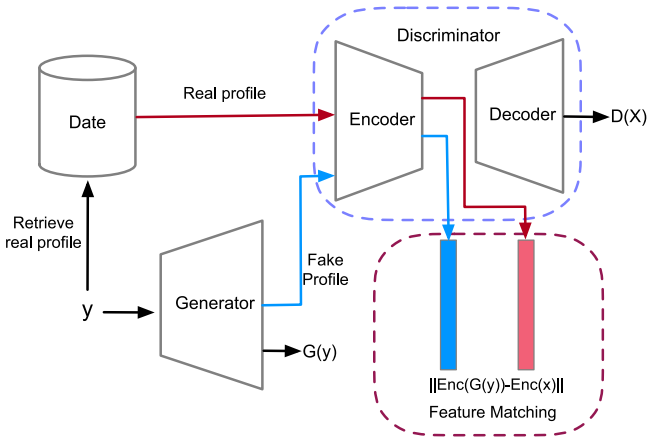

Fig. 19. The diagram of GAN-based matrix factorization architecture. An autoencoder implements the discriminator function in a GAN model. The user information build restriction is added. The recommendation work is finished by adding feature matching to the GAN.
Source: Adopted from [78].

users' feedback on these items in the past, the top-N recommendation is a question of recommending a subset of items from I that each user u is more likely to like. Organize past feedback into the shape $|U| \times |I|$ of a user rating matrix (URM). Each cell (u, i) of URM represents the user's feedback on the product. In this case, if the user is interested in item i, the value of cell (u, i) of URM is 1, otherwise, it is 0. The user record file is stored in each line of the URM module, and the project description file is stored in each column. The GANMF model, which is based on the GAN model to implement some method advancements, is depicted in Fig. 19. An additional constraint is introduced to the generator and the classifier discriminator in the GAN model is transformed into an autoencoder. The generator is modified by user label y during model training to produce the appropriate user files. Distinguishing between created user files and the actual user files kept in the URM is the discriminator's job. The experimental findings demonstrate that adding a feature matching mechanism to the GAN model can significantly enhance model performance. It also shows that GAN is a promising technology for recommendation systems.

To deal with the text data anonymization, Weggenmann et al. proposed an end-to-end differential private variational autoencoder model [79]. After the experimental evaluation of online reviews, it effectively reduces the risk of re-identification against author attribution attacks, while retaining the content of the text. It also shows that decoding the confusing latent vectors can produce coherent, high-quality, and human-readable output text. Because of the nature of VAEs, this method also has a good migration ability. Xu et al. proposed a semi-supervised sequence variational self-encoder for semi-supervised text classification [80]. By taking the category labels of unlabeled samples as discrete latent variables, this method maximizes the lower bound of likelihood variation of samples, thus implicitly deducing the potential category distribution of unlabeled samples, and by solving the autoregressive problem of sequence decoder, it can be applied to text classification.

An intelligent medical learning model is suggested by Chandru et al. [81] to assist patients in the clinical disease diagnosis process. For the purpose of clustering and predictive modeling of medical data, the model combines Work2vec with an autoencoder. The model also includes a long-term memory (LSTM) module and a convolutional neural network (CNN) module for interpreting text presented as ordered series. In order to derive contextual interdependencies of ordered data, the characteristics of both models are integrated. Fig. 20 shows the structure of the method. Extraction of term frequency (TF) and inverse document frequency (IDF) is the function of the multi-layer denoising autoencoder. The emblems used for classification are produced by this procedure. An attention mechanism is included in the two LSTM models for comparison prediction to improve accuracy. The type of entrance has an impact on the weight of this attention layer. If the patient is hospitalized via an outpatient department (OD), one set of weights is used, and if they are admitted via an emergency department, a different set of parameters are utilized. However, when the patient chooses the OD entry option, the LSTM pays closer attention to the patient's testimony and chooses a set of parameter values on its own. Therefore, they combine two specialized neural networks into one. Finally, the purpose of predictive diagnosis is achieved.

For sentiment analysis and text classification, As shown in Fig. 21, Bhaskaran et al. designed a new and modified red deer algorithm (MRDA) extreme learning machine sparse autoencoder (ELMSAE) model [82]. ELMSAE model is used for sentiment classification of text data. In addition, in the process of classification, TF-IDF is used to vectorize features. The parameters of the ELMSAE model are optimized and adjusted by MRDA technology. Experiments show that the MRDA-ELMSAE method has great advantages in dealing with emotion analysis problems.

Dahmani et al. used different neural structures to synthesize emotional language and studied the application of unsupervised learning technology in emotional language modeling [83]. By reconstructing emotional representation, the synthesized speech was made more continuous and flexible. They use Conditional Variational Automatic Coding (CVAE) architecture to learn the potential expression of emotions and to generate characteristic

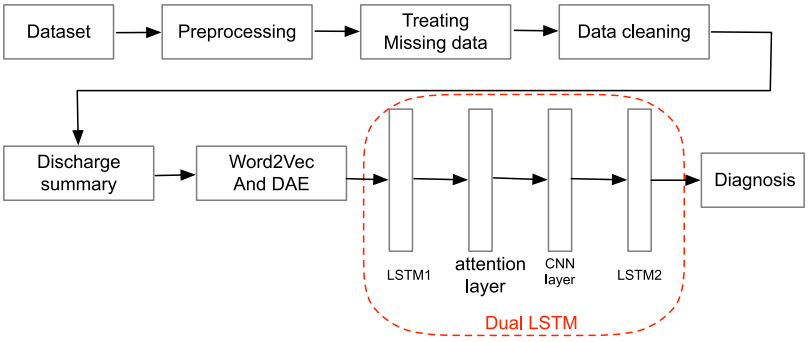

Fig. 20. The steps for implementing the diagnosis method. The first step is to preprocess and sanitize the medical text data. After the encoder's word frequency statistics processing, the data is entered into a dual LSTM for processing, which concludes the task of intelligent diagnosis.

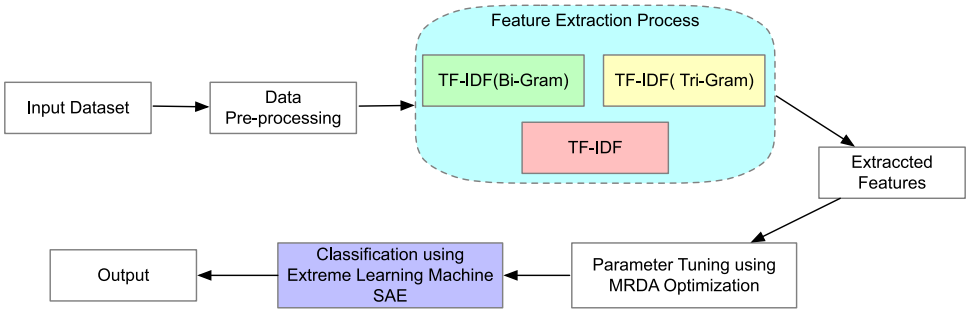

Fig. 21. The steps for sentiment analysis and text classification method. After data preprocessing, vectorization and feature extraction are carried out. The modified red deer algorithm is used to optimize and adjust the parameters, and the features are input into the extreme learning machine sparse autoencoder model for text classification.

expressive speech. The overlapping degree between emotion latent clusters is calculated by the probability measurement method, to select the best emotion latent cluster parameters. By manipulating the latent vectors, we can produce subtle differences in specific emotions and obtain phonetic expressions with different emotions.

4.4. Other applications

Besides the above-mentioned application fields, autoencoders are also used in image reconstruction [84,85], image enhancement [86,87], network security [88,89], financial analysis [90] and signal enhancement [91,92], etc. Through a large number of literature studies, it can be found that according to the characteristics of different research fields, there will be different ways to improve the autoencoder, which can be summarized as follows: (1) According to the changes in research fields, change the loss function of the autoencoder and add specific penalties. (2) Autoencoder is combined with conventional feature extraction methods to extract features, such as various filters. (3) Adjust the feature extraction method of an autoencoder, such as a convolution autoencoder. The convolution operation preserves the spatial information characteristics of two-dimensional data well. (4) Combine the autoencoder with other well-behaved models, and establish the relationship between the models by some methods. These improved models make autoencoders play an increasingly important role in people's lives.

5. Comparison and discussion

Table 2 summarizes and compares the above-mentioned autoencoders, and lists the characteristics (including advantages and disadvantages) of various automatic encoders, which can be used for researchers or applications to quickly understand the characteristics of this method. The main function of the autoencoder is to automatically extract effective features, so the hidden layer expression needs to be robust to the input noise. Through the study of the autoencoder and its improved model, it can be found that the main way to improve the autoencoder is to add different regularization terms based on the loss function. The purpose of adding regular terms is to constrain the hidden layer expression, to achieve different effects, such as SAD and CAE. The deep autoencoder needs to be trained layer by layer, so some autoencoders that shorten the training time, such as K-SAE, were born. For the images, the convolution autoencoder has great advantages.

As shown in Table 3, we extracted some experimental results of autoencoder classification on MNIST data set from the literature [93]. We can see that the convolution autoencoder is the best for image classification. It also shows that spatial features have a great influence on the classification effect of images. DAE realizes the robustness of the model by adding noise processing. CAE, SAE, etc. increase the robustness of the model in the hidden layer expression, so the classification effect of SAE is better than that of DAE. How to improve the robustness of implicit expression is an important direction to improve the autoencoder.

In recent years, generative models have developed rapidly in the fields of computer vision and natural language processing. Generative models are mainly classified into three categories: The first is the generative model which calculates the approximate distribution of likelihood functions by variational or sampling methods, such as VAE. The second is the generative adversarial network, which uses the learning ability of the neural network to fit the distance between two distributions, such as the GAN network. The third is to transform the likelihood function to simplify the calculation, mainly including the flow model [94] and autoregressive model, such as PixelRNN, etc. Table 2 lists some improved generative models based on VAE and their characteristics. It can be seen from the literature that most of the improvements of VAE are to change the variational lower bound. Therefore, the variational lower bound has a great influence on the generative ability.

As the simplest generative model, VAE has limited ability to generate data due to its shortcomings. For image data, the generative ability of VAE is not as good as that of the GAN-based model. Some researchers combine VAE with GAN to improve the ability of data generation, such as CVAE-GAN. After the addition of the GAN model, the generative ability has been improved, but its shortcomings have also been introduced, such as an overly complicated model and unstable training process, resulting in Model Collapse. In chapter 4, we can see that VAE is usually used as a feature extractor in the image field, but in the field of natural language processing, the VAE model only needs a simple structure to generate smooth text data. Therefore, it will be a good research direction to discover the advantages of VAE in the field of natural language processing and improve it.

The autoregressive neural networks express joint probability distribution by conditional probability product. It has a great influence on data generation, such as PixelCNN and PixelNN. Table 4 shows the comparison results of some generated models on the CIFAR-10 data set [101]. It can be seen that the generative model of the autoregressive network has a good performance. Therefore, researchers combine VAE with an autoregressive network model to improve the ability of image generation, such as PixelVAE. The ability of data generation is improved, but the introduction of an autoregressive network leads to an increase in the amount of computation required for training and sample generation. PixelVAE++ model is to increase the generation capacity and change the structure of the decoder, to achieve the purpose of compressing the computation. NAVE uses multi-scale

Table 2
Comparison of several autoencoder models. The main characteristics (Including advantages and disadvantages) of different autoencoder models.

Auto-encoder name	Characteristics of the model
DAE	Adding noise to the input data makes the model robust.
	High computational cost and lack of scalability to high-dimensional features
SAE	Add sparse regularization term to obtain sparse expression of signal
CAE	Add contraction regularization term
	The model will not be affected by small changes in input data
MDAE	Considers not only the sensitivity of reconstruction function to hidden
	layer expression, but also the sensitivity of hidden layer expression to the input signal.
DIAE	Add regularization items of intra-class distance and inter-class distance.
	Enhance the distinguishability of the extracted features.
LMAE	Increase the distance between different categories in the hidden space
	to improve the distinguishing ability.
LCSAE	The sparse constraint of SAE was changed.
	Use similar features to encode similar inputs.
CoAE	The convolution layer and the pooling layer are used to replace the full connection layer.
	The two-dimensional space features of the image are preserved.
VAE	Data generation model with a simple structure for easy calculation
	Insufficient expressiveness for complex models
CVAE	The category control is realized, and the data can be generated directionally.
	The generated images have poor quality and diversity.
VFAE	Separate noise from hidden variable information. Using MMD as a regularization term.
	The generated category control is realized, but the image quality is not high.
VLAE	Fast training and strong generating ability. Not applicable to serialization models.
CVAE-GAN	High-quality images of different categories can be generated.
	Hidden variable coding has a great influence on category information.
IFCVAE-GAN	Generate images of finer-grained categories. The complex structure and high time complexity.
	The structure is complex and the model is unstable.
CRVAE	High-quality images can be generated. Suitable for complex and serialized models.
	High structure complexity and long training time.
NVAE	Using depthwise separable convolutions and residual parameterization of normal distributions
	to generate high-quality big pictures, the training time is shortened.
DCVAE	The features generated based on different modality information are unified into the final synthetic features.
	The data enhancement effect is obvious.
PixelVAE++	Combine PixelCNN++ architecture with VAEs to generate high-quality large images.
	The calculation amount is compressed.
WAE	Introducing Wasserstein distance to generate good-quality images
	The generative effect of large-size images needs to be improved.
CWAE	Introducing Cramer-Wold distance to generate good-quality images
	Training is faster and more stable; Insensitive to the change of training parameters
KAE	Feature extraction is interpretable. The effect is influenced by the choice of the kernel function.
KLAE	Kernel method is used instead of nonlinear activation function to realize classification.
	This method can only be used to generate features, not to synthesize data.

Table 3
Comparison of experimental results of some autoencoders on MNIST data set. Its evaluation index is the error rate.

Autoencoder name	Error rate
AE	1.78
ReAE	1.68
DAE	1.28
MDAE	1.37
CAE	1.14
SAE	0.97
CoAE	0.71

convolution and residual structure to improve the generation ability and reduce the calculation amount. The NLL index value of NAVE is very close to that of PixelVAE++. This also pushes the generative effect of VAE to a new height. It can also be seen from Table 4 that for the VAE model, flow model, and autoregressive network, the generative capacity increases in turn. Generally, the generative models with autoregressive network components have higher accuracy. With the enhancement of generative ability, the calculation amount of the model is also increasing. How to reduce the calculation amount of the autoregressive network will also be a development direction of the autoregressive generation model.

Some scholars are studying the kernel of autoencoders, such as KAE and KLAE. This autoencoder has a simple structure and has achieved good results for classification tasks. Compared with other autoencoder models, it has a better explanation for feature extraction. For the KAE, the choice of kernel function is a difficult problem. Different parameter settings have a great influence on the results. Therefore, how to optimize the parameter selection process is the improvement direction of KAE. KLAE can only generate features, but not data. This will also be a direction of improvement. The study of this problem will also be a direction for its improvement.

Table 4
Comparison results of some important generation models on CIFAR-10 data set. The evaluation criteria of model performance are Frechet Inception Distance (FID) and negative log-likelihood (NLL). Lower is better in all cases.

Autoencoder name	FID	NLL
CWAE	120.02	–
WAE	129.37	–
NVAE	51.71	2.91
RAE [95]	74.16	–
DC-VAE	21.4	–
VFlow [96]	–	2.98
Flow++ [97]	–	3.08
DVAE++ [98]	–	3.38
VLAE	–	2.95
IAF-VAE [99]	–	3.11
PiexlRNN	–	3.00
PiexlCNN	–	3.14
PiexlCNN++ [100]	–	2.92
PiexlVAE++	–	2.90

For future research, the autoencoder has achieved good results in many fields, but there are still more problems to be solved. Therefore, the future improvement direction of the autoencoder model may be as follows:

- The training of complex multi-layer autoencoders is too time-consuming. It is also more demanding on the hardware. To address this problem, one can consider introducing some compression algorithms, such as pruning algorithms to remove some useless nodes, to achieve a streamlined network structure. Alternatively, the construction of the autoencoder can be improved so that the structure of the autoencoder is lightweight. In addition, the training time can be reduced by improving the way the autoencoder is trained and by introducing distributed optimization algorithms.
- Autoencoders and their deep structures are prone to overfitting when the amount of data is smaller than it should be, due to their complex structure. Currently, small sample learning is a hot topic in the field of deep learning. For autoencoders based on a small sample, learning will be a development direction.
- Accurate unsupervised learning is the most desirable learning model. Currently, for most autoencoders, especially those with multi-layer structures, a supervised learning step is added to the application to adjust the parameters of the whole network. How to achieve true unsupervised learning is also a problem that will be addressed in the future.
- For the construction of the autoencoder, the choice of some parameters also needs to be thought about, such as the number of hidden layers, the number of nodes, etc. The manner of selecting the most suitable parameters has a great impact on the final result of the model. Currently, some researchers try to combine genetic algorithms with the construction of autoencoders. It would be meaningful to use some optimization algorithms to find the optimal parameters.
- For the improvement of variational autoencoder, how to separate information effectively is the most researched direction. Many improved models weaken the interference between unrelated features, to improve the accuracy of the table method. So how to separate information more efficiently will be one of the development trends of the variational autoencoder.
- Some scholars describe deep neural network models as black boxes. These models do not clearly illustrate how important features are extracted and associated. Therefore, the interpretability of deep autoencoder will become a development direction in the future. For example, KAE may be a good interpretable method.

6. Conclusion

In recent years, autoencoders have played an important role in many fields. Researchers have deeply studied the autoencoder and its improved algorithm and put forward many improved models. In this paper, various autoencoders and related improved models were comprehensively expounded. This paper mainly introduced the principles and structures of various improved autoencoders and classified and summarized them according to their different characteristics. The primary characteristics of the autoencoder and its application status in different fields were also explained. Through comparison and discussion, the existing shortcomings and the future development direction were summarized. Future research will make the autoencoder even more powerful and advantageous, and provide even more intelligent services for the benefit of human society.

7. Survey methodology

In this survey, the papers we searched were obtained from Scopus, Google Scholar, and arXiv databases. We searched for papers with the topic of autoencoder, and the filtering conditions were “auto-encoder” or “autoencoder” keywords. After the papers are screened, we select the relevant papers, which we then describe according to their number of citations, publication time, titles, and related research fields. Some papers with a low number of citations and those with little influence were excluded. For papers with similar methods and fields, we choose papers with a large number of citations.

CRediT authorship contribution statement

Pengzhi Li: Complete the manuscript of the paper, Writing – original draft. **Yan Pei:** Conceptualization, Methodology, Supervision, Writing – review & editing. **Jianqiang Li:** Data curation, Ideological guidance.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgment

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

[1] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444, http://dx.doi.org/10.1038/nature14539.
[2] D.H. Hubel, T.N. Wiesel, Receptive fields of single neurones in the cat’s striate cortex, J. Physiol. 148 (3) (1959) 574, http://dx.doi.org/10.1113/jphysiol.1959.sp006308.
[3] G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507, https://science.org/doi/10.1126/science.1127647.

[4] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput. 1 (4) (1989) 541–551, http://dx.doi.org/10.1162/neco.1989.1.4.541.
[5] G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (7) (2006) 1527–1554, http://dx.doi.org/10.1162/neco.2006.18.7.1527.
[6] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, Nature 323 (6088) (1986) 533–536, http://dx.doi.org/10.1038/323533a0.
[7] H. Bourlard, Y. Kamp, Auto-association by multilayer perceptrons and singular value decomposition, Biol. Cybernet. 59 (4) (1988) 291–294, http://dx.doi.org/10.1007/BF00332918.
[8] P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol, Extracting and composing robust features with denoising autoencoders, in: Proceedings of the 25th International Conference on Machine Learning, 2008, pp. 1096–1103, http://dx.doi.org/10.1145/1390156.1390294.
[9] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, L. Bottou, Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion, J. Mach. Learn. Res. 11 (12) (2010) 3371–3408, URL https://dl.acm.org/doi/10.5555/1756006.1953039.
[10] A. Ng, Sparse Autoencoder, in: CS294A Lecture Notes, vol. 72, (no. 2011) 2011, pp. 1–19.
[11] A. Makhzani, B. Frey, K-sparse autoencoders, 2013, http://dx.doi.org/10.48550/arXiv.1312.5663.
[12] S. Rifai, G. Mesnil, P. Vincent, X. Muller, Y. Bengio, Y. Dauphin, X. Glorot, Higher order contractive auto-encoder, in: Machine Learning and Knowledge Discovery in Databases, 2011, pp. 645–660, http://dx.doi.org/10.1007/978-3-642-23783-6_41.
[13] S. Rifai, P. Vincent, X. Muller, X. Glorot, Y. Bengio, Contractive auto-encoders: Explicit invariance during feature extraction, in: Proceedings of the 28th International Conference on International Conference on Machine Learning, ICML '11, 2011, pp. 833–840, URL https://dl.acm.org/doi/abs/10.5555/3104482.3104587.
[14] M. Chen, Z. Xu, K. Weinberger, F. Sha, Marginalized denoising autoencoders for domain adaptation, in: Proceedings of the 29th International Coference on International Conference on Machine Learning, 2012, pp. 1627–1634, URL https://dl.acm.org/doi/10.5555/3042573.3042781.
[15] M. Chen, K. Weinberger, F. Sha, Y. Bengio, Marginalized denoising auto-encoders for nonlinear representations, in: Proceedings of the 31st International Conference on Machine Learning, Vol. 32, No.2, 2014, pp. 1476–1484, URL https://dl.acm.org/doi/10.5555/3044805.3045057.
[16] C. Tao, H. Pan, Y. Li, Z. Zou, Unsupervised spectral–spatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification, IEEE Geosci. Remote Sens. Lett. 12 (12) (2015) 2438–2442, http://dx.doi.org/10.1109/LGRS.2015.2482520.
[17] R. Ranjan, V.M. Patel, R. Chellappa, Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition, IEEE Trans. Pattern Anal. Mach. Intell. 41 (1) (2017) 121–135, http://dx.doi.org/10.1109/TPAMI.2017.2781233.
[18] D.P. Kingma, M. Welling, Auto-encoding variational bayes, 2013, http://dx.doi.org/10.48550/arXiv.1312.6114.
[19] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, Adversarial autoencoders, 2016, http://dx.doi.org/10.48550/arXiv.1511.05644, arXiv.
[20] C. Louizos, K. Swersky, Y. Li, M. Welling, R. Zemel, The variational fair autoencoder, 2015, http://dx.doi.org/10.48550/arxiv.1511.00830.
[21] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, W. Zuo, Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 945–954, http://dx.doi.org/10.1109/CVPR.2017.107.
[22] J. Bao, D. Chen, F. Wen, H. Li, G. Hua, CVAE-GAN: fine-grained image generation through asymmetric training, in: 2017 IEEE International Conference on Computer Vision, ICCV, 2017, pp. 2764–2773, http://dx.doi.org/10.1109/ICCV.2017.299.
[23] L. Mescheder, S. Nowozin, A. Geiger, Adversarial variational Bayes: Unifying variational autoencoders and generative adversarial networks, in: Proceedings of the 34th International Conference on Machine Learning, Vol. 70, PMLR, 2017, pp. 2391–2400, URL https://dl.acm.org/doi/10.5555/3305890.3305928.
[24] A. Creswell, Y. Mohamied, B. Sengupta, A.A. Bharath, Adversarial information factorization, 2017, http://dx.doi.org/10.48550/arXiv.1711.05175.
[25] X. Chen, D.P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, P. Abbeel, Variational lossy autoencoder, 2016, http://dx.doi.org/10.48550/arXiv.1611.02731.
[26] A. Van Oord, N. Kalchbrenner, K. Kavukcuoglu, Pixel recurrent neural networks, in: Proceedings of the 33rd International Conference on Machine Learning, Vol. 48, PMLR, 2016, pp. 1747–1756, URL https://dl.acm.org/doi/10.5555/3045390.3045575.
[27] W. Shang, K. Sohn, Y. Tian, Channel-recurrent autoencoding for image modeling, in: 2018 IEEE Winter Conference on Applications of Computer Vision, WACV, 2018, pp. 1195–1204, http://dx.doi.org/10.1109/WACV.2018.00136.
[28] J. Masci, U. Meier, D. Cireşan, J. Schmidhuber, Stacked convolutional auto-encoders for hierarchical feature extraction, in: International Conference on Artificial Neural Networks, Springer, 2011, pp. 52–59, http://dx.doi.org/10.1007/978-3-642-21735-7_7.
[29] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, P. Abbeel, InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets, in: Advances in Neural Information Processing Systems, Vol. 29, Curran Associates, Inc., 2016, http://dx.doi.org/10.48550/arXiv.1606.03657.
[30] I. Gulrajani, K. Kumar, F. Ahmed, A.A. Taiga, F. Visin, D. Vazquez, A. Courville, PixelVAE: A latent variable model for natural images, in: 5th International Conference on Learning Representations, ICLR 2017, 2017, http://dx.doi.org/10.48550/arXiv.1611.05013.
[31] A. van den Oord, N. Kalchbrenner, K. Kavukcuoglu, Pixel recurrent neural networks, in: Proceedings of the 33rd International Conference on Machine Learning, Vol. 48, 2016, pp. 1747–1756, http://dx.doi.org/10.48550/arXiv.1601.06759.
[32] H. Sadeghi, E. Andriyash, W. Vinci, L. Buffoni, M.H. Amin, PixelVAE++: Improved PixelVAE with discrete prior, 2019, http://dx.doi.org/10.48550/arXiv.1908.09948.
[33] G. Parmar, D. Li, K. Lee, Z. Tu, Dual contradistinctive generative autoencoder, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2021, pp. 823–832, http://dx.doi.org/10.1109/CVPR46437.2021.00088.
[34] A. Vahdat, J. Kautz, NVAE: A deep hierarchical variational autoencoder, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, 2020, pp. 19667–19679, URL https://dl.acm.org/doi/abs/10.5555/3495724.3497374.
[35] Y. Zhang, S. Huang, X. Peng, D. Yang, Dizygotic conditional variational AutoEncoder for multi-modal and partial modality absent few-shot learning, 2021, http://dx.doi.org/10.48550/arXiv.2106.14467, arXiv preprint arXiv:2106.14467.
[36] I. Tolstikhin, O. Bousquet, S. Gelly, B. Schoelkopf, Wasserstein auto-encoders, in: International Conference on Learning Representations, 2018, http://dx.doi.org/10.48550/arXiv.1711.01558.
[37] S. Knop, P. Spurek, J. Tabor, I. Podolak, M. Mazur, S. Jastrzebski, Cramer-wold auto-encoder, J. Mach. Learn. Res. 21 (1) (2020) 6594–6621, URL https://dl.acm.org/doi/abs/10.5555/3455716.3455880.
[38] Y. Pei, Autoencoder using kernel method, in: 2017 IEEE International Conference on Systems, Man, and Cybernetics, SMC, IEEE, 2017, pp. 322–327, http://dx.doi.org/10.1109/SMC.2017.8122623.
[39] V.Q. Dang, Y. Pei, A study on feature extraction of handwriting data using kernel method-based autoencoder, in: 2018 9th International Conference on Awareness Science and Technology, ICAST, IEEE, 2018, pp. 1–6, http://dx.doi.org/10.1109/ICAwST.2018.8517169.
[40] F. Karl Pearson, LIII. On lines and planes of closest fit to systems of points in space, Lond. Edinb. Dublin Philos. Mag. J. Sci. 2 (11) (1901) 559–572, http://dx.doi.org/10.1080/14786440109462720.
[41] B. Schölkopf, A. Smola, K.-R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Comput. 10 (5) (1998) 1299–1319, http://dx.doi.org/10.1162/089976698300017467.
[42] A. Majumdar, Kernelized linear autoencoder, Neural Process. Lett. 53 (2) (2021) 1597–1614, http://dx.doi.org/10.1007/s11063-021-10467-0.
[43] G. Ramachandra, Least square variational Bayesian autoencoder with regularization, 2017, http://dx.doi.org/10.48550/arXiv.1707.03134.
[44] J. Xie, Y. Fang, F. Zhu, E. Wong, Deepshape: Deep learned shape descriptor for 3d shape matching and retrieval, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1275–1283, http://dx.doi.org/10.1109/CVPR.2015.7298732.
[45] W. Liu, T. Ma, Q. Xie, D. Tao, J. Cheng, LMAE: a large margin auto-encoders for classification, Signal Process. 141 (2017) 137–143, http://dx.doi.org/10.1016/j.sigpro.2017.05.030.
[46] W. Luo, J. Yang, W. Xu, T. Fu, Locality-constrained sparse auto-encoder for image classification, IEEE Signal Process. Lett. 22 (8) (2015) 1070–1073, http://dx.doi.org/10.1109/LSP.2014.2384196.
[47] F. Wang, X. Liu, B. Dun, G. Den, Q. Han, H. Li, Application of kernel auto-encoder based on firefly optimization in intershaft bearing fault diagnosis, J. Mech. Eng. 55 (7) (2019) 58–64, http://dx.doi.org/10.3901/JME.2019.07.058.
[48] E. Pintelas, P. Pintelas, A 3D-CAE-CNN model for deep representation learning of 3D images, Eng. Appl. Artif. Intell. 113 (2022) 104978, http://dx.doi.org/10.1016/j.engappai.2022.104978.
[49] M. Abdolahnejad, P.X. Liu, A deep autoencoder with novel adaptive resolution reconstruction loss for disentanglement of concepts in face images, IEEE Trans. Instrum. Meas. 71 (2022) 1–13, http://dx.doi.org/10.1109/TIM.2022.3165261.

[50] N. Sobahi, B. Ari, H. Cakar, O.F. Alcin, A. Sengur, A new signal to image mapping procedure and convolutional neural networks for efficient schizophrenia detection in EEG recordings, IEEE Sens. J. 22 (8) (2022) 7913–7919, http://dx.doi.org/10.1109/JSEN.2022.3151465.
[51] J. Zhao, L. Hu, Y. Dong, L. Huang, S. Weng, D. Zhang, A combination method of stacked autoencoder and 3D deep residual network for hyperspectral image classification, Int. J. Appl. Earth Obs. Geoinf. 102 (2021) 102459, http://dx.doi.org/10.1016/j.jag.2021.102459.
[52] F. Kong, S. Zhao, Y. Li, D. Li, Y. Zhou, A residual network framework based on weighted feature channels for multispectral image compression, Ad Hoc Netw. 107 (2020) 102272, http://dx.doi.org/10.1016/j.adhoc.2020.102272.
[53] J. Guo, Y. Li, S. Dong, W. Zhang, Innovative method of crop classification for hyperspectral images combining stacked autoencoder and CNN, Nongye Jixie Xuebao/Trans. Chin. Soc. Agric. Mach. 52 (12) (2021) 225–232, http://dx.doi.org/10.6041/j.issn.1000-1298.2021.12.024.
[54] F. Khozeimeh, D. Sharifrazi, N.H. Izadi, J.H. Joloudari, A. Shoeibi, R. Alizadehsani, J.M. Gorriz, S. Hussain, Z.A. Sani, H. Moosaei, et al., Combining a convolutional neural network with autoencoders to predict the survival chance of COVID-19 patients, Sci. Rep. 11 (1) (2021) 1–18, http://dx.doi.org/10.1038/s41598-021-93543-8.
[55] R. Vankayalapati, A.L. Muddana, Denoising of images using deep convolutional autoencoders for brain tumor classification, Revue Intell. Artif. 35 (6) (2021) 489–496, http://dx.doi.org/10.18280/ria.350607.
[56] J. Geng, H. Wang, J. Fan, X. Ma, Deep supervised and contractive neural network for SAR image classification, IEEE Trans. Geosci. Remote Sens. 55 (4) (2017) 2442–2459, http://dx.doi.org/10.1109/TGRS.2016.2645226.
[57] G. Cheng, P. Zhou, J. Han, Duplex metric learning for image set classification, IEEE Trans. Image Process. 27 (1) (2018) 281–292, http://dx.doi.org/10.1109/TIP.2017.2760512.
[58] Y. Wang, H. Yao, S. Zhao, Auto-encoder based dimensionality reduction, Neurocomputing 184 (2016) 232–242, http://dx.doi.org/10.1016/j.neucom.2015.08.104.
[59] L. Deng, The MNIST database of handwritten digit images for machine learning research [best of the web], IEEE Signal Process. Mag. 29 (6) (2012) 141–142, http://dx.doi.org/10.1109/MSP.2012.2211477.
[60] J. Neves, R. Tolosana, R. Vera-Rodriguez, V. Lopes, H. Proença, J. Fierrez, GANprintR: Improved fakes and evaluation of the state of the art in face manipulation detection, IEEE J. Sel. Top. Sign. Proces. 14 (5) (2020) http://dx.doi.org/10.1109/JSTSP.2020.3007250.
[61] Y. Wei, X. Hu, Pedestrian anomaly detection method using autoencoder, in: 2021 2nd International Conference on Intelligent Computing and Human-Computer Interaction, ICHCI, IEEE, 2021, pp. 126–129, http://dx.doi.org/10.1109/ICHCI54629.2021.00033.
[62] Y. Feng, D.J.X. Ng, A. Easwaran, Improving variational autoencoder based out-of-distribution detection for embedded real-time applications, ACM Trans. Embed. Comput. Syst. 20 (5s) (2021) 1–26, http://dx.doi.org/10.1145/3477026.
[63] H. Hanafi, A. Pranolo, Y. Mao, CAE-COVIDX: automatic covid-19 disease detection based on x-ray images using enhanced deep convolutional and autoencoder, Int. J. Adv. Intell. Inform. 7 (1) (2021) 49–62, http://dx.doi.org/10.26555/ijain.v7i1.577.
[64] D. Cotroneo, L. De Simone, P. Liguori, R. Natella, Enhancing the analysis of software failures in cloud computing systems with deep learning, J. Syst. Softw. 181 (2021) 111043, http://dx.doi.org/10.1016/j.jss.2021.111043.
[65] C. Xu, Q. Liu, M. Ye, Age invariant face recognition and retrieval by coupled auto-encoder networks, Neurocomputing 222 (2017) 62–71, http://dx.doi.org/10.1016/j.neucom.2016.10.010.
[66] J. Guo, L. Wang, D. Zhu, C. Hu, Compact convolutional autoencoder for SAR target recognition, IET Radar Sonar Navig. 14 (7) (2020) 967–972, http://dx.doi.org/10.1049/iet-rsn.2019.0447.
[67] L. Wen, L. Gao, X. Li, A new deep transfer learning based on sparse auto-encoder for fault diagnosis, IEEE Trans. Syst. Man Cybern. 49 (1) (2017) 136–144, http://dx.doi.org/10.1109/TSMC.2017.2754287.
[68] Y. Chen, H. Zhang, Y. Wang, Y. Yang, X. Zhou, Q.J. Wu, MAMA net: multi-scale attention memory autoencoder network for anomaly detection, IEEE Trans. Med. Imaging 40 (3) (2021) 1032–1041, http://dx.doi.org/10.1109/TMI.2020.3045295.
[69] Z. Yang, D. Gjorgjevikj, J. Long, Y. Zi, S. Zhang, C. Li, Sparse autoencoder-based multi-head deep neural networks for machinery fault diagnostics with detection of novelties, Chin. J. Mech. Eng. 34 (1) (2021) 1–12, http://dx.doi.org/10.1186/s10033-021-00569-0.
[70] W. Gao, S.-P. Qiao, R.-J. Wai, M.-F. Guo, A newly designed diagnostic method for mechanical faults of high-voltage circuit breakers via SSAE and IELM, IEEE Trans. Instrum. Meas. 70 (2021) 1–13, http://dx.doi.org/10.1109/TIM.2020.3011734.
[71] X. Cai, Q. Jing, B. Peng, Y. Zhang, Y. Wang, J. Tang, Automatic traffic state recognition based on video features extracted by an autoencoder, Math. Probl. Eng. 2022 (2022) 2850111, http://dx.doi.org/10.1155/2022/2850111.
[72] Y. Li, X. Zhang, H. Li, Q. Zhou, X. Cao, Z. Xiao, Object detection and tracking under complex environment using deep learning-based LPM, IET Comput. Vis. 13 (2) (2019) 157–164, http://dx.doi.org/10.1049/iet-cvi.2018.5129.
[73] S. Yang, K. Yu, F. Cao, L. Liu, H. Wang, J. Li, Learning causal representations for robust domain adaptation, IEEE Trans. Knowl. Data Eng. 35 (3) (2021) 2750–2764, http://dx.doi.org/10.1109/TKDE.2021.3119185.
[74] Y. Zheng, G. Chen, M. Huang, Out-of-domain detection for natural language understanding in dialog systems, IEEE/ACM Trans. Audio Speech Lang. Process. 28 (2020) 1198–1209, http://dx.doi.org/10.1109/TASLP.2020.2983593.
[75] Y. Zhang, F. Ye, D. Xiong, X. Gao, LDNFSGB: prediction of long non-coding rna and disease association using network feature similarity and gradient boosting, BMC Bioinformatics 21 (1) (2020) 377, http://dx.doi.org/10.1186/s12859-020-03721-0.
[76] Y.K. Lee, J.G. Park, Multimodal unsupervised speech translation for recognizing and evaluating second language speech, Appl. Sci. 11 (6) (2021) 2642, http://dx.doi.org/10.3390/app11062642.
[77] Z. Mai, G. Wu, K. Luo, S. Sanner, Attentive autoencoders for multifaceted preference learning in one-class collaborative filtering, in: 2020 International Conference on Data Mining Workshops, ICDMW, IEEE, 2020, pp. 165–172, http://dx.doi.org/10.1109/ICDMW51313.2020.00032.
[78] E. Dervishaj, P. Cremonesi, GAN-based matrix factorization for recommender systems, in: Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing, 2022, pp. 1373–1381, http://dx.doi.org/10.1145/3477314.3507099.
[79] B. Weggenmann, V. Rublack, M. Andrejczuk, J. Mattern, F. Kerschbaum, DP-VAE: Human-readable text anonymization for online reviews with differentially private variational autoencoders, in: Proceedings of the ACM Web Conference 2022, 2022, pp. 721–731, http://dx.doi.org/10.1145/3485447.3512232.
[80] W. Xu, Y. Tan, Semisupervised text classification by variational autoencoder, IEEE Trans. Neural Netw. Learn. Syst. 31 (1) (2019) 295–308, http://dx.doi.org/10.1109/TNNLS.2019.2900734.
[81] A. Chandru, K. Seetharam, Processing of clinical notes for efficient diagnosis with dual LSTM, Int. J. Adv. Comput. Sci. Appl. 13 (2) (2022) 400–407, http://dx.doi.org/10.14569/IJACSA.2022.0130247.
[82] R. Bhaskaran, S. Saravanan, M. Kavitha, C. Jeyalakshmi, S. Kadry, H.T. Rauf, R. Alkhammash, Intelligent machine learning with metaheuristics based sentiment analysis and classification, Comput. Syst. Sci. Eng. 44 (1) (2022) 235–247, http://dx.doi.org/10.32604/csse.2023.024399.
[83] S. Dahmani, V. Colotte, V. Girard, S. Ouni, Learning emotions latent representation with CVAE for text-driven expressive audiovisual speech synthesis, Neural Netw. 141 (2021) 315–329, http://dx.doi.org/10.1016/j.neunet.2021.04.021.
[84] S. Park, H. Kim, FaceVAE: Generation of a 3D geometric object using variational autoencoders, Electronics 10 (22) (2021) 2792, http://dx.doi.org/10.3390/electronics10222792.
[85] S. Wang, J. Lv, Z. He, D. Liang, Y. Chen, M. Zhang, Q. Liu, Denoising auto-encoding priors in undecimated wavelet domain for MR image reconstruction, Neurocomputing 437 (2021) 325–338, http://dx.doi.org/10.1016/j.neucom.2020.09.086.
[86] D.-H. Zhang, Intelligent transport surveillance memory enhanced method for detection of abnormal behavior in video, J. Adv. Transp. 2022 (2022) 5631281, http://dx.doi.org/10.1155/2022/5631281.
[87] A. Chartsias, G. Papanastasiou, C. Wang, S. Semple, D.E. Newby, R. Dharmakumar, S.A. Tsaftaris, Disentangle, align and fuse for multimodal and semi-supervised image segmentation, IEEE Trans. Med. Imaging 40 (3) (2021) 781–792, http://dx.doi.org/10.1109/tmi.2020.3036584.
[88] S. Dong, Y. Xia, T. Peng, Network abnormal traffic detection model based on semi-supervised deep reinforcement learning, IEEE Trans. Netw. Serv. Manag. 18 (4) (2021) 4197–4212, http://dx.doi.org/10.1109/TNSM.2021.3120804.
[89] P.-F. Gimenez, J. Roux, E. Alata, G. Auriol, M. Kaâniche, V. Nicomette, RIDS: Radio intrusion detection and diagnosis system for wireless communications in smart environment, ACM Trans. Cyber-Phys. Syst. 5 (3) (2021) 1, http://dx.doi.org/10.1145/3441458.
[90] S. Yoo, S. Jeon, S. Jeong, H. Lee, H. Ryou, T. Park, Y. Choi, K. Oh, Prediction of the change points in stock markets using DAE-LSTM, Sustainability 13 (21) (2021) 11822, http://dx.doi.org/10.3390/su132111822.
[91] K.-C. Liu, K.-H. Hung, C.-Y. Hsieh, H.-Y. Huang, C.-T. Chan, Y. Tsao, Deep-learning-based signal enhancement of low-resolution accelerometer for fall detection systems, IEEE Trans. Cogn. Dev. Syst. 14 (3) (2022) 1270–1281, http://dx.doi.org/10.1109/TCDS.2021.3116228.
[92] S. Huang, M. Zhang, Y. Gao, Z. Feng, MIMO radar aided mmwave time-varying channel estimation in MU-MIMO V2X communications, IEEE Trans. Wireless Commun. 20 (11) (2021) 7581–7594, http://dx.doi.org/10.1109/TWC.2021.3085823.

[93] F.-N. Yuan, L. Zhang, J.-T. Shi, X. Xia, G. Li, Theories and applications of auto-encoder neural networks: A literature survey, Jisuanji Xuebao/Chin. J. Comput. 42 (1) (2019) 203–230, http://dx.doi.org/10.11897/SP.J.1016.2019.00203.
[94] L. Dinh, D. Krueger, Y. Bengio, NICE: Non-linear independent components estimation, in: 3rd International Conference on Learning Representations, ICLR 2015 - Workshop Track Proceedings, 2015, http://dx.doi.org/10.48550/arXiv.1410.8516.
[95] P. Ghosh, M.S.M. Sajjadi, A. Vergari, M. Black, B. Scholkopf, From variational to deterministic autoencoders, in: International Conference on Learning Representations, 2020, http://dx.doi.org/10.48550/arXiv.1903.12436.
[96] J. Chen, C. Lu, B. Chenli, J. Zhu, T. Tian, Vflow: More expressive generative flows with variational data augmentation, in: 37th International Conference on Machine Learning, Vol. PartF168147-3, ICML 2020, PMLR, 2020, pp. 1638–1647, http://dx.doi.org/10.48550/arXiv.2002.09741.
[97] J. Ho, X. Chen, A. Srinivas, Y. Duan, P. Abbeel, Flow++: Improving flow-based generative models with variational dequantization and architecture design, in: Proceedings of the 36th International Conference on Machine Learning, Vol. 97, PMLR, 2019, pp. 2722–2730, http://dx.doi.org/10.48550/arXiv.1902.00275.
[98] A. Vahdat, W. Macready, Z. Bian, A. Khoshaman, E. Andriyash, DVAE++: Discrete variational autoencoders with overlapping transformations, in: Proceedings of the 35th International Conference on Machine Learning, Vol. 80, PMLR, 2018, pp. 5035–5044, http://dx.doi.org/10.48550/arXiv.1802.04920.
[99] D. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, M. Welling, Improved variational inference with inverse autoregressive flow, Adv. Neural Inf. Process. Syst. 29 (2016) 4743–4751, URL https://dl.acm.org/doi/10.5555/3157382.3157627.
[100] T. Salimans, A. Karpathy, X. Chen, D. Kingma, PixelCNN++: Improving the PixelCnn with discretized logistic mixture likelihood and other modifications, in: 5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings, 2017, http://dx.doi.org/10.48550/arXiv.1701.05517.
[101] The CIFAR-10 dataset, Can. Inst. Adv. Res. URL https://www.cs.toronto.edu/~kriz/cifar.html.

Pengzhi Li Ph.D. student, Graduate School of Computer Science and Engineering, University of Aizu, Japan. Major research topics include machine learning, deep learning, image recognition, and the kernel method. He is a member of IEEE.

Yan Pei Associate Professor and Doctoral Supervisor, Computer Science Division, University of Aizu, Japan. He received a doctorate in engineering from Kyushu University, Japan. His main research topics include machine learning, computational intelligence, kernel method, and evolutionary computation. He is a senior member of IEEE, IEEE CIS, IEEE SMC, and ACM, and a member of the Japanese society for evolutionary computation. He is an executive board member of the international association of swarm and evolutionary intelligence and technical committee chair of soft computing in the IEEE SMC society.

Jianqiang Li Professor and Doctoral Supervisor, School of Software Engineering, Beijing University of Technology, China. He received a doctorate from Tsinghua University, China. His main research topics include machine learning, deep learning, natural language processing, and image recognition. He is a senior member of IEEE.