Artificial Intelligence Review (2024) 57:28
https://doi.org/10.1007/s10462-023-10662-6

Autoencoders and their applications in machine learning: a survey

Kamal Berahmand[1] · Fatemeh Daneshfar[2] · Elaheh Sadat Salehi[3] · Yuefeng Li[1] · Yue Xu[1]

Published online: 3 February 2024

Abstract

Autoencoders have become a hot researched topic in unsupervised learning due to their ability to learn data features and act as a dimensionality reduction method. With rapid evolution of autoencoder methods, there has yet to be a complete study that provides a full autoencoders roadmap for both stimulating technical improvements and orienting research newbies to autoencoders. In this paper, we present a comprehensive survey of autoencoders, starting with an explanation of the principle of conventional autoencoder and their primary development process. We then provide a taxonomy of autoencoders based on their structures and principles and thoroughly analyze and discuss the related models. Furthermore, we review the applications of autoencoders in various fields, including machine vision, natural language processing, complex network, recommender system, speech process, anomaly detection, and others. Lastly, we summarize the limitations of current autoencoder algorithms and discuss the future directions of the field.

Keywords Deep learning · Dimensionality reduction · Feature extraction · Unsupervised learning · Autoencoder · Bottleneck layer · Reconstruction loss · Autoencoder application

✉ Fatemeh Daneshfar
f.daneshfar@uok.ac.ir

Kamal Berahmand
kamal.berahmand@hdr.qut.edu.au

Elaheh Sadat Salehi
e.salehi@cse.shirazu.ac.ir

Yuefeng Li
y2.li@qut.edu.au

Yue Xu
yue.xu@qut.edu.au

[1] School of Computer Science, Faculty of Science, Queensland University of Technology (QUT), Brisbane, Australia

[2] Department of Computer Engineering, University of Kurdistan, Sanandaj, Iran

[3] Department of Electrical and Computer Engineering, University of Shiraz, Shiraz, Iran

List of symbols

X	The input
X'	The reconstructed output
$\hat{X}'$	The noisy input
Z	The hidden representation of the input data
L	The graph Laplacian matrix
W	Non-negative matrices (basis vectors)
H	Non-negative matrices (coefficients or activations)
W_e	The encoder weight matrix
W_d	The decoder weight matrix
D	The distances matrix between neighbors
N	The number of data points
E	The expectation operator
λ	The regularization parameter
$KL(.\Vert.)$	The Kullback–Leibler divergence
$p(.)$	The probability distribution
$q(.)$	The approximate probability distribution of p(.)
$f(.)$	The encoder function
$g(.)$	The decoder function
$tr(.)$	The trace of the matrix
$D(.)$	The discriminator's output for a real data point
$G(.)$	The generator's output for the latent variable
$\Vert.\Vert$	The 2-norm of a vector
$\Vert.\Vert_F$	The Frobenius norm
$\Vert X - X'\Vert_F^2$	The reconstruction loss

Abbreviations

AA	Adversarial Autoencoder
AAE	Adversarial Autoencoder
AE	Autoencoder
AGAE	Adversarial Graph Autoencoder
BAE	Bayesian Autoencoder
BCE	Binary Cross-Entropy
BiRNNAE	Bidirectional Autoencoder
CAE	Convolutional Autoencoder
CAE	Convolutional Autoencoder
CNN	Convolutional Neural Network
CVAE	Convolutional Variational Autoencoder
CSAE	Convolutional Sparse Autoencoder
DAE	Denoising Autoencoder
DVAE	Disentangled Variational Autoencoder
GAE	Graph Autoencoder
GAAE	Graph Attentional Autoencoder
GCN	Graph Convolution Network
GMAE	Graph Masked Autoencoder
GPU	Graphics Processing Unit
GRUAE	GRU Autoencoder
ISOMAP	Isometric Feature Mapping

LAE	Laplacian Autoencoder
L2,1-RAE	L2,1 Robust Autoencoder
LDA	Linear Discriminant Analysis
LSTM	Long Short-Term Memory
LSTMAE	LSTM Autoencoder
LSRAE	Label and Sparse Regularized Autoencoder
MAE	Masked Autoencoder
MLP	Multi-Layer Perceptron
M-DAE	Marginalized Denoising Autoencoder
NLP	Natural Language Processing
NMF	Non-Negative Matrix Factorization
NN	Neural Network
OAE	Orthogonal Autoencoder
PCA	Principal Component Analysis
RAE	Robust Autoencoder
ReLU	Rectified Linear Unit
SAE	Stacked Autoencoder
SDMAE	Self-Distilled Masked AutoEncoder
SGD	Stochastic Gradient Descent
SSVAE	Semi-supervised Variational Autoencoder
SSAE	Semi-supervised Autoencoders
t-SNE	T-Distributed Stochastic Neighbor Embedding
VAE	Variational Autoencoder
VGAE	Variational Graph Autoencoder

1 Introduction

Dimension reduction is crucial in machine learning for simplifying complex data sets (Van Der Maaten et al. 2009), reducing computational complexity (Ray et al. 2021), and mitigating the curse of dimensionality (Talpur et al. 2023), ultimately improving model performance and interpretability. Dimension reduction encompasses two primary approaches: feature selection (Solorio-Fernández et al. 2022), which involves choosing a subset of the most informative features from the original data-set to reduce dimensionality while maintaining interpretability; and feature extraction (Li et al. 2022), a method where new, lower-dimensional features are derived from the original data to capture essential patterns and relationships.

Feature extraction comprises both linear and nonlinear techniques that transform the original data into a lower-dimensional representation. Linear feature extraction such as Factor Analysis (FA) (Garson 2022), Linear Discriminant Analysis (LDA) (Balakrishnama and Ganapathiraju 1998), Principal Component Analysis (PCA) (Abdi and Williams 2010) and Non-negative Matrix Factorization (NMF) (Lee and Seung 2000) involves transforming the input data into a new set of features using linear combinations of the original input features (Wang et al. 2023).

Linear methods are relatively straightforward and computationally efficient. They often provide interpretable results, making it easier to understand the importance of each feature, and are effective when the underlying relationships in the data are approximately linear. However, they capture global correlations, and result in information loss, particularly when the data contains non-linear relationships or interactions between features (Wang et al. 2023). They are also sensitive to outliers, can be computationally expensive, particularly when dealing with high-dimensional data. Their linear projections can be difficult to interpret, and they can be prone to overfitting when the number of input features is significantly greater than the number of observations available (Jha et al. 2023).

In contrast, nonlinear feature extraction utilizes nonlinear transformations of the input features to generate a new feature set that can more effectively capture the underlying patterns present in the data (Wang et al. 2022). By mapping the data into a higher-dimensional feature space, nonlinear methods can find patterns that are not apparent in the original feature space, even when the number of features significantly exceeds the number of samples.

Nonlinear methods also can capture complex relationships between the input features and output variables without the need for domain knowledge or prior assumptions about the data and often leads to better predictive performance (Wang et al. 2022). Manifold-based feature extraction is a nonlinear technique, that relies on the assumption that high-dimensional data can be embedded in a low-dimensional space without losing important information. This is achieved by finding a non-linear mapping that preserves the structure of the data (Li et al. 2022). Some common manifold-based techniques include, ISOMAP (Ding et al. 2022), Locally Linear Embedding (LLE) (Miao et al. 2022) and t-SNE (t-distributed Stochastic Neighbor Embedding) (Meyer et al. 2022). These techniques may not always capture the global structure of the data and its performance is highly dependent on hyperparameter settings.

Another effective method to extract complex, hierarchical, and high-level features from nonlinear data is deep learning. Deep learning models can automatically learn abstract and high-level features, enabling better data representation from raw data and reducing the need for handcrafted feature engineering. They can be used for end-to-end feature extraction and task-specific modelling including image classification, object detection, Natural language Processing (NLP), and speech recognition. In this context, there are several deep learning-based nonlinear feature extraction techniques, some of which are: Convolutional Neural Networks (CNNs) (Molaei et al. 2022), Recurrent Neural Networks (RNNs) (Shi et al. 2022), and Autoencoders (AEs) (Bank et al. 2023). Deep learning models like CNNs and RNNs often require large amounts of labelled data for training and its training can

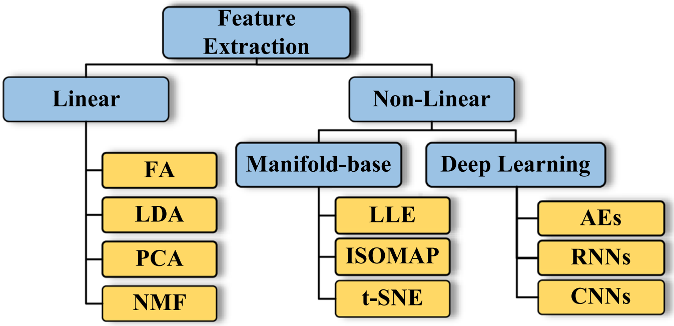


Fig. 1 Categorization of feature extraction methods into linear and non-linear approaches

Table 1 Methods for dimensionality reduction

Method Type	Method	Loss function L	Description
Linear	FA (Shrestha 2021)	$\min(-0.5\log\lvert\Psi\rvert$ $+\operatorname{tr}(S\Psi^{-1})$ $+0.5dp\log(2\pi))$	Explains patterns of correlations among observed variables by uncovering underlying latent factors
	PCA (Hasan and Abdulazeez 2021)	$\max_{W^TW=I}\operatorname{trace}(W^TAW)$	Optimizes the projection of data onto its principal components by maximizing the variance along those components
	LDA (Li et al. 2020)	$\min\left(\frac{w^TS_bw}{w^TS_ww}\right)$	Maximizes the separation between classes while minimizing the variance within each class
	NMF (Wang et al. 2023)	$\min_{W,H\geq 0}\lVert X-WH\rVert_F^2$	Decomposes a non-negative matrix into two lower-dimensional non-negative matrices
Nonlinear	LLE (Miao et al. 2022)	$\min\left(\sum_i \lvert\lvert x_i-\sum_j w_{ij}x_j\rvert\rvert^2\right)$	Seeks to preserve the local linear relationships between data points in a lower-dimensional space
	ISOMAP (Ding et al. 2022)	$\min\left(\lVert D-\hat{D}\rVert^2\right)$	Constructs a low-dimensional representation of data while preserving the geodesic distances between data points on a manifold-like structure
	t-SNE (Meyer et al. 2022)	$\min\left(\sum\sum p_{ij}\log\left(\frac{p_{ij}}{q_{ij}}\right)\right)$	Preserves the pairwise similarity relationships between data points in a lower-dimensional space
	AE (Bank et al. 2023)	$\min\left(\lVert X-X'\rVert_F^2\right)$	Aims to encode and subsequently decode data, facilitating dimensionality reduction and feature extraction
	RNN (Shi et al. 2022)	–	Captures temporal dependencies from sequential data passed recursively through hidden layers
	CNN (Molaei et al. 2022)	–	Processes structured grid data, by applying convolutional layers to automatically extracted features

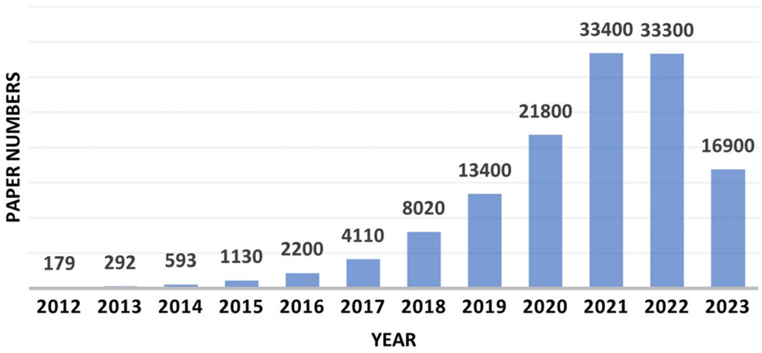


Fig. 2 All published papers in gScholar, Web of Science and arxiv since 2012 with keywords "Autoencoders" and "Machine Learning"

be computationally expensive, requiring powerful hardware. Figure 1 and Table 1 shows various feature extraction methods and their loss functions.

AEs are neural networks that use back propagation algorithm for feature learning. They are primarily used for unsupervised learning tasks, which means they do not require labelled data during training. In contrast, CNNs and RNNs are often used for supervised or semi-supervised tasks, which rely on labelled data. This makes AEs suitable for situations where labelled data is scarce or expensive to obtain (Bank et al. 2020). Furthermore, AEs automatically learn relevant features from the data without the need for manual feature engineering which can save significant time and effort in pre-processing. This encourages the AEs to capture the crucial characteristics of the input data in its encoding, thereby learning a meaningful representation of the data in the latent code (Liu et al. 2023).

AEs also provide a multitude of benefits additionally to dimensionality reduction across various machine learning and data analysis applications mainly used in complex high-dimensional data. They are equally valuable in the context of data compression, where they can efficiently encode information for storage or transmission, making them particularly beneficial for resource-constrained applications. Furthermore, they excel in anomaly detection by quantifying the reconstruction error; instances with elevated reconstruction errors are flagged as anomalies, aiding in the identification of outliers or irregularities within the data (Bank et al. 2020). Data denoising is another strength of AEs. AEs can be trained to eliminate noise or irrelevant information from input data, enhancing data quality. Beyond these applications feature learning, AEs foster a deeper understanding of data through the creation of meaningful representations. They also find practical utility in semantic embedding for NLP and information retrieval tasks and effectively reducing file sizes without compromising quality in image and signal compression (Liu et al. 2023). Furthermore, AEs contribute to privacy preservation techniques, such as differential privacy, by protecting sensitive data while enabling analysis and insights. In addition to these applications, AEs are instrumental in reducing data storage requirements, enhancing interpretability by revealing essential data features, and demonstrating robustness by generalizing well to new data and effectively handling noisy or incomplete data-sets (Liu et al. 2023). Overall, AEs stand as versatile and indispensable tools, offering an extensive array of applications across diverse domains and problem types in machine learning and data analysis.

However, AEs offer a powerful set of capabilities but also come with certain drawbacks that should be considered. One of the main drawbacks of using AEs is that they are sensitive to the choice of hyperparameters, such as the number and size of layers, the learning rate, the loss function, and the regularization. These hyperparameters can affect the performance and the quality of the autoencoder, and may require trial and error or grid search to find the optimal values (Bank et al. 2020). Another common concern with AEs is their lack of robustness. They can be sensitive to noisy data, outliers, and variations in input, which can lead to suboptimal representations and reconstructions (Singh and Ogunfunmi 2022). AEs can be prone to overfitting, especially when trained on limited data. Additionally, they may not inherently preserve the spatial or temporal locality of data during training. This can be problematic for tasks where preserving the local structure is essential, such as image segmentation or sequence modeling (Liu et al. 2023). Furthermore, AEs tend to capture lower-order features and may struggle to represent complex, higher-order relationships in the data. This limitation can impact their performance on tasks that require understanding intricate dependencies (Miuccio et al. 2022).

In recent years, substantial research efforts have been dedicated to addressing these drawbacks through advancements in deep learning and AE techniques. Some of the presented architectures in this area include regularization AEs, robust AE, generative AE, convolutional AE, recurrent AE, semi-supervised AE, graph AE and masked AE. These improvements, as demonstrated in Fig. 2, have caused that the use of autoencoder algorithms in machine learning has gained increasing interest over the years. The graph shows the trend of papers published in the field of "autoencoder" and “machine learning” since 2012, revealing that over 90% of all indexed papers were published between 2018 and 2023.

Despite being an important area of research, there is currently a lack of comprehensive studies exploring the applications of AE algorithms in machine learning on a wide scale. While existing review papers have examined specific themes, there has been no comprehensive review conducted. In Table 2, we compare our contribution in this paper to the descriptions of existing review papers in the field.

To this knowledge gap, our review will focus on addressing three key research questions:

- What are the different types of AE algorithms that have been developed and utilized in machine learning applications?
- What are the main methodological frameworks and the latest achievements in the application of AE algorithms?
- What are the gaps and future directions in this field, and how can they be addressed to enhance the effectiveness of AE algorithms in machine learning applications?

This review paper represents a significant endeavor to systematically categorize the diverse array of applications of AEs within the domain of machine learning. Furthermore, it embarks on the crucial endeavor of not only elucidating the advantages and challenges associated with these applications but also unraveling the existing frameworks that underpin this evolving field. In this pioneering exploration, we offer the following noteworthy contributions:

- New taxonomy. In this paper, we propose a comprehensive new taxonomy that categorizes major and modern AE methods within the realm of machine learning into distinct categories in recent years.

Table 2 Comparison of our article with the previous review or survey articles

Paper	Year	Brief description	Aspects not considered
Sagha et al. (2017)	2017	The article provides a comprehensive review of existing literature and studies that have utilized stacked denoising autoencoders for sentiment analysis, highlighting their strengths and limitations	Categorization of autoencoder taxonomies and applications. Comprehensiveness
Charte et al. (2018)	2018	It explains how autoencoders can be used for nonlinear feature fusion and provides guidelines for designing an autoencoder-based feature fusion system. The paper also provides a taxonomy of autoencoders and discusses various software tools that can be used for implementing them	Comprehensive analysis of autoencoder applications in ML
Zhang et al. (2020)	2020	The paper provides a comprehensive overview of the use of autoencoders in building recommender systems. The authors discuss the motivation for using autoencoders, the architecture and training methods used, and the evaluation metrics used to measure their effectiveness. It also presents a detailed analysis of various applications of autoencoder-based recommender systems in different domains	Autoencoders taxonomy and techniques in ML. Comprehensiveness
Girin et al. (2020)	2020	This study provides a detailed overview and analysis of the recent development and application of dynamical variational autoencoders (DVAEs) in the field of machine learning. The paper also explores the different types of DVAEs and their applications in various fields	Categorization of autoencoder taxonomies and applications. Comprehensiveness
Pereira et al. (2020)	2020	The paper provides a comprehensive review of various techniques and applications of autoencoders for missing data imputation. It highlights the advantages and limitations of autoencoder-based approaches for each application and provides insights into future research directions	Publicly available framework discussion. Comprehensiveness
Bank et al. (2020)	2020	This paper introduces and explains the concept of autoencoders in machine learning. It describes the architecture of autoencoders and various types of autoencoders. Finally, the paper provides an overview of the advantages and limitations of autoencoders and identifies future research directions	Publicly available framework discussion. Comprehensive

Table 2 (continued)

Paper	Year	Brief description	Aspects not considered
Pratella et al. (2021)	2021	The review discusses several types of autoencoders, the advantages, and disadvantages of each algorithm, and provides examples of how they can be applied to rare disease diagnosis	Autoencoder applications in ML
Song et al. (2021)	2021	It proposes the use of autoencoders as a technique for network intrusion detection. The authors conduct experiments on a dataset of network traffic, comparing the performance of autoencoders to traditional anomaly detection techniques	Applications of autoencoder techniques in ML
Qian et al. (2022)	2022	The article provides an overview of fault detection and diagnosis, and then discusses the use of autoencoders for feature extraction in industrial processes. It covers different types of autoencoders, and how they can be used for fault detection and diagnosis	Comprehensiveness. Autoencoder techniques in ML
Shankar and Parsana (2022)	2022	The paper provides an overview and empirical comparison of different NLP models and introduces and empirically applies autoencoder models in the marketing domain	Categorization of autoencoder taxonomies and applications. Comprehensiveness
Singh and Ogunfunmi (2022)	2022	The paper provides an overview of VAEs and their applications in source separation, finance, and bio-signal processing. It provides examples of each application and discusses the advantages and limitations of using VAEs in these contexts	Categorization of autoencoder taxonomies. Autoencoder applications in ML
Zhang et al. (2022)	2022	The survey likely provides an overview of the development, applications, and advancements of masked autoencoders in self-supervised learning, shedding light on their effectiveness, challenges, and future directions	Publicly available framework discussion. Masked autoencoder applications in ML.
Bank et al. (2023)	2023	The paper likely discusses the theory and practical applications of autoencoders in more detail, providing insights into how they can be effectively used in data science tasks	Publicly available framework discussion. Comprehensiveness. Applications of autoencoder techniques in ML in detail

Table 2 (continued)

Paper	Year	Brief description	Aspects not considered
Our paper	2023	It provided a comprehensive analysis of autoencoders, organizing them into distinct categories based on their architecture. In addition, the various taxonomies of autoencoders were explored in depth. The study reviewed prior applications of autoencoders in the machine learning domain, categorizing them according to the tasks they were utilized for. Furthermore, publicly accessible software and platforms for constructing and developing autoencoders were assessed. Finally, the open issues and challenges of autoencoders in machine learning were thoroughly investigated	

- Comprehensive overview. We not only provide an exhaustive review of the variations within each AE category but also offer detailed descriptions and unified schematic representations. Our in-depth exploration of each approach includes elucidating key equations and presenting pertinent performance comparisons.
- Abundant resources. We curate and present a valuable collection of AE resources, encompassing open-source code repositories for select reviewed methods, widely recognized benchmark datasets, and performance assessments across datasets with varying label rates.
- Future trends. we pinpoint unresolved challenges and explore potential directions for future research, drawing insights from recent seminal studies in this field.

This paper is organized as follows. Section 2 provides a concise overview of the structure and hyperparameter in AEs. Section 3 discusses various taxonomies of AEs that have been proposed in the literature. In Sect. 4, we review previous applications of AEs in the machine learning domain, categorizing them according to the task they were used for. In Sect. 5, we review explore publicly available software and platforms that can be used to construct and develop AEs the performance of various autoencoders. Section 6 is dedicated to discussing future directions in the field. Finally, in Sect. 7, we present our conclusions based on the insights gathered from our analysis.

2 Background of autoencoder

AE is a fundamental building block that can be used hierarchically to create deep models. They organize, compress, and extract high-level features, allowing unsupervised learning and the extraction of non-linear features (Chen and Guo 2023). Autoencoders have advantages over Restricted Boltzmann Machines (RBMs) as they can learn more complex data representations. RBMs are widely used for generating various data types, including images (Hinton et al. 2006). RBMs are a type of Boltzmann Machine (BM) that learns a probability distribution from inputs (Chen and Guo 2023). The main difference between Autoencoders, RBMs, and BMs lies in their architectures. AEs have an encoder and a decoder, while RBMs consist of visible and hidden layers. Boltzmann Machines (BMs) are more general and fully connected, making them less tractable compared to RBMs. AEs are feed-forward neural networks, allowing information to flow in one direction. In contrast, RBMs and BMs are generative models capable of generating new samples from the learned distribution.

2.1 Vanilla autoencoder

The concept of AE was initially introduced in a research paper by Rumelhart (1985). AEs are a type of neural network designed for learning and reconstructing input data. In unsupervised learning, the primary goal is to obtain an "informative" data representation. AEs encode input data into a compressed and semantically meaningful form and then decode it to faithfully reconstruct the original input data (Bank et al. 2023). The term "vanilla" is used to describe the simplest form of autoencoder, which has no additional complexities or architectural variations. A vanilla autoencoder typically consists of an input layer, one or more hidden layers, and an output layer (Zhang et al. 2016). You can visualize the structure of a vanilla autoencoder in Fig. 3.

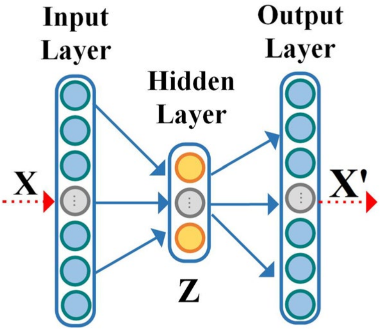


Fig. 3 illustrates the structure of an autoencoder, where X represents the input data of the input layer, Z represents the data in the hidden layer, and X' represents the reconstructed output data in the output layer

During the encoding step, an AE maps an input vector X to a code vector Z using an encoding function f_θ. In the decoding step, it maps the code vector Z back to the output vector X', aiming to reconstruct the input data using a decoding function g_θ. AEs adjust the network's weights (W) through fine-tuning, achieved by minimizing the reconstruction error L between X and the reconstructed data X'. This reconstruction error acts as a loss function used to optimize the network's parameters (Chai et al. 2019). The objective function of an AE can be written as:

$$\min_\theta J_{AE}(\theta) = \min_\theta \sum_{i=1}^{n} l(x_i, x_i') = \min_\theta \sum_{i=1}^{n} l(x_i, g_\theta(f_\theta(x_i))) \tag{1}$$

where x_i represents the ith dimension of the training sample, x_i' represents the ith dimension of the output data, and n is the total amount of training data. The term "l" refers to the reconstruction error between the input and output, defined as:

$$L(X, X') = \sum_{i=1}^{n} \|X_i - X_i'\|^2 \tag{2}$$

The encoder and decoder mapping functions are $Z = f_\theta(X) = s(WX + b)$ and $X' = g_\theta(Z) = s(W'Z + b')$, where "s" is a non-linear activation function like sigmoid or ReLU. W and W' are weight matrices, and b and b' are bias vectors. During training, the weights and biases of the autoencoder are adjusted to minimize the reconstruction error using an optimization algorithm like stochastic gradient descent. Once trained, the encoding function can create low-dimensional representations of new input data (Z), while the decoding function can reconstruct the original data from the low-dimensional representation (X').

2.2 Stack autoencoder

Traditional AE typically employs a single-layer encoder, making it challenging to extract deep features. To enhance feature extraction, one effective strategy is to deepen the neural network structure. By employing a layer-wise learning approach, multiple basic autoencoders can be stacked together to form a Stacked Autoencoder (SAE), allowing for the extraction of complex data features. The training process of each individual autoencoder involves learning a condensed data representation, with the final output obtained by combining the outputs of these individual autoencoders. Typically, training a

Stacked Autoencoder follows a layer-wise approach (Hoang and Kang 2019; Hinton et al. 2006). After training layer 1, it serves as the input for training layer 2. When evaluating the reconstruction loss, it is assessed relative to layer 1 rather than the input layer. The encoding process can be mathematically represented as follows:

$$a^k = f(W_e^k a^{k-1} + b_e^k), \quad k = 1 : n \tag{3}$$

in which k represents the k-th autoencoder, a_k represents the encoding outcome of the k-th autoencoder, and when $k = 1$, $a_0 = x$ denotes the input data. The decoding process can be mathematically represented as follows:

$$c^k = f(W^{n-(k-1)} c^{k-1} + b^{n-(k-1)}), \quad k = 1 : n \tag{9}$$

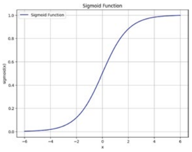
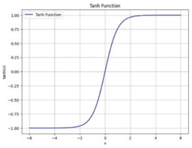
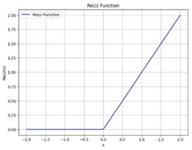
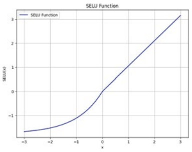
when $k = 1$, $c_0 = a_n$, and when $k = n$, $c_n = \hat{x}$ represents the reconstructed data of the input variable x (Hoang and Kang 2019).

2.3 Hyperparameters in autoencoder

Autoencoders come with various hyperparameters that must be defined prior to training, and their values can significantly influence the model's performance. It's crucial to understand that certain hyperparameters are usually set before training and remain constant, while others can be dynamically tuned during training to optimize the model's performance. Selecting and adjusting hyperparameters often involves experimentation and validation to achieve the best results for a particular task. The following outlines the most common hyperparameters in autoencoders:

- **Number of Hidden Layers:** The quantity of hidden layers within the autoencoder defines its network depth and its capacity to capture intricate data patterns. This parameter is configured before training. While adding more hidden layers can enhance the model's representational power, it may also introduce optimization challenges and elevate the risk of overfitting.
- **Number of Neurons in Each Layer:** The number of neurons in each layer governs the network's data representation capacity and is typically set before training. A higher count of neurons can amplify the network's capacity but might also elevate the risk of overfitting and complicate the optimization process.
- **Size of Latent Space:** Adjusting the size of the bottleneck layer permits fine-tuning the balance between model complexity and performance. This parameter is set prior to training.
- **Activation Function:** The activation function utilized in the bottleneck layer plays a pivotal role in the autoencoder's performance. To optimize the autoencoder's performance, the bottleneck layer activation function should be tailored before training. These functions determine the network's nonlinearity and its ability to learn intricate data patterns. Common activation functions employed in bottleneck layers encompass sigmoid, tanh, ReLU, and SELU. Further details, including their equations, outputs, and output curves, are outlined in Table 3.
- **Objective Function:** The objective function, also known as the loss function, is a critical element of an autoencoder, serving to train the network by minimizing the distinction between input and output data. It gauges the dissimilarity between the

Table 3 Activation functions

Activation Function	Equation	Output	Output Curve
Sigmoid	$f(x) = \frac{1}{1+e^{-x}}$	[0,1]	
Tanh	$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	[−1,1]	
ReLU	$f(x) = \max(0, x)$	$[0, \infty]$	
SELU	$f(x) = \lambda \begin{cases} x & \text{if } x > 0 \\ \alpha e^x - \alpha & \text{if } x \leq 0 \end{cases}$	$[-2, \infty]$	

input and output data, and the autoencoder is trained to diminish this dissimilarity. The selection of the objective function hinges on the data type and the specific application and is generally determined before training. Common objective functions used in autoencoders include:

– **Mean Squared Error (MSE):** This is the predominant objective function in autoencoders, measuring the average squared difference between input and output data. MSE is defined by formulas (1):

$$L_{\mathrm{AE}}(X, X') = \min\left(\|X - X'\|_F^2\right) \tag{4}$$

– **Binary Cross-Entropy(BCE):** BCE employed when the input data is binary (0 or 1), this function measures the difference between predicted and actual output in terms of binary cross-entropy loss. Cross-entropy is defined in formulas (2):

$$L_{\mathrm{AE}}(X, X') = -\sum_{i=1}^{n}\left(x_i \log(x_i') + (1 - x_i)\log(1 - x_i')\right) \tag{5}$$

When choosing an autoencoder loss function, consider the problem's unique needs. MSE suits regression tasks, offering robustness against outliers but sensitivity to data scaling. BCE is for binary classification but can be numerically unstable near 0 or 1 probabilities. The choice depends on the problem and task requirements. MSE is the most prevalent autoencoder loss function, quantifying input–output discrepancies in the latent space.

- **Optimization Algorithm:** Autoencoders utilize optimization algorithms to minimize the objective function during training. These algorithms adjust network weights and biases to train the autoencoder effectively. The choice of the optimization algorithm is made prior to training but may involve hyperparameter tuning during training. Several optimization techniques can be employed to train autoencoders, with the most notable ones being Stochastic Gradient Descent (SGD), Adam, and Adagrad. Further elaboration on each of these methods is provided below.

  - **Stochastic Gradient Descent (SGD):** A widely used algorithm that updates network parameters after processing small batches of data. While computationally efficient, it may converge slowly for complex models and datasets. Careful tuning of initial learning rates is often needed.
  - **Adam:** Combines features of SGD with adaptive learning rates and momentum to accelerate convergence and reduce the risk of getting stuck in local minima. Requires tuning of hyperparameters like beta1 and beta2 and is suitable for non-stationary and noisy objectives.
  - **Adagrad:** An adaptive algorithm adjusting learning rates based on parameter update frequency. Effective for sparse data, it can lead to quick convergence but may also converge prematurely and face challenges with non-convex optimization.

  The choice of optimization algorithm depends on dataset size, model complexity, loss function type, and computational resources. Each has its advantages and disadvantages, so selecting the right one is crucial for optimal performance.

- **Learning Rate:** The learning rate, a hyperparameter, dictates the step size during optimization. It influences weight and bias updates and the convergence speed of the objective function. High values can cause overshooting, while low ones may lead to local minima trapping. The learning rate is preset but may be adjusted with schedules during training for better convergence.
- **Number of Epochs:** Epochs are training iterations, representing full dataset passes. More epochs can enhance model accuracy but risk overfitting. The ideal count depends on dataset size and problem complexity. Learning rate initially set may require modification if convergence isn't reached or early stopping is used to curb overfitting.
- **Batch Size:** Batch size, in each optimization iteration, affects gradient noise and optimization efficiency. Smaller sizes yield noisier gradients but faster, memory-efficient optimization. Larger sizes offer stable gradients but slower, memory-intensive optimization. Batch size is determined beforehand but can be adjusted during training for optimization and memory use.

These interconnected hyperparameters necessitate careful selection for optimal performance, often requiring experimentation despite the time investment, crucial for building an effective autoencoder model.

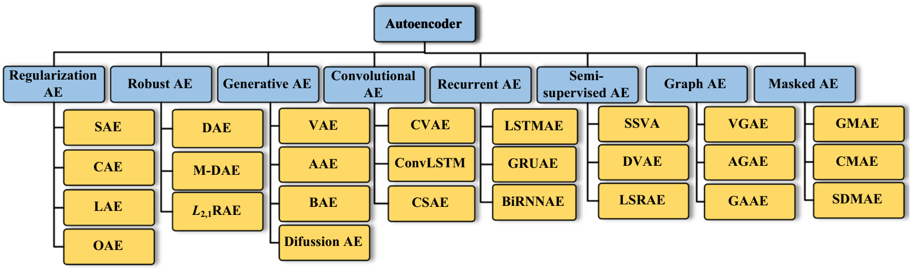


Fig. 4 Taxonomy of Autoencoder architectures categorized by network structure

3 Autoencoder taxonomy

Autoencoders, frequently employed in unsupervised learning, excel in dimensionality reduction tasks. They adeptly capture intricate, non-linear data relationships, enabling a hierarchical transformation of high-dimensional input into a lower-dimensional latent space. Autoencoders exhibit remarkable flexibility, allowing for customization across diverse data types and tasks by adjusting their architecture or objective functions. Over the past decade, a myriad of autoencoder variants has emerged, as illustrated in Fig. 4.

Autoencoder enhances feature discrimination through the incorporation of regularization techniques. Robust autoencoder aims to fortify the encoded data against noise or outliers, enhancing its ability to handle noisy or corrupted input data. The generative autoencoder specializes in learning a generative model using the extracted encoded representations, enabling the generation of new data samples closely resembling the distribution of the training data. Convolutional autoencoders replace fully connected layers with convolutional layers in both the encoder and decoder, making them particularly well-suited for image data by excelling at capturing spatial relationships within the data. Recurrent autoencoders leverage recurrent layers, such as LSTM or GRU, in both the encoder and decoder, proving invaluable for sequence data by capturing temporal dependencies within the information. Semi-supervised autoencoders harness the power of both labeled and unlabeled data to enhance model performance and generalization, demonstrating their value in scenarios with limited labeled data or resource constraints. Graph autoencoder leverages graph structures to learn data representations by processing graph-structured inputs and utilizing graph convolutional layers, allowing for the effective modeling of complex data dependencies. Masked autoencoders represent a straightforward autoencoding technique designed to reconstruct the original signal from its partially observed form.

The breadth of autoencoder models and their specialization options empowers fine-tuning for various applications. The adaptability of the autoencoder architecture and objective functions underscores their ability to be tailored to specific use cases, establishing them as indispensable tools for machine learning researchers and developers. In the following sections, we provide detailed explanations for each category.

3.1 Regularized autoencoder

Regularized Autoencoder (RAE) is a neural network architecture that extracts a compressed representation of input data while enforcing regularization constraints. These constraints encourage the formation of a discriminative, low-dimensional feature space. By incorporating different regularization techniques into the autoencoder, it becomes possible to create specialized models with desired properties, such as sparsity, manifold structure, or orthogonality.

3.1.1 Sparse autoencoder

Sparse Autoencoder (SAE) (Ng 2011) is characterized by having a limited number of simultaneously active neural nodes, as it aims to learn a sparse representation of input data by incorporating a sparsity constraint into the loss function. Its objective is to minimize the disparity between input data and reconstructed data while adhering to constraints on the sparsity of the latent representation. The loss function in a Sparse Autoencoder (SAE) comprises two components: the reconstruction loss and the sparsity loss, represented as follows:

$$L_{\text{SAE}}(X, X') = \min\left(\|X - X'\|_F^2 + \lambda \text{KL}(p \parallel q)\right) \tag{6}$$

where $\text{KL}(p \parallel q)$ calculates the Kullback–Leibler divergence between a target sparsity parameter (p) and the estimated average activation of each neuron (q) during training, defined as

$$\sum p \log\left(\frac{p}{q}\right) + (1-p)\log\left(\frac{1-p}{1-q}\right) \tag{7}$$

This combined penalty term encourages the model to acquire a sparse representation, wherein only a limited number of neurons are active for each input.

3.1.2 Contractive autoencoder

Contractive Autoencoder (CAE) (Rifai et al. 2011) is an autoencoder that aims to produce similar representations for similar input data by adding a penalty term to the loss function. This penalty term, based on the Frobenius norm of the Jacobian matrix of the encoder concerning the input data, encourages local stability in the learned representation. The primary objective of the CAE is to minimize the difference between the input data and the reconstructed data while taking the penalty term into account, promoting similarity in representations for similar input data. The overall loss function of CAE includes the reconstruction loss and a penalty term as follows:

$$L_{\text{CAE}}(X, X') = \min\left(\|X - X'\|_F^2 + \lambda \|J_F(X)\|_F^2\right) \tag{8}$$

where $\|J_F(X)\|_F^2$ represents the squared Frobenius norm of the Jacobian matrix of the encoded representation concerning the input data. This norm measures the sensitivity of the encoded representation to small variations, calculated as:

$$\|J_F(X)\|_F^2 = \sum_{i,j}\left(\frac{\partial h_j(X)}{\partial X_i}\right)^2 \tag{9}$$

3.1.3 Laplacian autoencoder

The standard Autoencoder may not emphasize the relationships between nearby data points during its learning process, which can lead to extracted features lacking crucial information about the data's internal structure. In contrast, the Laplacian Autoencoder prioritizes preserving the distances between neighboring data points, effectively capturing the significant internal structure within the data. Inspired by this concept, the Laplacian Autoencoder (LAE) (Jia et al. 2015) was introduced to facilitate the generation of lower-dimensional representations for Autoencoders. This approach ensures that the learned representations incorporate essential local structural information, enhancing their suitability for specific data analysis tasks. The loss function for the Laplacian Autoencoder is defined as follows:

$$L_{LAE}(X, X') = \min \left(\|X - X'\|_F^2 + \lambda \mathrm{tr}(Z' L Z) \right) \tag{10}$$

where matrix L, known as the graph Laplacian, is calculated based on how similar pairwise are in the latent space. This calculation typically involves techniques like using k-nearest neighbor graphs or Gaussian kernels.

3.1.4 Orthogonal autoencoder

Orthogonal Autoencoder (OAE) (Wang et al. 2019) is designed to enhance the orthogonality of learned embeddings, leading to more discriminative and diverse feature representations. Unlike the standard Autoencoder, OAE introduces a regularization term known as the orthogonal reconstruction error into the reconstruction loss function. This term promotes orthogonality among latent features, thereby improving class discriminability. The OAE loss function can be expressed as follows:

$$L_{OAE}(X, X') = \min \left(\|X - X'\|_F^2 + \lambda \|Z^T Z - I\|_F^2 \right) \tag{11}$$

where I is the identity matrix, Z^T represents the transpose of the compressed representation Z, and λ is a penalization parameter. Notably, setting λ to zero yields a conventional autoencoder.

3.2 Robust autoencoder

Robust Autoencoder (RAE) is utilized to enhance the robustness of autoencoders when dealing with noisy or corrupted input data. They prove especially valuable in situations where the input data exhibits noise, outliers, or imperfections. These issues are commonplace in real-world datasets, including those found in healthcare, finance, and sensor networks, where RAEs can effectively handle the data's imperfections while retaining its valuable information. Three primary variants of robust autoencoders include Denoising Autoencoder, Marginalized Denoising Autoencoder, and $L_{2,1}$ Autoencoder.

3.2.1 Denoising autoencoder

Denoising Autoencoder (DAE) (Vincent et al. 2010) is designed to reconstruct clean data from noisy input by introducing noise during training. The primary objective is to minimize the dissimilarity between the clean data and the reconstructed output. DAE training involves intentionally corrupting input data with various forms of noise and then minimizing the difference between the original clean input data and the reconstructed clean data. This process allows the DAE to discern valuable features within the input data while disregarding noise and irrelevant aspects. The DAE loss function is expressed as follows:

$$L_{DAE}(X, X') = \min \left(\|X - \hat{X}'\|_F^2 \right) \tag{12}$$

where X represents the clean input data, and $\hat{X}'$ denotes the noisy input data.

3.2.2 Marginalized denoising autoencoder

Marginalized Denoising Autoencoder (M-DAE) (Chen et al. 2012) is a specialized version of the Denoising Autoencoder (DAE) designed to handle datasets with missing or incomplete features. Like the standard DAE, the M-DAE is a neural network crafted to reconstruct clean input data from noisy versions. It achieves this by restoring clean data from corrupted counterparts, where input data X is intentionally subjected to random corruption. Each feature has a probability p of being set to 0, creating these corrupted versions referred to as $\hat{X}_i$. The primary goal of the M-DAE is to minimize a specific loss function represented as:

$$L_{\text{M-DAE}}(X, X') = \min \left(\frac{1}{m} \sum_{i=1}^{m} \|X - \hat{X}' W\|_F^2 \right) \tag{13}$$

where W signifies the learned transformation matrix, and m represents the total number of input examples.

The M-DAE seeks the best solution for W, which can be expressed mathematically as:

$$W = E[Q^{-1}]E[P] \tag{14}$$

this equation calculates $E[Q^{-1}]$ based on the inverse of the expected Q and $E[P]$ using the expected P. These expectations are calculated using specific formulas involving the covariance matrix of the uncorrupted data X.

3.2.3 $L_{2,1}$ robust autoencoder

$L_{2,1}$ Robust Autoencoder ($L_{2,1}$-RAE) (Li et al. 2018) is a modified version of the Robust Autoencoder (RAE) designed to enhance the autoencoder's resilience when dealing with noisy or corrupted input data. This enhancement is achieved through the use of a specific type of regularization known as $\mathrm{L}_{2,1}$ regularization. $\mathrm{L}_{2,1}$ regularization encourages the learned features to possess specific properties. Notably, it promotes feature sparsity, meaning that most features consist of zeros, and robustness, enabling them to handle scenarios with data outliers or noise. The mathematical expression of the $L_{2,1}$-RAE loss function is given as follows:

$$L_{2,1RAE}(X, X') = \min\left(\|X - X'\|_F^2 + \lambda \cdot \|Z\|^{2,1}\right) \tag{15}$$

where $\|Z\|_{2,1}$ represents the L2,1-norm of the latent representations, which emphasizes both sparsity and robustness in these learned features.

3.3 Generative autoencoder

Generative Aautoencoder (GAE) differs from traditional autoencoders by focusing on learning the underlying probability distribution of data rather than just dimensionality reduction. This enables GAE to generate new data samples that resemble the training data, making them valuable for tasks like image or text generation. Examples of GAE include Variational Autoencoders, Adversarial Autoencoders, Bayesian Autoencoder and Diffusion Autoencoder.

3.3.1 Variational autoencoder

Variational Autoencoder (VAE) (An and Cho 2015) is a type of autoencoder that learns to represent data in a lower-dimensional latent space and generate new data samples that resemble the input. Unlike traditional autoencoders, VAEs are generative models that can capture the underlying distribution of input data. In a VAE, the encoder maps input data to a posterior distribution $q(Z|X)$ instead of a fixed latent representation Z. During reconstruction, Z is sampled from this distribution and passed through a decoder. The regularization loss in VAE encourages $q(Z|X)$ to match a specific distribution, often a standard Gaussian. The VAE loss function is defined as:

$$L_{\mathrm{VAE}} = -E(q(Z|X))\left[\log[p(X|Z)]\right] + \mathrm{KL}(q(Z|X)||p(Z)) \tag{16}$$

the first term measures the difference between the original input data ($p(X|Z)$) and the data reconstructed by the decoder. The second term, a regularization component, quantifies the KL divergence between $q(Z|X)$ and $p(Z)$, typically a standard Gaussian distribution. This loss function guides VAE training to balance accurate data reconstruction with a structured latent space for generative purposes.

3.3.2 Adversarial autoencoder

Adversarial Autoencoder (AAE) (Makhzani et al. 2015) is a specialized type of autoencoder designed to align its learned latent representations with a desired prior distribution. It consists of three main parts: an encoder, a decoder, and a discriminator. The encoder and decoder work together to create data that can deceive the discriminator, which is trained to distinguish between real input data and fake data produced by the decoder. The adversarial loss in AAE assesses its ability to generate data that resembles the original input data distribution. The discriminator aims to maximize its accuracy in telling real and generated data apart, while the decoder aims to minimize the discriminator's accuracy. The overall loss function for AAE is expressed as:

$$L_{AAE}(X,X') = \min\left(\|X - X'\|_F^2 + \log(D(X)) + \log(1 - D(G(Z)))\right) \tag{17}$$

where $G(z)$ is the decoder function that converts the latent representation back to the original input data, and $D(X)$ represents the discriminator's output for the original input data. The term $\log(1 - D(G(Z)))$ reflects the discriminator's output for data generated by the decoder.

3.3.3 Bayesian autoencoder

Bayesian Autoencoder (BAE) (Yong and Brintrup 2022) is a probabilistic AE that models all parameters, in contrast to the Variational Autoencoder (VAE) that mainly models the latent layer. BAE combines a Gaussian likelihood for data reconstruction with an isotropic Gaussian prior for parameter uncertainty. The loss function maximizes data likelihood and minimizes model complexity. The BAE loss function is defined as:

$$\log p(x|\theta) = -\left(\frac{1}{D}\sum_{i=1}^{D}\frac{1}{2\sigma_i^2}(x_i - x_i')^2 + \frac{1}{2}\log\sigma_i^2\right) \tag{18}$$

where σ_i^2 is the variance of the Gaussian distribution, and $\log p(x|\theta)$ represents the log-likelihood of observing the original data x given the model parameters θ. It quantifies data reconstruction through squared errors and variances while promoting model simplicity. The training objective is to maximize this log-likelihood while minimizing regularization to find optimal parameters θ for effective data pattern and uncertainty capture.

3.3.4 Diffusion autoencoder

Diffusion Autoencoder (DiffusionAE) (Preechakul et al. 2022) is a specialized type of autoencoder designed for generative modeling tasks. It draws inspiration from diffusion models and is engineered to capture intricate data distributions. In this framework, data is subjected to a progressive denoising process, allowing the model to grasp complex data patterns effectively. A fundamental element of DiffusionAE is its employment of a unique loss function known as the Diffusion Probabilistic Loss. This loss function guides the training by modeling how data evolves over time. Mathematically, the loss function is represented as:

$$L(X,X') = -\log P(X|X') \tag{19}$$

in which $P(X|X')$ signifies the conditional probability of observing the original data X when given the reconstructed data X'. During training, the primary objective is to minimize this loss, driving the Diffusion Autoencoder to generate X' that closely resembles the original data X.

3.4 Convolutional autoencoder

Convolutional Autoencoder (CAE) (Seyfioğlu et al. 2018) employs convolutional layers instead of fully connected layers in both the encoder and decoder. The encoder uses these layers to create a compact representation from input images, while the decoder employs deconvolution layers for image reconstruction. CAEs are particularly effective for image data, as they excel at capturing spatial dependencies, which refer to the patterns and relationships among pixels or locations within individual images or data frames. They find wide-ranging applications in tasks such as image denoising, inpainting, segmentation, and super-resolution.

3.4.1 Convolutional variational autoencoder

convolutional variational autoencoder (CVAE) (Semeniuta et al. 2017) is a significant variant of Convolutional Autoencoders (CAEs) that incorporates probabilistic modeling, allowing for the generation of new data samples. In a CVAE, input images undergo a reconstruction process where a latent variable, denoted as Z, is sampled from a Gaussian distribution, and subsequently passed through a decoder. This decoder employs convolutional and upsampling layers to reconstruct the original image. The loss function in CVAE, similar to Variational Autoencoder (VAE), is defined as:

$$L_{\mathrm{CVAE}} = -E(q(Z|X))\left[\log[p(X|Z)]\right] + \mathrm{KL}(q(Z|X)||p(Z)) \tag{20}$$

the first term measures the difference between the original image and its reconstruction by the decoder, while the second term encourages the latent representation $q(Z|X)$ to follow a standard Gaussian distribution through KL divergence regularization, ensuring a structured latent space for effective generative capabilities.

3.4.2 Convolutional LSTM autoencoder

Convolutional LSTM (ConvLSTM) (Luo et al. 2017) is an advanced neural network architecture specialized in spatiotemporal data analysis. It combines convolutional structures with recurrent operations, allowing it to capture both spatial dependencies and temporal relationships in data. This design employs 3D tensors, with the last two dimensions representing spatial dimensions (e.g., rows and columns). ConvLSTM excels in tasks involving both spatial and temporal patterns, such as precipitation nowcasting and video analysis. It utilizes a unique loss function to make predictions based on neighboring cells, consistently outperforming traditional RNNs and contemporary algorithms in various spatiotemporal forecasting applications. The overall loss function for a ConvLSTM can be defined as follows:

$$L_{\mathrm{ConvLSTM}}(X, X') = \min \sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{t=1}^{T} \left(||X_{ijt} - X'_{ijt}||_F^2 \right) \tag{21}$$

where N is the number of spatial rows in the data, M is the number of spatial columns in the data, T is the number of time steps in the sequence, X_{tij} represents the ground truth value at spatial location (i, j) at time step t, and X'_{tij} represents the predicted value at spatial location (i, j) at time step t.

3.4.3 Convolutional sparse autoencoder

Convolutional Sparse Autoencoder (CSAE) (Luo et al. 2017) is a neural network architecture that combines convolutional autoencoder principles with techniques to induce sparsity, such as max-pooling and feature channel competition. This integration simplifies the training process by eliminating the need for complex optimization procedures. CSAE includes a sparsifying module designed to create sparse feature maps. This module retains the highest value and its corresponding position within each local subregion before performing unpooling, primarily through max pooling. The loss function used in CSAE, which quantifies the disparities between the original input and the reconstructed output, relies on the Frobenius norm and is defined as follows:

$$L_{\mathrm{CSAE}}(X, X') = \min \sum_{l=1}^{L} \left(\|X^{(l)} - X'^{(l)}\|_F^2 \right) \tag{22}$$

$$X'^{(l)} = \sum_{i=1}^{d} \left(\mathrm{rot}(W_i, 180) * Z_i^l \right) + c_i \tag{23}$$

$$Z^l = G_{p,s}\left(Z_i^{(l)}\right) = G_{p,s}\big(f(W_i \cdot X^{(l)} + b_i)\big) \tag{24}$$

where l is the number of layers, $X^{(l)}$ represents the original input at layer l, $X'^{(l)}$ represents the reconstructed output at layer l, d is the number of feature channels, Z_i^l is the ith sparsified feature map, and $G_{p,s}(X)$ represents the sparsifying operator, involving max-pooling and unpooling operations to create sparse feature maps.

3.5 Recurrent autoencoder

RNNs (Medsker and Jain 2001) are designed for processing sequential data, like time series where the current state (h^t) relies on the previous state (h^{t-1}). Vanilla RNNs have a limitation of short-term memory, leading to gradient problems in long sequences. To address this, LSTM equipped with three gates (forget gate, input gate, and output gate), and GRU networks consist of two gates (update gate and reset gate) were introduced. These architectures incorporate self-loops to effectively manage gradients over extended sequences, addressing the vanishing or exploding gradient issue. Recurrent Autoencoder is an autoencoder that incorporates recurrent layers, such as LSTM or GRU, within both the encoder and decoder components.

3.5.1 Long short term memory autoencoder

LSTM Autoencoder (LSTMAE) (Nguyen et al. 2021) is an advanced variation of the recurrent autoencoder, specifically designed to capture representations from sequential data. In this architecture, both the encoder and decoder components are built using LSTM units, a type of recurrent layer. The encoder LSTM takes in a sequence of vectors, which can represent images or features. In contrast, the decoder LSTM reconstructs the original input sequence, often in reverse order. The MSE loss function computes the average squared differences between the input and the reconstructed output at each time step. The formula for MSE loss is as follows:

$$L_{\mathrm{LSTMAE}}(X, X') = \min \left(\|X - X'\|_F^2 \right) \tag{25}$$

where X represents the clean input sequence and X' represents the reconstructed output sequence.

3.5.2 Gated recurrent unit autoencoder

GRU Autoencoder (GRUAE) (Dehghan et al. 2014) employs GRU units in both the encoder and decoder parts. Unlike LSTM, GRU has a simpler architecture with only two gates: the update and reset gates. This architectural simplicity can lead to easier training and faster processing while still capturing long-term dependencies in input sequences. The formulation of a GRU Autoencoder is similar to that of an LSTM Autoencoder, making it flexible and effective for modeling sequential data,

$$L_{\mathrm{GRUAE}}(X, X') = \min\left(\|X - X'\|_F^2\right) \tag{26}$$

where X represents the clean input sequence and X' represents the reconstructed output sequence.

3.5.3 Bidirectional autoencoder

Bidirectional Autoencoder (BiRNNAE) (Marchi et al. 2015) is a neural network designed for unsupervised learning from sequential data. It utilizes bidirectional RNNs like LSTM or GRU in both the encoder and decoder parts. While traditional RNNs only consider information in one direction, bidirectional RNNs incorporate knowledge from both forward and backward directions, improving their grasp of temporal relationships. The BiRNN-AE aims to minimize the squared reconstruction error between the original input sequence and the generated sequence during training. To represent the input data efficiently, it combines the final hidden states from all encoder layers. This compact representation can be valuable for various downstream tasks involving sequential data. The loss function for BiRNN-AE is the Mean Squared Error (MSE) loss, which can be mathematically expressed as:

$$L_{\mathrm{BiRNNAE}}(X, X') = \min \frac{1}{T} \sum_{t=1}^{T} \left(\|X_t - X'_t\|_F^2\right) \tag{27}$$

where T is the sequence length, X_t represents the input at time step t, and X'_t represents the reconstructed output at time step t.

3.6 Semi-supervised autoencoder

Semi-supervised Autoencoders (SSAE) is autoencoder model that utilize both labeled and unlabeled data to enhance feature learning, especially in scenarios with limited labeled data. The primary objective of SSAE is to leverage the available labeled data to facilitate the extraction of crucial latent features, which can subsequently be applied to tasks such as clustering or classification (Yang et al. 2022). This approach proves highly advantageous when dealing with a scarcity of labeled data, as it enables the exploitation of abundant unlabeled data, a common occurrence in real-world applications. In the following section, we delve into an explanation of the three methods of semi-supervised Autoencoder.

3.6.1 Semi-supervised variational autoencoder

Semi-supervised Variational Autoencoder (SSVAE) (Xu et al. 2017) is a category of generative models employed in semi-supervised learning scenarios. In SSVAE, the encoder responsible for generating the latent variable, denoted as z, is defined as $q_{\phi}(z|x, y)$. This implies that the latent variable z is parameterized by both input data x and label y. The decoder, on the other hand, generates samples from the distribution $p_{\theta}(x|y, z)$. The label predictive distribution $q_{\phi}(y|x)$ is determined by a classification network. Notably, the label y is also considered a latent variable and plays a role in generating a sample x in conjunction with z. The loss function for SSVAE is mathematically expressed as follows:

$$\begin{aligned} L_{\mathrm{SSVAE}} = & -\mathbb{E}_{q_{\phi}(z|x,y)}[\log p_{\theta}(x|y, z)] \\ & - \log p_{\theta}(y) + \mathrm{KL}(q_{\phi}(z|x, y)||p(z)) \end{aligned} \tag{28}$$

in which the first term represents the expectation of the conditional log-likelihood of the latent variable z, the second term denotes the log-likelihood associated with y, and the third term quantifies the Kullback–Leibler divergence between the prior distribution $p(z)$ and the posterior distribution $q_{\phi}(z|x, y)$.

3.6.2 Disentangled variational autoencoder

Disentangled Variational Autoencoder (DVAE) (Higgins et al. 2016) is a sophisticated generative model designed to untangle complex data representations. By incorporating specific graphical model structures and distinct encoding factors, it can effectively separate and capture meaningful information. This model leverages neural networks within a graphical framework to capture relationships among observed and unobserved variables. To optimize its performance, it employs a conditional probability factorization, $q(y, z|x)$, which is different from traditional approaches. This change requires advanced variational inference methods. In essence, Disentangled VAEs are adept at modeling intricate data patterns, making them valuable for various machine learning tasks. Mathematically, they use a loss function expressed as:

$$\mathbb{E}_{q(y,z|x)}(\log p(x|y, z) + \log p(y) + \log p(z) - \log q(y|x, z) - \log q(z|x)) \tag{29}$$

In simpler terms, this loss function guides the model to generate data resembling real-world data while considering the relationships between observed and latent variables.

3.6.3 Label and sparse regularized autoencoder

Label and Sparse Regularized Autoencoder (LSRAE) (Chai et al. 2019) is a novel approach that combines label and sparse regularizations with autoencoders to create a semi-supervised learning method. This method effectively leverages the strengths of both unsupervised and supervised learning processes. On one hand, sparse regularization selectively activates a subset of neurons, enhancing the extraction of localized and informative features. This unsupervised learning process helps uncover underlying data concepts, improving generalization. On the other hand, label regularization enforces the extraction of features aligned with category rules, leading to improved categorization accuracy. The objective function of LSRAE is defined as follows:

$$L_{\mathrm{LSRAE}}(X, X') = \min\left(\|X - X'\|_F^2 + KL(p \parallel q) + \sum_{i=1}^{d}\sum_{j=1}^{l}(W_{ij})^2 + \sum_{i=1}^{n}\|L - T\| \right) \quad (30)$$

where the first term ensures precise data reconstruction, the second term promotes sparsity within the hidden layer, facilitating efficient feature extraction. The third term acts as a safeguard against overfitting by penalizing excessive weights. Lastly, the fourth term enhances classification accuracy by quantifying the label error. Here, L denotes the actual label, and T represents the desired label.

3.7 Graph autoencoder

Graph Autoencoder (GAE) (Pan et al. 2018) is a power method for reducing the dimensionality of graph data, enhancing efficiency in graph analytics. It takes a graph as input and outputs a condensed vector representation that captures its essential feature. Within GAE, the encoder converts the input graph into a lower-dimensional vector, which the decoder uses to recreate the original graph. The model aims to minimize the dissimilarity between input and output graphs while capturing essential graph features. The loss function for GAE is defined as:

$$L_{\mathrm{GAE}}(X, X') = \min\left(\|X - X'\|_F^2\right) \quad (31)$$

where X' is computed from the inner product of the hidden representation Z and its transpose Z^T using the logistic sigmoid function $\sigma(ZZ^T)$. $Z = GCN(F, X)$, obtained through the Graph Convolutional Network (GCN) applied to the node features matrix F, is based on the input data X.

3.7.1 Variational graph autoencoder

Variational Graph Autoencoder (VGAE) (Kipf and Welling 2016) is a framework for learning interpretable latent representations of graph-structured data. It employs a probabilistic approach to encode graph information effectively. VGAE consists of two essential components: an encoder and a decoder. The encoder utilizes a Graph Convolution Network (GCN) to transform graph nodes into a lower-dimensional latent space. It generates latent variables z_i for each node by sampling from Gaussian distributions. These latent variables capture crucial structural information of the graph. The decoder functions as a generative model, aiming to reconstruct the original graph structure using the latent variables z_i. It estimates the likelihood of connections (edges) between nodes based on their corresponding latent vectors.The VGAE loss function combines a reconstruction term and a regularization term to guide the learning process effectively:

$$L_{\mathrm{VGAE}} = -E(q(Z|F, X))[\log[p(X|Z)]] + \mathrm{KL}(q(Z|F, X)||p(Z)) \quad (32)$$

where $q(Z|F, X)$ represents the encoding distribution, $p(X|Z)$ models the likelihood of the adjacency matrix given the latent variables, and $\mathrm{KL}(q(Z|F, X)||p(Z))$ quantifies the divergence between the encoding distribution and the prior distribution governing the latent variables Z.

3.7.2 Adversarial graph autoencoder

Adversarial Graph Autoencoder (AGAE) (Pan et al. 2018) leverages adversarial training to acquire a lower-dimensional representation of the input graph. It employs an encoder to map graph nodes to this lower-dimensional space and a decoder to reconstruct the original graph. AGAE integrates an adversarial component, akin to a discriminator, to ensure the learned embeddings preserve the graph structure. This unsupervised model combines autoencoder-based reconstruction with adversarial training to generate high-quality graph representations. The AGAE loss function is defined as follows:

$$L_{\text{AGAE}} = E_{(H\sim p_z)}[\log D(Z)] + E_X[\log(1 - D(G(F, X)))] \tag{33}$$

where $G(\cdot)$ represents the generator, and $D(\cdot)$ signifies the discriminator. The discriminator's role is to distinguish between the real input graph, p_z, and the reconstructed graph generated by the generator $G(F, X)$.

3.7.3 Graph attention autoencoder

Graph Attentional Autoencoder (GAAE) (Salehi and Davulcu 2019) is a variant of graph autoencoders that combines Graph Attention Network (GAT) with GAE. It employs attention mechanisms to weigh the importance of neighboring nodes and edges during the reconstruction process. In essence, GAAE aims to learn a low-dimensional representation of a graph while preserving its structural information using attention mechanisms. The GAAE loss function is defined as follows:

$$L_{\text{GAAE}} = \min\left(\|X - \text{Sigmoid}(ZZ^T))\|_F^2\right) \tag{34}$$

in which Z represents the hidden layer representation of node v. The calculation of $Z_i^{(l)}$ is based on the formula:

$$Z_i^{(l)} = \sigma\left(\sum_{j\in N_i} a_{ij} W^{(l-1)} Z_j^{(l-1)}\right) \tag{35}$$

where N_i denotes the set of neighbors of node v_i, and $W^{(l-1)}$ represents the learnable parameter matrix. The attention coefficient a_{ij} is computed using the following formula:

$$a_{ij} = \frac{\exp(\delta M_{ij}(\mathbf{a}^T[W\mathbf{x}_i \| W\mathbf{x}_j]))}{\sum_{r\in N_i} \exp(\delta M_{ir}\mathbf{a}^T([W\mathbf{x}_i \| W\mathbf{x}_r]))} \tag{36}$$

where M represents topological weights, and δ is the LeakyReLU activation function.

3.8 Masked autoencoders

Masked AE (MAE) is a variant of autoencoder used for sequence modeling, particularly in vision and NLP. It operates by taking a sequence of data and randomly masking or hiding some of the elements. The model's task is to predict the masked or missing elements based on the context provided by the unmasked portions. This training approach enables MAE to generate coherent and contextually appropriate text or videos, making them valuable for tasks like text completion (Zhang et al. 2022), text generation (Zhang et al. 2023,) language modeling, image captioning (Alzu'bi et al. 2021) and data augmentation (Xu et al. 2022).

3.8.1 Graph masked autoencoder

Graph Masked Autoencoder (GMAE) (Hou et al. 2022) is a simplified and cost-effective approach for self-supervised graph representation learning. Unlike most GAEs that focus on reconstructing graph structures, GMAE's core emphasis is on feature reconstruction through masking. Additionally, GMAE departs from using MSE, opting for the cosine error, which benefits cases where feature magnitudes vary, common in graph node attributes. The primary objective of GMAE is to reconstruct the masked features of nodes, $V' \subset V$, given the partially observed node signals. Formally, for GMAE, the Loss function is as follow, where it is averaged over all masked nodes,

$$L_{\mathrm{GMAE}} = \min \frac{1}{|V'|} \sum_{v_i \in V'} \left(1 - \frac{x_i^T z_i}{\|x_i\| \cdot \|z_i\|} \right)^{\gamma}, \quad \gamma \geq 1 \tag{37}$$

3.8.2 Contrastive masked autoencoder

Contrastive Masked Autoencoders (CMAE) (Huang et al. 2022) is a novel self-supervised pre-training method designed to enhance the learning of comprehensive and versatile vision representations. CMAE comprises two distinct branches: the online branch, characterized by an asymmetric encoder-decoder configuration, and the target branch, featuring a momentum-updated encoder. During the training process, the online encoder is tasked with reconstructing original images from latent representations of masked images with positional embeddings added. The loss uses cosine similarity ρ between (y_s^p and z_t^p) and negative ρ_j^- pairs. The final objective function is as follow,

$$L_{\mathrm{CMAE}} = \min \left(\|Y_m - Y'_m\|_F^2 + \lambda \log \frac{\left(- \exp(\frac{\rho_-^j}{\tau}) \right)}{\exp(\frac{\rho_j^-}{\tau}) + \sum_{j=1}^K \exp(\frac{\rho_j^-}{\tau})} \right) \tag{38}$$

3.8.3 Self-distillated masked autoencoder

Self-Distilled Masked AutoEncoder (SDMAE) (Chen et al. 2022) is composed of two branches: a student branch equipped tasked with reconstructing missing information, and a teacher branch responsible for generating latent representations of masked tokens. In this approach, a student network f_θ trained through gradient descent using $\hat{x}$ as inputs and a teacher network f_ϕ. Based on the MAE method, a value normalization function is proposed for the teacher outputs as $\overline{f_\phi(x_i)}$. This function calculates the mean and standard deviation of feature values within a patch. Subsequently, the optimization objective involves minimizing the normalized teacher features with the output features of the student decoder, utilizing feature cosine similarity as follow,

Table 4 Various autoencoder methods including details on their respective improvements and utilized loss functions

Method	Improvement	Loss function
SAE	Learns a more compact and informative representation of the data	$\min\left(\|X - X'\|_F^2 + \lambda \mathrm{KL}(p \parallel q)\right)$
CAE	Learns a mapping that is robust to small input variations	$\min\left(\|X - X'\|_F^2 + \lambda \|J_F(X)\|_F^2\right)$
LAE	Learns a low-dimensional data representation while preserving the local structure	$\min\left(\|X - X'\|_F^2 + \lambda \mathrm{tr}(Z'LZ)\right)$
OAE	Enforcing orthogonality among latent features, enhancing class discriminability	$\min\left(\|X - X'\|_F^2 + \lambda \|Z^T Z - I\|_F^2\right)$
DAE	Introduces noise to input and reconstructs the output from the original clean input	$\min\left(\|X - \hat{X}'\|_F^2\right)$
M-DAE	Reconstructs clean data from noisy data where some of the features are missing	$\min\left(\frac{1}{m}\sum_{i=1}^{m} \|X - \hat{X}'W\|_F^2\right)$
2,1RAE	Enhances resilience to noisy data using $L_{2,1}$ regularization, encouraging feature sparsity and robustness	$\min\left(\|X - X'\|_F^2 + \lambda \cdot \|Z\|^{2,1}\right)$
VAE	Learns the input data distribution and generates new data points from this distribution	$-E(q(Z\|X))\left[\log[p(X\|Z)]\right] + \mathrm{KL}(q(Z\|X)\|\|p(Z))$
AAE	Learns the input data structure and generates new data points similar to them	$\min\left(\|X - X'\|_F^2 + \log(D(X)) + \log(1 - D(G(Z)))\right)$
BAE	Combining Gaussian likelihood and isotropic Gaussian prior for effective data pattern and uncertainty capture	$-\left(\frac{1}{D}\sum_{i=1}^{D} \frac{1}{2\sigma_i^2}(x_i - x_i')^2 + \frac{1}{2}\log\sigma_i^2\right)$
DiffusionAE	a specialized generative model, employing the Diffusion Probabilistic Loss for training	$-\log P(X\|X')$
CVAE	Integrating convolutional layers and probabilistic modeling, using a Gaussian latent variable and KL divergence regularization	$-E(q(Z\|X))\left[\log[p(X\|Z)]\right] + \mathrm{KL}(q(Z\|X)\|\|p(Z))$
ConvLSTM	combines convolution and recurrent layers for spatiotemporal data	$\min\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{t=1}^{T}\left(\|X_{ijt} - X'_{ijt}\|_F^2\right)$
CSAE	Combines the convolutional layers of a CNN with the sparsity constraint of a SAE	$\min\sum_{l=1}^{L}\left(\|X^{(l)} - X'^{(l)}\|_F^2\right)$
LSTMAE	Uses LSTM units in the encoder and decoder parts of the network	$\min\left(\|X - X'\|_F^2\right)$
GRUAE	Uses GRU units in the encoder and decoder parts of the network	$\min\left(\|X - X'\|_F^2\right)$
BiRNNAE	Using bidirectional RNNs to minimize squared reconstruction error with an MSE loss for sequential data	$\min\frac{1}{T}\sum_{t=1}^{T}\left(\|X_t - X_t'\|_F^2\right)$
SSVAE	Combining log-likelihood terms for latent variables and Kullback–Leibler divergence	$-\mathbb{E}_{q_\phi(z\|x,y)}[\log p_\theta(x\|y,z)] - \log p_\theta(y) + \mathrm{KL}(q_\phi(z\|x,y)\|\|p(z))$
DVAE	A unique loss function for capturing complex data patterns and relationships between observed and latent variables	$\mathbb{E}_{q(y,z\|x)}(\log p(x\|y,z) + \log p(y) + \log p(z) - \log q(y\|x,z) - \log q(z\|x))$

Table 4 (continued)

Method	Improvement	Loss function
LSRAE	Combining sparse and label regularizations with autoencoders to improve feature extraction and categorization accuracy	$\min\left(\lVert X-X'\rVert_F^2+KL(p\parallel q)+\sum_{i=1}^{d}\sum_{j=1}^{l}(W_{ij})^2+\sum_{i=1}^{n}\lVert L-T\rVert\right)$
VGAE	Using a probabilistic approach, combining an Encoder and a Decoder guided by a loss function with a reconstruction term	$-E(q(Z\mid F,X))[\log[p(X\mid Z)]]+\mathrm{KL}(q(Z\mid F,X)\mid\mid p(Z))$
AGAE	Using adversarial training with encoder and decoder components to create compact graph representations	$E_{(H\sim p_z)}[\log D(Z)]+E_X[\log(1-D(G(F,X)))]$
GAAE	Using attention mechanisms to reconstruct graphs effectively. Its loss emphasizes preserving structural information	$\min\left(\lVert X-\mathrm{Sigmoid}(ZZ^T))\rVert_F^2\right)$
GMAE	Prioritizing feature reconstruction through masking and employs cosine error	$\min\frac{1}{\lvert V'\rvert}\sum_{v_i\in V'}\left(1-\frac{x_i^T z_i}{\lVert x_i\rVert\cdot\lVert z_i\rVert}\right)^{\gamma},\quad \gamma\geq 1$
CMAE	Uses Improving vision representations with online and target branches, online encoder reconstructs masked images, using cosine similarity loss	$\min\left(\lVert Y_m-Y'_m\rVert_F^2+\lambda\log\frac{\left(-\exp(\frac{\rho_-^j}{\tau})\right)}{\exp(\frac{\rho_j^-}{\tau})+\sum_{j=1}^{K}\exp(\frac{\rho_j^-}{\tau})}\right)$
SDMAE	Utilizing student and teacher branches to reconstruct missing information	$\min(\log q_{\psi}(\hat{x}\mid\tilde{x})$

$$L_{\mathrm{SDMAE}} = \min(\log q_{\psi}(\hat{x}|\tilde{x})) \approx \min \frac{\sum_{i=1}^{n} m^{i}\overline{f_{\phi}(x_i)}f_{\theta}(\hat{x})}{\sqrt{\sum_{i=1}^{n} m^{i}(\overline{f_{\phi}(x_i)})^{2}}\sqrt{\sum_{i=1}^{n} m^{i}(f_{\theta}(\hat{x}))^{2}}} \tag{39}$$

Table 4 presents a comprehensive summary of different autoencoder methods, offering insights into the specific enhancements each method brings to the table as well as the loss functions they employ for optimization.

4 Application autoencoder

AEs have been widely used in various domains, including computer vision, natural language processing, complex network analysis, recommenders, anomaly detection, speech recognition, and more. Different types of autoencoder architectures have been proposed to address specific challenges and improve performance in these domains. For example, convolutional autoencoders are commonly used in image processing tasks, while recurrent autoencoders are well-suited for sequential data processing. In addition, variational autoencoders have been developed for generating new data samples and improving model generalization. Although each architecture has its own advantages and limitations, it is important to consider the specific requirements of the application domain when selecting an appropriate architecture. Figure 5 provides an overview of the applications of autoencoders in various domains, which can be used as a starting point for selecting an appropriate architecture. However, further research is needed to investigate which architectures are more suitable for which application categories and which architectures are more popular in specific domains.

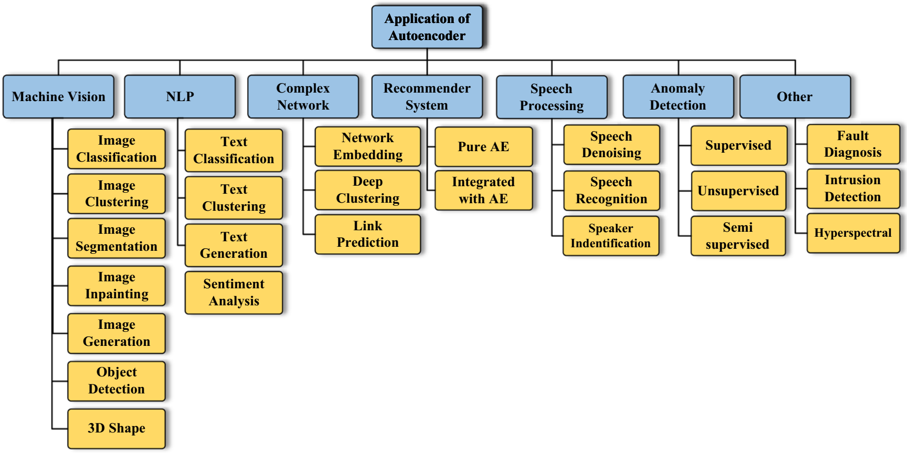


Fig. 5 The process of creating the consensus matrix, including the generation of random walks of different lengths and their combination

4.1 Machine vision

Machine vision utilizes computer algorithms and software to analyze and interpret images or video data, aiming to enable machines to understand and interact with the visual world (Jain et al. 1995). AEs play a vital role in various machine vision applications by learning to extract meaningful image features and reducing data dimensionality. These applications encompass tasks such as image classification (Vincent et al. 2010), image clustering (Guo et al. 2017), image segmentation (Myronenko 2019), image inpainting (Bertalmio et al. 2000), image generation (Vahdat and Kautz 2020), object detection (Liang et al. 2018), and 3D shape analysis (Todd 2004).

AEs are instrumental in image classification. Methods like Semi-supervised stacked distance autoencoder (Hou et al. 2020) enhance feature representation by incorporating semi-supervised learning, utilizing both labeled and unlabeled data to learn inter-data point distances. Deep Convolutional Autoencoders (DCAE) aid in semi-supervised classification, as seen in Geng et al. (2015), where they pre-train on unlabeled Synthetic Aperture Radar (SAR) images and fine-tune using labeled data for high-resolution SAR images classification.

AEs are also valuable in image clustering, where they learn compressed image representations for grouping similar images in the latent space. This technique involves training a clustering algorithm like K-means on the latent space, as described in references Song et al. (2013) and Yang et al. (2017). Additionally, AEs can be used for unsupervised image clustering, making them suitable for scenarios with limited labeled data.

AEs are instrumental in image segmentation, with a wide array of applications that enhance the precision and efficiency of this critical computer vision task. By learning meaningful feature representations from image data, AEs provide a valuable foundation for distinguishing objects and boundaries in images. Their capability for dimensionality reduction streamlines the processing of high-resolution images, making segmentation algorithms computationally more tractable (Zhang et al. 2019). AEs also excel in noise reduction, eliminating unwanted artifacts from images, which is pivotal for accurate segmentation (Tripathi 2021). They are integral in semantic segmentation (Ohgushi et al. 2020), where they classify each pixel in an image, and instance segmentation (Lin et al. 2020), distinguishing individual object instances. Furthermore, AEs contribute to medical image segmentation (Ma et al. 2022), aiding in the precise identification of structures and anomalies in healthcare images. Overall, AEs substantially elevate the accuracy and efficiency of image segmentation tasks, encompassing a range of applications that extend from object recognition to medical diagnosis.

AEs find significant applications in the domain of image inpainting, a process of reconstructing missing or corrupted parts of an image. They excel at capturing complex patterns and textures within images, making them invaluable for this task. AEs, particularly VAEs and GANs, offer high-quality inpainting results by learning to generate realistic and coherent content to fill in the gaps (Tian et al. 2023; Han and Wang 2021). They effectively model the underlying structures and features of images, ensuring that the inpainted regions seamlessly blend with the surrounding content.

AEs find versatile applications in image generation tasks, contributing to the creation of high-quality and diverse visual content. They serve as a foundational component in generative models, VAEs and GANs, enabling the synthesis of realistic and novel images (Huang and Jafari 2023). AEs are essential in encoding and decoding operations, effectively generating images with specific features, styles, and content (Xu et al. 2019). They also play a vital role in style transfer, where they transform images to adopt the artistic characteristics of other images or styles (Kim et al. 2021).

AEs play a role in object detection by extracting valuable features from images or video frames, improving detection accuracy. Convolutional AEs are used to learn compressed image representations that enhance the performance of object detection algorithms, such as Region-based Convolutional Neural Networks (R-CNN) (Ding et al. 2019). VAE further enhanes object detection accuracy, as seen in the integration of VAE with You Only Look Once (YOLO) (Redmon et al. 2016).

In the domain of 3D shape analysis, AEs learn compressed representations for tasks like shape generation, completion, and retrieval. Achieving a disentangled latent representation that separates various factors of variation is a challenge. Recent research introduces methods like Split-AE (Saha et al. 2022) and 3D Shape Variational Autoencoder Latent Disentanglement (Foti et al. 2022), addressing this challenge. Other approaches employ deep learning features for 3D shape retrieval by projecting 3D shapes into 2D space and utilizing AEs for feature learning (Zhu et al. 2016). Additionally, architectures like point-cloud AEs combined with VAEs are explored to partition the latent space and enhance 3D shape analysis (Aumentado-Armstrong et al. 2019).

While AEs offer valuable capabilities in various machine vision applications, their effectiveness often depends on the specific task and dataset characteristics, and they may be complemented by specialized models in certain scenarios.

4.2 NLP

NLP is a field that explores how computers can understand and work with human language in speech or text form to perform useful tasks (Chowdhary and Chowdhary 2020). This area mainly concentrates on methods for handling text data, including tasks like categorizing text (text classification) (Kowsari et al. 2019), grouping similar texts together (text clustering) (Aggarwal and Zhai 2012), generating new text (text generation) (McKeown 1992), and assessing the sentiment expressed in text (sentiment analysis) (Medhat et al. 2014). To tackle the complexities of working with textual data, researchers have developed advanced models, often incorporating AEs. These models have proven effective in addressing the challenges associated with processing text data (Li et al. 2023).

AEs play a versatile role in text classification tasks, offering feature learning to capture crucial patterns in text data (Guo et al. 2023; Ye et al. 2022), dimensionality reduction for efficient processing of high-dimensional text features (Le et al. 2023; Che et al. 2020), noise reduction to clean and enhance noisy text (García-Mendoza et al. 2022; Che et al. 2020), and semi-supervised learning for improved classification using limited labeled data (Wu et al. 2019; Xu et al. 2017). They also excel in topic modeling by uncovering underlying themes within text documents (Paul et al. 2023; Smatana and Butka 2019), aid in anomaly detection to identify unusual patterns (Gorokhov et al. 2023; Bursic et al. 2019), and enable coherent text generation (Semeniuta et al. 2017; Zhao et al. 2021). Their adaptability and versatility make them indispensable tools in NLP and text analysis, enhancing various aspects of text classification. Another application of AE in the field of NLP is text clustering. In this context, AEs have been applied to organize text documents into meaningful groups. One approach utilizes stacked AEs, combining them with k-means clustering to effectively group text documents into meaningful clusters

(Hosseini and Varzaneh 2022). In Deep Embedded Clustering (DEC), AEs play a pivotal role by initializing feature representations of data points and serving as the foundation for similarity computations during the clustering process. The embeddings learned by AEs are jointly optimized with cluster assignments, thereby enhancing the overall quality of clustering results (Xie et al. 2016; Daneshfar et al. 2023). AEs also provide a solution to the challenges of short text clustering. They address the sparsity problem in short text representations by employing low-dimensional continuous representations or embeddings like Smooth Inverse Frequency (SIF) embeddings. Here, the encoder maps the input short texts to a lower-dimensional continuous representation, and the decoder strives to reconstruct the input from this representation. AEs are used to encode and reconstruct these SIF embeddings, resulting in improved short text clustering quality (Hadifar et al. 2019).

4.3 Complex network

Autoencoders have emerged as valuable tools in complex network analysis, playing a pivotal role in transforming and enhancing network data for various tasks, including network embedding (Cui et al. 2018), deep clustering (Berahmand et al. 2023), and link prediction (Martínez et al. 2016). These applications harness the capability of autoencoders to capture complex, non-linear relationships within network data, enabling more effective and insightful analyses.

Network embedding involves learning compact representations of nodes and edges in a network. Autoencoders excel in this task by seeking optimal non-linear functions to preserve intricate graph structures. For instance, the Structural Deep Network Embedding (SDNE) method (Wang et al. 2016) employs a deep autoencoder approach to address challenges such as high non-linearity, structure preservation, and sparsity. It utilizes multiple non-linear layers to preserve neighbor structures of nodes, enhancing the depth of representation learning. Another method, DNGR (Cao et al. 2016), captures both the weighted graph structure and nodes' non-linear characteristics by employing a random surfing model inspired by PageRank. This approach constructs node representations through a weighted transition probability matrix and employs stacked denoising autoencoders for latent representation learning. Additionally, the adversarial framework ARGA (Pan et al. 2018) aims to balance graph structure reconstruction and enforcing latent code adherence to a prior distribution, producing robust graph representations.

Deep clustering focuses on dividing a network into meaningful clusters of nodes with similar attributes or behaviors. The Marginalized Graph Autoencoder (MGAE) augments autoencoder-based representation learning with GCN to achieve deep node representations (Wang et al. 2017). Shaohua Fan et al. introduce the One2Multi graph autoencoder (Fan et al. 2020), which learns node embeddings by reconstructing multiple graph views using one informative graph view and content data. This approach effectively captures shared feature representations and optimizes cluster label assignments and embeddings through self-training and autoencoder-based reconstruction. In contrast, the N2D method (McConville et al. 2021) simplifies deep clustering by replacing the clustering network with an alternative framework, reducing the complexity of typical deep clustering algorithms.

Link prediction aims to predict missing or future connections in a network based on observed data. In this context, the Heterogeneous Hypergraph Variational Autoencoder (HeteHG-VAE) transforms Heterogeneous Information Networks (HINs) into heterogeneous hypergraphs, capturing both high-order semantics and complex relationships while preserving pairwise topology (Fan et al. 2021). Bayesian deep generative frameworks are used to learn deep latent representations, improving link prediction in HINs. Another method (Salha et al. 2019) inspired by Newtonian gravity extends the graph autoencoder and VAE frameworks to address link prediction in directed graphs, effectively reconstructing directed graphs from node embeddings. Lastly, the Multi-Scale Variational Graph Autoencoder (MSVGAE) introduces a novel graph embedding framework that leverages graph attribute information through self-supervised learning (Guo et al. 2022).

In conclusion, autoencoders are versatile tools for intricate network analysis, contributing significantly to tasks such as network embedding, deep clustering, and link prediction by capturing complex patterns, enhancing representations, and enabling precise predictions.

4.4 Recommender system

Autoencoders find valuable applications in recommendation systems, which aim to suggest items to users based on their historical behavior or preferences. Recommender systems play a pivotal role in various domains, including e-commerce, social media, and online content platforms, offering personalized recommendations to users (Zhang et al. 2019). However, traditional recommender systems grapple with the challenges posed by the immense volume, complexity, and dynamic nature of information (Zhang et al. 2020).

The concept behind autoencoder-based recommender systems involves using AEs to acquire a lower-dimensional representation of both items and users. This representation can subsequently predict a user's preferences for items they haven't yet interacted with. Autoencoder-based recommender systems fall into two categories: pure autoencoder models and integrated autoencoder models, depending on the model architecture employed (Zhang et al. 2020).

In pure autoencoder models, the autoencoder serves as the sole architecture for recommendation. These models rely exclusively on user-item interaction data and/or item features to learn a compressed representation of the data, enabling personalized recommendations. Examples of pure autoencoder models include the Collaborative Denoising Autoencoder (CDAE) (Wu et al. 2016) and Deep Content-based Autoencoder (DCAE) (Van den Oord et al. 2013). CDAE is tailored for collaborative filtering data, where user-item interactions form a sparse matrix. It learns low-dimensional representations of users and items by reconstructing missing entries in the matrix. In contrast, DCAE handles content-based data, representing items as feature vectors. This model learns low-dimensional representations of items by reconstructing the original feature vectors (Wang et al. 2015). Additional examples include Collaborative Filtering Neural Network (CFN) (Strub et al. 2016, 2015), Hybrid Collaborative Recommendation via Semi-Autoencoder (HCRSAE) (Zhang et al. 2017), and Imputation-boosted Denoising Autoencoder (IDAE) (Lee and Lee 2017). Each model has its specific strengths and limitations, rendering them suitable for distinct recommendation scenarios.

In integrated autoencoder models, the autoencoder collaborates with other recommendation models, such as matrix factorization or neural network-based models, to enhance recommendation accuracy. These models use the autoencoder to learn a compressed representation of the data, which is then integrated with other models to generate recommendations (Strub et al. 2016). Examples of integrated autoencoder models include the Hybrid Collaborative Content-based Autoencoder (HCCAE) (Zhang et al. 2017), Variational Autoencoders for Collaborative Filtering (VAE-CFs) (Liang et al. 2018), and Neural Collaborative Autoencoder (NCAE) (He et al. 2017). HCCAE combines the learned representations with other recommendation models, while NCAE utilizes a neural network to generate recommendations directly from the learned representations. These models leverage additional information such as content features, social relationships, or visual data to enhance their recommendations. Each model possesses unique characteristics and objectives, making them suitable for addressing various challenges like cold start problems, sequential data, semantic information, or visual styles.

4.5 Anomaly detection

While AEs have the ability to learn complex patterns in data and detect anomalies that are not easily identifiable, it has been widely used in the field of anomaly detection (Pang et al. 2021). An anomaly detection model can be used to detect a fraudulent transaction or any highly imbalanced supervised tasks (Chandola et al. 2009). AEs can be used in supervised (Alsadhan 2023), unsupervised (Lopes et al. 2022), and semi-supervised (Akcay et al. 2018; Ruff et al. 2019) anomaly detection tasks.

In supervised anomaly detection, AEs are trained on both normal and anomalous data. The AE is first trained on normal data to learn the underlying patterns and features of normal data. Then, the AE is fine-tuned on the combined normal and anomalous data to capture the difference between normal and anomalous data. During training, the objective is to minimize the reconstruction error between the input and the output of the AE. After training, the reconstruction error of the test data is compared to a threshold. If the reconstruction error is above the threshold, the input data is classified as anomalous (Pang et al. 2021). This approach combines the feature learning capabilities of AEs with the discriminative power of supervised classifiers, enhancing the accuracy of anomaly detection in real-world applications, including fraud detection (Alsadhan 2023; Debener et al. 2023; Fanai and Abbasimehr 2023), network security (Ghorbani and Fakhrahmad 2022; Lopes et al. 2022), and fault detection (Ding et al. 2022; Ying et al. 2023) in industrial processes.

In unsupervised tasks, the idea is to train AEs on only sample data of one class (majority class). This way the network is capable of re-constructing the input with good or less reconstruction loss. Now, if a sample data of another target class is passed through the AE network, it results in comparatively larger reconstruction loss, a threshold value of reconstruction loss (anomaly score) can be decided, larger than that can be considered an anomaly (Sakurada and Yairi 2014). This inherent ability to capture complex data representations without labeled anomalies makes AEs effective in detecting anomalies, whether in cyber-security for identifying network intrusions (Lopes et al. 2022; An et al. 2022; Lewandowski and Paffenroth 2022), in manufacturing for spotting defects (Papananias et al. 2023; Sudo et al. 2021), or in finance for fraud detection (Du et al. 2022; Jiang et al. 2023; Kennedy et al. 2023). The versatility of AEs and their capacity to adapt to diverse data types contribute to their widespread use in unsupervised anomaly detection scenarios, enhancing system security and reliability.

AEs have been employed effectively in semi-supervised anomaly detection by capitalizing on their capacity to learn rich data representations (Zhou et al. 2023). In this context, a portion of the training data is labeled as normal, while the majority remains unlabeled. The AE is trained to reconstruct the normal data accurately, and during this process, it learns to capture the underlying structure and features of the normal class. When presented with new, unlabeled data, the AE endeavors to reconstruct it (Ruff et al.

2019). Anomalies, which deviate significantly from the learned normal patterns, result in high reconstruction errors. By setting a suitable threshold on the reconstruction error, anomalies can be effectively detected. This semi-supervised approach minimizes the need for extensive labeled anomaly data and has proven effective in various domains, including fraud detection (Charitou et al. 2020; DeLise 2023; Dzakiyullah et al. 2021), network security (Dong et al. 2022; Hara and Shiomoto 2020; Hoang and Kim 2022; Thai et al. 2022), and quality control (Cacciarelli et al. 2022; Sae-Ang et al. 2022), where labeled anomalies are often scarce.

4.6 Speech processing

Speech processing is focused on enabling machines to understand and interpret human speech with the ultimate objective of creating systems that facilitate natural and intuitive interaction between humans and machines (Hickok and Poeppel 2007). AEs have found numerous applications in speech processing, especially in speech denoising (Bhangale and Kothandaraman 2022; Tanveer et al. 2023), speech recognition (Kumar et al. 2022; Sayed et al. 2023), speech representation (Alex and Mary 2023; Seki et al. 2023), speech compression (Li et al. 2021; Srikotr 2022), feature representation (Shixin et al. 2022; Tian et al. 2022), and speech emotion recognition (Dutt and Gader 2023; Gao et al. 2023).

Speech denoising is a vital process aimed at eliminating unwanted noise from speech signals (Azarang and Kehtarnavaz 2020). AEs have emerged as a powerful tool for this task, where the objective is to enhance the quality of speech by removing noise (Hosseini et al. 2021). In the denoising AE framework, the model is trained using noisy speech samples, with the noisy speech serving as the input and the corresponding clean speech as the target. Through this training, the AE becomes adept at reconstructing noise-free speech from noisy inputs, enabling it to effectively denoise unseen speech signals. The encoder component of the AE extracts informative features from the noisy speech, while the decoder component reconstructs the clean speech based on these extracted features. Denoising AEs have demonstrated remarkable efficacy in mitigating various types of noise in speech signals, including background noise, reverberation, and distortion.

Speech recognition is the process of converting spoken words into text or commands that a computer can understand and execute (Gaikwad et al. 2010). AEs can be used in speech recognition as a pre-processing step for feature extraction. The AE can learn to encode the raw audio signals into a more compact and meaningful representation of the speech signal, which can then be used as input to a speech recognition model. This can improve the accuracy and efficiency of speech recognition systems, especially in noisy or variable acoustic environments (Sayed et al. 2023; Wubet and Lian 2022). Additionally, AEs can be used for speaker identification, where the AE can learn to distinguish between different speakers based on their speech patterns (Liao et al. 2022; Rituerto-González and Peláez-Moreno 2021). A popular approach is using a CNN as the encoder to extract local features from the audio signal, and a RNN as the decoder to capture the temporal dependencies in the speech signal, with the output of the RNN decoder able to transcribe the speech signal (Palaz and Collobert 2015; Rusnac and Grigore 2022).

4.7 Other

Autoencoders have diverse applications in fault diagnosis, intrusion detection, and hyperspectral imaging. They help detect faults in systems, identify network intrusions, and enhance the analysis of hyperspectral data for applications like remote sensing. Different autoencoder versions are tailored to meet specific challenges in these domains.

4.7.1 Fault diagnosis

Fault diagnosis is the process of identifying, isolating, and characterizing faults or anomalies in a system or machine. It involves analyzing the behavior of the system or machine and identifying any deviations from normal or expected behavior. Fault diagnosis is critical in various fields, including manufacturing, automotive, aerospace, and healthcare, as it can help prevent failures, reduce downtime, and improve safety and reliability (Gao et al. 2015). Autoencoders have demonstrated significant potential in fault diagnosis applications. By training an autoencoder on normal data, it can detect deviations from the norm, indicating the presence of a fault or anomaly. To use an autoencoder for fault diagnosis, the initial step is to collect a dataset of normal operating conditions for the system or equipment. This dataset is then employed to train the autoencoder to learn the normal data patterns. Subsequently, it can be applied to new data for fault diagnosis by identifying deviations from these learned patterns citeyang2022autoencoder.

One crucial aspect of using autoencoders for fault diagnosis is selecting an appropriate anomaly detection threshold. Typically, this threshold is determined based on the distribution of the reconstruction error for normal data. Any data that produces a reconstruction error exceeding the threshold is flagged as an anomaly (Ma et al. 2018). Autoencoders are effective for fault diagnosis because they can autonomously learn intricate patterns and recognize deviations from those patterns, eliminating the need for explicit feature engineering. This capability makes them well-suited for detecting subtle anomalies that might be challenging to identify using traditional fault diagnosis methods (Lei et al. 2020).

4.7.2 Intrusion detection

The process of intrusion detection involves continuous monitoring of a system or network to identify and respond to instances of malicious activity or breaches of established policies. Its purpose is to detect anomalous behavior or indicators of potential attacks to prevent or mitigate any potential damage (Farahnakian and Heikkonen 2018). Al-Qatf et al. (2018) have proposed a deep autoencoder-based intrusion detection system that utilizes enhanced representative features to enhance intrusion detection accuracy. The autoencoder extracts representative features from network traffic data, which are subsequently employed to train a classification model for intrusion detection. Another technique to improve intrusion detection systems is the use of Stacked Sparse Autoencoders (SSAE). Yan and Han (2018) utilize SSAE, which is trained on a combination of normal and attack traffic to uncover underlying patterns in network traffic data. These extracted features serve as the basis for training a classifier to detect attacks.

Autoencoders can play a significant role in automatic feature extraction for intrusion detection systems. Kunang et al. (2018) propose a method in which an autoencoder is employed to extract relevant features from raw network traffic data. These extracted features are then used as input for a classifier, such as a Support Vector Machine (SVM), to distinguish between normal and malicious traffic. Compared to traditional rule-based or signature-based methods, autoencoders have the potential to enhance the accuracy and efficiency of intrusion detection systems (Ieracitano et al. 2020).

4.7.3 Hyperspectral imaging

AEs find wide-ranging applications in hyperspectral image analysis due to their ability to learn concise representations of high-dimensional data. Hyperspectral imaging is a potent technique for capturing detailed spectral information about objects or scenes. It involves multi-dimensional data where each pixel contains a spectrum of reflectance or radiance values across numerous narrow, contiguous spectral bands (Jaiswal et al. 2023).

AEs are employed for various tasks in managing hyperspectral data, including hyperspectral data compression (Minkin et al. 2021), hyperspectral unmixing (Książek et al. 2022), blind hyperspectral unmixing (Palsson et al. 2022), and dimensionality reduction (Zabalza et al. 2016). In data compression, AEs condense hyperspectral data while retaining crucial information, facilitating subsequent analysis and processing. Hyperspectral unmixing entails decomposing a hyperspectral image into its constituent parts, referred to as endmembers. AEs play a pivotal role in reconstructing the spectral profiles of these identified components (endmembers) and determining their proportional mixing amounts (abundances). This is indispensable for enhancing the efficiency of hyperspectral analysis and classification tasks (Su et al. 2019). Blind hyperspectral unmixing involves deconstructing the recorded spectrum of a pixel into a mixture of endmembers while simultaneously discerning the proportions or fractions of these endmembers within the pixel. Training an AE on hyperspectral images results in a lower-dimensional representation of the data, rendering it more manageable for subsequent analysis (Petersson et al. 2016).

5 Autoencoder libraries and practical applications

The development and availability of open-source libraries for various versions of AEs have greatly facilitated research in this field. Three popular libraries that are widely used for building and training autoencoder models are TensorFlow, PyTorch, and Keras. Each of these libraries has its strengths and is preferred by different segments of the machine learning and deep learning community. Table 5 presented in this section provides a comprehensive overview of the source code for our proposed category of AE variants. Researchers can access these code repositories to implement and test different versions of AEs, and to compare their performance on various tasks. For instance, one could use the available code to train a variational AE for image reconstruction or a graph attention AE for node embedding. These libraries are not only useful for research but also for practical applications, as they enable practitioners to easily deploy pre-trained models on their own datasets. Table 6 presents a comprehensive overview of various AE

Table 5 AE Models and their corresponding years of publication, programming languages, and code repositories

Subsection	Model	Year	Language	Code Repository
ReAE	SAE	2011	Python	https://github.com/siddharth-agrawal/Sparse-Autoencoder
	CAE	2011	Python	https://github.com/avijit9/Contractive_Autoencoder_in_Pytorch
	LAE	2015	Python	https://github.com/IlMioFrizzantinoAmabile/Laplacian_Autoencoder
	OAE	2019	Python	https://github.com/Ghost———Shadow/orthogonal-autoencoder
RoAE	DAE	2010	Python	https://github.com/jatinshah/ufldl- utorial/tree/master/assignment2
	M-DAE	2012	Python	https://github.com/douxu896/mSDA
	L_2,1-RAE	2018	Python	–
GAE	VAE	2015	Python	https://github.com/keras-team/keras/blob/master/examples/
	AAE	2015	Python	https://github.com/Naresh1318/Adversarial_Autoencoder
	BAE	2021	Python	https://github.com/bangxiangyong/bae-anomaly-uncertainty
	DiffusionAE	2022	Python	https://github.com/phizaz/diffae
CAE	CVAE	2015	Python	https://github.com/o-tawab/Variational-Autoencoder-pytorch
	ConvLSTM	2016	Python	https://github.com/AnthonySMaida/convLSTM-autoencoder
	CSAE	2017	Python	https://github.com/CyprienGille/Sparse-Convolutional-AutoEncoder
RAE	LSTMAE	2016	Python	https://github.com/iwyoo/LSTM-autoencoder
	GRUAE	2014	Python	https://github.com/satolab12/GRU-Autoencoder
	BiRNNAE	2015	Python	https://github.com/ecdraayer/Bidirectional_Autoencoder
SSAE	SSVAE	2017	Python	https://github.com/gcolmenarejo/asva
	DVAE	2016	Python	https://github.com/AndrewSpano/Disentangled-Variational-Autoencoder
	LSRAE	2019	Python	–
GaAE	VGAE	2016	Python	https://github.com/tkipf/gae
	AGAE	2018	Python	https://github.com/GRAND-Lab/ARGA
	GAAE	2019	Python	https://github.com/sktoyo/cancerGATE
MAE	GMAE	2022	Python	https://github.com/THUDM/GraphMAE.
	CMAE	2022	Python	https://github.com/ZhichengHuang/CMAE
	SDMAE	2022	Python	https://github.com/AbrahamYabo/SdAE

models and their diverse applications in machine learning. Each model is associated with specific applications, datasets, methodology, evaluation metrics, and performance results. Notable applications include feature learning, dimensionality reduction, graph-based data representation, generative modeling, anomaly detection, and sequential data analysis. The evaluation metrics vary depending on the application but commonly include error rates, accuracy, precision, recall, F1 score, Area Under the Curve (AUC), and more. These AEs demonstrate their effectiveness in tasks ranging from image classification and sentiment analysis to graph representation learning and acoustic novelty detection, showcasing their versatility in addressing a wide array of machine learning challenges across various domains.

Table 6 AE Models and their corresponding applications

AE model	Application	Methodology	Dataset	Performance
SAE (Ng 2011)	Sparse and Discriminative Feature Learning.	Image classification	MNIST	Error rate = 1.35
		Fault diagnosis	CWRU	ACC = 100
CAE (Rifai et al. 2011)	Feature Extraction and Dimensionality Reduction.	Feature extraction and classification	CIFAR MNIST	Error rate = 47.86 Error rate = 1.14
LAE (Jia et al. 2015)	Graph-based data representation learning.	Manifold generalization	MNIST CIFAR-10	Error rate = 0.98 Error rate = 45.41

OAE (Wang et al. 2019)	Discriminative and diverse feature representations	Data clustering	MNIST	ACC = 95.4
				NMI = 90

DAE (Vincent et al. 2010)	Robust Feature Extraction.	Data classification	MNIST	Error rate = 1.21
M-DAE (Chen et al. 2012)	Anomaly detection	Sentiment analysis	Amazon reviews	Transfer rate = 1.1

$L_{2,1}$-RAE (Li et al. 2018)	Outlier detection	Unsupervised feature learning	MNIST
			Reuters-21578
				ACC = 97.66
				ACC = 82.92

VAE (An and Cho 2015)	Generative modeling.	Anomaly detection	MNIST	AUC ROC = 91.7 AUC PRC = 51.7
AAE (Makhzani et al. 2015)	Generative modeling	Semi-supervised classification Unsupervised clustering	MNIST SVHN	Error rate = 0.85 Error rate = 4.1
BAE (Yong and Brintrup 2022)	probabilistic modeling and generative modeling tasks.	Anomya Detection	ODDS	Uncertainity estimation = 85.6
DiffusionAE (Preechakul et al. 2022)	Generative modeling. Capture intricate data distributions.	Attribute manipulation and conditional generation	CelebA	FID = 5.3
CVAE (Semeniuta et al. 2017)	image generation and image representation learning	Text Generation	Penn Treebank dataset	Rec = 58.5

ConvLSTM (Luo et al. 2017)	Preserve spatial/temporal representation in sequentional data	Anomaly detection	MNIST
			Avenue
				AUC = 99.9
				AUC = 77

CSAE (Luo et al. 2017)	Feature Learning with Perceive locality.	Image Classification	MNIST Caltech-101	Error rate = 0.57 ACC = 66.7

LSTMAE (Nguyen et al. 2021)	Capture representations from sequential data.	Forecasting and anomaly detection	C-MAPSS	ACC = 98.36
				F-score = 96.98

Table 6 (continued)

AE model	Application	Methodology	Dataset	Performance

GRUAE (Dehghan et al. 2014)	Sequential data reconstruction	Determining Parent-Offspring Resemblance	Family 101
			KinFaceW-II
				Precision = 81.5
				Precision = 74.5

BiRNNAE (Marchi et al. 2015)	Capture contextual information from both of sequence directions	Acoustic novelty detection	PASCAL CHiME	Precision = 94.7
				Recall = 92.0

SSVAE (Xu et al. 2017)	Data representation.	Text classification	IMDB AGNews	Error rate = 7.6 Error rate = 7.68
DVAE (Higgins et al. 2016)	Disentanglement representation learning in complex data.	Unsupervised disentanglement representations	celebA	ACC = 83.9
LSRAE (Chai et al. 2019)	Extract the potential features to improve classification.	Image classification	MNIST	ACC = 98.33

VGAE (Kipf and Welling 2016)	Graph-based generative modeling.	Link prediction	Cora	ACC = 63.8
				NMI = 45

AGAE (Pan et al. 2018)	Graph-Based Anomaly Detection.	Link prediction	Cora	AUC = 92.4
				AP = 92.6

GAAE (Salehi and Davulcu 2019)	Graph representation learning.	Node classification	Cora	ACC = 83.2
GMAE (Hou et al. 2022)	Sequence modeling and text generation.	Node classification	Cora	Micro-f = 84.2
CMAE (Huang et al. 2022)	Data augmentation	Image classification, data augmentation	ImageNet-1k	ACC = 85.3
SDMAE (Chen et al. 2022)	Generate high descriptive capability for MAE	Image classification	ImageNet-1k	ACC = 84.1

6 Future directions

Despite in-depth research on autoencoders and their improved algorithms in recent years, the following issues still need to be addressed.

6.1 Semi-supervised and self-supervised learning in autoencoder

Autoencoders, a prominent tool in unsupervised learning, primarily function without the need for labeled data. However, a significant research gap lies in exploring their adaptability to semi-supervised learning paradigms. This entails investigating methodologies for integrating labeled information into the training process, potentially enhancing their performance when only limited labeled data is available. Additionally, another intriguing avenue for exploration is the incorporation of self-supervised learning techniques within autoencoder frameworks. Such an endeavor aims to allow autoencoders to autonomously learn meaningful representations from unlabeled data, reducing their reliance on extensive labeled datasets. Addressing these aspects could significantly expand the applicability and effectiveness of autoencoders across various real-world scenarios with limited labeled data resources.

6.2 Hypergraph autoencoder

Autoencoders have proven effective in preserving the non-linear structure of data due to their deep learning capabilities. However, they face a challenge in preserving higher-order neighbors in complex datasets. While autoencoders can address the former concern, they may not inherently handle the latter. To bridge this gap, integrating hypergraph-based representations of data into the autoencoder framework emerges as a potential solution. By transforming the data into a hypergraph and feeding it as input to the autoencoder, it may be possible to preserve the critical high-order neighbor relationships. This approach holds promise for enhancing the utility of autoencoders in scenarios where preserving intricate data dependencies is crucial, potentially leading to improved performance across various applications.

6.3 Tuning parameter with reinforcement learning

Constructing an autoencoder involves crucial decisions about parameters like the number of hidden layers and nodes, which significantly influence the model's final performance. While parameter selection is essential, the process of identifying the most suitable configuration can be challenging. In current research efforts, some have explored leveraging reinforcement learning techniques in conjunction with autoencoder construction. This novel approach aims to optimize autoencoder parameters efficiently, potentially enhancing model performance. The integration of reinforcement learning into parameter tuning represents an evolving research gap that holds promise for automating and improving the autoencoder design process.

6.4 Handling multi-modal and heterogeneous data with autoencoders

Autoencoders are proficient at capturing patterns in data, especially in scenarios involving different types of data sources or modalities, like text, images, and numerical features, which make data structures more complex. The current challenge lies in effectively handling such multi-modal and heterogeneous datasets. Existing autoencoder models may struggle to efficiently capture and integrate the information present in these intricate datasets. As a result, there is a research gap in developing autoencoder variants or techniques that can adeptly manage multi-modal and heterogeneous data, leading to more comprehensive and valuable data representations. Addressing this gap has the potential to significantly enhance the applicability of autoencoders in various real-world applications.

7 Conclusion

Autoencoders have become a focal point in unsupervised learning due to their remarkable ability to uncover data features and serve as a valuable dimensionality reduction tool. This paper has conducted a thorough examination of autoencoders, covering their fundamental principles and a detailed classification of models based on unique characteristics. We have also explored their use in various areas, from computer vision to natural language processing, highlighting their adaptability. During this study, we've recognized both the advantages and occasional drawbacks of autoencoders. By classifying and summarizing these models based on their unique traits, we've revealed possible directions for future enhancements and innovations. This insight paves the way for further progress in the field.

In summary, autoencoders have an important role in the field of machine learning, and their significance is continuously growing. They have the remarkable ability to find valuable insights in data and create smart results, which can greatly impact various areas. We expect an ongoing journey of progress and important developments in the field of autoencoders, ultimately leading to the creation of even more powerful and intelligent solutions that benefit society as a whole. Autoencoders are positioned to foster innovation and shape the future of machine learning.

Author contributions KB and FD has made a substantial contribution to the concept of the article and drafted the article, ES has made an analysis of the article data, and YL and YX has revised the article.

Funding The authors have not disclosed any funding.

Data availability The data that support the findings of this study are available from the corresponding author upon reasonable request.

Declarations

Conflict of interest The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

Abdi H, Williams LJ (2010) Principal component analysis. Wiley interdisciplinary reviews: computational statistics 2(4):433–459

Aggarwal CC, Zhai C (2012) A survey of text clustering algorithms. Mining Text Data, 77–128

Akcay S, Atapour-Abarghouei A, Breckon TP (2018) Ganomaly: Semi-supervised anomaly detection via adversarial training. In: Computer Vision-ACCV 2018: 14th Asian conference on computer vision, Perth, Australia, December 2-6, 2018, Revised Selected Papers, Part III 14, Springer, pp 622–637

Alex SB, Mary L (2023) Variational autoencoder for prosody-based speaker recognition. ETRI J 45(4):678–689

Al-Qatf M, Lasheng Y, Al-Habib M, Al-Sabahi K (2018) Deep learning approach combining sparse autoencoder with SVM for network intrusion detection. IEEE Access 6:52843–52856

Alsadhan N (2023) A multi-module machine learning approach to detect tax fraud. Comput Syst Sci Eng 46(1):241–253

Alzu'bi A, Albalas F, Al-Hadhrami T, Younis LB, Bashayreh A (2021) Masked face recognition using deep learning: a review. Electronics 10(21):2666

An J, Cho S (2015) Variational autoencoder based anomaly detection using reconstruction probability. Special Lecture IE 2(1):1–18

An P, Wang Z, Zhang C (2022) Ensemble unsupervised autoencoders and gaussian mixture model for cyberattack detection. Inform Process Manag 59(2):102844

Aumentado-Armstrong T, Tsogkas S, Jepson A, Dickinson S (2019) Geometric disentanglement for generative latent shape models. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 8181–8190

Azarang A, Kehtarnavaz N (2020) A review of multi-objective deep learning speech denoising methods. Speech Commun 122:1–10

Balakrishnama S, Ganapathiraju A (1998) Linear discriminant analysis-a brief tutorial. Inst Signal Inform Process 18(1998):1–8

Bank D, Koenigstein N, Giryes R (2020) Autoencoders. arXiv preprint arXiv:2003.05991

Bank D, Koenigstein N, Giryes R (2023) Autoencoders. Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook 353–374

Bank D, Koenigstein N, Giryes R (2023) Autoencoders. Machine learning for data science handbook: Data mining and knowledge discovery handbook, pp 353–374

Berahmand K, Li Y, Xu Y (2023) DAC-HPP: deep attributed clustering with high-order proximity preserve. Neural Comput Appl pp 1–19

Bertalmio M, Sapiro G, CasellesV, Ballester C (2000) Image inpainting. In: Proceedings of the 27th annual conference on computer graphics and interactive techniques, pp 417–424

Bhangale KB, Kothandaraman M (2022) Survey of deep learning paradigms for speech processing. Wireless Pers Commun 125(2):1913–1949

Bursic S, Cuculo V, D'Amelio A (2019) Anomaly detection from log files using unsupervised deep learning. In: International symposium on formal methods, Springer, pp 200–207

Cacciarelli D, Kulahci M, Tyssedal J (2022) Online active learning for soft sensor development using semi-supervised autoencoders. arXiv preprint arXiv:2212.13067

Cao S, Lu W, Xu Q (2016) Deep neural networks for learning graph representations. In: Proceedings of the AAAI conference on artificial intelligence, vol. 30

Chai Z, Song W, Wang H, Liu F (2019) A semi-supervised auto-encoder using label and sparse regularizations for classification. Appl Soft Comput 77:205–217

Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: a survey. ACM Comput Surv 41(3):1–58

Charitou C, Garcez Ad, Dragicevic S (2020) Semi-supervised gans for fraud detection. In: 2020 international joint conference on neural networks (IJCNN), IEEE, pp 1–8

Charte D, Charte F, García S, del Jesus MJ, Herrera F (2018) A practical tutorial on autoencoders for nonlinear feature fusion: taxonomy, models, software and guidelines. Inform Fus 44:78–96

Che L, Yang X, Wang L (2020) Text feature extraction based on stacked variational autoencoder. Microprocess Microsyst 76:103063

Chen S, Guo W (2023) Auto-encoders in deep learning-a review with new perspectives. Mathematics 11(8):1777

Chen Y, Liu Y, Jiang D, Zhang X, Dai W, Xiong H, Tian Q (2022) Sdae: Self-distillated masked autoencoder. In: European conference on computer vision, Springer, pp 108–124

Chen M, Xu Z, Weinberger K, Sha F (2012) Marginalized denoising autoencoders for domain adaptation. arXiv preprint arXiv:1206.4683

Chowdhary K, Chowdhary K (2020) Natural language processing. Fundamentals of artificial intelligence, pp 603–649

Cui P, Wang X, Pei J, Zhu W (2018) A survey on network embedding. IEEE Trans Knowl Data Eng 31(5):833–852

Daneshfar F, Soleymanbaigi S, Nafisi A, Yamini P (2023) Elastic deep autoencoder for text embedding clustering by an improved graph regularization. Expert Syst Appl 121780

Debener J, Heinke V, Kriebel J (2023) Detecting insurance fraud using supervised and unsupervised machine learning. J Risk Insurance

Dehghan A, Ortiz EG, Villegas R, Shah M (2014) Who do i look like? determining parent-offspring resemblance via gated autoencoders. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1757–1764

DeLise T (2023) Deep semi-supervised anomaly detection for finding fraud in the futures market. arXiv preprint arXiv:2309.00088

Ding L, Liu G-W, Zhao B-C, Zhou Y-P, Li S, Zhang Z-D, Guo Y-T, Li A-Q, Lu Y, Yao H-W et al (2019) Artificial intelligence system of faster region-based convolutional neural network surpassing senior radiologists in evaluation of metastatic lymph nodes of rectal cancer. Chin Med J 132(04):379–387

Ding S, Keal CA, Zhao L, Yu D (2022) Dimensionality reduction and classification for hyperspectral image based on robust supervised Isomap. J Ind Prod Eng 39(1):19–29

Ding Y, Zhuang J, Ding P, Jia M (2022) Self-supervised pretraining via contrast learning for intelligent incipient fault detection of bearings. Reliab Eng Syst Saf 218:108126

Dong Y, Chen K, Peng Y, Ma Z (2022) Comparative study on supervised versus semi-supervised machine learning for anomaly detection of in-vehicle can network. In: 2022 IEEE 25th international conference on intelligent transportation systems (ITSC), IEEE, pp 2914–2919

Du X, Yu J, Chu Z, Jin L, Chen J (2022) Graph autoencoder-based unsupervised outlier detection. Inf Sci 608:532–550

Dutt A, Gader P (2023) Wavelet multiresolution analysis based speech emotion recognition system using 1d CNN LSTM networks. IN: IEEE/ACM Transactions on audio, speech, and language processing

Dzakiyullah NR, Pramuntadi A, Fauziyyah AK (2021) Semi-supervised classification on credit card fraud detection using autoencoders. J Appl Data Sci 2(1):01–07

Fan H, Zhang F, Wei Y, Li Z, Zou C, Gao Y, Dai Q (2021) Heterogeneous hypergraph variational autoencoder for link prediction. IEEE Trans Pattern Anal Mach Intell 44(8):4125–4138

Fanai H, Abbasimehr H (2023) A novel combined approach based on deep autoencoder and deep classifiers for credit card fraud detection. Expert Syst Appl 217:119562

Fan S, Wang X, Sh, C, Lu E, Lin K, Wang B (2020) One2multi graph autoencoder for multi-view graph clustering. In: Proceedings of the web conference 2020, pp 3070–3076

Farahnakian F, Heikkonen J (2018) A deep auto-encoder based approach for intrusion detection system. In: 2018 20th international conference on advanced communication technology (ICACT), IEEE, pp 178–183

Foti S, Koo B, Stoyanov D, Clarkson MJ (2022) 3d shape variational autoencoder latent disentanglement via mini-batch feature swapping for bodies and faces. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 18730–18739

Gaikwad SK, Gawali BW, Yannawar P (2010) A review on speech recognition technique. Int J Comput Appl 10(3):16–24

Gao Z, Cecati C, Ding SX (2015) A survey of fault diagnosis and fault-tolerant techniques-part I: fault diagnosis with model-based and signal-based approaches. IEEE Trans Ind Electron 62(6):3757–3767

Gao Y, Wang L, Liu J, Dang J, Okada S (2023) Adversarial domain generalized transformer for cross-corpus speech emotion recognition. IEEE Trans Affect Comput. https://doi.org/10.1109/TAFFC.2023.3290795

García-Mendoza J-L, Villaseñor-Pineda L, Orihuela-Espina F, Bustio-Martínez L (2022) An autoencoder-based representation for noise reduction in distant supervision of relation extraction. J Intell Fuzzy Syst 42(5):4523–4529

Garson GD (2022) Factor analysis and dimension reduction in R: a social Scientist's Toolkit. Taylor & Francis, New York

Geng J, Fan J, Wang H, Ma X, Li B, Chen F (2015) High-resolution SAR image classification via deep convolutional autoencoders. IEEE Geosci Remote Sens Lett 12(11):2351–2355

Ghorbani A, Fakhrahmad SM (2022) A deep learning approach to network intrusion detection using a proposed supervised sparse auto-encoder and SVM. Iran J Sci Technol Trans Electr Eng 46(3):829–846

Girin L, Leglaive S, Bie X, Diard J, Hueber T, Alameda-Pineda X (2020) Dynamical variational autoencoders: a comprehensive review. arXiv preprint arXiv:2008.12595

Gorokhov O, Petrovskiy M, Mashechkin I, Kazachuk M (2023) Fuzzy CNN autoencoder for unsupervised anomaly detection in log data. Mathematics 11(18):3995

Guo X, Liu X, Zhu E, Yin J (2017) Deep clustering with convolutional autoencoders. In: Neural information processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part II 24, Springer, pp 373–382

Guo Z, Wang F, Yao K, Liang J, Wang Z (2022) Multi-scale variational graph autoencoder for link prediction. In: Proceedings of the Fifteenth ACM international conference on web search and data mining, pp 334–342

Guo Y, Zhou D, Ruan X, Cao J (2023) Variational gated autoencoder-based feature extraction model for inferring disease-Mirna associations based on multiview features. Neural Netw

Hadifar A, Sterckx L, Demeester T, Develder C (2019) A self-training approach for short text clustering. In: Proceedings of the 4th workshop on representation learning for NLP (RepL4NLP-2019), pp 194–199

Han C, Wang J (2021) Face image inpainting with evolutionary generators. IEEE Signal Process Lett 28:190–193

Hara K, Shiomoto K (2022) Intrusion detection system using semi-supervised learning with adversarial auto-encoder. In: NOMS 2020-2020 IEEE/IFIP network operations and management symposium, IEEE, pp 1–8

Hasan BMS, Abdulazeez AM (2021) A review of principal component analysis algorithm for dimensionality reduction. J Soft Comput Data Min 2(1):20–30

He X, Liao L, Zhang H, Nie L, Hu X, Chua T-S (2017) Neural collaborative filtering. In: Proceedings of the 26th international conference on world wide web, pp 173–182

Hickok G, Poeppel D (2007) The cortical organization of speech processing. Nat Rev Neurosci 8(5):393–402

Higgins I, Matthey L, Pal A, Burgess C, Glorot X, Botvinick M, Mohamed S, Lerchner A (2016) beta-vae: Learning basic visual concepts with a constrained variational framework. In: International conference on learning representations

Hinton GE, Osindero S, Teh Y-W (2006) A fast learning algorithm for deep belief nets. Neural Comput 18(7):1527–1554

Hoang D-T, Kang H-J (2019) A survey on deep learning based bearing fault diagnosis. Neurocomputing 335:327–335

Hoang T-N, Kim D (2022) Detecting in-vehicle intrusion via semi-supervised learning-based convolutional adversarial autoencoders. Veh Commun 38:100520

Hosseini S, Varzaneh ZA (2022) Deep text clustering using stacked autoencoder. Multimedia tools and applications 81(8):10861–10881

Hosseini M, Celotti L, Plourde E (2021) Speaker-independent brain enhanced speech denoising. In: ICASSP 2021-2021 IEEE international conference on acoustics, speech and signal processing (ICASSP), IEEE, pp 1310–1314

Hou L, Luo X-Y, Wang Z-Y, Liang J (2020) Representation learning via a semi-supervised stacked distance autoencoder for image classification. Front Inform Technol Electron Eng 21(7):1005–1018

Hou Z, Liu X, Cen Y, Dong Y, Yang H, Wang C, Tang J (2022) Graphmae: Self-supervised masked graph autoencoders. In: Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining, pp 594–604

Huang G, Jafari AH (2023) Enhanced balancing GAN: minority-class image generation. Neural Comput Appl 35(7):5145–5154

Huang Z, Jin X, Lu C, Hou Q, Cheng M-M, Fu D, Shen X, Feng J (2022) Contrastive masked autoencoders are stronger vision learners. arXiv preprint arXiv:2207.13532

Ieracitano C, Adeel A, Morabito FC, Hussain A (2020) A novel statistical analysis and autoencoder driven intelligent intrusion detection approach. Neurocomputing 387:51–62

Jain R, Kasturi R, Schunck BG et al (1995) Machine vision, vol 5. McGraw-hill New York, New York

Jaiswal G, Rani R, Mangotra H, Sharma A (2023) Integration of hyperspectral imaging and autoencoders: benefits, applications, hyperparameter tunning and challenges. Comput Sci Rev 50:100584

Jha S, Shah S, Ghamsani R, Sanghavi P, Shekokar NM (2023) Analysis of RNNs and different ML and DL classifiers on speech-based emotion recognition system using linear and nonlinear features. CRC Press, Boca Raton, pp 109–126

Jia K, Sun L, Gao S, Song Z, Shi BE (2015) Laplacian auto-encoders: an explicit learning of nonlinear data manifold. Neurocomputing 160:250–260

Jiang S, Dong R, Wang J, Xia M (2023) Credit card fraud detection based on unsupervised attentional anomaly detection network. Systems 11(6):305

Kennedy RK, Salekshahrezaee Z, Villanustre F, Khoshgoftaar TM (2023) Iterative cleaning and learning of big highly-imbalanced fraud data using unsupervised learning. J Big Data 10(1):106

Kim S, Jang H, Hong S, Hong YS, Bae WC, Kim S, Hwang D (2021) Fat-saturated image generation from multi-contrast MRIs using generative adversarial networks with Bloch equation-based autoencoder regularization. Med Image Anal 73:102198

Kipf TN, Welling M (2016) Variational graph auto-encoders. arXiv preprint arXiv:1611.07308

Kowsari K, Jafari Meimandi K, Heidarysafa M, Mendu S, Barnes L, Brown D (2019) Text classification algorithms: a survey. Information 10(4):150

Książek K, Głomb P, Romaszewski M, Cholewa M, Grabowski B, Búza K (2022) Improving autoencoder training performance for hyperspectral unmixing with network reinitialisation. In: International Conference on Image Analysis and Processing, pp. 391–403. Springer

Kumar S, Rath SP, Pandey A (2022) Improved far-field speech recognition using joint variational autoencoder. arXiv preprint arXiv:2204.11286

Kunang YN, Nurmaini S, Stiawan D, Zarkasi A, et al (2018) Automatic features extraction using autoencoder in intrusion detection system. In: 2018 international conference on electrical engineering and computer science (ICECOS), IEEE, pp 219–224

Le T-D, Noumeir R, Rambaud J, Sans G, Jouvet P (2023) Adaptation of autoencoder for sparsity reduction from clinical notes representation learning. IEEE J Trans Eng Health Med

Lee J-w, Lee J (2017) Idae: Imputation-boosted denoising autoencoder for collaborative filtering. In: Proceedings of the 2017 ACM on conference on information and knowledge management, pp2143–2146

Lee D, Seung HS (2000) Algorithms for non-negative matrix factorization. Adv Neural Inform Process Syst 13

Lei Y, Yang B, Jiang X, Jia F, Li N, Nandi AK (2020) Applications of machine learning to machine fault diagnosis: a review and roadmap. Mech Syst Signal Process 138:106587

Lewandowski B, Paffenroth R (2022) Autoencoder feature residuals for network intrusion detection: Unsupervised pre-training for improved performance. In: 2022 21st IEEE international conference on machine learning and applications (ICMLA), IEEE, pp 1334–1341

Li Y-J, Wang S-S, Tsao Y, Su B (2021) Mimo speech compression and enhancement based on convolutional denoising autoencoder. In: 2021 Asia-pacific signal and information processing association annual summit and conference (APSIPA ASC), IEEE, pp 1245–1250

Li F, Zuraday J, Wu W (2018) Sparse representation learning of data by autoencoders with 1ˆ sub 1/2ˆ regularization. Neural Netw World 28(2):133–147

Li H, Zhang L, Huang B, Zhou X (2020) Cost-sensitive dual-bidirectional linear discriminant analysis. Inf Sci 510:283–303

Li Z, Huang H, Zhang Z, Shi G (2022) Manifold-based multi-deep belief network for feature extraction of hyperspectral image. Remote Sens 14(6):1484

Li X, Li C, Rahaman MM, Sun H, Li X, Wu J, Yao Y, Grzegorzek M (2022) A comprehensive review of computer-aided whole-slide image analysis: from datasets to feature extraction, segmentation, classification and detection approaches. Artif Intell Rev 55(6):4809–4878. https://doi.org/10.1007/s10462-021-10121-0

Liang D, Krishnan RG, Hoffman MD, Jebara T (2018) Variational autoencoders for collaborative filtering. In: Proceedings of the 2018 World Wide Web Conference, pp 689–698

Liao L, Cheng G, Ruan H, Chen K, Lu J (2022) Multichannel variational autoencoder-based speech separation in designated speaker order. Symmetry 14(12):2514

Lin C-C, Hung Y, Feris R, He L (2020) Video instance segmentation tracking with a modified vae architecture. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 13147–13157

Li P, Pei Y, Li J (2023) A comprehensive survey on design and application of autoencoder in deep learning. Appl Soft Comput 110176

Liu Y, Ponce C, Brunton SL, Kutz JN (2023) Multiresolution convolutional autoencoders. J Comput Phys 474:111801

Lopes IO, Zou D, Abdulqadder IH, Ruambo FA, Yuan B, Jin H (2022) Effective network intrusion detection via representation learning: a denoising autoencoder approach. Comput Commun 194:55–65

Luo W, Li J, Yang J, Xu W, Zhang J (2017) Convolutional sparse autoencoders for image classification. IEEE Trans Neural Netw Learn Syst 29(7):3289–3294

Luo W, Liu W, Gao S (2017) Remembering history with convolutional lstm for anomaly detection. In: 2017 IEEE international conference on multimedia and expo (ICME), IEEE pp 439–444

Ma M, Sun C, Chen X (2018) Deep coupling autoencoder for fault diagnosis with multimodal sensory data. IEEE Trans Ind Inf 14(3):1137–1145

Makhzani A, Shlens J, Jaitly N, Goodfellow I, Frey B (2015) Adversarial autoencoders. arXiv preprint arXiv:1511.05644

Ma S, Li X, Tang J, Guo F (2022) Eaa-net: Rethinking the autoencoder architecture with intra-class features for medical image segmentation. arXiv preprint arXiv:2208.09197

Marchi E, Vesperini F, Eyben F, Squartini S, Schuller B (2015) A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional lstm neural networks. In: 2015 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 1996–2000. IEEE

Martínez V, Berzal F, Cubero J-C (2016) A survey of link prediction in complex networks. ACM Comput Surv 49(4):1–33

McConville R, Santos-Rodriguez R, Piechocki RJ, Craddock I (2021) N2d:(not too) deep clustering via clustering the local manifold of an autoencoded embedding. In: 2020 25th international conference on pattern recognition (ICPR), IEEE, pp 5145–5152

McKeown K (1992) Text generation. Cambridge University Press, Cambridge

Medhat W, Hassan A, Korashy H (2014) Sentiment analysis algorithms and applications: a survey. Ain Shams Eng J 5(4):1093–1113

Medsker LR, Jain L (2001) Recurrent neural networks. Design Appl 5(64–67):2

Meyer BH, Pozo ATR, Zola WMN (2022) Global and local structure preserving GPU t-SNE methods for large-scale applications. Expert Syst Appl 201:116918

Miao J, Yang T, Sun L, Fei X, Niu L, Shi Y (2022) Graph regularized locally linear embedding for unsupervised feature selection. Pattern Recogn 122:108299

Minkin A (2021) The application of autoencoders for hyperspectral data compression. In: 2021 international conference on information technology and nanotechnology (ITNT), IEEE, pp 1–4

Miuccio L, Panno D, Riolo S (2022) A wasserstein GAN autoencoder for SCMA networks. IEEE Wireless Commun Lett 11(6):1298–1302

Molaei S, Ghorbani N, Dashtiahangar F, Peivandi M, Pourasad Y, Esmaeili M (2022) Fdcnet: presentation of the fuzzy CNN and fractal feature extraction for detection and classification of tumors. Comput Intell Neurosci 2022

Myronenko A (2019) 3d mri brain tumor segmentation using autoencoder regularization. In: Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part II 4, Springer, pp 311–320

Ng A et al (2011) Sparse autoencoder. CS294A Lecture Notes 72(2011):1–19

Nguyen HD, Tran KP, Thomassey S, Hamad M (2021) Forecasting and anomaly detection approaches using LSTM and LSTM autoencoder techniques with the applications in supply chain management. Int J Inf Manage 57:102282

Ohgushi T, Horiguchi K, Yamanaka M (2020) Road obstacle detection method based on an autoencoder with semantic segmentation. In: proceedings of the Asian conference on computer vision

Palaz D, Collobert R (2015) Analysis of CNN-based speech recognition system using raw speech as input. Report, Idiap

Palsson B, Sveinsson JR, Ulfarsson MO (2022) Blind hyperspectral unmixing using autoencoders: a critical comparison. IEEE J Sel Topics Appl Earth Observ Remote Sens 15:1340–1372

Pang G, Shen C, Cao L, Hengel AVD (2021) Deep learning for anomaly detection: a review. ACM Comput Surv 54(2):1–38

Pang G, Shen C, Cao L, Hengel AVD (2021) Deep learning for anomaly detection: a review. ACM Comput Surv 54(2):1–38

Pan S, Hu R, Long G, Jiang J, Yao L, Zhang C (2018) Adversarially regularized graph autoencoder for graph embedding. arXiv preprint arXiv:1802.04407

Pan S, Hu R, Long G, Jiang J, Yao L, Zhang C (2018) Adversarially regularized graph autoencoder for graph embedding. arXiv preprint arXiv:1802.04407

Papananias M, McLeay TE, Mahfouf M, Kadirkamanathan V (2023) A probabilistic framework for product health monitoring in multistage manufacturing using unsupervised artificial neural networks and gaussian processes. Proc Inst Mech Eng Part B: J Eng Manufact 237(9):1295–1310

Paul D, Chakdar D, Saha S, Mathew J (2023) Online research topic modeling and recommendation utilizing multiview autoencoder-based approach. IEEE Trans Comput Soc Syst

Pereira RC, Santos MS, Rodrigues PP, Abreu PH (2020) Reviewing autoencoders for missing data imputation: technical trends, applications and outcomes. J Artif Intell Res 69:1255–1285

Petersson H, Gustafsson D, Bergstrom D (2016) Hyperspectral image analysis using deep learning-a review. In: 2016 sixth international conference on image processing theory, tools and applications (IPTA), IEEE, pp 1–6

Pratella D, Ait-El-Mkadem Saadi S, Bannwarth S, Paquis-Fluckinger V, Bottini S (2021) A survey of autoencoder algorithms to pave the diagnosis of rare diseases. Int J Mol Sci 22(19):10891

Preechakul K, Chatthee N, Wizadwongsa S, Suwajanakorn S (2022) Diffusion autoencoders: Toward a meaningful and decodable representation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10619–10629

Qian J, Song Z, Yao Y, Zhu Z, Zhang X (2022) A review on autoencoder based representation learning for fault detection and diagnosis in industrial processes. Chemometrics Intell Lab Syst, 104711

Ray P, Reddy SS, Banerjee T (2021) Various dimension reduction techniques for high dimensional data analysis: a review. Artif Intell Rev 54(5):3473–3515. https://doi.org/10.1007/s10462-020-09928-0

Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 779–788

Rifai S, Vincent P, Muller X, Glorot X, Bengio Y (2011) Contractive auto-encoders: Explicit invariance during feature extraction. In: Proceedings of the 28th international conference on international conference on machine learning, pp 833–840

Rituerto-González E, Peláez-Moreno C (2021) End-to-end recurrent denoising autoencoder embeddings for speaker identification. Neural Comput Appl 33(21):14429–14439

Ruff L, Vandermeulen RA, Görnitz N, Binder A, Müller E, Müller K-R, Kloft M (2019) Deep semi-supervised anomaly detection. arXiv preprint arXiv:1906.02694

Rumelhart DE, Hinton GE, Williams RJ, et al (1985) Learning internal representations by error propagation. Institute for Cognitive Science, University of California, San Diego La

Rusnac A-L, Grigore O (2022) CNN architectures and feature extraction methods for EEG imaginary speech recognition. Sensors 22(13):4679

Sae-Ang B-I, Kumwilaisak W, Kaewtrakulpong P (2022) Semi-supervised learning for defect segmentation with autoencoder auxiliary module. Sensors 22(8):2915

Sagha H, Cummins N, Schuller B (2017) Stacked denoising autoencoders for sentiment analysis: a review. Wiley Interdiscip Rev Data Min Knowl Discov 7(5):1212

Saha S, Minku LL, Yao X, Sendhoff B, Menzel S (2022) Split-ae: An autoencoder-based disentanglement framework for 3d shape-to-shape feature transfer. In: 2022 international joint conference on neural networks (IJCNN), IEEE, pp 1–9

Sakurada M, Yairi T (2014) Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis, pp. 4–11

Salehi A, Davulcu H (2019) Graph attention auto-encoders. arXiv preprint arXiv:1905.10715

Salha G, Limnios S, Hennequin R, Tran V-A, Vazirgiannis M (2019) Gravity-inspired graph autoencoders for directed link prediction. In: Proceedings of the 28th ACM international conference on information and knowledge management, pp 589–598

Sayed HM, ElDeeb HE, Taie SA (2023) Bimodal variational autoencoder for audiovisual speech recognition. Mach Learn 112(4):1201–1226

Seki S, Kameoka H, Tanaka K, Kaneko T (2023) Jsv-vc: Jointly trained speaker verification and voice conversion models. In: ICASSP 2023-2023 IEEE international conference on acoustics, speech and signal processing (ICASSP), IEEE, pp 1–5

Semeniuta S, Severyn A, Barth E (2017) A hybrid convolutional variational autoencoder for text generation. arXiv preprint arXiv:1702.02390

Seyfioğlu MS, Özbayoğlu AM, Gürbüz SZ (2018) Deep convolutional autoencoder for radar-based classification of similar aided and unaided human activities. IEEE Trans Aerosp Electron Syst 54(4):1709–1723

Shankar V, Parsana S (2022) An overview and empirical comparison of natural language processing (NLP) models and an introduction to and empirical application of autoencoder models in marketing. J Acad Mark Sci 50(6):1324–1350

Shi D, Zhao C, Wang Y, Yang H, Wang G, Jiang H, Xue C, Yang S, Zhang Y (2022) Multi actor hierarchical attention critic with RNN-based feature extraction. Neurocomputing 471:79–93

Shixin P, Kai C, Tian T, Jingying C (2022) An autoencoder-based feature level fusion for speech emotion recognition. Digital Commun Netw

Shrestha N (2021) Factor analysis as a tool for survey analysis. Am J Appl Math Stat 9(1):4–11

Singh A, Ogunfunmi T (2022) An overview of variational autoencoders for source separation, finance, and bio-signal applications. Entropy 24(1):55

Smatana M, Butka P (2019) Topicae: a topic modeling autoencoder. Acta Polytechnica Hungarica 16(4):67–86

Solorio-Fernández S, Carrasco-Ochoa JA, Martínez-Trinidad JF (2022) A survey on feature selection methods for mixed data. Artif Intell Rev 55(4):2821–2846. https://doi.org/10.1007/s10462-021-10072-6

Song Y, Hyun S, Cheong Y-G (2021) Analysis of autoencoders for network intrusion detection. Sensors 21(13):4294

Song C, Liu F, Huang Y, Wang L, Tan T (2013) Auto-encoder based data clustering. In: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications: 18th Iberoamerican Congress, CIARP 2013, Havana, Cuba, November 20-23, 2013, Proceedings, Part I 18, pp 117–124. Springer

Srikotr T (2022) The improved speech spectral envelope compression based on VQ-VAE with adversarial technique. Thesis

Strub F, Mary J, Gaudel R (2016) Hybrid collaborative filtering with autoencoders. arXiv preprint arXiv:1603.00806

Strub F, Mary J, Philippe P (2015) Collaborative filtering with stacked denoising autoencoders and sparse inputs. In: NIPS workshop on machine learning for ecommerce

Su Y, Li J, Plaza A, Marinoni A, Gamba P, Chakravortty S (2019) DAEN: deep autoencoder networks for hyperspectral unmixing. IEEE Trans Geosci Remote Sens 57(7):4309–4321

Sudo T, Kanishima Y, Yanagihashi H (2021) A study of anomalous sound detection using autoencoder for quality determination and condition diagnosis. IEICE Tech. Rep. 121(284):20–25

Talpur N, Abdulkadir SJ, Alhussian H, Hasan MH, Aziz N, Bamhdi A (2023) Deep neuro-fuzzy system application trends, challenges, and future perspectives: a systematic survey. Artif Intell Rev 56(2):865–913. https://doi.org/10.1007/s10462-022-10188-3

Tanveer M, Rastogi A, Paliwal V, Ganaie M, Malik A, Del Ser J, Lin C-T (2023) Ensemble deep learning in speech signal tasks: a review. Neurocomputing 126436

Thai HH, Hieu ND, Van Tho N, Do Hoang H, Duy PT, Pham V-H (2022) Adversarial autoencoder and generative adversarial networks for semi-supervised learning intrusion detection system. In: 2022 RIVF international conference on computing and communication technologies (RIVF), IEEE, pp 584–589

Tian Y, Xu Y, Zhu Q-X, He Y-L (2022) Novel stacked input-enhanced supervised autoencoder integrated with gated recurrent unit for soft sensing. IEEE Trans Instrum Meas 71:1–9

Tian H, Zhang L, Li S, Yao M, Pan G (2023) Pyramid-VAE-GAN: transferring hierarchical latent variables for image inpainting. Comput Visual Med pp 1–15

Todd JT (2004) The visual perception of 3d shape. Trends Cogn Sci 8(3):115–121

Tripathi M (2021) Facial image denoising using autoencoder and UNET. Herit Sustain Dev 3(2):89–96

Vahdat A, Kautz J (2020) Nvae: a deep hierarchical variational autoencoder. Adv Neural Inf Process Syst 33:19667–19679

Van den Oord A, Dieleman S, Schrauwen B (2013) Deep content-based music recommendation. Adv Neural Inform Process Syst 26

Van Der Maaten L, Postma EO, van den Herik HJ et al (2009) Dimensionality reduction: a comparative review. J Mach Learn Res 10(66–71):13

Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol P-A, Bottou L (2010) Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. J Mach Learn Res 11(12)

Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol P-A, Bottou L (2010) Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. J Mach Learn Res 11(12)

Wang W, Yang D, Chen F, Pang Y, Huang S, Ge Y (2019) Clustering with orthogonal autoencoder. IEEE Access 7:62421–62432

Wang G, Karnan L, Hassan FM (2022) Face feature point detection based on nonlinear high-dimensional space. Int J Syst Assurance Eng Manag 13(Suppl 1):312–321

Wang D, Cui P, Zhu W (2016) Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 1225–1234

Wang D, Li T, Deng P, Zhang F, Huang W, Zhang P, Liu J (2023) A generalized deep learning clustering algorithm based on non-negative matrix factorization. ACM Trans Knowledge Discovery Data

Wang C, Pan S, Long G, Zhu X, Jiang J (2017) Mgae: Marginalized graph autoencoder for graph clustering. In: Proceedings of the 2017 ACM on conference on information and knowledge management, pp 889–898

Wang H, Wang N, Yeung D-Y (2015) Collaborative deep learning for recommender systems. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, pp1235–1244

Wu C, Wu F, Wu S, Yuan Z, Liu J, Huang Y (2019) Semi-supervised dimensional sentiment analysis with variational autoencoder. Knowl-Based Syst 165:30–39

Wubet YA, Lian K-Y (2022) Voice conversion based augmentation and a hybrid CNN-LSTM model for improving speaker-independent keyword recognition on limited datasets. IEEE Access 10:89170–89180

Wu Y, DuBois C, Zheng AX, Ester M (2016) Collaborative denoising auto-encoders for top-n recommender systems. In: Proceedings of the Ninth ACM international conference on web search and data mining, pp 153–162

Xie J, Girshick R, Farhadi A (2016) Unsupervised deep embedding for clustering analysis. In: International conference on machine learning, PMLR, pp 478–487

Xu W, Keshmiri S, Wang G (2019) Adversarially approximated autoencoder for image generation and manipulation. IEEE Trans Multimed 21(9):2387–2396

Xu H, Ding S, Zhang X, Xiong H, Tian Q (2022) Masked autoencoders are robust data augmentors. arXiv preprint arXiv:2206.04846

Xu W, Sun H, Deng C, Tan Y (2017) Variational autoencoder for semi-supervised text classification. In: Proceedings of the AAAI conference on artificial intelligence, vol. 31

Yan B, Han G (2018) Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system. IEEE Access 6:41238–41248

Yang B, Fu X, Sidiropoulos ND, Hong M (2017) Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In: International conference on machine learning, pp 3861–3870. PMLR

Yang X, Song Z, King I, Xu Z (2022) A survey on deep semi-supervised learning. IEEE Trans Knowl Data Eng

Ye H, Zhang W, Nie M (2022) An improved semi-supervised variational autoencoder with gate mechanism for text classification. Int J Pattern Recognit Artif Intell 36(10):2253006

Ying LJ, Zainal A, Norazwan MN (2023) Stacked supervised auto-encoder with deep learning framework for nonlinear process monitoring and fault detection. In: AIP conference proceedings, vol. 2785. AIP Publishing

Yong BX, Brintrup A (2022) Bayesian autoencoders with uncertainty quantification: Towards trustworthy anomaly detection. Expert Syst Appl 209:118196

Zabalza J, Ren J, Zheng J, Zhao H, Qing C, Yang Z, Du P, Marshall S (2016) Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging. Neurocomputing 185:1–10

Zhang Y, Zhang E, Chen W (2016) Deep neural network for halftone image classification based on sparse auto-encoder. Eng Appl Artif Intell 50:245–255

Zhang S, Yao L, Sun A, Tay Y (2019) Deep learning based recommender system: a survey and new perspectives. ACM Comput Surv 52(1):1–38

Zhang R, Yu L, Tian S, Lv Y (2019) Unsupervised remote sensing image segmentation based on a dual autoencoder. J Appl Remote Sens 13(3):038501–038501

Zhang G, Liu Y, Jin X (2020) A survey of autoencoder-based recommender systems. Front Comp Sci 14:430–450

Zhang G, Liu Y, Jin X (2020) A survey of autoencoder-based recommender systems. Front Comp Sci 14:430–450

Zhang S, Yao L, Xu X, Wang S, Zhu L (2017) Hybrid collaborative recommendation via semi-autoencoder. In: Neural information processing: 24th international conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part I 24, Springer, pp 185–193

Zhang C, Zhang C, Song J, Yi JSK, Kweon IS (2023) A survey on masked autoencoder for visual self-supervised learning

Zhang C, Zhang C, Song J, Yi JSK, Zhang K, Kweon IS (2022) A survey on masked autoencoder for self-supervised learning in vision and beyond. arXiv preprint arXiv:2208.00173

Zhang C, Zhang C, Song J, Yi JSK, Zhang K, Kweon IS (2022) A survey on masked autoencoder for self-supervised learning in vision and beyond. arXiv preprint arXiv:2208.00173

Zhao K, Ding H, Ye K, Cui X (2021) A transformer-based hierarchical variational autoencoder combined hidden Markov model for long text generation. Entropy 23(10):1277

Zhou F, Wang G, Zhang K, Liu S, Zhong T (2023) Semi-supervised anomaly detection via neural process. IEEE Trans Knowl Data Eng

Zhu Z, Wang X, Bai S, Yao C, Bai X (2016) Deep learning representation using autoencoder for 3d shape retrieval. Neurocomputing 204:41–50

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Terms and Conditions