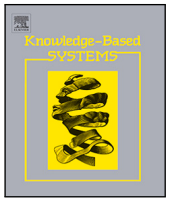
Knowledge-Based Systems 285 (2024) 111315

Contents lists available at ScienceDirect

Knowledge-Based Systems

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



Deep clustering framework review using multicriteria evaluation

Frédéric Ros [a,*], Rabia Riad [b], Serge Guillaume [c]

[a] Laboratory PRISME, Orléans University, France
[b] Laboratory PETI, Ibnou Zohr University, Morocco
[c] ITAP, Univ Montpellier, INRAE, Montpellier SupAgro, Montpellier, France

ARTICLE INFO

Keywords:
Clustering
Domain representation
Unsupervised learning
Supervised learning
Deep learning

ABSTRACT

The application of clustering has always been an important method for problem-solving. In the era of big data, most classical clustering methods suffer from the curse of dimensionality and scalability issues. Recently, deep clustering models have garnered more attention due to their capabilities in dealing with complex, high-dimensional, and large-scale datasets. They offer intriguing perspectives owing to their outstanding representative capacity and fast inference speed. The remaining major problem in clustering scenarios with high-dimensional data revolves around determining an appropriately compressed representation that semantically preserves cluster structures. Without labels, defining an objective function to encourage a suitable representation becomes a critical question. After several years of stagnation, impressive results have been achieved in the last two years. This paper proposes a comprehensive and up-to-date review of deep clustering methods. We first introduce the basic concepts shared by several deep clustering algorithms, available network architectures, and optimization strategies. Then, a detailed review is presented for each family by analyzing their most representative algorithms. These algorithms are then assessed based on their classification accuracy and from a multi-criteria perspective to aid investigators in selecting the most appropriate solution. Finally, an overview of the diversity of tasks and application domains is provided, and current issues and challenges are discussed.

1. Introduction

Clustering is an unsupervised process that consists of organizing items according to their similarity so that similar objects belong to the same group and are different from those that belong to other groups. It is one of the major unsupervised learning techniques and has been applied to many fields. Many clustering algorithms have been proposed in recent decades [1]. To be efficient they must include outlier and noise management. In the big data era dimensionality reduction of the input space and scalability of the algorithms are of prime concern. The increasing dimensionality degrades the performance of machine learning algorithms, including clustering ones.

The curse of dimensionality is a fundamental issue, particularly for clustering. The first method to appear for dimensionality reduction was feature selection in the raw data space [2–4]. This process can be based on attribute redundancy or relevance to a given objective. For a feature to be helpful, it must possess a certain level of consistency and/or semantic information. When dealing with data signals or images, selecting higher-level features containing semantic information is necessary as a single piece of information is usually insufficient. These techniques proved to be limited when dealing with high-dimensional data, especially for unsupervised tasks since the task is unassisted by sample labels, and there is still a lack of consensus on the correct optimization objective. Another popular technique is based on input space transforms such as Independent Component Analysis (ICA), Local Linear Embedding (LLE) [5] or Isomap [6]: the idea is to build a reduced number of latent components based on a linear combination of the initial variables. As an example using Principle Component Analysis (PCA), the latent variables are orthogonal to each other and designed to explain a decreasing part of the input variance. Such input space transforms result in a loss of interpretability.

Neural network techniques have also been used to produce reduced representation spaces: Self-organizing maps (SOM) [7], Hebbian Learning [8] and Deep Belief Network (DBN) [9] were the first neural network based methods able to discover hidden structures in unlabeled data. Deep neural networks are part of this history but brought significant improvements thanks to powerful convolutional and pooling layers combined with self-learning strategies. The major problem in unsupervised scenarios remains the question of good representation, and clustering cannot be done without transforming/reducing the original feature space in a latent space. Without labels, what should be

* Corresponding author.
E-mail addresses: frederic.ros@univ-orleans.fr (F. Ros), r.riad@uiz.ac.ma (R. Riad).

https://doi.org/10.1016/j.knosys.2023.111315
Received 25 February 2023; Received in revised form 10 December 2023; Accepted 20 December 2023
Available online 3 January 2024

represented is not even clear. How can one define an objective function able to capture the semantics of the data? In any data distribution, two major factors are generally tracked: variance and entanglement [10]. Thanks to the development of deep learning, such feature transformation can be achieved by using Deep Neural Networks and a strategy where clustering and feature learning can be done simultaneously. This kind of clustering is referred to as deep clustering. Deep clustering methods combine feature extraction, dimensionality reduction, and clustering in an end-to-end model in which the deep neural networks learn suitable representations in order to adapt to the assumptions and criteria of the clustering module.

After ten years of research, deep clustering algorithms have proved to be efficient in achieving a large diversity of tasks in many application domains, especially in the field of image analysis. The time has come to take stock of the field as deep clustering architectures have now reached a high level of development and learning strategies have very recently been significantly improved by using pseudo-supervised data to handle unsupervised tasks. The unprecedented success can explain the increasing attention paid to deep learning the method has achieved in solving high-dimensional data problems, but also by the high level of automation that it includes: the increased complexity, compared to classical techniques, did not lead to a similar increase of human involvement. Two research directions proved useful in coping with huge data sets. First, transfer learning allows the reuse of trained models for a specific task, but also the labeling of unknown data using either a small sample of labeled data or data augmentation techniques. Research work also yielded new proposals in network architecture, e.g., Siamese neural networks or generative models, as well as new regularization functions.

Categorically, all deep clustering algorithms share the common goal of constructing a representation space conducive to the application of clustering tasks. However, their distinction lies in two pivotal facets: the approach used to derive this representation space and the strategy employed to organize and optimize these tasks. These methods can be broadly classified into four architectural families, each inherently influencing the process and being more or less suitable for different types of data.

Historically, the first one was the Autoencoders (AE) [11]. An AE aims to reconstruct the input data by learning an efficient representation of the sample itself. It includes two main components: the encoder maps the input data to a lower-dimensional latent space and the decoder rebuilds the original data from this reduced space. AEs are often used for dimensionality reduction and feature learning. They may include a convolutional layer and are then called Convolutional Autoencoders (CAE). For clustering purposes, they use a variety of losses to encourage data points to form tightly packed and well-separated clusters in the latent space. Inspired by Deep-embedded clustering (DEC) [12], several deep clustering methods have been proposed and have obtained promising preliminary results even if there is no guarantee that clusters in the latent space correspond to semantic clusters in the input space.

The second family is a set of methods based on a common architecture often referred to as the 'backbone' (ResNet18, ResNet34...). It is hereafter called CNN, which stands for Convolutional Neural Networks [13]. Their common objective is discrimination, either classification, recognition, or segmentation. Deep clustering based on CNN has recently reached a milestone thanks to the development of smart contrastive learning [14], as well as self-supervised learning, approaches [15]. These learning strategies are based on the idea that the data themselves contain inherent features that provide supervision for training the model. Then, using so-called pretext tasks, they allow the network to learn high-level features and, therefore, obtain semantically meaningful representations from unlabeled data. While initially designed for processing image data, their impact has extended beyond various other data types.

The third category consists of methods based on Graph Neural Networks (GNNs) [16–18]. Graphs are an integral part of our environment, representing real-world objects through their interconnected relationships. A group of objects and the connections that bind them together are inherently modeled as a graph. A typical model defined to solve graph-based problems either operates on the original graph adjacency matrix [19] or on a derived vector space [20]. Graphs for unsupervised tasks have been investigated for a long time [21–23] and can be of two types: structure-based (community-based, and structurally equivalent clustering) and attribute-based clustering (labels and observed links are considered to cluster nodes). Conventional methods for graph processing, including widely-used techniques like random walks [24] and matrix factorization [25], usually begin with an initial step in which the graph is transformed into a more straightforward data representation, such as a vector or a sequence of real numbers. Over the years, researchers have been refining neural networks tailored for graph data, often referred to as GNNs, which can work directly with input graphs via a diffusion process, effectively incorporating their connectivity into the processing framework. GNNs have found success in various domains with graph-based datasets, performing well in a range of graph analysis tasks, including node classification and link prediction. However, the complexity of graph data has posed significant challenges to existing machine learning algorithms, noticeably representation learning on graphs is far more complex than image data since there is no spatial locality in graphs. Deep clustering on graphs has proved to be more resilient to advances in GNNs [26] but very promising approaches [27–29] have recently emerged motivated by the achievements of many GNN-based methods in encoding graph structures, as well as advancements in generative, adversarial, and contrastive learning schemes.

The fourth family is "generative models". Generative Adversarial Networks (GAN) [30–32] generate new pattern instances that resemble training samples by discovering and learning the regularities in input patterns. They learn the complex data distribution by playing a min–max game. The generator is trained to create new patterns, while the discriminator model learns to differentiate between genuine patterns and those generated by the generator. Variational Autoencoders (VAE) [33], even if they inherit the AE architecture, belong to this group as they are trained to estimate the data distribution. They are capable of generating *nice* samples thanks to their powerful latent representation of great worth for knowledge discovery tasks and are used in clustering methods based on these generative approaches.

In specific contexts, especially those involving sequences and time series data, the primary architectures can be enriched by incorporating additional elements such as RNNs and, more recently, Transformers [34]. Notably, these supplementary architectures are worth mentioning, even though recent advancements in deep clustering are more closely associated with image data and not yet relevant in the time domain [35]. Recurrent Neural Networks (RNNs) and their diverse adaptations [36], notably Long Short-Term Memory (LSTM), bidirectional LSTM, Gated Recurrent Unit (GRU) networks, and bidirectional GRU were originally designed for the processing of time series data and sequence-related scenarios [37] such as natural language processing (NLP), video classification, and speech recognition. These neural architectures have enjoyed extensive use in supervised learning, yet they are still emerging in the domains of unsupervised clustering and in the early phases of deep clustering. Transformers, as well as Transformer-based architectures like GPT [38] and BERT [39], are characterized by their encoder–decoder structure predominantly reliant on attention mechanisms. They have set state-of-the-art (SOTA) records in sequence-to-sequence tasks across a range of NLP-supervised applications and Vision Transformers (ViTs) [40] have sparked significant interest in the field of computer vision while still evolving in unsupervised scenarios [41].

Many reviews of standard clustering are available [1,42–44], as well as abundant literature on clustering time series, trajectory data, or

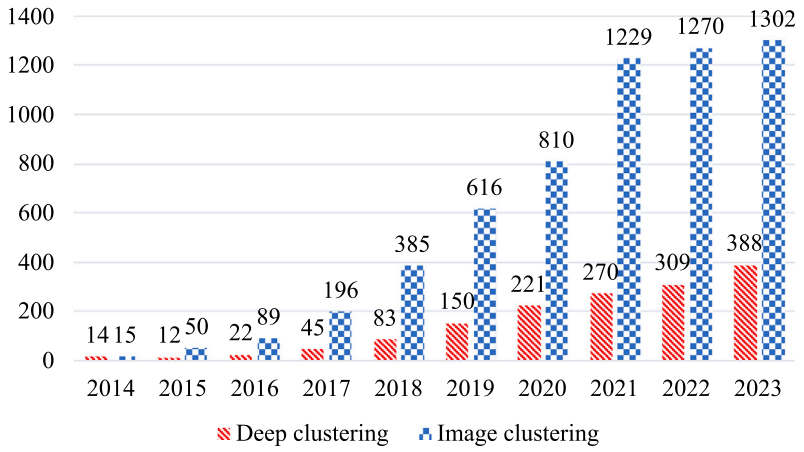


Fig. 1. Deep clustering papers from 2014 to 2023 (published version or accepted papers, based on title/source Scopus).

spatiotemporal data, a few for general deep clustering [45–50] and also several for specific domains such as [51–55]. During the last few years, deep clustering has attracted increasing attention from scientists, as shown in Fig. 1. Simultaneously, works involving clustering and images are in great progress. Nevertheless, before 2020, disruptive innovations in deep clustering were scarce, with a predominant focus on tasks related to image data.

State-of-the-art methods generated excellent results on simple datasets but encountered more difficulties on complex or unfriendly datasets, as the latent space is not semantically discriminative enough.

Impressive progress has been made recently, and even the latest reviews are not up-to-date, as the majority of works primarily encompass methods predating the year 2020, thereby overlooking more recent breakthroughs in the past three years. The present survey aims to improve existing reviews, and

our contribution can be summarized as follows:

- Providing an encompassing overview of essential concepts vital for understanding the key innovations propelling contemporary advancements, this perspective extends beyond specific algorithms and architectures, offering valuable insights particularly beneficial for young researchers, a viewpoint not commonly explored in existing reviews.
- A dual-level taxonomy encompassing "architecture" and "process", demonstrating their complementary nature. Most publications focus on a single taxonomy, such as architecture [46,56], representation learning [45] or given data sources [50].
- A user-friendly and intuitive exposition of selected prominent and representative techniques within each category, aiming to facilitate comprehension, in contrast to the typically formal or concise descriptions found in other reviews.
- An exhaustive and all-encompassing examination of the current state of Graph Neural Networks (GNNs). GNN-based architectures are discussed in one general review paper [50], albeit lacking comprehensive details.
- Examination of innovations from an algorithmic perspective, extending to 2022–2023, backed by a substantial collection of relevant references. This review is kept current by incorporating the latest groundbreaking papers. Notably, the most recent review papers include no more than five references from 2022.
- A concise comparative analysis, emphasizing the most relevant methods for processing both image and text data across several recognized benchmarks, a unique feature that distinguishes it from other reviews.
- Exploration of a multi-criteria approach. The representative algorithms are synthetically described to highlight the different kinds of innovations. They are analyzed not only according to their classification accuracy but also considering other criteria that are important to the user, a unique characteristic not observed in other reviews.
- Categorization of technologies across tasks and domains, along with a synthetic and comprehensive state-of-the-art review. Notably, one review paper specifically addresses applications in various domains.
- Several issues and challenges are explored, with a particular emphasis on the unsupervised paradigm, a unique aspect not typically covered in other reviews.

The rest of the paper is organized as follows. The next section describes the basic concepts that are shared by several deep clustering algorithms, starting from the input data domain representation, and then focusing on automatic learning components and strategies. In the second part of the section the available network architectures are introduced, including the proposition of the four families, and a focus on the loss functions that are of major importance for the training is proposed. Section 3 details the four families by analyzing their most representative algorithms. In Section 4, the algorithms, organized by family, are compared according to their classification accuracy but also to other criteria such as the type of data they can deal with, the attention they pay to semantics, or the skills required for their use. An overview of the diversity of tasks performed by these algorithms and the application domains is given. The distribution of the latter by family is also provided. This section ends with the issues and remaining challenges that are likely to be addressed in the near future. Finally, Section 7 summarizes the main conclusions.

2. Overview of the deep clustering area

This section starts by presenting basic concepts that are especially useful and essential for non-experts and young researchers to be familiar with the domain. It is followed by an overall presentation of the deep clustering framework.

2.1. Basic concepts

We begin by providing an overview of "Representation Learning", a fundamental concept that underlies deep clustering approaches and addresses the challenges posed by the curse of dimensionality in the field of clustering. Next, we introduce key concepts that have contributed to the remarkable achievements of recent deep learning approaches. While these concepts are interconnected, they primarily revolve around novel learning strategies, such as contrastive and self-supervised learning, which heavily rely on data augmentations and the concept of pretext tasks. Pretext tasks serve as secondary objectives for the network

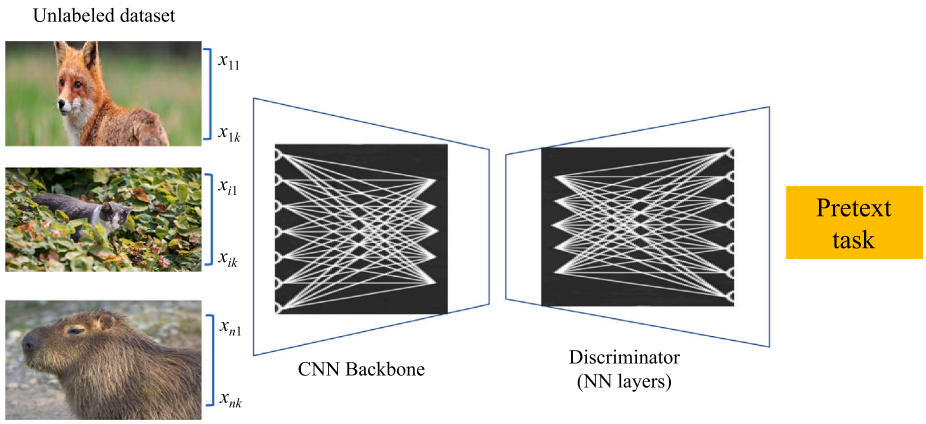


Fig. 2. Self-supervision pretext task training: different views of each input pattern (x_{i1}, ..., x_{ik}) are learned by learning objective functions of pretext tasks.

to solve, without being the primary objectives themselves. It is worth noting that contrastive learning is a specific type of self-supervised learning that focuses on learning representations by contrasting similar and dissimilar pairs of samples. On the other hand, self-supervision encompasses a broader range of techniques that train models on pretext tasks to learn meaningful representations without explicit supervision. Furthermore, we gradually introduce these concepts, accompanied by a brief overview of techniques such as self-paced learning, regularization, and attention mechanisms. These aspects collectively dominate the influential literature in the field.

2.1.1. The essential topic: Domain representation

Representation learning [57–60] is a fundamental concept in machine learning and artificial intelligence that involves learning meaningful and informative representations or features from raw data. The goal is to capture the underlying structure and patterns in the data for use in various tasks such as classification, clustering, and prediction. Representation learning aims to map representations to other representations and generate dense and compact learned representations that can be generalized to similar data modalities. Deep learning algorithms, including deep clustering algorithms, are powerful because they primarily perform representation learning. Good representations [61] are expressive, meaning they can capture a vast number of possible input configurations with a reasonably-sized learned representation. One of the challenges in representation learning is the absence of a clear objective or target for training, unlike tasks such as classification. Nevertheless, the central objective remains the preservation of information to discover and capture latent semantic structures in the data. This enables models to gain a better understanding of the underlying concepts and relationships and make more informed decisions. Semantics, in general, refer to the meaning or interpretation of language elements. The relation between semantics and representation learning lies in the fact that representation-learning techniques aim to capture semantic information in learned representations. Whether in natural language processing or computer vision, deep learning models learn hierarchical representations that encode semantic information about the content of the data. These learned representations not only solve the immediate task but also possess generalization properties, making them useful for other downstream tasks such as object detection, segmentation, and pose estimation. Transfer learning is a popular approach that leverages the knowledge gained from one task to solve another task with limited annotations. In summary, representation learning plays a crucial role in capturing and encoding semantic information from data, enabling models to understand and utilize the underlying meaning and relationships between different elements. This has broad applications in various domains and contributes to the advancement of machine learning and artificial intelligence.

2.1.2. Self-supervised learning

The idea of Self-supervised learning (SSL) [15] consists of deriving pseudo labels by analyzing how different parts of the data interact to learn semantically meaningful features from unlabeled data. These pseudo labels are obtained in a semi-automated manner, without human input. As they are supervisory signals, they allow the application of supervised learning techniques. Pseudo-labeling is then the process of using the labeled data model to predict labels for unlabeled data. During self-supervised training, the representations are learned through solving automatically generated pretext tasks. In this context, the creation of different versions, often called “views”, of a given data point through data augmentation is an essential part of various self-supervised approaches (cf. Fig. 2). There is a plethora of self-supervised representations on the market using diverse pre-text tasks and data augmentations. While some of them are well-recognized in the image field, there are no real standard techniques, and most of them are based on heuristics or trials.

Self-supervised learning and unsupervised learning methods can be considered complementary learning techniques as neither needs labeled datasets. Unsupervised learning can be considered as the superset of self-supervised learning as it does not have any feedback loops. On the contrary, self-supervised learning has a lot of supervisory signals that act as feedback in the training process. An easier way to put it is that the unsupervised learning technique focuses primarily on the model and not on the data whereas the self-supervised learning technique works the other way around. To sum up, unsupervised learning methods are good at clustering and dimensionality reduction, while self-supervised learning is a pretext method for regression and classification tasks, which can contribute to clustering.

The core procedure of SSL is first designing a domain-specific pretext task and training the networks on the pretext task, such that the learned representations can be more discriminative and applicable.

Differently, semi-supervised methods aim at solving such problems by using labeled and unlabeled information to measure correlation.

SSL revitalizes and outperforms in various domains. It excels at learning valuable representations from unlabeled data. Therefore, it does not require explicit labels.

2.1.3. Pretext task

The aim of the pretext task, also known as a supervised task, is to guide the model to learn intermediate representations of data. Once we pick a pretext task, data augmentation generally takes place: the data augmentation process encourages the network to learn hidden representations by solving the pretext task and helping the model to capture the essential features in the latent space. A pretext task acts as a proxy that makes the network learn useful representations that can be used to ease the process of learning different downstream tasks and help in understanding the underlying structural meaning. They are designed

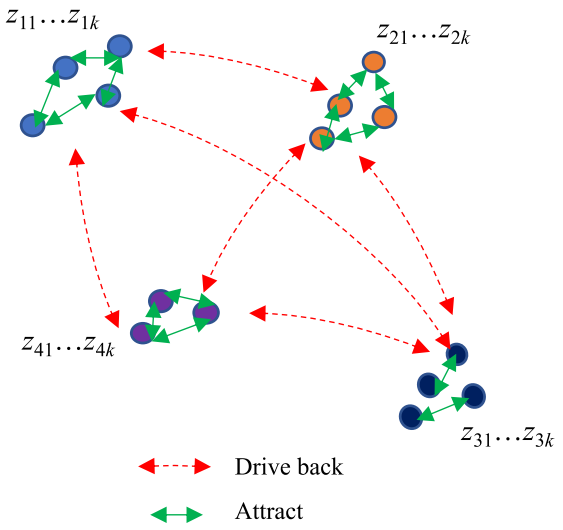


Fig. 3. DNA of contrastive learning: positive pairs (same color) need to be concentrated in the projective space z while being separated from negative pairs.

by directly generating supervisory signals from the raw images without manual labeling and aim to learn well-pre-trained representations for downstream tasks, such as image classification, object detection, and semantic segmentation. The complexity of the data augmentation, or pretext task, must challenge the model. Pretext tasks in the image field are relative positioning, rotation, colorization, jigsaw puzzle or Mutual information and instance discrimination, and so on. Creating pretext tasks resembles the process of hand-designing features for a classifier. It is not clear which pretext tasks work and why they work. It is essential to identify the right pretext task. It has to help the model to be invariant to transformations of one data point while remaining discriminative to other data points. Recent self-supervised learning methods have deviated to a general approach that involves a kind of instance discrimination pretext task combined with contrastive-based loss functions. Popular pretext tasks include the reconstruction pretext and the adversarial objective [15,62,63].

2.1.4. Contrastive learning

Contrastive learning [63] has emerged as an effective technique for enhancing deep clustering performance. It typically generates positive sample pairs and negative sample pairs via data augmentations. Positive sample pairs represent similar samples that should have similar representations, they are aligned. Empirically, they are often obtained by taking two independently randomly augmented versions of the same sample. Negative pairs refer to unrelated samples and then should be separated. Contrastive methods aim to maximize the agreement between positive pairs and minimize the agreement between negative pairs. In other words, they learn representations by enforcing similar elements to be equal and dissimilar elements to be different (see Fig. 3) The idea behind this is to obtain invariance to irrelevant details or transformations by decreasing the distance between positive pairs while increasing the distance between negative pairs for solving the tasks. There is a non-productive effect in the case of negative pairs of the same "class" as the contrastive objective undesirably pushes them apart.

This technique enhances the performance of tasks by using the principle of contrasting samples against each other to learn attributes that are common between data classes and attributes that set one data class apart from another. Its corresponding pretext task is that the features encoded from multi-views of the same input are similar to each other. The core insight behind these methods is to learn multi-view invariant representations. This is done by minimizing pretext task objectives, the specificity being that inputs and "labels" are derived from the unlabeled data, especially by using data augmentations. In recent months, an explosion of unsupervised Deep Learning methods based on these principles has been seen. It can be used for supervised scenarios but also for self-supervised ones [14,64] that focus more on the knowledge discovery task. The principle has been increasingly investigated [65–70], and the results obtained appear very promising. It is still an active area of research as it might be challenging to choose the "right" and appropriate transformations for a specific task. The most competitive deep clustering algorithms are now based on this technique.

2.1.5. Data augmentation

Data augmentation [71–73]is a process of artificially increasing the amount of data by generating new data points from existing data, generally via the modification of examples within the original dataset. This includes adding minor alterations to data or using machine learning models to apply various transformations under the domain knowledge. The common objective is to generate new data points without changing the semantic characteristics of the data (cf Fig. 4).

Data augmentation can be applied to all machine learning applications where acquiring quality data is challenging. The challenge lies in determining the appropriate modifications that should be under control: strong augmentations may alter the sample identity of the positives, while weak augmentation produces easy positives/negatives, resulting in nearly-zero loss and ineffective learning. Ideally, the transformations should be adapted to better capture the data semantics. Advanced models for data augmentation include adversarial machine learning, GANs, and neural style transfer. Data augmentation constitutes a form of weak supervision currently used in contrastive and self-supervised learning strategies.

2.1.6. Self-paced learning

In machine learning, how to select training samples to learn more effective models is an active research topic. The idea behind Self-paced learning is to better guide the clustering process by selecting suitable samples adaptively along the clustering process. It is based on the concept of curriculum learning [74,75] which consists of a gradual introduction of the difficulties: first learning simple knowledge, followed by learning more difficult and professional knowledge. Self-paced learning works under the assumption that easy samples with smaller losses ought to be selected in the early stage while complex samples with larger losses will be supposed to be selected later or not. In the process of self-paced learning, the initial step involves selecting a subset of samples with minimal construction errors for training, aiming to obtain accurate training models. Additional samples are then incorporated by gradually increasing the threshold value to enhance the generalization ability of the training model until the established model achieves stability. This trick is widely used in the deep clustering area such as in [76].

2.1.7. Regularization techniques

Regularization [77,78] is a set of methods that both optimize the learning of a Deep Learning model and counter overfitting to prevent lack of generalization. Most deep clustering methods include regularization terms. These methods also offer an alternative way to learn classifications for data sets with a large number of features but a small sample size. They trim the space of features directly during classification. In other words, regularization effectively shuts down the influence of unnecessary features and influences the learning process. Regularization can be incorporated either into the error criterion or directly into the model. Regularization can be implemented in multiple ways by modifying the loss function (L1, L2, and entropy regularization are the most common) to penalize the model for nonzero weights, to force the capture of a more effective feature representation for input data (sparsity [79,80]). The modification can have the objective of influencing the training approach itself (dropout [81]) by adding a

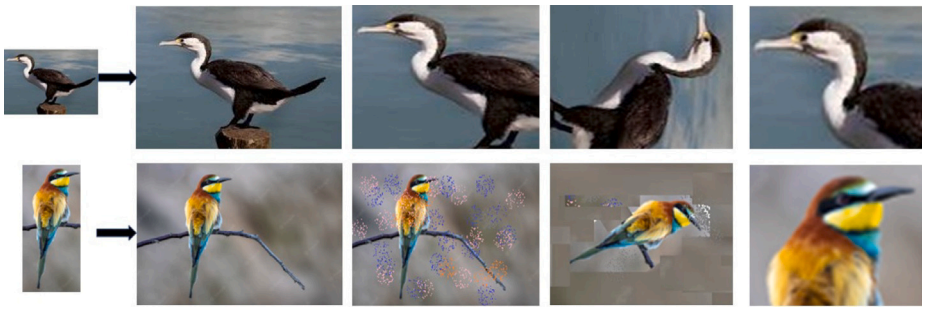


Fig. 4. Concept of data augmentation.

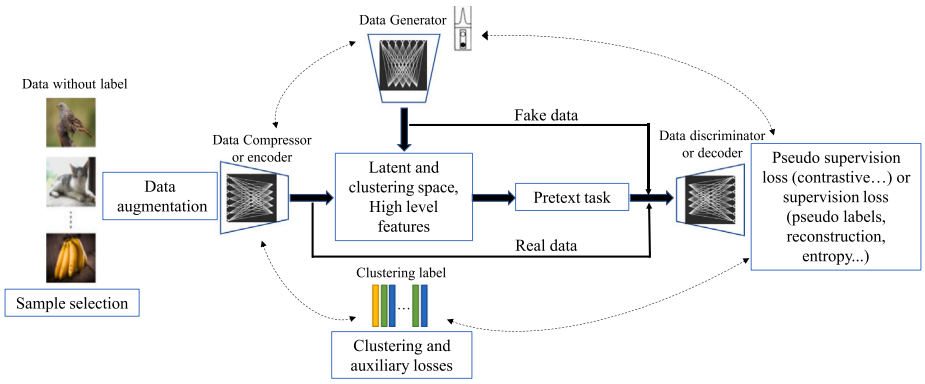


Fig. 5. A synthetic conceptual view of deep clustering pipeline.

penalty to the activations of the neurons aiming at simplifying the training phase. Lastly, auxiliary losses (cluster losses, data augmentation losses) are usually added to drive the learning. The cluster regularization loss is very popular in the deep clustering area. It forces the network to preserve suitable discriminant information from the data in the representations.

2.1.8. Attention mechanism

Attention mechanisms enable the quantification of interdependencies among input elements, allowing the learning system to focus its attention on the elements that are crucial to the training task. Attention mechanisms are often integrated as a component or foundational structure in modern Deep Neural Network (DNN) models, most notably in Transformers [34]. They have garnered significant attention recently, not only for their outstanding performance, sometimes surpassing classical convolutional approaches, but also for their efficiency in terms of computation time. These mechanisms were originally introduced in the context of Natural Language Processing (NLP) [82] and later extended to applications in computer vision [83]. However, their adoption in unsupervised learning has been relatively limited, typically confined to specific scenarios, such as their use in graph clustering [84] or spatial attention within image analysis [85].

2.2. The deep clustering framework

Traditional clustering algorithms give discouraging results on large-scale complex datasets due to the inferior capability of representation learning. The core idea of Deep clustering methods is to exploit the representations learned by neural networks (via CNN, AE, VAE, GNN, GAN) to face the issue of the curse of dimensionality (cf. Fig. 5). Instead of clustering the samples directly in the original input space X, they are transformed with a nonlinear mapping $f_\theta : X \to Z$ where θ are learnable parameters and $Z \in \mathbb{R}^K$ is the learned or embedded feature space. Within the reduced space, parameter optimization can be performed via classical clustering or by iterating between computing an auxiliary target distribution and minimizing clustering loss. To face issues with high-dimensional spaces, Deep clustering successes first depend on the quality of the representations, *i.e.*, the capacity to reduce the original space dimension while capturing the essential data information at a higher level of abstraction in a concise representation. Ideally, each latent factor should represent an underlying dimension of variation between samples that explains variation in multiple dimensions of the measurement space. Obtaining a so-called disentangled representation [86] is not straightforward as it is assumed to decompose the original parameters having non-linear effects in the measurement space. In addition, there is no metric to measure the disentangled quality for real-world problems as the generative factors are unknown. This is the main reason that has led researchers to develop specific learning strategies for deep clustering instead of directly applying advanced traditional clustering approaches limited to low-dimensional spaces. Therefore, the learning strategy including the tricks involved as well as the use of various combined losses are extremely important as they can influence the latent representation, and more generally contribute to the optimization of the clustering process. Deep clustering solutions are based on both discriminative and generative models.

2.2.1. Taxonomy based on architectures

The basic idea of deep clustering is to use a deep neural network to learn an embedding of the original data and carry out clustering on the learned embedding. Among various deep architectures, one simple yet effective neural network is the auto-encoder as by construction it presents a way to contrast input information in a reduced latent space. Therefore, many methods applied an auto-encoder to extract the latent embedding for clustering. Since the convolutional neural network has demonstrated promising performances in many tasks, especially in image processing tasks, CNN has also been used in deep clustering via transfer learning and more recently by generating pseudo-labels for semantic clustering. Since these methods generate high-quality pseudo-labels, they achieve state-of-the-art performance. Another famous unsupervised deep architecture is the generative model like

Generative Adversarial Networks (GAN) or Variational Auto-Encoder (VAE). These generative methods [31,87] were adapted to deep clustering such as InfoGan [88], ClusterGan [89] (GAN family) or VaDE [90], GMM-VAE [91] deep (VAE family). A generative adversarial network (GAN) empirically learns the map that transforms the latent variables into the complex data distribution by playing a min–max game. Different from GAN and VAE, some methods tried to generate pseudo-labels for semantic clustering.

In summary, clustering methods that take advantage of deep learning techniques for jointly learning hidden features of the data are referred to as deep clustering (DC) models. DC algorithms can be classified into four major categories: three are related to the use of specific domain representations, and one is conceptually different as it is generative-oriented.

- Autoencoder-based models including Convolutional Autoencoders (AE/CAE).
- CNN-based models
- Graph-based models
- Generative model-based methods.

The CNN-based models represent a family of methods based on a given architecture often referred to as the 'backbone' (ResNet18, ResNet34...) initially devoted to a discriminative task (direct classification, recognition, segmentation...) and then a straightforward objective. The representation provided by Deep Autoencoders (AE/CAE) is based on the idea of data reconstruction, the one provided by Convolution Neural Networks relies on the high-level features obtained with supervised training in large-scale datasets (transfer knowledge, domain adaptation. self-supervision...) and the one provided by generative approaches relies on the powerful latent representation obtained via the estimation of the posterior distribution of inputs. More recently, via self-supervised learning and contrasting techniques, the representation is learned by guiding the network based on pseudo labels computed from the raw input data via pre-designed tasks (the pretext tasks), which do not require annotated data.

It is noteworthy that graph-based models are currently emerging in the field of deep clustering, whereas RNN-based models are still in the early stages of development. Furthermore, it is important to emphasize that a significant number of these models integrate convolutional layers.

2.2.2. Optimization strategies and losses

Deep Clustering learning strategies can be classified into four broad families according to this taxonomy:

- Sequential multistep Deep Clustering approaches: these approaches have two basic steps. The first stage involves learning a latent representation of the input data, followed by clustering on this deep or latent representation in the second step. The representation vector contains all the important information of the given data point; hence, clustering on the representation vectors yields better results. Any classical clustering algorithms can be applied. The disadvantage in this category is the mismatch problem between data representation and clustering. Specifically, the clustering algorithm does not participate in representation learning, which will lead to the blindness of representation learning. The learned DNNs do not necessarily output reduced-dimension data that are suitable for clustering.
- Closed-loop multistep Deep Clustering approaches: instead of a sequential scheme, this family has two key phases that alternate in an iterative loop rather than being conducted in a single feed-forward linear approach. They jointly train the network to learn better features and use the clustering results to direct the network training. With autoencoders, this category simultaneously optimizes clustering and reconstruction losses for preserving the local structures (DEC, IDEC, JULES, DCCM, DDC, ... seen later). These approaches have difficulties taking into consideration the semantic sample relationships that existed in both local and global features. The local structure preservation cannot be guaranteed by the clustering loss. Thus the feature transformation may be misguided, leading to the corruption of embedded space.
- Joint Deep Clustering approaches: Instead of two independent processes for representation learning and clustering, this family of approaches includes a step where the representation learning is intimately associated with the clustering or other pretext objectives. Tight coupling is usually achieved by optimizing a combined or joint loss function that promotes good reconstruction while accounting for some sort of data grouping, clustering, or codebook representation. The main thrust to advance deep clustering is self-supervised representation learning, including contrastive learning and non-contrastive learning. Via Contrastive learning, the idea is to exploit the discriminative representations, learned from contrastive learning by performing instance-wise discrimination using the InfoNCE loss that aims to pull together the positive pair (x; x+) from two different data augmentations of the same instance and push x away from M negative examples of other instances. The goal is to assist the downstream clustering tasks or simultaneously optimize representation learning and clustering (e.g SCAN, SimCLR, MoCO, IDFD, GCC, WCL, ADC, IIC, IMSAT, PICA, DCDC described later). Contrastive methods often require comparing each example with many other examples in order to work well prompting the question of whether using negative pairs is necessary. The idea of Non-contrastive learning is to involve the alignment term using the representations of one augmented view to predict another (idea of BYOL [92]). They rely on using auxiliary handcrafted prediction tasks (image inpainting, jigsaw puzzle, geometric transformations...) to learn their representation. These methods are being outperformed by contrastive methods.
- The fourth category empirically learns the map that transforms the latent variables to the complex data distribution by playing a min–max game. This category, referred to as VAE and GAN architectures, is different from the others. Generative models have attracted increasing interest from the community, and have been developed mainly in two directions: VAE-based models that learn the data distribution via maximum likelihood estimation (MLE) and GAN-based methods that train a generator via adversarial learning.

Losses can be divided into Non-clustering loss and clustering loss [46]. The general formulation shown in Eq. (1) is as follows:

$$L(\theta) = \gamma L_c(\theta) + (1-\gamma)L_n(\theta) \tag{1}$$

where θ represents hyper parameters of the network, $L_c(\theta)$ and $L_n(\theta)$ respectively the clustering and non-clustering loss. γ is a constant that acts for the trade-off. If γ is set to 0, then the non-cluster loss function is used for training the network. If γ is set to 1, then the clustering prediction is used for training the network. If $0 \leq \gamma \leq 1$, then both the losses are used for training the network. Finding the balance between clustering and non-clustering losses remains tricky.

Concerning the first category (L_c), the different clustering losses are No clustering loss, Cluster classification loss, K-Means loss, Agglomerative clustering loss, Cluster assignment hardening, Balanced assignments loss, Locality-preserving loss, and Group sparsity loss. Concerning the second, this type of loss (e.g. regularization and self-augmentation loss) is specific to the clustering method and the clustering-friendliness of the learned representations. They usually enforce a desired constraint on the learned model, which guarantees that the learned representation preserves important information (e.g. spatial relationships between features). The network loss can be the reconstruction loss of an autoencoder (AE) $L_n = L_{rec}$. For a Convolutional Neural Network (CNN), the representation learning process is exactly

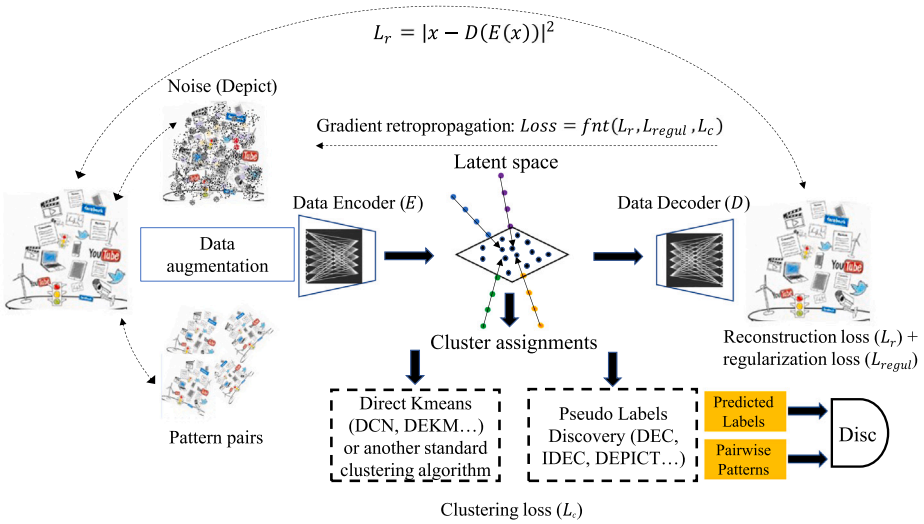


Fig. 6. Autoencoder architecture with various learning schemes.

the same in a supervised manner, but after, for example, pseudo-label generation, then $L_n = L_{cnn}$. For GNNs, the main idea is the preservation of the graph structure, then $L_n = L_{gnn}$ or $L_n = L_{gnn} + L_{rec}$ as AE concepts are commonly hybridized in deep graph clustering. The variational loss of a variational encoder (VAE) $L_n = L_{vae}$ or the adversarial loss of a generative adversarial network (GAN) $L_n = L_{gan}$. For GAN-based or VAE-based deep clustering, the network loss and the clustering loss are usually incorporated together.

3. Detailed overview of the four deep clustering families

In this section, the different families are studied and the most representative approaches are detailed.

3.1. Autoencoder-based models (AE/CAE)

Autoencoders are a kind of unsupervised neural network that aims to reconstruct the input data by learning an efficient representation of it through an encoder and a decoder. The encoder maps the input data to a lower-dimensional representation, known as the bottleneck or latent space, and the decoder reconstructs the original data from this bottleneck. Autoencoders are often used for dimensionality reduction and feature learning.

3.1.1. Autoeconder for clustering

The neural network can be used to extract the latent features of the sample, and then the sample can be grouped into different clusters with the traditional clustering algorithms. The deep clustering algorithm can solve the problems caused by excessive dimensionality through the nonlinear dimensionality reduction of the encoder. Most of the existing deep clustering strategies share two simple concepts. The first concept is that deep embedded representations are favorable to clustering. The second concept is that clustering assignments can be used as a supervisory signal to learn embedded representations. Based on that, the existing deep clustering methods can be classified into two main families.

- Two-stage work that applies clustering after having learned a representation. Most of the earlier AE-based deep clustering approaches used this scheme. To learn more robust features, several autoencoder variants were proposed such as adding sparsity and contractive constraints on the hidden representation as well as the use of denoising autoencoders.
- Approaches that jointly optimize the feature learning and clustering using pseudo-labels with different fine-tuned alternatives. The family of algorithms that jointly perform clustering on the embedded features of an autoencoder are mainly referred to DEC and its variants.

Earlier deep clustering methods are based on autoencoders as they can be used to identify features that are sufficient for reconstructing the data. They have the ability to generalize models better by reducing the dimensions of data through a latent space while maintaining high-quality representation as well. There are several variants of autoencoders that aim at learning the key factors of similarities in the embedded space with respect to the data semanticity. The reconstruction loss function unites them all, and their primary difference from one another is how the encoding operation is limited. To be precise, a regularization term is added in the loss function with the aim of preventing the encoders from overfitting by making the representation as sensitive as possible with respect to changes in input. A sparse autoencoder is simply an autoencoder whose training criterion involves a penalty in the loss function penalizing activations of hidden layers so that only a few nodes are encouraged to activate when a single sample is fed into the network. The variants and advances stem from the strategy of including constraints via regularization, sparsity techniques, and the use of specific losses. The latter aims at encouraging data points to form tightly packed and well-separated clusters in the latent space. Fig. 6 represents the general AE/CAE scheme as well as some of its variants.

As one of the earliest deep clustering algorithms, Deep Embedding Network (DEN) [93] uses an AE to learn clustering-oriented representations from raw data, and then performs clustering with k-means. To achieve the clustering-oriented representations, two constraints (sparsity and locality-preserving constraints) are imposed on the learned representations of the deep embedding network. Deep Clustering Network (DCN) [94] is very representative of the autoencoder family approaches. It learns representations that are amenable to the K-means algorithm. It pre-trains the autoencoder and then jointly optimizes the reconstruction loss and K-means loss with alternating cluster assignments. DCN defines its objective (Eq. (2)) as :

$$\text{Loss} = \min \sum_{i=1}^{n} L_r(D(E(x_i)), x_i) + \lambda \|(E(x_i) - M s_i)\|_2^2 \quad (2)$$

where E and D are encoder and decoder functions respectively with their proper hyperparameters, s_i is the assignment vector of data point i which has only one non-zero element, and M_k, denotes the centroid of

the kth cluster. The function $L_r()$ is a certain loss function that measures the reconstruction error.

DEC was considered for years as the benchmark for comparing the performance of deep-learning clustering approaches. It uses deep neural networks to learn feature representations and cluster assignments continuously. In addition, it optimizes the clustering objectives by mapping the data space to a lower-dimensional feature space. In DEC pre-training, encoder and decoder parameters are initialized for a few epochs with reconstruction loss. After that, the encoder network is removed, and the decoder network is fine-tuned by optimizing KL divergence between soft cluster assignment and auxiliary distribution. Overall, DEC is a self-training clustering process that continuously refines the data representations while performing cluster assignments. Discriminately Boosted Clustering (DBC) [95] is the convolutional autoencoder version of DEC. Researchers have proposed many variants based on the auto-encoder, IDEC [96], DCN [94], N2D [97], COAE [98] and ASPC-DA [99]. For example, COAE [98] is based on the assumption that orthogonality is beneficial to enhance the discriminability and representability of the embedding. An orthogonality regularization term is added to the reconstruction cost function. The idea behind N2D is to first learn an embedding of the data, then learn the manifold of the autoencoded data, and finally transform the data into a form that can be easily clustered. DEPICT [100] and DEKM [101] are presented in more detail as they are representative of the state of the art of this family of methods that inherited the advances of the pioneer algorithms such as DEC, IDEC...

3.1.2. DEPICT: Deep embedded regularized clustering

DEPICT [100] utilizes correlations between an instance and its "self" to construct positive sample pairs. More colloquially, the category of an instance should be the same as its slightly mutated form. DEPICT leverages a convolutional autoencoder for learning embedded representations (Y) and their clustering assignments. It consists of two parts, *i.e.*, a convolutional autoencoder for learning the embedding space ($X \rightarrow Y$), and a multinomial logistic regression layer functioning as a discriminative clustering model.

For jointly learning the embedding space and clustering, DEPICT employs an alternating approach to optimize a unified objective function. Similar to DEC, DEPICT has a relative cross-entropy (KL divergence) objective function (cf. Eq. (3)). In addition, the loss function of DEPICT has a regularization term, which makes it possible to explicitly impose the size of each cluster based on some prior knowledge.

$$\begin{aligned} &\text{Loss} = \text{KL}(Q \parallel P) + \text{KL}(f \parallel u) \\ &\text{Loss} = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{K} p_{ij} \log\left(\frac{p_{ij}}{q_{ij}}\right) + \frac{1}{n}\sum_{j=1}^{K} f_j \log\left(\frac{f_j}{p_j}\right) \end{aligned} \tag{3}$$

where n is the number of samples and K is the number of categories having:

$$\begin{aligned} f_k &= P(Y = k) = \sum_{i=1}^{n} q_{ik} \\ q_{ik} &= P(y_i = k|z_i, \Theta) = \frac{\exp(\theta_k^\top z_i)}{\sum_{k'=1}^{K} \exp(\theta_{k'}^\top z_i)} \\ p_{ik} &= \frac{q_{ij}^2 / \sum_{i=1}^{N} q_{ij}}{\sum_{j=1}^{K}(q_{ij}^2 / \sum_{i=1}^{N} q_{ij})} \end{aligned} \tag{4}$$

where $z_i \in \mathbb{R}^{d_z}$ is the embedding feature that has a much lower dimension compared to the input data $x_i \in \mathbb{R}^{d_x}$, $\Theta = [\theta_1, \ldots, \theta_k] \in \mathbb{R}^{d_z \times K}$ are the soft-max function parameters, f_k is the empirical cluster distribution, *i.e.* the frequency of the clusters, and p_{ik} indicates the probability of the ith sample belonging to the kth cluster and emphasizes the role of those "confident assignments". The clustering loss subsequently compels the current distribution q to align with the target distribution p, adhering to the pioneering concept introduced by DEC.

This avoids situations where most of the data points are assigned to a few clusters, but this prior knowledge assumed by DEPICT is not always well-adapted for pure unsupervised problems.

Moreover, after a pre-training phase, they also changed the AE, to use a denoising version with masking noise on each layer (*i.e.* with dropout layers). They then extend the classical reconstruction loss to be computed as the sum of the reconstruction at each depth of the autoencoder. All the layers of the encoder and decoder contribute to the reconstruction loss instead of just the input and output layers.

The clustering ($\text{KL}(Q \parallel \tilde{P})$) and regularization loss force the model to have invariant features with respect to noise.

$$\text{Loss} = \sum_{i=1}^{n}\sum_{k=1}^{K} q_{ik} \log \tilde{p}_{ik} + \frac{1}{n}\sum_{i=1}^{n}\sum_{l=0}^{l-1} \frac{1}{|z_i^l|} \| z_i^l - \hat{z}_i^l \|_2^2 \tag{5}$$

z^l is the clean features of the lth layer (the input for $l = 0$) and $\hat{z}^l$ is the lth reconstruction layer output.

3.1.3. DEKM: Deep Embedded K-means clustering

Like DEC and IDEC, DEKM [101] uses an autoencoder to transform the original space into an embedding space to reduce the dimensionality before clustering. Then, it discards the decoder and optimizes the representation for better clustering. The representation optimization of DEKM does not optimize the Kullback–Leibler (KL) divergence between the cluster distribution and an auxiliary target distribution. Instead, DEKM optimizes the representation by reducing its entropy.

DEKM alternately optimizes representation learning and clustering. To better reveal the cluster structure, the embedding space is transformed into a new space via an orthonormal transformation matrix which contains the eigenvectors of the within-class scatter matrix of K-means.

To summarize, DEKM has three steps:

- Generating an embedding space by an autoencoder using the reconstruction loss L_r

$$L_r = \sum_{k=1}^{K} \| x_i - g(f(x_i)) \|^2 \tag{6}$$

  where x_i is the ith data point, $f(x_i)$ is the output of the encoder $f(\cdot)$, and $g(x_i)$ is the reconstructed output of the decoder $g(\cdot)$.
- Detecting clusters in the embedding space by K-means. Its objective function L_c is as follows:

$$L_c = \sum_{i=1}^{k}\sum_{h \in C_i} \| h - \mu_i \|^2 \tag{7}$$

  where h is a data point in the embedding space, k is the number of clusters, C_i denotes the set of data points assigned to the ith cluster, and $\mu_i = \frac{1}{|C_i|}\sum_{h \in C_i} h$ denotes the centroid of the ith cluster. To reveal the cluster structures in the embedding space, the embedding space H is transformed into a new space by an orthonormal transformation matrix V ($Y = VH$).

$$\begin{aligned} L_c &= \sum_{i=1}^{k}\sum_{h \in C_i} \| Vh - V\mu_i \|^2 \\ L_c &= \sum_{i=1}^{k}\sum_{h \in C_i} (h - \mu_i)^\top V^\top V (h - \mu_i) \end{aligned} \tag{8}$$

  The above equation can be further written as:

$$\begin{aligned} &L_c = \text{Trace}(V S_w V^\top) \quad \text{with} \\ &S_w = \sum_{i=1}^{k}\sum_{h \in C_i} (h - \mu_i)^\top (h - \mu_i) \end{aligned} \tag{9}$$

  where S_w is the within-class scatter matrix of K-means.

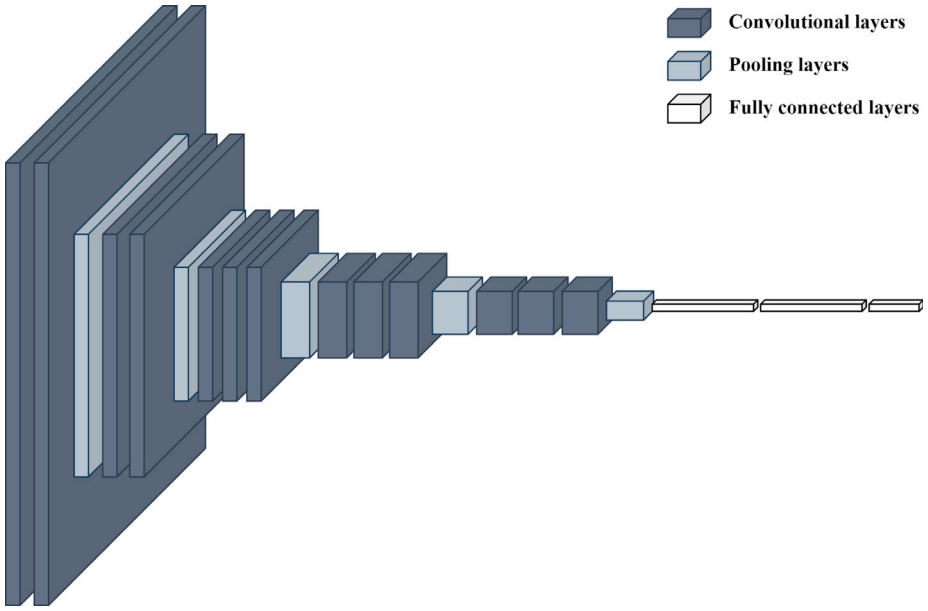


Fig. 7. The core architecture from inputs to high-level features (Backbone).

- Optimizing the representation to increase the cluster-structure information. K-means is first performed in the embedding space H to get S_w, and then eigen decomposes S_w to get V. Finally, the embedding space is transformed into a new space Y that reveals the cluster-structure information.
  Eq. (8) can be rewritten as follows:

$$L_c = \sum_{i=1}^{k} \sum_{h \in C_i} \|y - m_i\|^2 \tag{10}$$

  where $y = Vh$ and $m_i = V\mu_i$, V contains the eigenvectors of S_w sorted in ascending order w.r.t. their eigenvalues. The goal is to move the data points in clusters found by K-means so that they are closer to their centroids in consistent with Eq. (7), which is further minimized. Instead of moving data points closer to their respective centroids in each dimension of Y, authors in DEPICT endorse a more selective, greedy strategy. This involves initially replicating y as y' and subsequently replacing only the last dimension of y' with the corresponding dimension from m_i. The representation in Y is then optimized as follows:

$$L_c = \sum_{i=1}^{k} \sum_{y \in C_i} \|y - y'\|^2 \tag{11}$$

The last two steps are alternately repeated to generate better embedding space and clustering results.

3.1.4. Summary analysis

This category probably represented the most prominent deep clustering approach until 2020. Despite their success, existing methods generate the target distribution with only the information of the autoencoder. As the latter may not represent the semantics of the original data, the knowledge discovery task cannot be optimal.

3.2. CNN-based models

These models [13] differ from their 'backbone' (cf. Fig. 7) and have a common objective of discrimination, either supervised classification, recognition or segmentation. Pre-trained convolutional neural networks, or ConvNets, have become the building blocks in most computer vision applications. They produce excellent general-purpose features that can be used to improve the generalization of models learned on a limited amount of data and were adapted for clustering purposes.

3.2.1. CNN-based models for clustering

A group of Deep clustering methods such as Deep adaptive image clustering (DAC) [102], DeepCluster [103], or others exploit the representations learned by neural networks and have made significant progress on high-dimensional data. They leverage the architecture of CNNs as prior to cluster images. Starting from the initial feature representations, the clusters are iteratively refined by deriving the supervisory signal from the most confident samples, or through cluster reassignments calculated offline. Methods that rely on the initial feature representations of the network are generally sensitive to initialization, or prone to degenerate solutions, thus requiring special mechanisms (e.g. pretraining, cluster reassignment, and feature cleaning) to avoid those situations.

Other methods such as Invariant Information Clustering (IIC) [104], Information Maximizing Self Augmented Training (IMSAT) [105] learn the representations for clustering by constructing a data correlation regulation, such as positive sample pairs, self-supervised guidance, and mutual information. Straightforward approaches estimate data correlation with the representations obtained from pre-trained models and then learn a classical clustering model. However, because of the independence of the two stages, the pre-learned representations may not fully explore the semantic structures of the data, resulting in a sub-optimal clustering solution. Some approaches construct positive data-pair information with data-augmentation methods, but such approaches tend to be over-fitting and have difficulty learning the discriminative information due to the lack of negative sample pairs. Some methods learn clustering models by self-supervised learning, which uses previous clustering predictions to guide later model learning. But, the performance of these methods depends strongly on the performance of the initial model. In general, poor performance of the initial model leads to a poor clustering assignment.

Before 2020, instance discrimination methods based on CNN architectures and self-supervision learning approaches shared a common weakness: the representation is not encouraged to encode the semantic structure of data. This problem arises because instance-wise contrastive learning treats two samples as a negative pair as long as they are from different instances, regardless of their semantic similarity. This is magnified by the fact that thousands of negative samples are generated to form the contrastive loss, leading to many negative pairs that share similar semantics but are undesirably pushed apart in the embedding space.

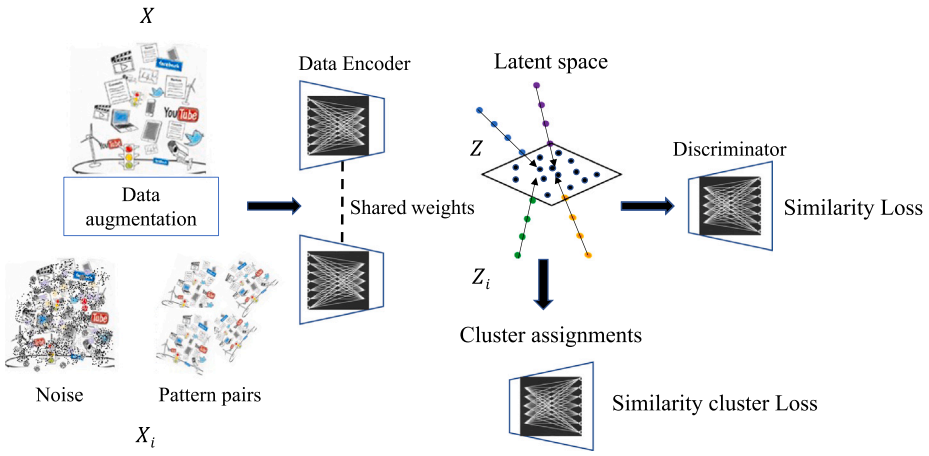


Fig. 8. Novel generation of deep learning algorithms based on CNN.

Very recent proposals (cf. Fig. 8) have considered this issue and achieved very promising results. The contrastive learning paradigm shows its power in unsupervised representation learning [65,106]. It first constructs positive and negative pairs for each instance and then projects them into a subspace to maximize the similarities of positive pairs and minimize those of the negative ones. SimCLR [14] constructs positive and negative pairs through augmentations within mini-batches. MoCo [107] recasts contrastive learning as a dictionary lookup task by building a dynamic dictionary with a queue and a moving-averaged encoder. To avoid the efforts in building negative pairs, BYOL [92] and EDCN [108] replace negative pairs with an online predictor that prevents the network from collapsing into trivial solutions. Others such as AdCo [109] directly learn negative samples in an adversarial manner. Barlow Twins [106] performs contrastive learning from a redundancy-reduction perspective. PICA [110] learns the most semantically plausible data separation by maximizing the partition confidence of the clustering solution. Semantic Clustering by Adaptive Nearest neighbors (SCAN) [70] conducts a pretext task of contrastive learning to mine the nearest neighbors via the search for similar samples across the whole dataset, encouraging the model to output the same labels for similar instances. The Nearest Neighbor Matching (NNM) [111] method extends SCAN by matching samples with their nearest neighbors from both local and global levels. Similarly, Nearest Neighbor Contrastive Clustering (NNCC) [112] fuses contrastive learning with neighbor relation mining to obtain more semantically meaningful representations. The most recent deep clustering methods, all based on the contrastive learning paradigm with variants, have achieved state-of-the-art performance, e.g., SPICE [113], ProPos [114], (TCL) [115], EDCN [108]

The state-of-the-art methods will be synthesized in the following section, with a more detailed presentation of four selected methods here: IMSAT [105], the SimCLR framework [14], PICA [110], and ProPos [114]. These methods are highlighted for their influence and representation of the current state-of-the-art.

3.2.2. IMSAT

In IMSAT [105], data augmentation is used to model the invariance of learned data representations. IMSAT maximizes the mutual information between the input and output of the model. More specifically, data points are mapped into their discrete representations by a deep neural network and regularized it by encouraging its prediction to be invariant to data augmentation. The predicted discrete representations then exhibit the invariance specified by the augmentation. The best augmentation of the prediction can be calculated with Self-Augmented Training (SAT) which uses data augmentation to impose the intended invariance on the data representations. SAT penalizes representation dissimilarity between the original data points and augmented ones. The idea is to consider random deviation to be added to the input from a predefined noise distribution. Essentially, SAT penalizes representation dissimilarity between the original data points and augmented ones using Random Perturbation Training (RPT) [116] or Virtual Adversarial Training (VAT) [117]. With RPT, a random perturbation r from a predefined noise distribution is added to the input randomly, which is a naive way to do augmentation. With (VAT), the deviation is assigned so that the model cannot assign it from the same cluster. Regularization using local perturbation is based on the idea that it is preferable for data representations to be locally invariant (i.e., remain unchanged under local perturbations on data points).

Let X and Y denote the domains of inputs and discrete representations, respectively. Given training samples, $\{x_1, x_2, \ldots, x_n\}$, the task of discrete representation learning is to obtain a function, $f : X \rightarrow Y$, that maps similar inputs into similar discrete representations. Given a data point x, an augmented training example $T(x)$ is generated where $T : X \rightarrow X$ denotes a pre-defined data augmentation function. The goal is that the cross-entropy between $p(y|x)$ and $p(y|T(x))$ is minimized.

Let $x \in X$ and $Y \in \{0, \ldots, K-1\}$ (K is the number of clusters) denote random variables for data and cluster assignments, respectively.

$$\text{Loss} = \Big(\sum_{i=1}^{n} R_{\text{sat}}\big(\theta, x_i, T(x_i)\big)\Big) - \lambda\big(H(Y) - H(Y|X)\big) \tag{12}$$

where R_{sat} is the SAT regularization, $H(\cdot)$ and $H(\cdot|\cdot)$ are entropy and conditional entropy, respectively and λ a tradeoff parameter. Increasing the marginal entropy $H(Y)$ encourages the cluster sizes to be uniform. Decreasing the condition entropy $H(Y|X)$ encourages unambiguous cluster assignments.

3.2.3. SimCLR

SimCLR framework [14] is based on the main idea of self-supervision and data augmentation. An image is taken and random transformations are applied to it to get a pair of two augmented images x_i and x_j. Each image in that pair is passed through an encoder such as Resnet-50 or others with the same hyperparameters to get representations. The representations h_i and h_j of the two augmented images are then passed through a series of fully connected layers to apply non-linear transformation and project it into a representation z_i and z_j (cf. Fig. 9). The task is to maximize the similarity between these two representations z_i and z_j for the same image based on the cosine similarity (cf. Eq. (13)).

$$cos_{sim}(z_i, z_j) = \langle z_i \cdot z_j \rangle = \frac{z_i^{\top} z_j}{\|z_i\| \|z_j\|} \tag{13}$$

SimCLR uses a contrastive loss called NT-Xent loss (Normalized Temperature-Scaled Cross-Entropy Loss) as seen in Eq. (14).

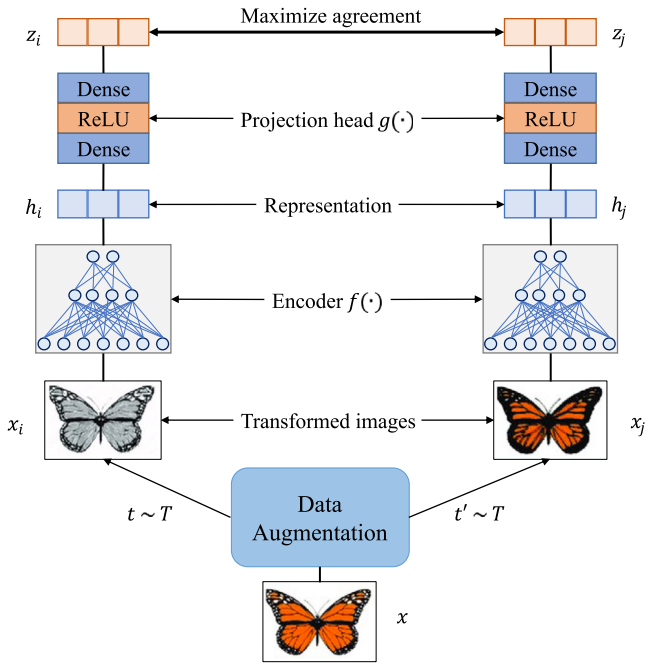


Fig. 9. SimCLR flow formalization. SIMCLR maximizes the similarity between two representations z_i and z_j for the same image.

A minibatch of N examples is sampled, and the contrastive prediction task on pairs of augmented examples is derived from the minibatch, resulting in $2N$ data points. Negative examples are not sampled explicitly. Instead, given a positive pair, the other $2(N-1)$ augmented examples within a minibatch are treated as negative examples. The contrastive loss encourages the positive pairs to have high similarity scores while pushing the negative pairs to have low similarity scores. To stabilize the learning process and improve performance, SimCLR applies L_2 normalization to the projected representations and introduces a temperature parameter in the contrastive loss. The temperature parameter scales the similarity scores, effectively controlling the contrastive learning objective's sensitivity. Then the loss function for a positive pair of examples (i, j) is defined as:

$$L_{ij} = -\log \frac{\exp\langle z_i \cdot z_j \rangle / \tau}{\sum_{k=1}^{2N} \mathbb{1}_{k \neq i} (\exp\langle z_i \cdot z_k \rangle / \tau)} \tag{14}$$

where $\langle \cdot \rangle$ stands for the dot product or cosine similarity (see Eq. (13)), τ is the temperature parameter, and $\mathbb{1}_{k \neq i}$ is an indicator function evaluating to 1 if $k \neq i$.

The resulting loss for the batch is:

$$\text{Loss} = \frac{1}{2n} \sum_{k=1}^{2n} L(2k-1, 2k) + L(2k, 2k-1) \tag{15}$$

Both positive and negative pairs are made from the same mini-batch. If the mini-batch size is n, n pairs of positive pairs are generated by augmentation. As a known weakness, SimCLR needs large batches to guarantee a sufficient number of negative pairs. Multiple positive pairs can be present in the same batch while considered negative. SimCLR requires big hardware infrastructures and is still considered state-of-the-art for self-supervised learning, outperforming competitive methods on ImageNet.

3.2.4. PICA: PartItion confidence maximisation

PICA [110] is based on a partition uncertainty index (PUI) that measures how a deep CNN is capable of interpreting and partitioning the target image data. The core concept of the algorithm revolves around achieving high visual similarity among instances belonging to the same semantic classes. However, a notable challenge arises when apparent similarity significantly impacts the distribution of features. This challenge becomes particularly evident when positive sample pairs are assigned to different clusters, resulting in decreased intra-cluster compactness and reduced inter-cluster diversity. As a consequence, this leads to lower partition confidence. Ultimately, the level of partition confidence plays a critical role in establishing semantic plausibility.

One of the key strengths of PICA lies in its focus on encouraging the model to learn the clusters that exhibit the highest levels of confidence across various potential solutions. This approach aims to identify a partition that reflects a coherent separation between different classes in a way that aligns closely with the true underlying semantics. The ultimate goal is to achieve a partitioning where each cluster corresponds directly to a specific ground truth class.

To achieve this, PICA employs several core techniques. It introduces a partition uncertainty index that captures the uncertainty associated with the assignment of data points to clusters. This index is made differentiable and approximated stochastically, enabling its integration into the training process. Additionally, PICA employs a meticulously designed objective loss function that minimizes the aforementioned uncertainty index. The combination of these components allows PICA to seamlessly integrate with traditional deep neural networks and supports efficient mini-batch-based model training.

Let be $P = [p_1, \ldots, p_n] \in \mathbb{R}^{K \times n}$ the prediction matrix of all the n input image patterns. Each element $p_{i,j}$ specifies the predicted probability of the ith image assigned to the jth cluster among K. $q_j = [p_{1,j}, \ldots, p_{n,j}] \in \mathbb{R}^{1 \times n}$ is the jth row of P that collects the probability values of all the images for the jth cluster, which summarizes the ASV (Assignment Statistics of that cluster) over the whole target data. (PUI) is formulated as the ASV cosine similarity set of all the cluster pairs through the matrix $M_{PUI} \in \mathbb{R}^{K \times K}$ matrix, where K is the number of clusters:

$$M_{PUI}(j_1, j_2) = \cos(q_{j1}, q_{j2}) = \frac{(q_{j_1} \times q_{j_2})}{(\|q_{j_1}\|_2 \times \|q_{j_2}\|_2)} \tag{16}$$

where $j_1, j_2 \in [1, \ldots, K]$

The learning objective of PICA is then to minimize the PUI (except the diagonal elements) which is supposed to provide the most confident clustering solution at its minimum. A stochastic approximation of PUI is considered to avoid using the entire target data set. Instead of using all the images, at each training iteration, a random subset of them is used. PICA is a global clustering solution measurement, different from most existing deep clustering methods that leverage some local constraints on individual samples or sample pairs without global solution-wise learning guidance.

The overall objective function of PICA is formulated as:

$$L = L_{ce} + \lambda L_{ne} \tag{17}$$

where λ is a weight parameter, L_{ce} is the common cross-entropy loss function and L_{ne} is the negative entropy of the cluster size distribution.

$$\begin{aligned} L_{ce} &= \frac{1}{K} \sum_{j=1}^{K} -\log(m_{j,j}) \\ m_{j,j'} &= \frac{\exp(M_{PUI}(j, j'))}{\sum_{k=1}^{K} \exp(M_{PUI}(j, k))}, \quad j' \in [1 \ldots K] \end{aligned} \tag{18}$$

$$\begin{aligned} L_{ne} &= \log(K) - H(Z), \quad Z = [z_1, \ldots, z_K] \\ z_j &= \frac{\sum_{j=1}^{K} q_j^t}{\sum_{k=1}^{K} \sum_{j=1}^{K} q_k^t} \end{aligned} \tag{19}$$

where $H(\cdot)$ is the entropy of a distribution, and Z is L_1 normalized soft cluster size distribution with each element computed as z_j. Using $\log(K)$ is to ensure non-negative loss values. PICA can be introduced in standard deep network models and end-to-end trainable without bells and whistles. The advantages of PICA over a wide range of state-of-the-art deep clustering approaches have been demonstrated by extensive experiments on challenging object recognition benchmarks.

3.2.5. ProPos

Prototype Scattering and Positive Sampling (ProPos) [114] is an end-to-end deep clustering method with prototype scattering and positive sampling implementable (as a backbone) in a CNN architecture such as the ResNet family. It is mainly based on two novel ideas. First, considering that different prototypes/clusters are truly negative pairs, ProPos performs contrastive learning over prototypical representations, in which two augmented views of the same prototypes are positive pairs and different prototypes are negative pairs. The idea is to maximize the between-cluster distance so as to learn uniform representations towards well-separated clusters. It is formalized by a Prototype Scattering Loss (L_{psl}).

Specifically, assume that K prototypes have been obtained from one view in the embedding space, $\{\mu_1, \mu_2, \dots, \mu_K\}$, and another K prototypes from another view, $\{\mu'_1, \mu'_2, \dots, \mu'_K\}$, L_{psl} is defined as follows:

$$\begin{aligned} L_{psl} &= \frac{1}{K}\sum_{k=1}^{K} -\log \frac{\frac{\exp\,(\mu_k^\top \mu'_k)}{\tau}}{\frac{exp\,(\mu_k^\top \mu'_k)}{\tau} + \sum_{j=1,j\neq k}^{K} \frac{\exp\,(\mu_k^\top \mu_j)}{\tau}} \\ L_{psl} &\approx \frac{1}{K}\sum_{k=1}^{K} -\frac{\mu_k^\top \mu_j}{\tau} + \frac{1}{K}\sum_{k=1}^{K} \log \sum_{j=1,j\neq k}^{K} \frac{\exp\,(\mu_k^\top \mu_j)}{\tau} \end{aligned} \tag{20}$$

The first term is related to the prototypical alignment and the second is the prototypical uniformity. The cluster centers μ_k and μ'_k are computed within a mini-batch B as follows:

$$\begin{aligned} \mu_k &= \frac{\sum_{x\in B} p(k|x) f(x)}{\|\sum_{x\in B} p(k|x) f(x)\|_2} \\ \mu'_k &= \frac{\sum_{x\in B} p(k|x) f'(x)}{\|\sum_{x\in B} p(k|x) f'(x)\|_2} \end{aligned} \tag{21}$$

where $p(k|x)$ is the cluster assignment posterior probability.

Second, to improve the within-cluster compactness, one augmented view of the instance is aligned with the randomly sampled neighbors of another view that are assumed to be truly positive pairs in the embedding space. This refers to Positive Sampling Alignment (PSA) which takes into account the neighboring samples in the embedding space, improving the within-cluster compactness.

The key step of PSA is to sample the neighboring examples v. A natural way is modeling the representation of one augmented view of an instance as a continuous distribution in the embedding space. Thanks to its simplicity, a Gaussian distribution is introduced, which can be formulated as follows:

$$v \sim N(f(x), \sigma^2 I) \tag{22}$$

where 'I' is the identity matrix and σ is a positive hyperparameter controlling how many samples around one view can be treated as positive pairs with another view. Based on the reparameterization trick v is reformulated as follows allowing backpropagation:

$$v = f(x) + \sigma\epsilon \quad \text{with} \quad \epsilon \sim N(0, I) \tag{23}$$

Then, L_{psa} can be formalized as follows:

$$\begin{aligned} L_{\text{psa}} &= \|g(v) - f'(x^+)\|_2^2 \\ L_{\text{psa}} &= \|g(f(x) + \sigma\epsilon) - f'(x^+)\|_2^2 \end{aligned} \tag{24}$$

where $g(\cdot)$, $f(\cdot)$, and $f'(\cdot)$ are the predictor, online, and target networks, respectively; $g(f(x))$ and $f'(x^+)$ are l_2-normalized.

ProPos is optimized in an expectation–maximization (EM) framework, in which two steps are iteratively performed: E-step for estimating the instance pseudo-labels via spherical K-means and M-step for minimizing a cost L that combines two costs to obtain well-separated clusters and within-cluster compactness as shown in Eq. (25):

$$L = L_{psa} + \lambda_{psl} L_{psl} \tag{25}$$

where λ_{psl} controls the balance between two loss components.

There are only two hyper-parameters in the loss function, one in L_{psa} and the loss weight λ_{psl}.

3.2.6. Synthesis analysis

In synthesis, state-of-the-art models are based on a self-supervised learning paradigm where ground-truth labels are obtained automatically based on the natural groupings of the dataset. One approach is to use clustering results as pseudo labels to guide the pair construction. The other, which is more direct and commonly used, is to treat each instance as a class represented by a feature vector, and data pairs are constructed through data augmentations. Despite highly promising empirical results in deep clustering particularly with image datasets, its effectiveness remains poorly understood from a theoretical perspective [118]. There are many design choices present in the formulation which can affect the quality of the representations learned such as the number of negative examples per sample k, the architecture selection, the distribution of positive pairs (x, x+), and the hyper-parameters of the optimization algorithm among others.

3.3. Graph neural network models

Graphs permeate our daily lives, with real-world entities frequently characterized by their interconnections. Graphs are an effective way to represent networks of entities and their relationships due to their widespread prevalence in our world. Many data types showing non-Euclidean underlying anatomy naturally conform to graph structures, including social networks, citation networks, molecular interactions, and physical and biological systems. Furthermore, even data formats traditionally unrelated to graphs such as images and textual information, can be transformed into a graph-based framework for more comprehensive analysis.

3.3.1. Graph definition

A graph G is defined as a pair (N, E), where N is a set of n nodes or vertices ($n = |N|$), and E is a set of m edges or arcs connecting these nodes ($m = |E|$): $E \subseteq \{(u, v) \mid u, v \in N\}$. Centrality measures the importance of a node in a graph, while density measures the ratio of the actual number of edges and the maximum possible number of one.

G can be direct or indirect. In the case of a directed graph, each edge (u, v) has a direction, specifying a head and a tail. The indegree of a node is the number of edges leading into that node and its outdegree, the number of edges leading away from it. In contrast, undirected graphs do not have a defined ordering between u and v, meaning (u, v) is equivalent to (v, u). The degree of a node in an undirected graph is the number of edges incident on it. Degree matrix D is a diagonal matrix defining the number of connections of nodes. The adjacency matrix A is a $n \times n$ matrix with $A_{ij} = 1$ if $e_{ij} \in E$ and $A_{ij} = 0$ if $e_{ij} \notin E$. If connections are weighted, the weighted matrix W is considered. Laplacian matrix L is $L = D - A$ or $L = D - W$ in a weighted matrix. Directly applying adjacency graphs to large-scale data is impractical due to high computational and memory costs. The core idea is to choose a group of data points (called Anchors) to approximate the original adjacency graph structure [119,120]. A graph is considered acyclic if there is no path, which is a sequence of interconnected edges, that starts and ends at the same node; otherwise, it is referred to as cyclic. Attributes or feature vectors (with dimensions d and c, respectively) can be used to label nodes and edges within the graph. X^l, where $X^l \in \mathbb{R}^{n\times d}$ is a node feature matrix with $x_v \in \mathbb{R}^d$ representing the feature vector of a node v. X^e, where $X^e \in \mathbb{R}^{m\times c}$, it is an edge feature matrix with $x^e_{v,u} \in \mathbb{R}^c$ representing the feature vector of an edge (v, u).

A graph can also be dynamic or temporal [121] meaning that it changes (nodes, edges, and attributes) over time. The dynamic graph is formally defined as $G_t = (N_t, E_t)$ and $G_{t+1} = (N_{t+1}, E_{t+1})$ with $G_t \neq G_{t+1}$, where G_t represents the graph at t, and G_{t+1} represents the graph at $t + 1$. Temporal graphs [122] continue to be an emerging field. The exploration of deep clustering for temporal graphs has not been fully developed yet [123–125].

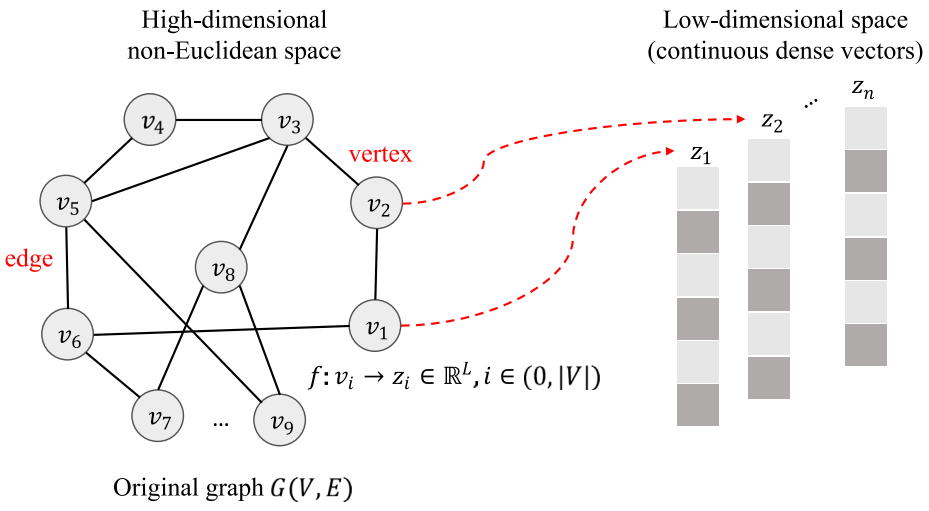


Fig. 10. Node embedding.

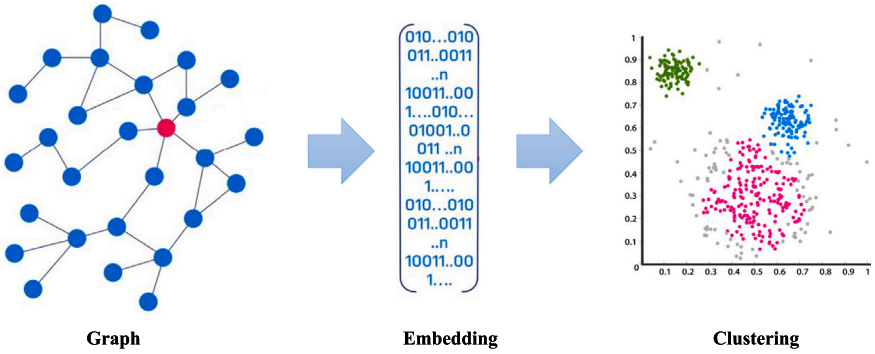


Fig. 11. Classic two-step scheme of clustering with graphs.

3.3.2. Graph for clustering

Graphs serve as versatile tools employed for a diverse array of tasks, spanning graph-level (holistic graph analysis), node-level (node-level taxonomy), and edge-level (specific edge assignments) strategies. Graph clustering, which includes community detection and group segmentation, refers to clustering data in graph form [126–128]. The goal is to seek to partition nodes (node clustering task) according to their similarity within a graph into distinct groups through an unsupervised approach. Given a set of communities $C = \{C_1, C_2, \ldots, C_k\}$ each community C_k is a partition of G. Communities are disjoint if $\forall(k, k')$ $C_k \bigcap C_{k'} = \varnothing$. The similarity usually considers either topological features (*e.g.*, features extracted from the graph), or other characteristics related to the nodes and edges of the graph (*e.g.*, additional information that may be associated with the nodes and edges) or both of them. Most traditional methods [129] of community detection are based on statistical inference and conventional machine learning. However, with complex and large data sets, relying solely on straightforward connection-based information can often result in suboptimal community detection outcomes, as noted in recent studies [130].

A highly related task of graph clustering is network embedding [24], a fundamental encoding task that aims to learn latent representations (namely embedding) for nodes of a graph while preserving relevant network properties (Fig. 10).

It means that the similarity S for the nodes (u, v) in the original space (G) and in the latent space (E) are comparable ($S_G(u, v) \approx S_E(u, v)$). The main idea is to minimize the loss function $L = \sum_{(u,v)\in V} \|S_G(u, v) - S_E(z_u, z_v)\|_2^2$. Early graph clustering methods only utilize graph structure to cluster nodes. These methods are often based on Markov random walk techniques such as Deep-Walk [20], which generate node sequences with truncated random walks, and then obtain node embeddings with the Skip-gram model. They decode statistics of random walks. Matrix factorization techniques are also employed aiming to preserve various structural properties within these embedding vectors. These methods can be computationally expensive, there is no parameter sharing and a serious issue is that there is no semantic information considered. This means that integrating features on the nodes will be difficult in the encoder defined. To better analyze the graph data, attributed graph clustering methods [131] have been developed which can describe and represent many real-world applications such as social networks, and citation networks. They utilize two sides of information in attributed graphs to cluster nodes. Recent studies have resorted to deep learning techniques to learn compact representation to exploit the rich information of both the content and structure data [132]. Based on the learned graph embedding, simple clustering algorithms such as K-means are applied.

Most of these embedding-based methods are two-step approaches (Fig. 11). The drawback is that the learned embedding may not be the best fit for the subsequent graph clustering task, and the graph clustering task is not beneficial to graph embedding learning. The flaws of these methods have contributed to the motivation for introducing something like convolution therefore a neural network in a graph representation which is known as GNNs (Graph Neural Networks).

3.3.3. GNNs

The initial work of deep clustering on graphs and their models are relatively simple usually following the two-step paradigm, *i.e.*, constructing a 'good' graph structure and achieving the message passing for signals supported on the learned graph. They however do not take into account the multiple views and different degrees of influence among

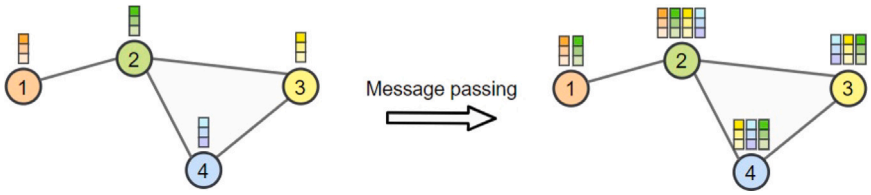


Fig. 12. Diffusion mechanism via message passing.

node neighbors, so the various information implied in the feature representation they learn is relatively insufficient and the learned graph structure may be unreliable. GNNs [132,133] (*cf.* Fig. 12) are grounded in an information diffusion mechanism. Their distinct advantage in community detection, compared to other machine learning methods that rely on either an adjacency matrix or a node attribute matrix, is their ability to encode feature representations of high-dimensional data using deep learning techniques. Traditional deep learning methods are primarily designed for handling straightforward sequences and grids, functioning within the confines of Euclidean spaces. In CNNs, a sliding $n \times n$ pixel window is applied, and various transformations are performed to represent this information as a new cell. These transformations fall short for complex structures like graphs: the number of neighbors changes, the distance between nodes changes, the number of features can vary, the nodes can have different meanings and attributes (heterogeneous graph) and node ordering can change.

Graph Neural Networks (GNNs) introduce a breakthrough concept: they generate node embeddings by leveraging information from local neighborhoods, the idea being to represent each entity in vector spaces. This key idea draws inspiration from the successful strategy of Convolutional Neural Networks (CNNs) for gathering neighborhood information. After embedding, the vector representation can be applied to further tasks like link prediction, clustering, and recommendation. The different types of GNNs are then mostly a variation of Convolutional Neural Networks extending deep learning approaches for graph data. Deep learning-based community detection has been a new emerging branch. A graph is processed by a set of units, each one corresponding to a node of the graph, which are linked according to the graph connectivity. The units update their states and exchange information until they reach a stable equilibrium. The output of a GNN is then computed locally at each node on the base of the unit state. The diffusion mechanism is constrained in order to ensure that a unique stable equilibrium always exists. Broadly, there are two approaches for building graph convolutional neural networks, Spectral GCNs and Spatial GCNs [132]. Spectral GCNs are based on principles of spectral graph theory. More specifically, the graph processing is based on the Eigen-decomposition of graph Laplacian (resulting in potentially intense computations), which is used to compute the Fourier transform of a graph signal, through which graph filtering operations are defined. Spatial GCNs define convolutions directly on graph data and try to capture information by aggregating information from neighboring nodes through shared weights. These approaches require the definition of an operator that works with different-sized neighborhoods and the challenge is to maintain the weight-sharing property of CNNs.

The process of sharing information in GNN is known as "Message Passing" which has become the most important component in GNN. Each layer in a network will be considered as a single round of message passing. And, with each round of message passing, more and more information gets passed around in the graph. The first step is that each node creates a feature vector x_v that represents the message it wants to send to all its neighbors. In the second step, the messages are sent to the neighbors, so that a node receives one message per adjacent node. The message-passing phase runs for T time steps (Fig. 12) in several steps:

- At $t = 0$ initialize $h_v = x_v$
- For each node v in the graph represented by x_v, gather all the neighboring node embeddings (or messages) and aggregate all messages via an aggregate function $f_{\text{aggregate}}$ such as a mean aggregation, a max, etc.

$$m_v^{t+1} = f_{\text{aggregate}}(h_w^t | w \in N(v)) \text{ (general)} \tag{26}$$

  where h_v^t is the hidden state at each node in the graph, and N_v is the neighborhood of v.
- All pooled messages are passed through an update function f_{update}, usually a learned neural network.

$$h_v^{t+1} = f_{\text{update}}(m_v^{t+1}, h_w^t) \tag{27}$$

  Changing the permutation of the nodes does not change the node's neighbors the information is aggregated from a node's neighborhood regardless of ordering.

By stacking messages passing GNN layers together (k times), a node can eventually incorporate information from across the entire graph. After k layers, a node has information about the nodes that are k steps away from it. The simplest rule of GCN's layer-wise propagation can be formalized in a matrix way as follows:

$$H^{(t+1)} = \sigma\left(D^{-\frac{1}{2}}\ \overline{A}\ D^{\frac{1}{2}} H^{(t)} W^{(t)}\right) \tag{28}$$

where $H^0 = X$, $W^{(l)}$ is the weight matrix of layer t, $\overline{A} = A + I_n$ identity matrix, $D_{ij} = \sum_j a_{ij}$ serves as normalization and $\sigma(\cdot)$ is the activation function.

Attention mechanisms are popular in many sequence-based tasks. They allow for dealing with variable-sized inputs, focusing on the most relevant parts of the input to make decisions. They have recently been introduced in the graph area [134] with the idea of computing the hidden representations of each node in the graph. This is achieved by attending to its neighbors, following a self-attention strategy, and then assigning different importances to nodes within the same neighborhood. The self-attention outputs undergo LeakyReLU non-linearity (L_{ReLU}) and softmax operations to produce attention scores.

$$v_i^l = \sigma\left(\sum_{j \in N(i)} \alpha_{ij}\ v_j^{l-1}\ W^{l-1}\right) \tag{29}$$

where

$$\begin{aligned} \alpha_{ij} &= f(v_i, v_j, W) = \frac{\exp(\rho_{ij})}{\sum_{k \in N(i)} \exp(\rho_{ik})} \\ \rho_{ij} &= L_{\text{ReLU}}\left(\vec{a}^{\top}[(v_i^{l-1}\ W^{l-1}) \parallel (v_j^{l-1}\ W^{l-1})]\right) \end{aligned} \tag{30}$$

α_{ij} is a soft max function. A weighted average is used instead of a simple average (Fig. 13) and || stands for the concatenator operator. In Graph attention networks (GAT), W and a are the learning parameters. Generalizing to h layers gives:

$$v_i^l = \Big\|_h^H\ \sigma\left(\sum_{j \in N(i)} \alpha_{ij}^h\ v_j^{l-1}\ W^{h,(l-1)}\right) \tag{31}$$

The main difference between GCNs and GATs is that in GCN the aggregation is performed via the adjacency matrix and in GAT via α_{ij}.

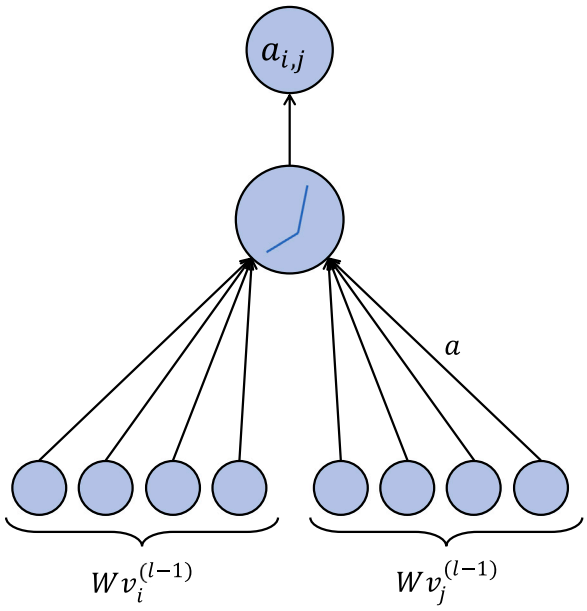


Fig. 13. Attention mechanism.

3.3.4. GNNs for clustering

The following section synthesizes the main state-of-the-art methods, followed by the presentation of three complementary and representative techniques.

While Graph Neural Networks (GNNs) have demonstrated remarkable success in various graph analysis tasks, including node classification and link prediction, they have shown greater resilience to improvements in deep clustering methods [26].

The general approach is to learn the graph embedding in a purely unsupervised way in an end-to-end framework. These algorithms exploit edge-level information in two ways. One simple way is to adopt an autoencoder framework where the encoder employs graph convolutional layers to embed the graph into the latent representation upon which a decoder is used to reconstruct the graph structure [135,136]. Autoencoders (AEs) (and the variants) are the most commonly used in graph clustering and a loss function $L[x, Dec(Enc(x))]$ can be defined to maximize the likelihood between source data x and decoded data $Dec(Enc(x))$. Graph autoencoder (GAE) constructs an encoder by using GCN and a decoder based on structure reconstruction, which has been treated as a powerful embedding method. However, GAE is vulnerable to perturbations and adversarial attacks. A subtle perturbation in the graph structure or node feature may lead to an apparent change in embedding and degradation for further tasks. Since the graphs in real life are often noisy and unfaithful, improving the robustness of GAE is important. Recent advancements in this field are rooted in generative, adversarial, and contrastive learning techniques, drawing inspiration from the progress made in the image field.

Contrastive learning has emerged in graph representation learning since 2019–2020 obeying the principle of mutual information (MI) maximization. It pulls the representations of samples with shared semantic information closer while pushing the representations of irrelevant samples away. A pioneer AGE [137] conducts contrastive learning by a designed adaptive encoder. GCA [131] proposed an adaptive augmentation with incorporating various priors for topological and semantic aspects of the graph. Besides, in the field of graph classification, AD-GCL [138] proposed a learnable augmentation for the edge level while neglecting the augmentations on the node level. MVGRL [139] generates two augmented graph views by utilizing the graph diffusion matrix as the augmented view. Different from the above algorithms, SCAGC [140] conducts random edge perturbation to construct the augmented view. Regarding feature operation, both DCRN [141] and SCAGC execute augmentations on node attributes through attribute corruption aiming to alleviate the collapsed representation by reducing correlation in both sample and feature levels. AGC-DRR [142] follows the graph contrastive learning framework and exploits the dual redundancy reduction mechanism to solve the corresponding information redundancy problem caused by the InfoMax operation. To fully explore the hidden high-order structure information of graph data, AHDDC [143] adopts a hierarchical GCN to the graph clustering task, and SLAPS [144] constructs a homogeneous node graph at the graph level and contrasts it.

Adversarial training (AT) improves the robustness of the model by adding some adversarial samples to the training target. For graph embedding, AT generates a series of adversarial structures and features in the neighborhood of the original graph and then considers the loss minimization problem based on these adversarial samples. ARGA and ARVGA [136,145] introduce adversarial training that forces the node embedding to match the known prior distribution based on the minimum reconstruction error. GraphGAN [146] provides negative samples by simulating data distribution instead of random negative sampling. In AGAE (Adversarial Graph Autoencoders) [147], the authors incorporate ensemble clustering into the deep graph embedding process and introduce an adversarial regularizer to guide the training of the autoencoder and discriminator. Most recent studies have predominantly focused on two-step approaches, but a limitation is that the resulting embeddings may not be optimally suited for the clustering task. To address this, DAEGC [84] proposes a goal-oriented graph attention auto-encoder clustering architecture to sufficiently explore both feature and structure information and also introduces the self-supervised clustering loss as a guide to controlling the clustering label assignment. JANE [148] enables embedding methods to effectively benefit from adversarial learning via joint correction between real data (including embedding, topology, and features) and fake data. SDCN [149] proposes a dual self-supervision mechanism that integrates the auto-encoder and the graph auto-encoder into a unified framework. On this basis, DFCN [150] designs a structure and feature information fusion module to enhance the quality of node representation by integrating the complementary information that is learned from the auto-encoder and the graph convolutional network. On the basis of DFCN, DCRN [141] implements a dual correlation reduction network to preserve discriminative information by reducing the information correlation at both sample and feature levels.

3.3.5. SDCN: Structural Deep Clustering Network

SDCN [149] consists of an autoencoder and a Graph Convolutional Network (GCN) module, grounded in two fundamental concepts. The initial idea involves propagating information in the GCN module from two sources to enrich its representation: the data generated by the autoencoder itself and the relationships between data points derived from a KNN Graph. The DNN and GCN modules have the same number of layers L. The secondary concept involves the implementation of a dual self-supervised mechanism to integrate these two diverse deep neural architectures and direct the overall model update. This approach draws inspiration from pioneering algorithms such as DEC. Let $H^{(l)}$ be the representation learned by the lth layer of the autoencoder and $Z^{(l)}$ by the lth of GCN.

$$\begin{aligned} Z^{(l-1)} &= (1-\epsilon)Z^{(l-1)} + \epsilon H^{(l-1)} \\ Z^{(l)} &= \phi_{GCN}\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{\frac{1}{2}}Z^{(l-1)}W_{GCN}^{(l-1)}\right) \\ H^{(l)} &= \phi_{AE}\left(W_{AE}^{(l)}H^{(l-1)} + b_{AE}^{(l)}\right) \end{aligned} \tag{32}$$

where ϵ is a balance coefficient and $\tilde{D}^{-\frac{1}{2}}\ \tilde{A}\ \tilde{D}^{\frac{1}{2}}$ is the normalized adjacency matrix having:

$$\begin{aligned} Z^{(1)} &= (1-\epsilon)Z^{(l-1)} + \epsilon H^{(l-1)} \\ Z^{(l)} &= \phi_{GCN}\left(\tilde{D}^{-\frac{1}{2}}\ \tilde{A}\ \tilde{D}^{\frac{1}{2}} Z^{(l-1)} W^{(1)}_{GCN}\right) \\ Z &= \text{softmax}\left(\tilde{D}^{-\frac{1}{2}}\ \tilde{A}\ \tilde{D}^{\frac{1}{2}} Z^{(L)} W^{(L)}_{GCN}\right) \end{aligned} \tag{33}$$

The SDCN Loss comprises the reconstruction objective, the cluster objective, and the classification one as follows:

$$\begin{aligned} \text{Loss} &= L_{res} + \alpha L_{clu} + \beta L_{GCN} \\ L_{res} &= \frac{1}{n}\sum_{i}^{n} \|x_i - \hat{x}_i\|_2^2 \\ L_{clu} &= KL(P \parallel Q) = \sum_{i}^{n}\sum_{j}^{n} p_{ij} \log \frac{p_{ij}}{q_{ij}} \\ L_{gen} &= KL(P \parallel Z) = \sum_{i}^{n}\sum_{j}^{n} p_{ij} \log \frac{p_{ij}}{z_{ij}} \end{aligned} \tag{34}$$

where α and β are weight parameters, $Q = [q_{ij}]$ is the distribution of the assignments of all samples, and $P = [p_{ij}]$ is the target distribution resulting from Q with the aim to make data representation closer to cluster centers. Self-supervision occurs as a result of the interdependence between the calculation of P from Q and the subsequent supervision of the update of the distribution Q by P.

3.3.6. DAEGC: Embedded Graph Clustering

Deep Attentional Embedded Graph Clustering (DAEGC) [84] extracts graph structure and attribute information in the encoding part utilizing a multi-layer encoder based on an attention mechanism. In the decoding part, DAEGC designs a self-training module, which guides the model to learn the feature representation for the clustering task by selecting some clustering soft labels with high confidence.

Let a graph $G = \{V, E, X\}$ with the attribute values $X = \{x_1, \ldots, x_n\}$ ($x_i \in \mathbb{R}^m$). The attention mechanism allows us to weigh the importance of neighbors in the node representation.

$$z_i^{l+1} = \sigma\left(\sum_{j \in N_i} \alpha_{ij} W z_j'\right) \tag{35}$$

where z_i^{l+1} is the output representation of node i, N_i the neighbors of i, α_{ij} is the importance of neighbor j to node i, and σ is a nonlinearity function. α_{ij} is represented as a single-layer feedforward neural network of the concatenation of x_i and x_j with weight vector $\vec{a} \in \mathbb{R}^{2m'}$.

$$\begin{aligned} c_{ij} &= \vec{a}^{\top}[W x_i \parallel W x_j] \\ \alpha_{ij} &= \text{softmax}_j(e_{ij}) = \frac{\exp(c_{ij})}{\sum_{r \in N_i} \exp(c_{ir})} \end{aligned} \tag{36}$$

having $x_i = z_i^0$ and $z_i = z_i^{(2)}$.

The latent embedding contains both content and structure information.

The inner product decoder (cf. Eq. (37)) is considered to predict the links between nodes through the reconstructed structure matrix of the graph $\hat{A}$.

$$\hat{A}_{ij} = \text{sigmoid}(z_i^{\top} z_j) \tag{37}$$

The reconstruction minimizes the difference between A and $\hat{A}$.

$$L_r = \sum_{i=1}^{n} \text{loss}(A_{ij}, \hat{A}_{ij}) \tag{38}$$

$$L_c = KL(P \parallel Q) = \sum_{i}^{n}\sum_{j}^{n} p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{39}$$

where q_{ij} measures the similarity between node embedding z_i and cluster embedding μ_j. As for SDCN, it is measured using a Student's distribution.

$$\begin{aligned} q_{ij} &= \frac{(1 + \|z_i - \mu_j\|^2)^{-1}}{\sum_k^K (1 + \|z_i - \mu_k\|^2)^{-1}} \\ p_{ij} &= \frac{q_{ij}^2 / \sum_i q_{ij}}{\sum_k^K (q_{ik}^2) / \sum_i (q_{ik})} \end{aligned} \tag{40}$$

where K is the number of clusters. The autoencoder is trained without the self-optimize clustering part to obtain a meaningful embedding z as described in Eq. (35).

Self-optimizing clustering is then performed to improve this embedding. The total objective function is defined as:

$$\text{Loss} = L_r + \alpha L_c \tag{41}$$

where α is a weight parameter.

To obtain the soft clustering assignment distributions of all the nodes Q through Eq. (40), the K-means clustering is performed once to obtain the initial center of the cluster centers that are updated based on the gradients of L_c concerning μ and z. The target distribution P works as "ground-truth labels" in the training procedure, but also depends on the current soft assignment Q which updates at every iteration.

3.3.7. SCGC: Simple Contrastive Graph Clustering

SCGC [151] is a recent Contrastive Graph Clustering following a classic scheme in two steps (*cf.* node embedding and clustering) with the aim of reducing the complexity of data augmentation and the consuming time. It comprises two stages, first an independent pre-processing stage involving a straightforward low-pass denoising operation for neighbor information aggregation and second a network backbone. The backbone is designed with a siamese network consisting of only two multilayer perceptrons (MLPs).

Let $G = \{X, A\}$ denotes an undirected graph, D is the degree matrix, and $\hat{A} = A + I$ is the renormalization trick, the symmetric normalized graph Laplacian matrix is defined as:

$$\tilde{L} = \hat{D}^{-\frac{1}{2}} \hat{L} \hat{D}^{-\frac{1}{2}} \tag{42}$$

Let be $H = I - \tilde{L}$ the Laplacian filter. By stacking up t layers with this filter, the smoothed attribute matrix is as follows:

$$X_s = \left(\prod_{i=1}^{t} H\right) X = H^t X \tag{43}$$

Instead of constructing two different views of the same node with complex modification against graphs, two augmented views Z^{v_1} and Z^{v_2} of the same vertex are generated by crafting parameter unshared siamese encoders and perturbing the node embeddings of the second MLP with Gaussian noise directly ($N(0, \sigma)$).

$$\begin{aligned} Z^{v_1} &= \frac{z^{v_1}}{\|z^{v_1}\|_2}, & z^{v_1} &= \text{MLP}_1(X_s) \\ Z^{v_2} &= \frac{z^{v_2}}{\|z^{v_2}\|_2} + N(0, \sigma), & z^{v_2} &= \text{MLP}_2(X_s) \end{aligned} \tag{44}$$

A cross-view similarity matrix S, based on the cosine similarity between the ith node of the first view and the jth of the second view, is obtained as follows:

$$S_{ij} = Z_i^{v_1} \cdot (Z_j^{v_2})^{\top}, \quad \forall i, j \in [1, N], \tag{45}$$

SCGC is based on a cross-view structural consistency objective loss L aiming at forcing the cross-view similarity matrix S to approximate the self-looped adjacency matrix $\hat{A}$ and then enhancing clustering performance.

$$\begin{aligned} L &= \frac{1}{N^2}\sum (S - \hat{A})^2 \\ L &= \frac{1}{N^2}\left(\sum_i \sum_j \mathbb{1}_{ij}^1 (S_{ij} - 1)^2 + \sum_i \sum_j \mathbb{1}_{ij}^0 (S_{ij})^2\right) \end{aligned} \tag{46}$$

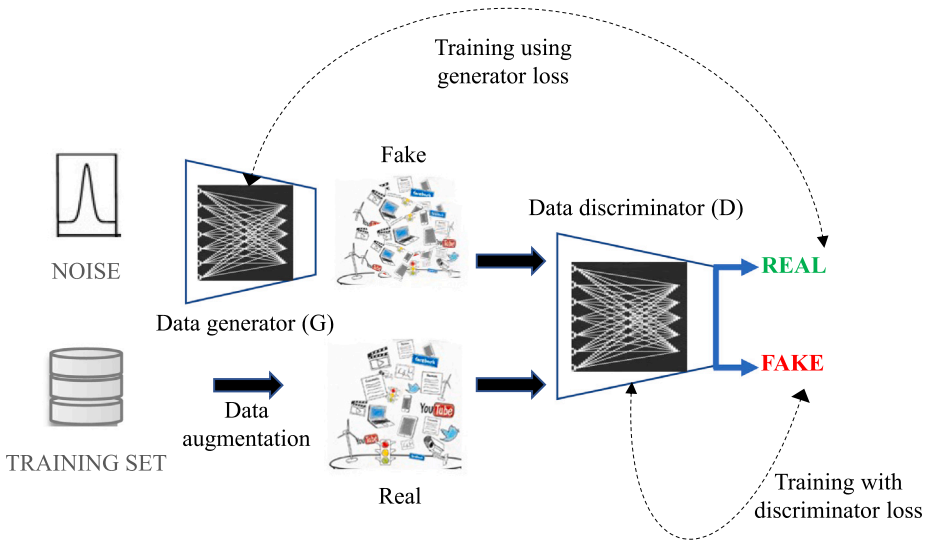


Fig. 14. GAN basic architecture.

Any clustering algorithm can then be applied to the resultant clustering embedding $Z = \frac{Z^{v1}+Z^{v2}}{2}$.

3.3.8. Synthetic analysis

Benefiting from the breakthroughs of GNNs on graph-structured data, GNNs are capable of organically integrating node attributes and graph structures in a united way and have emerged as a promising way for graph clustering. GNNs have shown significant superiority in graph-related tasks and applications, owing to their capability to explicitly encode node attributes and their interaction simultaneously, as well as implicitly learn high-order dependencies. Most graph-based methods are often based on Auto-encoders, contrastive learning, or random walk concepts. Deep clustering on graphs is an ongoing field and has proved to be more resilient to advances in GNNs [26]. However, even complex, very promising approaches [27–29] have recently emerged motivated by the achievements of many GNN-based methods in encoding graph structures, as well as some recent advancements in generative, adversarial, and contrastive learning schemes.

3.4. Generative models

Differently and maybe more appropriate for a knowledge discovery task, generative methods [31,87] such as InfoGan [88], ClusterGan [89] (GAN family) or such as VaDE [90], GMM-VAE [91] deep (VAE family) were adapted to deep clustering.

3.4.1. GANs for clustering

GANs [30,31] are generative models and have become the most popular deep generative model in recent years. They empirically learn the map that transforms the latent variables into the complex data distribution by playing a min–max game. The estimation of the posterior distribution of latent inference from the data is however an intractable problem in GAN models. They have shown a remarkable generation performance, especially in image synthesis. They generate new pattern instances that resemble training patterns by discovering and learning the regularities or patterns in input patterns. In other words, GANs aim to capture the underlying probability density of the training patterns. They are based on the adversarial idea [152]. Since [30], a number of GAN variants such as [153–156] have been proposed to improve the model structure, better control the outputs, extend the theory, adapt them for specific applications... As an example, in conditional GAN (cGAN) the generator and the discriminator are conditioned so that they know which type they are dealing with. The inherent strategy behind GANs can be assimilated to a self-supervised process which is a strong asset.

To perform the task, GANs include two neural network sub-models: a generator (G) that is trained to generate new patterns, and the discriminator (D) model that learns to distinguish true patterns from the output of the generator (cf. Fig. 14). The generator output is connected directly to the discriminator input. The discriminator classifies both real data and fake patterns from the generator. It is driven by a loss (cf. Eq. (47)) that penalizes the discriminator for misclassifying a real instance as fake or a fake instance as real. Through backpropagation, the discriminator's classification provides a signal that the generator uses to update its weights. The generator learns to make the discriminator classify its output as real. The generator takes random noise as its input, transforms it into a meaningful output, and is able to produce a wide variety of data, sampling from different places in the target distribution. The generator feeds into the discriminator neural network, and the discriminator produces the output we are trying to affect. The generator loss penalizes the generator for producing a sample that the discriminator network classifies as fake. Through backpropagation again, the weights of the discriminator neural network are modified but this time to optimize the generator loss and not the discriminator loss. The two models are then linked, and trained together but alternately; the parameters of one model are updated, while the parameters of the other are fixed. The GAN loss is as follows:

$$Gan_{\text{loss}} = \min_{\theta} \max_{\phi} \left[\mathbb{E}_{x\sim pdata}(\log(D_{\phi}(x))) + \mathbb{E}_{z\sim p(z)}(1 - \log(D_{\phi}(G_{\theta}(z))))\right] \tag{47}$$

ϕ and θ are the hyperparameters for the discriminator D, and the generator G and $\mathbb{E}$ is the distribution expectation.

The generator tries to fool the discriminator and improves with training. The discriminator tries to avoid being fooled, and its performance worsens because it cannot easily tell the difference between real and fake. The process is stopped when the discriminator model is sufficiently fooled, meaning that the generator model is generating plausible patterns. Because of the competition between the sub-models, GANs are known to be challenging to train.

In their raw formulation, GANs are unable to fully impose all the cluster properties of the real data onto the generated data, especially when the real data has skewed clusters. Data that are generated are not well controlled. Clustering accuracy performances on popular benchmarks such as STL10, CIFAR10, or USPS are about 30%. The difficulty in existing GANs concerns three main points:

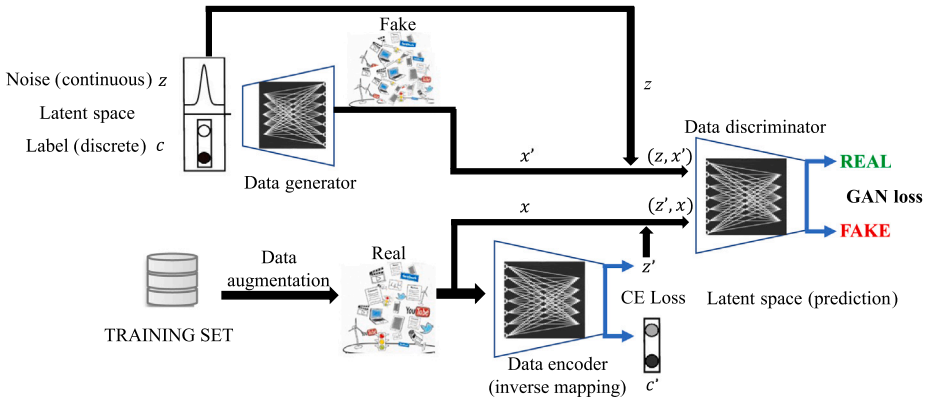


Fig. 15. ClusterGAN architecture.

- The generator and discriminator may not be optimal at the same time
- The generator cannot control the semantics of the generated samples
- GANs model the latent space as a simple unimodal distribution, ignoring the often more complicated implicit structure of the learned data distribution

GANs have however shown a very strong ability to capture complex data distributions, such as images and audio from raw features. They can then be useful for complex data clustering. GANs may improve clustering results without any prior information. If they are capable of generating "nice" samples it is essentially due to the powerful latent representation that is worth investigating to deliver a discovery task which is the cluster DNA. Because of its better performance in generating samples than autoencoders, researchers deduce that the powerful latent representation of GANs can improve clustering results.

InfoGAN [88] and ClusterGAN [76,89] are popular methods in the clustering GAN field. InfoGAN [88] is a pioneer of the derived models of GAN which learns and represents features contained in training data in a human-interpretable form without any label. It can learn disentangled representations by changing the latent space Z structure. Instead of only noise, a latent code is added as an input to G so that it can have meaningful effects on the generated samples. It trains a classifier and maximizes the mutual information between generated images and input vectors. It can be assimilated into a generative clustering model by decomposing data into noise and latent code to learn disentangled semantic information. This algorithm makes it possible to control the categories of generated samples according to the latent codes z_c and enhance the interpretability of GAN. While InfoGAN is a pioneer in learning disentangled representations, it is not explicitly designed for clustering which is the case for ClusterGAN. SIMI-ClusterGAN [157] is a two-stage approach aiming at improving ClusterGAN performances notably in the presence of imbalanced datasets while being representative of the state-of-the-art. Two other recent works improving ClusterGAN [108,158] are also worth to be mentioning. [158] proposed a Generative Adversarial Attention Clustering network based on Inverse autoencoder (IAE-ClusterGAN) while the originality of EDCN [108] is to use of Siamese Network to find a clustering-friendly embedding space to mine highly-reliable pseudo-supervised information.

The main principles of ClusterGAN, SIMI-ClusterGAN and EDCN are presented below.

3.4.2. ClusterGAN

ClusterGAN [76,89] is the first work to address the problem of clustering in the latent space of GAN based on the spirit of INFOGAN as the disentangling of latent space is obtained by a mixture of a continuous-discrete prior distribution. ClusterGAN has shown the ability to perform unsupervised clustering by joining an encoder with the concepts of GAN (Fig. 15), making it possible to obtain a mapping from the data space X to a lower-dimensionality space Z that could be clustering-friendly. The main ideas are as follows:

- Use of a mixture of discrete and continuous latent variables.
- Use of an explicit inverse-mapping network (encoder E) to obtain the latent variables given a data sample.
- Joint training of the GAN and the encoder with a clustering-specific loss.

In the first variant [76], clustering information was incorporated into GAN which resulted in an adversarial game among the generator, discriminator, and cluster. The second variant [89] was designed to perform clustering in the latent space by optimizing GAN and a clustering-specific loss.

ClusterGAN has however some weaknesses:

- The capability of the feature Extractor is not robust enough to noise permutation.
- Small distances between clusters of synthetic samples, *i.e.*, the distribution of representations in latent space are concentrated.
- It is difficult to preserve the consistency and cluster-specific information between real and synthetic samples.
- The method assumes uniformly distributed priors during the generation of modes, which is a restrictive assumption in real-world data and causes a loss of diversity in the generated modes.

3.4.3. EDCN, Siamese neural network

In a Siamese framework, there are two networks, commonly referred to as a student and teacher network, and their inputs are differently augmented views of the same object. Learning is carried out by maximizing the agreement between the outputs of the two networks.

EDCN (Unsupervised Discriminative feature learning via finding a Clustering-friendly embedding space) [108] is composed of a Feature Extractor, a Conditional Generator, a Discriminator, and a Siamese Network. Specifically, it utilizes two kinds of generated data based on adversarial training and the original data to train the Feature Extractor for learning effective latent representations. In addition, the Siamese network is adopted to find an embedding space that provides better affinity similarity.

Based on the self-supervised concept, the Siamese Network helps to find a clustering-friendly embedding space for reliable pseudo-supervised information mining. It is used to find a clustering-friendly embedding space for mining reliable pseudo-supervised information. This is applied to VAT and Conditional GAN to synthesize cluster-specific samples in unsupervised learning. VAT is adopted to synthesize samples with different levels of perturbation that can enhance the robustness of the Feature Extractor to noise and improve the lower-dimensional latent coding space discovered by the Feature Extractor.

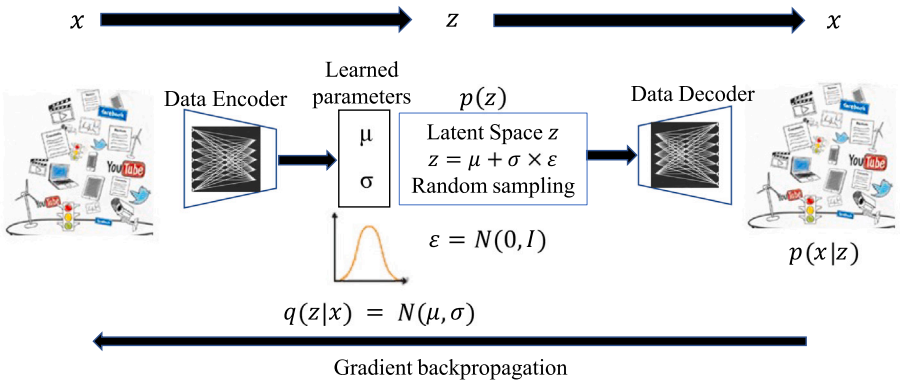


Fig. 16. VAE architecture.

The training of EDCN involves adversarial gaming between three players, which not only boosts performance improvement of the clustering but also preserves the cluster-specific information from the Siamese Network in synthesizing samples. While the effectiveness of EDCN has been empirically demonstrated, four losses are considered for the training and the authors adapt the 'backbone' architecture to the datasets to be handled. This makes the algorithm not really friendly for a non-expert investigator.

3.4.4. VAEs for clustering

A variational autoencoder inherits an autoencoder architecture that is trained to minimize the reconstruction error between the encoded-decoded data and the initial data. It can be defined as an autoencoder whose training is regularized to avoid overfitting and to ensure that the latent space has good properties that enable a generative process. In contrast to AE, a VAE makes strong assumptions regarding the distribution of latent variables. The main reason is that with AE there is no guarantee about the regularity of the organization of the latent space for autoencoders. Traditional autoencoders can lack continuity in the latent space, preventing interpolation between training points and, thus, its generative ability. There is a dependence on the distribution of the data in the initial space, the dimension of the latent space, and the architecture of the encoder. So, it is rather difficult to ensure, a priori, that the encoder will organize the latent space in a smart way compatible with the classic generative process.

Variational autoencoders (VAEs) [159] then attempt to remedy the limitation of AE by modeling the input probability distribution using Bayesian inference. They are based on Gaussian mixture models which assume that all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters, referring to the soft clustering concept [160]. The main goal of the process is to enforce a continuous, smooth latent space representation allowing the decoder to act as a generator (cf Fig. 16). With VAEs, each sample x must have its own corresponding posterior distribution so that the random hidden variable sampled from the posterior distribution can be restored to the corresponding reconstructed sample x' by the generator.

The term "variational" comes from the close relationship that exists between regularization and the variational inference method in statistics. VAEs make it possible to sample new data from the learned distribution and are also well-suited to provide interpretable/disentangled data representations in low-dimensional space. In order to introduce some regularization of the latent space and create unique clusters, a slight modification of the encoding-decoding process is carried out: instead of encoding an input as a single point, a distribution over the latent space is encoded. Maximum Likelihood Estimation (MLE) is used to train the latent variables. VAE maps the samples to the hidden variable z through the encoding process $q(z \mid x)$, assumes that the hidden variable obeys a given distribution, and draws samples from the hidden variable. This method can transform the likelihood function into the mathematical expectation under the hidden variable distribution. The extracted latent z ($q(z \mid x)$) must have similar a similar distribution to that of the prior distribution $p(z)$; the latter is usually imposed with a Gaussian form. This is expressed by a regularization term based on the Kullback–Leibler divergence. With VAE architectures, the cost is often named as the Evidence Lower Bound (ELBO) cost (cf. Eq. (48)) referring to the mathematical approximation of the marginal likelihood (MLE). The MLE concatenates the marginal likelihoods of individual data points but it is not directly tractable. It is then approximated by a lower bound solved by the ELBO. See mathematical details in [159,161]

Precisely, the usual cost is as follows:

$$ELBO_{\text{loss}} = \lambda_1 \min \mathbb{E}_q \left[\log\ p(x \mid z) - \log p(z)\right] - \lambda_2 \mathbb{E}_q \left[\log p(x \mid z)\right] \quad (48)$$

$\mathbb{E}$ stands for expectation under q. The first term is the KL divergence aiming at minimizing the difference between the log probability of z under the q distribution and the log probability of z under the p distribution. z is sampled many times (according to a batch size) in order to estimate the KL divergence that needs to be minimized ($q \simeq p$). This cost controls the dispersion to a certain extent. The second one is the reconstruction term: z is again sampled from q and is used to calculate the probability of seeing the input x given the sampled z. It needs to be maximized, explaining the negative sign. Costs are complementary and have to be well-balanced using λ_1 and λ_2.

The decoder with VAE models acts as a generative model while the encoder is a pure recognition model. Thus, VAE can learn data distributions in the latent space which is good for unsupervised learning tasks. The Gaussian prior remains a strong assumption that can hinder the subsequent clustering process from separating different groups effectively and may lead to crowding clusters. Several DC models turn to the Gaussian mixture prior to modeling the discrete clusters in the latent space, which has contributed to a more clustering-oriented VAE framework.

3.4.5. VaDE and GMVAE

Variational Deep Embedding (VaDE) [90] combines a generative architecture with the ability to cluster data points by capturing the statistical structure of the data (cf Fig. 17). It allows both clustering and data generation. Variational Deep Embedding (VaDE) uses a Gaussian mixture model (GMM) as the predefined distribution. A (GMM) is a probabilistic model that assumes that all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. It is the key idea. Note that VaDE extends VAE in that it uses a mixture of Gaussian models and also differs in the way that the latent space is exploited. VAE learns by minimizing the difference between the reconstructed and original data X when VaDE generates the set of clusters with a Multivariate Gaussian prior from data X, and latent z is generated from each cluster.

The VaDE loss function (cf. Eq. (49)) combines the VAE loss function with a term controlling the divergence between the cluster distribution

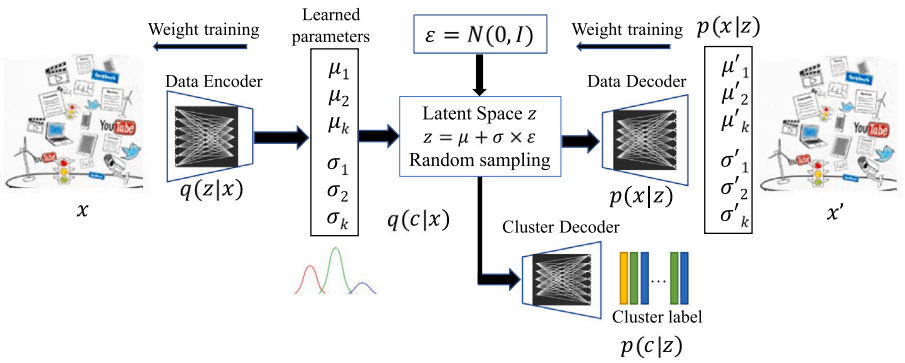


Fig. 17. VaDE architecture. The samples are generated using (μ_i, σ_i) and ϵ via the reparameterization trick also allowing back propagation training.

from the data and the latent space: the probability of being included in a cluster from data is approximately the same as the average probability of extracting clusters from z.

$$ELBO_{\text{VaDE}} = ELBO_{\text{VAE}} - KL(q(c \mid x), p(c \mid z)) \tag{49}$$

The data distribution is encoded as a GMM in the embedded space and the corresponding sampled embedding is decoded using the reparameterization trick [162] in order to allow optimization based on Stochastic Gradient Variational Bayes (SGVB) [163]. The GMM selects a fitting cluster that is subsequently transposed towards an observable embedding by a DNN. Instead of trying to learn a distribution, VaDE learns as many distributions as expected clusters. The GMM picks a cluster from which a latent embedding is generated to be then decoded via the DNN. While variational autoencoders are somewhat complex, the modifications introduced to disentangle their latent spaces are remarkably simple. In Gaussian Mixture VAE [91], the general idea is that the objective function optimized by a variational autoencoder applies a penalty on the latent space encoded by a neural network to make it match a prior distribution and that the strength and magnitude of this prior penalty can be changed to enforce less entangled representations [86].

3.4.6. Synthetic analysis

VAEs and GANs have been demonstrated to achieve satisfactory clustering performances and have been widely adopted in recent deep-learning studies. There are, however, typical issues associated with them for example, images generated by VAE are usually blurry, and the distribution assumption induces some limitations. In fact, they are used only moderately and GANs are more common in applications. GAN, however, lacks the encoder–decoder architecture and is hard to train, and algorithms such as ClusterGAN, SIMIClusterGAN, or even EDCN are complex. One of the main shared drawbacks is that they require large computing resources, are difficult to train, and rely on a complex theory. More optimized network architectures are needed for simplicity. It is worth noting that considerable research focuses on combining the two frameworks [164,165]

4. Overall evaluation

There are many deep clustering algorithms, making the choice difficult for the user. In this section, the different families are analyzed through the most representative algorithms, with an emphasis on the recent ones.

4.1. Main characteristics of representative algorithms

Tables 1 and 2 describe the main characteristics of selected representative algorithms.

4.2. Evaluation metrics

Cluster analysis involves assessing the partition that aligns best with the data structure. Various validation strategies, including internal and external measures, play a crucial role in result assessment. Internal validation demands no additional information about the data, and numerous Cluster Validity Indices (CVIs) have been proposed [181,182]. Noteworthy among these indices are Silhouette (SL), Calinski–Harabaz (CH), and Dunn. However, commonly used statistical measurements within internal indices exhibit constraints, and many popular internal CVIs demonstrate optimal performance only under specific clustering algorithms and simplistic dataset structures [183]. Furthermore, their applicability diminishes when confronted with data in high-dimensional spaces. The dynamic nature of this field is evident as novel CVIs continue to be proposed, underscoring the evolving landscape of cluster evaluation [184,185].

In the deep clustering community, there is currently no real consensus on the use of CVIs. It is usual to compare the partitions provided by a clustering algorithm with the classes using external indices [186] when the class labels are known.

This way is debatable because there is not necessarily a link between the internal structure of the data and a partition of the data with their class labels. It can only prove that a single semantic that underlies the input data can be retrieved using a clustering approach. In the absence of reliable CVIs in high-dimensional space, it remains, however, the currently adopted method of evaluation.

Popular indices are normalized mutual information (NMI) [187], Entropy (E), F-measure (F), adjusted Rand index (ARI) [188], and the most widely used one, clustering accuracy (ACC) [189]. ACC and NMI are detailed in the next section.

4.2.1. ACC

ACC can be expressed as follows:

$$\text{ACC} = \max_{m} \sum_{i=1}^{n} \frac{\mathbb{1}[y_i = m(c_i)]}{n} \tag{50}$$

where y_i is the ground truth, c_i the output vector generated, and m is the mapping function representing the set of possibilities for assigning observations in the classes from c_i.

4.2.2. NMI

NMI is a normalized value of mutual information so that the value of mutual information, which was initially unlimited, becomes within the range of values $[0, 1]$. Let U be the ground-truth label, and V be the label from the unsupervised algorithm. The value of the NMI can be determined in the following equation:

$$\text{NMI}(U, V) = \frac{\text{MI}(U, V)}{\sqrt{H(U)H(V)}} \tag{51}$$

$\text{MI}(U, V)$ is a function of mutual information between clusters V and the ground truth labels U. Normalization is done by dividing the value of the mutual information function by $\sqrt{H(U)H(V)}$.

Table 1
Top/representative state of-the-art deep clustering approaches per family (AE/CAE, CNN): synthetic description L_r: reconstruction loss, L_{clust}: clustering loss, KL: Kullback–Leibler divergence.

Family	Framework	Year	Loss	Innovation/characteristics
AE/CAE	DEPICT	2017 [100]	L_r + unsupervised cross-entropy loss	Convolutional noisy autoencoder and relative entropy minimization
	DCSPC	2021 [166]	L_r + L_{clust} +discriminative loss	Self-evolution training algorithm, representation and clustering jointly pseudo-supervision regularized
	SPC	2021 [167]	L_r+ cross-entropy loss	Selective pseudo-label clustering with Multi autoencoders
	N2D	2021 [97]	L_r combined with UMAP [168]	Local manifold learning on an autoencoded embedding
	DSSC-EAGF	2022 [169]	L_r+L_{clust} (KL) + a self-supervised loss	Self-supervised clustering with embedded adjacent graph features
	MEDEC	2023 [166]	L_r+L_{clust} (KL)+ L_{view} + L_{att}	Multi-view embedding + auto attention mechanism
CNN	JULES	2016 [170]	Agglomerative clustering loss (L_{clust})	Trains the representation by enforcing nearest clusters having similar features
	DAC	2017 [102]	Pairwise classification Loss	Self-adaptation learning
	IMSAT	2017 [105]	Regularization information maximization	Hybrid data augmentation, mutual information
	RDKM	2021 [171]	Kmeans+self-augmented training Loss	Hierarchical k-means through deep layers and regularization
	DeepCluster	2019 [103]	Hierarchical+self-supervised loss	Self-supervision and clustering alternately combined + Introduction of super class
	IIC	2019 [104]	Classification of functions for paired data samples	Invariant Information Clustering Mutual information
	SimCLR	2020 [14]	Contrastive loss based on cosine similarity	Contrastive learning to maximize agreement between 2 augmented versions of the same image
	MoCo	2020 [107]	Contrastive loss+ InfoNCE [172]	Self-supervised, use of a dictionary of keys Uses a Memory Bank and a Queue to store and sample batches
	SwAV	2020 [65]	Double contrastive loss	Simultaneously clustering while enforcing assigning a unique cluster for different image transformations memory optimization via a multi-crop approach
	PICA	2020 [110]	Cosine similarity loss (augmentation, perturbation)	Maximizing the "global" partition confidence
	SCAN	2020 [70]	Soft assignment + entropy losses	Self-supervision and Nearest neighbors mining
	BYOL	2020 [92]	Similarity loss (predictions and target projections)	Self-supervision without requiring positive or negative sample pairs
	PCL	2021 [173]	ProtoNCE loss (InfoNCE for contrastive learning)	Contrastive learning encoding clusters in the embedding space
	SPICE	2022 [113]	Cross entropy loss	Self-supervision: synergizes the similarity among instances and a semantic discrepancy between clusters
	NNM	2021[111]	Similarity global and local losses + entropy loss	Matching the nearest neighbors from local and global levels
	IDFD	2022 [174]	L_I (discrimination) + L_F (decorrelation) losses	Performonality constraint and feature decorrelation
	SIMSIAM	2022 [175]	Data augmentation, cosine similarity	Siamese neural network
	ProPos	2022 [114]	Prototype scattering loss + Positive alignment + uniformity losses	Prototype scattering and positive sampling Maximizes the between-cluster distance Minimizes the within-cluster compactness

4.3. Clustering accuracy on image data

The comparison is limited to the most popular benchmark datasets[1] that come from the image field. These data are of common use, including in recent proposals. The scores from different papers have been collected in the experimental result sections. The results are summarized in Table 3 and the values are the accuracy classification score (ACC). As the latter is not available for DSSC-EAGF, it was replaced by the normalized mutual information index (NMI) [187] that is roughly linearly linked to ACC ($NMI \simeq 0.9ACC$). Algorithms that lack scores for datasets are because the paper did not originally test on this dataset, nor was it evaluated by others. It was straightforward to get a "neutral" score due to the overall training complexity.

MINST is very popular but not really discriminative as it is relatively easy. All the novel approaches reach an accuracy of more than 95% and close to 100%. CIFAR 10 is more interesting as it is reputed to be more difficult. Novel algorithms based on CNN and self-supervision (since 2020) generally give better results than AE/CAE or generative approaches. Novel CNN methods are also particularly efficient with the ImageNet dataset. This dataset is not the most complex but embraces a lot of classes and is very large. The initial challenge for researchers was to reduce the accuracy gap between supervised and unsupervised learning strategies. All novel CNN methods give classification scores around 90%. EDCN outperforms all the others reaching a score close to 100%. STL10 is reputed to be a difficult and unbalanced dataset. Pioneer deep clustering methods achieve very low classification accuracies between 10% and 40%. Novel CNN methods give better results, especially SPICE and ProPoS. Note the IMSAT, which is older, performs very well, and novel generative approaches (SIMI-ClusterGan, and EDCN) are the winners. USPS and COIL are less benchmarked. For USPS, N2D and SPC are particularly relevant as well as EDCN and GAN approaches. For COIL, the winner is a generative approach (BigBiGAN) far ahead from DEPICT, RDKM and GMVAE. Clearly, Deep clustering methods outperform non-deep clustering methods. As an example, standard K-means achieves less than 60% of accuracy with MINST and less than 20% for the other datasets. This highlights the deep clustering contribution in managing high-dimensional data sets. RDKM extends K-Means, and it is not limited to image datasets. It improves K-Means's performances but its classification accuracy with the MINST dataset is low compared to novel CNN approaches.

In sum, there is no clear domination of one method against the others as the winner changes from one dataset to another. It can be,

[1] https://paperswithcode.com/dataset.

Table 2
Top/representative state of-the-art deep clustering approaches per family (GNNs, Generative): synthetic description (L_r: reconstruction loss, L_{gcn}: graph reconstruction loss, L_{clust}: clustering loss, KL: Kullback–Leibler divergence.

Family	Framework	Year	Loss	Innovation/characteristics
GNNs	SDNE	2016 [176]	Modified L_r + proximity + regularization losses	Embedding method that exploits the first-order and second-order proximity jointly to preserve network structure
	GAE/VGAE	2016 [135]	Variationnal lower bound	Based on VAE, a GCN encoder and an inner product decoder mixed
	DNGR	2016 [177]	L_r	Embedding method based on a random surfing model to capture structural information + a stacked denoising autoencoder
	GAT	2018 [134]	Graph-based loss function to optimize node representations	Compute hidden representations in the graph using self-attention by attending to its neighbors
	ARGA	2019 [136]	L_r + adversarial regularization cost	Incorporate an adversarial training scheme to regularize the latent codes to learn a robust graph representation
	DAEGC	2019 [84]	$L_r + L_{clust}$	A graph attention layer + a self-supervised clustering loss to control the clustering label assignment
	AGE	2020 [137]	Cross entropy loss	Attribute graph embedding, contrastive learning with Laplacian smoothing + an adaptive encoder for better node embeddings
	SDCN	2020 [149]	$L_r + L_{clust} + L_{gcn}$	A dual self-supervision mechanism with the auto-encoder and the graph auto-encoder within a unified framework
	JANE	2020 [148]	$\min_{(G,E)} - \max_{(D)}$ function like GAN	Jointly discriminate the real and fake combinations of topology, node features and embeddings
	DFCN	2021 [150]	L_r attribute + adjacent matrices	Siamese network Similar to SDCN with a structure and attribute
	SLAPS	2021 [144]	classification loss and L_r losses	Joint Learning of Adjacency and GNN Parameters, Self-supervision Information fusion module fusing the dual information
	DCRN	2022 [141]	MCE Loss to Identity Matrix	A dual reduction network to maintain information by reducing correlation at both sample and feature levels
	SCAGC	2022 [140]	Self consistent contrastive loss	Optimize intra-cluster and inter-cluster consistency representations
	SSGNN	2023 [29]	L_r + graph structure losses	Self-supervision and clustering alternately and layer by layer
	GC-SEE	2023 [178]	Cross entropy + L_r losses	Fusion graph convolution module + a graph encoder with attention
	DCAHR	2023 [179]	L_r(data+Adjancy matrix)+L_{clust}	Self-supervision and clustering alternately
	AHDDC	2023 [143]	$L_r + KL(P \parallel Z) + L_{clust}$ losses	Denoising autoencoder + attention in the GNN (KNN graph)
	SCGC	2023 [151]	L_r + contrastive + L_{clust} losses	Contrastive scheme with high time reduction by simplifying the data augmentation and the graph convolutional operation
Generatives	GMVAE	2017 [90]	Variational lower bound on the likelihood with priors	Gaussian mixture mode
	ClusterGAN	2019 [89]	GAN Loss + L_{clust}	Trains a GAN with a clustering-specific loss
	BigBiGAN	2019 [180]	GAN loss+ Encoder Loss	Adds an encoder, a generator and improves the discriminator
	SimiClusterGAN	2021 [157]	GAN loss+ 5 additional losses	Extends ClusterGAN via self-augmented information maximization, priors
	EDCN	2022 [108]	Conditional-GAN loss, VAT loss Siamese Network and collaborative losses	Integrated VAT and Conditional GAN into a unified framework, Siamese Network

however, said that autoencoder architectures, due to the reconstruction loss that imposes a strong constraint, appear to have more difficulties for complex datasets even using data augmentation approaches. This analysis has to be nuanced as the benchmark data sets are not statistically representative. Novel CNN approaches perform very well, but most of the time, they were specifically developed for image datasets. In the image field, there is an expertise that allows qualitative data augmentation which is not always the case for other fields. GAN models are very competitive for some data sets, especially SIMI-ClusterGan and EDCN which involve augmentation strategies but are more mitigated for others. Except for MINST, VAE approaches are less competitive.

4.4. Clustering accuracy on text data

Text clustering involves categorizing similar text from a collection, with varying levels of granularity, including document, paragraph, sentence, or phrase levels [190]. It is closely tied to text mining, whose significance is growing [191,192], especially with the expanding access to big data across digital platforms. The contribution to this field made by supervised methods is relatively well-known, this is less so for unsupervised learning. One of the initial processes during text clustering is to represent text in the form of a numeric vector before applying clustering or deep embedding clustering approaches. This representation aims to help discover and learn patterns from the data. Representation learning methods [193] commonly used are a bag of word methods such as Term Frequency-Inverse Document Frequency (TFIDF) and sequence of word methods such as word2vec and more recently Bidirectional Encoder Representations from LSTM and Transformers (BERT). Unlike feature-based text clustering algorithms, model-based clustering algorithms perceive the clustering process as a generative model. In the latent Dirichlet allocation (LDA) model, topics are initially generated from texts, after which words in the text are generated from these topics. In text clustering, the conventional approach involves representation learning followed by cluster analysis. However, the joint optimization of both tasks – representation learning and clustering – is still in its early stages.

4.4.1. Text datasets with graph structure

Performance comparison experiments have been conducted on five popular text datasets (Table 4) having graph structure, including ACM, DBLP, CiteSeer, PubMed, and Cora datasets. The methods in the experiments include AE and graph methods and are as follows:

- The AE-based methods such as AE, DEC, DBC, and IDEC convert the raw data to low-dimensional codes to learn feature representations by AE and then perform clustering over the learned latent embeddings.

Table 3
Top/representative state-of-the-art deep clustering approaches per family (AE/CAE first, then CNN and finally the Generative family): clustering accuracy with the most frequently used image benchmarks (the top three scores are in bold).

Framework	ImageNet 10	MINST	CIFAR 10	STL10	USPS	COIL
DEPICT [100]		0.965	0.326	0.371	0.838	0.667
DCSPC [166]		0.853	0.464	0.791	0.694	
SPC [167]		**0.990**			**0.984**	
N2D [97]		0.979			0.958	
DSSC-EAGF [169]		0.840*			0.716*	**0.750***
JULES [170]	0.334	0.964	0.275	0.288	0.950	
DAC [102]	0.527	0.978	0.522	0.471	0.614	
IMSAT [105]		**0.984**	0.456	**0.941**		
RDKM [171]		0.554				**0.680**
DeepCluster [103]	0.720	0.656	0.374	0.344	0.562	
IIC [104]		0.984	0.576	0.596		
SimCLR [14]	0.710					
MoCo [107]	0.711		0.776	0.728		
SwAV [65]	0.753			0.831		
SCAN [70]			**0.883**	0.809		
NNM [111]			0.843	0.808		
PCL [173]	0.617		0.874	0.41		
BYOL [92]	0.939		0.978	0.825		
SPICE [113]	0.921		0.838	0.910		
PICA [110]	0.870		0.696	0.712		
IDFD [174]	**0.954**		0.815	0.756		
SimSiam [175]	0.921		0.856	0.716		
ProPos [114]	**0.956**		**0.943**	0.867		
VADE-GMVAE [90]		0.945	0.336	0.317	0.566	0.441
ClusterGAN [89]		0.964	0.412	0.423	**0.970**	
BigBiGAN [180]	0.613	0.871	0.316		0.795	**0.936**
SIMI-ClusterGAN [157]		**0.986**	0.512	**0.954**	**0.951**	
EDCN [108]	**0.985**	0.544	0.482	**0.980**		

Table 4
Top/representative state-of-the-art deep clustering approaches per family (AE/CAE first, then GNNs): clustering accuracy with the most frequently used text benchmarks with graph structure (the top three scores are in bold).

Framework	ACM	DBLP	Citeeser	PupMed	Cora
DEC	0.843	0.582	0.559	0.601	0.499
IDEC	0.864	0.603	0.605	0.551	0.525
DBC	0.827		0.591	0.621	0.499
AE	0.861	0.577	0.591	0.619	0.384
VAE	0.697	0.439	0.392	0.547	0.367
VGAE	0.804	0.612	0.613	0.621	0.434
AGE			0.702	**0.711**	**0.768**
GAE	0.615	0.3981	0.395	0.568	0.402
VGAE	0615.	0.403		0.556	0.396
DAEGC	0.897	0.621	0.645	0.688	0.704
JANE			0.622	0.692	0.726
DFCN	0.911	0.760	0.695	0.689	0.569
SDCN	0.906	0.681	0.659	0.689	0.356
SCGC	0.898	0.672	0.710	0.631	**0.739**
SSGNN	**0.936**	0.776	0.714		
GC-SEE	0.917	0.792	0.709		**0.736**
DCAHR	**0.936**	**0.821**	**0.735**	**0.734**	0.725
DCRN	**0.919**	**0.796**	0.708	**0.698**	
ARGA	0.821	0.612	0.617	0.699	0.640
SCAGC	0.913	**0.794**			
SCGC	**0.926**	0.777	**0.731**		
SSGNN	**0.936**	0.776	0.714		

- The GCN-based methods such as GAE, VGAE, DAEGC, and ARGA. They adopt the GCN encoder to learn the node content and topological information for clustering.
- In addition, other hybrid methods combining AE and GCN boost the embedded representations for clustering such as SDCN, DFCN, and their variants. These methods integrate GCN with AE from different perspectives to train the clustering network jointly.

The selected methods are those for which results can be systematically compared with others, utilizing a minimum of three databases from the five datasets.

Three key observations can be discerned. Firstly, the efficacy of deep clustering methods, exemplified by DEC and IDEC, appears limited due to their omission of the underlying graph structure in graph data. Secondly, a notable enhancement in clustering performance is achieved by various deep reconstructive graph clustering methods, including GAE, DAEGC, SDCN, and DFCN. These approaches succeed by reconstructing graph information, thereby augmenting the discriminative capabilities of the networks. Thirdly, recent contrastive methods like SCGC and DCRN further elevate sample discriminative capabilities by bringing positive sample pairs closer and pushing away negative sample pairs. This results in the most promising overall performance. It is noteworthy that DCAHR consistently demonstrates top-tier performance across all databases. Its innovative approach lies in concurrently acquiring enriched semantic and structural representations layer by layer, contributing to its exceptional results.

4.4.2. *Text datasets without graph structure*

The field of text clustering is vast [194]. Many studies evaluate methods on very diverse datasets, making fair comparisons challenging. However, two widely used benchmarks are 20newsgroup and Reuters. They have been employed to compare classical and deep learning methods. Deep learning methods are based on autoencoder and graph architectures. They also incorporate various data representations such as LSTM [195], BiLSTM [196], and BERT. Four methods leverage data augmentation techniques such as back translation (BT) and random masking (RM) to enhance performance, as highlighted in [195]. The comparison in Table 5 aims to highlight the effectiveness of emerging deep clustering approaches in this domain.

While deriving overarching conclusions from two datasets poses challenges, except MDEC, which achieves the best performances, no single family unequivocally outperforms others, but valuable insights can still be gleaned. The superiority of deep learning methods over classical approaches is evident. Notably, the effectiveness of deep clustering methods, such as DEC and IDEC, may seem less pronounced compared to Graph Neural Network (GNN) methods, particularly exemplified by DCAHR. It is noteworthy to highlight the impact of the

Table 5
Top/representative state-of-the-art deep clustering approaches per family (non-deep learning methods, AE/CAE, GNN methods and ELMO/LSTM/BERT representations): clustering accuracy with the most frequently used text benchmarks without graph structure (the top three scores are in bold).

Framework	20newsgroup	Reuters
K-MEANS	0.1	0.540
LDA	0.372	0.549
DEC	0.308	0.756
IDEC	0.3241	0.5306
AE	0.417	0.537
VAE	0.136	0.613
MDEC (Multi views with AE)	**0.559**	**0.844**
GAE	0.1702	0.547
VGAE	0.1702	0.646
ARGA	0.183	**0.776**
SDCN	0.122	**0.776**
DFCN	0.156	0.756
DCAHR	0.269	**0.832**
ELMO (BiLSTM layers)+ K-MEANS	**0.481**	0.579
BERT + K-MEANS	0.419	0.471
LSTM SCL(BT)	0.302	0.55
LSTM SCL(RM)	0.235	0.503
BERT + SLC(BT)	**0.501**	0.617
BERT + SLC(RM)	0.441	0.644

contrastive learning mechanism using LSTM and BERT representations. BERT operates in a pre-trained and fine-tuning mode but its effectiveness diminishes when tasked with acquiring latent representations for domain-specific datasets. BERT + K-MEANS yields results akin to Autoencoder (AE) schemes but surpasses them with a contrastive learning scheme, outperforming those employing LSTM representations. Even with contrastive learning, LSTM and BERT representation fall short of the effectiveness achieved by Graph Neural Network (GNN) methods when applied to the Reuters dataset.

4.5. Multicriteria evaluation

While the classification criterion is central, other criteria have to be taken into account for the ordinary user. The selected algorithms are evaluated through seven criteria that are summarized in Table 6.

The first column indicates whether pre-processing is needed for data preparation (e.g. which data augmentation to choose), the need to select the appropriate backbone CNN architecture as well as the hyperparameter initialization. The second and third columns summarize the level of user expertise (Fair training) and computational load to run the algorithm. Some of them are very difficult to train due to the nature of the algorithm, the architecture, or the number of parameters to be considered. For these two columns, the higher the number of crosses the friendlier the algorithm: easy training and light computational load. The model type is reported in the fourth column: it can be discriminative (D), generative (G), or both. The following columns indicate some limitations of the corresponding algorithms: the ones based on the K-means process only produce hyperspherical clusters and are not able to handle arbitrarily shaped clusters (column #5); others were specifically designed for image data via smart dedicated tricks/expertise thus are not adapted to other data types or require specific additional investigations (column #6). Lastly, some algorithms were developed with semantic constraints, either in the design of the latent space using data augmentation or by defining a pretext task or cost function based on prior knowledge. The goal is to perform better with unfriendly data sets such as STL, which are mentioned in column #7. The comments about Table 6 are grouped in three sections, corresponding to the three families of methods.

4.5.1. AE/CAE

The two-stage training scheme used by the first generation of AE/CAE presents some limitations: the symmetric architecture restricts the network depth to maintain computational tractability. AE/CAE solutions often rely on K-means clustering. They are then generally less efficient regarding classification accuracy with non-friendly data sets. They also face difficulties in capturing original data semantics as they are constrained by the reconstruction cost. N2D uses a shallow clustering algorithm instead of a deeper network with the aim of identifying the underlying manifold and achieving a more cluster-friendly embedding. With DSSC-EAGF the AE benefits from additional knowledge: the spatial relationships induced by a graph. This requires a heavier computational load and makes it a bit trickier. AE/CAE can be used, for most of them, either as discriminative or generative models. These solutions are not limited to image data but are often limited to simple cluster shapes when based on the K-means process. They are generally reasonable in terms of memory and computational time.

4.5.2. CNN approaches

In contrast to AE/CAE, which try to learn robust cluster-oriented features for complex datasets, CNN methods only optimize a clustering loss. As a result, the depth of the network is not limited and supervised pre-trained architectures can be used by transfer learning to extract more discriminative features. They are discriminative models. Without reconstruction loss, there is the risk of learning a corrupted feature representation, thus the clustering loss should be well-designed.

The first algorithms reported in Table 6, JULES, DAC, IMSAT, DeepCluster and ICC are relatively simple methods that perform well on user-friendly image datasets. They are quite easy to tune and their computational load is reasonable. JULES, one of the first Deep Clustering algorithms, is an exception as it is based on a hierarchical process facing challenges related to computational and memory complexity. DAC and DeepCluster are able to handle arbitrary-shaped clusters. They take advantage of CNN architectures as priors for clustering images. DAC employs a straightforward binary pairwise classification as a clustering loss, supplemented by a regularization constraint and a self-paced strategy. Both methods are however sensitive to initialization. IMSAT and ICC are pioneers in the utilization of Self Augmentation Training. These methods learn a clustering function by maximizing the mutual information between an image and its augmentations. RDKM can be considered as a deep implementation of K-means. It is a straightforward method capable of handling diverse input data without the need for pre-processing tasks and outperforms K-means when dealing with high-dimensional data.

The remaining methods reported in Table 6 were proposed in the most recent years and are based on smarter self-supervision and contrastive learning schemes that have further improved their clustering performance, even with complex datasets. They all require some preprocessing setup (data augmentation and/or pretext task definition) and are all semantic-oriented. Many of them were specifically designed for image data and can handle arbitrary-shaped clusters. Most of these methods are sensitive to initialization as they rely on the initial feature representation. They often require specific skills: the choice of the pretext task for self-supervision purposes, and the sample selection for the different learning strategies. In self-supervision, selecting an appropriate pretext task that well aligns with the downstream task is essential. The choice and the level of augmentation strength are pivotal and data-dependent. Weak augmentation may improve the positive/negative distinguishability while a stronger one is likely to penalize positive sample identification. An optimized augmentation strategy can yield significant information gains, as demonstrated by the success of methods like SimCLR and SwAV compared to ICC or IMSAT.

These methods also require important resources: as an example, SimCLR uses 128 TPU (Tensor Processing Units) for large batch training in the self-supervised training phase. MoCo uses a momentum update

Table 6
Multi criteria assessment of the selected algorithms. D stands for Discriminative, G for Generative, "+" indicates the level of positive assessment. (AE/CAE first, CNN, GNN, and finally the Generative family).

Algorithm	Pre-process	Fair training	Computational load	Model type	Arbitrary shape	Image oriented	Semantics
DEPICT	√	+++	++	D and G	√		
DCSPC		++	+++	D and G			√
SPC		++	++	D			
N2D		+++	+++	D and G			
DSSC-EAGF		++	++	D and G	√		√
MEDEC		+	++	D	√		√
JULES		+	+	D	√	√	
DAC		++	+++	D		√	
IMSAT	√	++	++	D	√	√	√
RDKM		+++	+++	D			
DeepCluster	√	++	++	D		√	
IIC	√	+++	++	D	√		√
SimCLR	√	++	+	D	√	√	√
MoCo	√	+	+	D	√	√	√
SwAV	√	+	++	D	√	√	√
SCAN	√	++	++	D	√	√	√
NNM	√	++	++	D	√	√	√
PCL	√	++	++	D	√	√	√
BYOL	√	+	++	D	√	√	√
SPICE	√	++	++	D	√	√	√
PICA	√	++	++	D	√	√	√
IDFD	√	++	++	D			√
SIMSIAM	√	++	++	D	√		√
ProPos	√	+	++	D		√	√
SDNE		++	++	D	√		
DNGR		++	+	D			
VGAE		++	+	D and G			√
GAT		+	++	D			√
ARGA		+	+	D			√
DAEGC		+	+	D			√
AGE		+	+	D			√
SDCN		+	++	D			√
JANE	√	+	+	D and G			√
DFCN		+	+	D			√
SLAPS	√	+	+	D			√
DCRN	√	+	+	D			√
SCAGC	√	+	+	D			√
SSGNN		+	+	D			√
GC-SEE		+	+	D			√
DCAHR		+	+	D			√
AHDDC	√	+	+	D			√
SCGC	√	+	++	D			√
VaDE-GMVAE		+	++	D and G			
ClusterGAN		+	+	D and G	√		√
BigBiGAN		+	+	D and G	√		√
SIMI-ClusterGAN	√	+	+	D and G	√		√
EDCN	√	++	++	D and G	√		√

mechanism to create a queue of negative examples and a memory bank. Maintaining and updating representations in the memory bank may be both computationally expensive and challenging. PICA introduces a partition uncertainty index to enhance its focus on cluster semantics and three operations including color jitters, random rescale, and horizontal flip are adopted for data augmentation and perturbations. This improvement comes at the expense of more training complexity. Additionally, the performance of PICA is dependent on the selected backbone model. ProPos combines the advantages of both contrastive and non-contrastive methods, avoiding class collision issues. The approach needs a complex loss function and careful management of the balance between these techniques. This is done at a reasonable additional computational cost (about 10%) compared to the other approaches such as BYOL [114].

To summarize, the success of CNN approaches relies on a weak-supervision process but they cannot be applied without pre-processing work and require skills in the selection or design of the data augmentation scheme, the associated pretext task, and the clustering loss function.

4.5.3. GNN approaches

Contrasted with alternative models, graph approaches based on neural networks are gaining prominence in the realm of deep clustering. Despite the plethora of proposed variants in recent years, the majority of top methods amalgamate the use of an autoencoder (with its variations) for data reconstruction and incorporate techniques to streamline the representation of the initial graph while preserving its semantics. Relative to other deep clustering methods, graph-based techniques pose complexity for non-expert users. The latest approaches seamlessly integrate adversity mechanisms and self-supervision. As a result, any improvement in the outcomes adds to the original complexity, especially with the integration of these new mechanisms specifically devised to enhance performance. Furthermore, it is hard to investigate various deep network models due to their computational training time. To train convolutional graph neural networks, it is usually required to store both the complete graph data and the intermediate states of all nodes in memory. The full-batch training algorithm for ConvGNNs often faces challenges related to memory overflow, particularly when dealing with graphs containing millions of nodes. Finally, it is hard to design an end-to-end training model between deep feature extraction

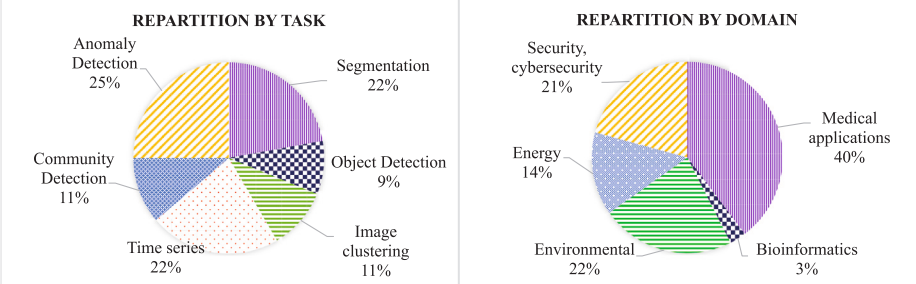


Fig. 18. Deep clustering applications: distribution by task and by domain (Scopus 2018–2023).

and GCN clustering modeling. In addition, the ability to interpret the results of deep learning models on graphs is critical in decision-making problems but interpretability for graph-based deep learning is even more challenging than other black-box models because graph nodes and edges are often heavily interconnected.

4.5.4. Generative approaches

These methods can be used both as discriminative or generative models and have proven to achieve amazing results on a large diversity of data. They include two distinct groups. VAE-based techniques benefit from mathematical foundations and theoretical guarantees because they minimize the variational lower bound on the marginal likelihood of data. However, they often rely on a Gaussian Mixture Model (GMM) prior, and optimizing the numerous hyperparameters involved may not be easy. The second group is based on GAN. These solutions have a high potential and offer great flexibility. The corresponding algorithms are not restricted to spherical-shaped clusters and pay attention to the data semantics. The first generation of GAN-based algorithms such as ClusterGAN do not need specific data preparation, but the best-performing ones do. In the case of SIMI-ClusterGAN the parameters used for the self-augmentation process are left to the user: that means 17 parameters, 5 of them dedicated to regularization. The improved performance is associated with complex architectures and heavy and difficult training as the convergence may be challenging due to the adversarial training itself and, even more challenging when aiming at performing a clustering task. The use of a Siamese network, in the generation of the representation space, eases the burden in the EDCN framework.

5. Applications

Deep clustering methods have achieved significant progress in unsupervised representation learning and clustering. They have been applied to various application domains to perform different tasks. Medical applications often deal with limited and unbalanced data sets. In the field of (Cyber) Security, they are mainly used for anomaly detection while the main issue in the Bioinformatics domain is to cope with unstructured and high-dimensional data. In environmental and energy domains deep clustering algorithms are useful to understand underlying phenomena from large, heterogeneous, and unstructured data. Additional insights for various fields will be provided in the following paragraphs. A synthetic view of deep clustering application domains and tasks is illustrated in Fig. 18. As the needs and data are likely to be different, it may be interesting to analyze the distribution by architectures and domains as shown in Fig. 19.

Concerning architecture, several key observations can be discerned:

- AE/CAEs are very popular and are used for most of the tasks in various fields, with the noticeable exception of image segmentation and object detection. They are still preponderant in image clustering as they are more appropriate for this task that can be done straightforwardly via the latent layer. Autoencoders are the core of all unsupervised deep anomaly detection models [197]. At the time this survey was conducted, they emerged as the dominant deep-learning algorithms in document clustering, collectively having more research studies than the combination of CNNs and RNNs.
  AE/CAEs are also often preferred for security and detection as the models run indifferently with different kinds of data.
- CNNs are preponderant in image analysis and particularly for segmentation and object detection problems due to the greater expertise available in data or this field.
  In document clustering tasks, their presence is comparatively lower. Although RNNs and their variants are the primary options for modeling time series data, CNNs sometimes show better performance in several applications such as [198].
- GNNs are comparatively more emerging but there are some applications in various domains such as NLP (Neural Network Processing), social networks, bioinformatics, and medical sciences. Applications on images are more related to topics such as object detection, question answering, and interaction detection but very few in image classification and clustering. Despite being more complex and computationally intensive than other architectures, it is noteworthy that the number of studies involving GNNs is on the rise in 2023.
- GAN models are used in various domains but more often with image data, especially in the medical field. VAE are used less than GAN methods, which usually generate more accurate data and hence achieve lower performance. Even if GAN models require specific skills, they are used in various applications and their principles are increasingly being integrated into other architectures through hybrid models.

5.1. Overview by task

Six tasks are relatively well-represented, with a slight dominance in anomaly detection that addresses all data types. 43% of tasks exclusively deal with image data, while others are non-exclusive. Nevertheless, this distribution of works does not precisely mirror the levels of innovation. Advancements in time series analysis are still in the early stages, accounting for only 22% of studies, in contrast to other areas of progress [35]. It is worth noting that Natural Language Processing is utilized across various tasks either alone or in hybridized forms.

5.1.1. Anomaly detection

Anomaly Detection, also referred to as Outlier Detection or Novelty Detection, represents a methodological approach employed for the discrimination of atypical instances or patterns within a dataset. Previous to the appearance of deep clustering techniques, density-based clustering methodologies had intensively acknowledged and faced the challenge of noise inherent to clustering. This recognition subsequently laid the foundation for the development of a subset of anomaly detection techniques based on density-based principles [199]. Deep clustering is now showing substantial promise in improving the building of an enhanced clustering space for the aim of anomaly detection. In contrast

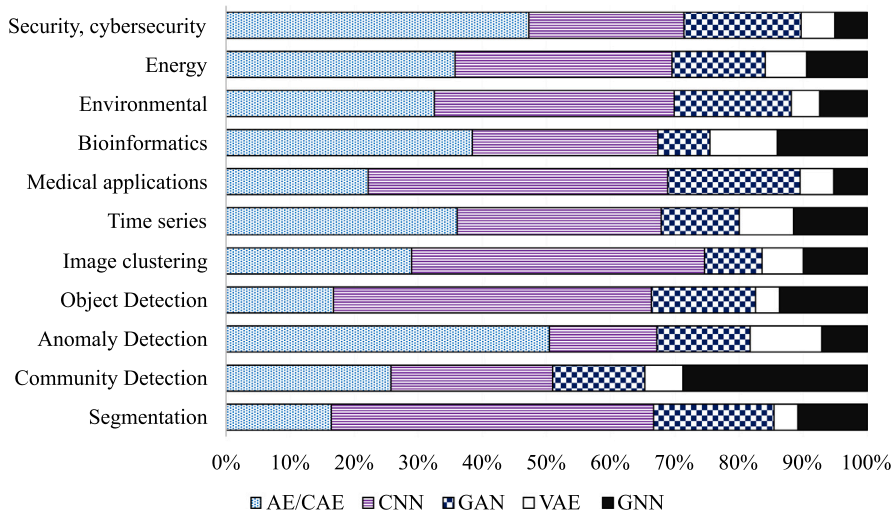


Fig. 19. Distribution by architecture and domain (Scopus 2018–2023).

to the traditional technique of post-clustering anomaly detection, recent efforts have focused on a more integrated approach [197,200–202] to reduce the impact on clustering, and anomaly detection can be further improved with better clustering results. The major goal of this paradigm is to identify and subsequently eliminate anomalous instances within a unified framework, to limit their disruptive impact on the clustering process.

In time-series data, anomalies refer to data points at specific time steps exhibiting unexpected behaviors that deviate significantly from their preceding time steps. Anomaly detection in multivariate time series data, as highlighted by Choi (2021) [203], presents a particular challenge due to the need to simultaneously account for temporal dependencies and inter-variable relationships, with many existing methods tailored to specific use cases. Numerous methods, typically integrating a variety of architectures (CNN, AE, LSTM, …), have emerged for specialized applications like fraud detection, cyber-intrusion detection, medical use, IoT, sensor networks, and video anomaly detection [204]. Graph anomaly detection [202] with deep learning has also received growing attention recently. These deep-learning methods have garnered significant attention in recent years due to their superior performance. However, nearly all the leading methods especially for video anomaly detection depend on large-scale training datasets, leading to extended training times. In addition, these approaches often exhibit an ad hoc nature. Since these methods are evaluated on diverse datasets, it is difficult to have a universal meta-analysis of their empirical performance [201]. Despite the substantial progress in deep learning across various machine learning domains, there remains a deficiency of generic deep learning methods for anomaly detection [197,205].

5.1.2. Segmentation

Image segmentation stands as one of the most important approaches for simulating human understanding of images, with the primary goal of dividing pixels into distinct and non-overlapping regions [206]. Segmentation problems can be classified into class-based problems, which use class labels to identify object classes, and partition-based problems, which segment input without class labels [207]. In the literature, various segmentation techniques rooted in deep learning have been presented and documented [208]. Deep clustering has been effectively applied to image segmentation, utilizing the clustered regions to generate comprehensive scene representations [209]. Xia and Kulis presented W-Net [210], an unsupervised image segmentation model based on two concatenated U-net architectures. As a result, the training technique is complex. A deep clustering model is developed [211] for unsupervised image segmentation, which comprises an autoencoder as a feature transformation subnetwork and a differentiable deep clustering subnetwork for dividing the image space into different clusters.

Significant attention has also been given to 3D and video segmentation in the field of deep clustering, with potential applications in medical image analysis, autonomous driving, robotics, and augmented reality [212]. An unsupervised segmentation method for 3D medical images is introduced in [213], addressing the challenge of supervised learning in handling large amounts of manually annotated data. The method consists of two phases: learning deep feature representations using joint unsupervised learning (JULE) and applying K-means to the deep representations, extending JULE to 3D medical images, and projecting cluster labels to the target image. Extending the success of deep learning-based image segmentation techniques to the video domain has become a recent research focus in computer vision [214]. While the most straightforward strategy is the naive application of an image semantic segmentation model in a frame-by-frame manner, the current research trend leans towards exploiting cross-frame relations. Almost all existing unsupervised learning-based video segmentation models are based on self-supervised learning methods, where the prior knowledge Z refers to pseudo labels derived from intrinsic properties of video data [215]. As a recent example, a two-stage framework called Improved Instance Contrastive Learning (CL) with Deep Clustering (ICDC) is proposed in [216]. By using Instance-CL and unsupervised clustering, this method can learn temporal video representations with high intra-class compactness. The approach introduces a consistency-preserving sampling strategy focusing on motion dynamics and uses K-means clustering to generate pseudo-labels for training the encoder.

5.1.3. Times series

Time series data is a series of data points recorded at regular intervals, and time series clustering methods are used to divide this data into segments with similar patterns. These segments have various analytical uses, such as exploring temporal patterns, reducing dimensionality, detecting outliers, anomaly detection [203], and performing similarity searches. However, traditional clustering algorithms face challenges [217] such as noise, high dimensionality, and high feature correlation. To overcome these challenges, deep learning methods can be designed to disentangle data manifolds and deal with learned features instead of raw data. Deep clustering allows neural networks to extract similar patterns in lower-dimensional space and find idealistic representative centers for distributed data. Since 2018, Deep Time-Series Clustering (DTSC) has received particular attention from different kinds of network architectures [36], such as deep auto-encoder (DAE), deep convolutional auto-encoder (DCAE), and recurrent neural networks (RNNs), including RNN auto-encoder (RNN-AE) or seq2seq

auto-encoder (S2S-AE). Neural network-based techniques like Long Short Term Memory (LSTM) [218] and their autoencoder-based variations, Generative Adversarial Networks (GANs), and attention networks need further exploration for formulating classical problems in time series clustering and data analysis. Lafabregue's study [35] suggests that simple autoencoder architecture with reconstruction-based pretext losses is the best for time series processing. In contrast, sophisticated deep learning frameworks primarily designed for image clustering do not yield performance improvements. To better address the temporal dimension, there is a need for adaptations or the development of novel approaches.

5.1.4. Object detection

Object detection is a crucial computer vision task that identifies specific object classes within digital images, aiming to create computational models for various computer vision applications. Deep learning approaches have greatly improved object detection technology, resulting in considerable advances in object detection and tracker performance. Several deep learning approaches have been proposed in the literature [219]. One of the most promising object detection techniques based on deep learning is the YOLO algorithm [220]. YOLO (You Only Look Once) is a real-time object detection technique that uses a single Convolutional Neural Network to assign probabilities to detected objects. Deep clustering-based unsupervised object detection techniques have also been introduced in the literature. While some of these techniques are based on deep generative models [221], others use variational autoencoders (VAE) [222]. Recently, Wang et al. [223] proposed Cut-and-LEaRn (CutLER), a simple and self-supervised approach for training unsupervised object detection. It generates coarse masks for multiple objects in an image, learns a detector on these masks, and self-trains the model on predictions. Some specific tasks of object detection are salient object detection, face detection, and pedestrian detection. The exploration of single-stage point-based 3D object detection networks remains relatively unexplored [224], even though several very recent methods based on GNNs and GANs have shown promising perspectives [225–227].

5.1.5. Image/Text clustering

Image. Supervised learning requires large labeled datasets, but manual labeling is expensive and limits applicability. Unsupervised methods are needed for image clustering relying on image feature representations and are challenging due to real-world image diversity. Traditional methods separate feature learning and clustering, while recent deep clustering methods usually mine specific data correlations as supervision signals during feature learning. The problem is that differences among data correlations are far from trivial, especially in unsupervised circumstances. Deep joint clustering, which combines representation learning with clustering, has potential. However, most existing methods suffer from poor discriminability in complex images and performance bottlenecks due to the lack of supervision [228]. On a brighter note, more recent advances based on self-supervised learning show promise.

Text. Text clustering involves the task of categorizing a collection of texts into groups, with the aim of ensuring that texts within the same group exhibit higher similarity compared to those in different groups. It is essential to many real-world applications, such as text mining, online text organization, and automatic information retrieval systems in various fields such as industry and medicine. Manual text clustering is a time-consuming and labor-intensive process. Traditional machine learning models often follow a two-step workflow: they begin by preprocessing and extracting meaningful features from text documents (text representation), which serves as the foundation for subsequent processing within a conventional classification framework. Common feature representation models include Bag-of-Words (BoW), Word and Sentence Embedding, Term Frequency-Inverse Document Frequency (TF-IDF), and more. Deep learning models have revolutionized various Natural Language Processing (NLP) tasks, enhancing language modeling for more extensive context aiming to learn feature representations and perform classification in an end-to-end manner. They possess the potential to uncover hidden patterns in data and exhibit superior transferability across different projects. Numerous prominent frameworks within natural language processing are grounded in RNN models like LSTM, GRU, and their bidirectional variants, as well as more recent advancements such as GNNs [229] and BERT which needs to be fine-tuned for specific tasks [39]. Drawing inspiration from image data research, the field has seen the emergence of self-supervised contrastive learning (SCL) as a novel approach, as recently introduced in NLP, demonstrating performance levels approaching those of supervised learning [195].

5.1.6. Community detection

Communities in networks reflect high-order proximities, such as similar opinions and behaviors. Community detection holds numerous promising applications, including the identification of like-minded users for recommendation systems, the detection of shared research interests in collaborative networks, the discovery of functionally associated proteins in protein-protein interaction networks, and the revelation of concealed relationships among network nodes. Essentially, community detection is most useful in situations when the goal is to identify closely related groups of people or objects that share characteristics. The following are some popular applications for community detection: Public Health, such as preventing epidemic spreading [230]. Missing link prediction and link analysis [231,232], which includes finding possible collaborators in cooperation networks or identifying hidden links between members of a criminal network, etc. Politics, community detection is used to track how particular politicians or political beliefs affect a particular social group. Deep learning algorithms for community detection have advanced significantly in recent years, owing to their benefits in processing high-dimensional network data. The most promising DL-based methods for community discovery are Graph Neural Networks (GNNs), where nodes are represented as single instances in the low-dimensional space. For this reason, it is getting harder to distinguish between the community detection and graph clustering techniques used today [233]. Numerous fields (recommendation systems, social network analysis, biochemistry...) and applications have already adopted graph clustering techniques even if few works integrate the community embedding into a deep learning model [234,235].

5.2. Overview by domain

The statistics undergo rapid changes annually. Although fields related to image data were initially predominant, the distribution has now become more balanced. Nevertheless, medical applications remain the most representative, often relying on image data. The field of bioinformatics has the least representation. Currently, only the fundamentals of deep learning are actively employed in bioinformatics research, particularly for supervised learning tasks. However, there are emerging works [51] for unsupervised tasks, often utilizing autoencoder-based architectures.

5.2.1. Medical applications

Through supervised learning, Convolutional Neural Networks (CNNs) have made important advances in the field of medical image processing [236]. However, manually annotating datasets is often a labor-intensive operation requiring specialized medical knowledge, making it difficult to use efficiently in real-world situations. Deep clustering algorithms have recently been presented to automatically categorize large-scale medical images [237]. Yan et al. [238] introduces a deep clustering approach that obtains discriminative embeddings and an initial clustering prediction from annotation-free WSI patches in a contrastive environment. Within the field of biological science, single-cell RNA sequencing (scRNA-seq) [239] produces a cell-gene

matrix as output, allowing the analysis of cellular populations, their behavior, and the possible identification of novel cell types. Deep learning has demonstrated its potential with promising outcomes across a range of tasks, including image classification, tissue classification, cancerous cell characterization, detection, and segmentation. Notably, the utilization of auto-encoders and Deep Belief Networks (DBN) has yielded significant insights. The predominant use of the intricate CNN and RNN models, while highly effective, still presents challenges in terms of implementation [240].

5.2.2. Environmental

The environmental challenges faced have been widely recognized. For instance, the limited data and understanding regarding the impact of various activities on surface water ecosystems present a critical challenge to water resource managers. Air pollution contributes to the pollution of the atmosphere and the deterioration of the environment and is responsible for many diseases causing a major threat to human public health. Extensive research into Machine Learning algorithms has been undertaken to gain a deeper insight into highly intricate environmental phenomena and to improve our monitoring and prevention efforts. Deep Learning algorithms have also become pivotal in addressing complex issues within environmental systems, ranging from earthquake prediction and weather forecasting to sustainability and environmental preservation [241]. AE/CAEs, CNNs, and Generative models are the dominant algorithms in environment and water management while the fusion of spatial and temporal data has been extensively studied using hybrid CNN and long short-term memory (LSTM) models. ConvNetQuake [242] is a CNN model designed for the detection and localization of earthquakes using seismogram data. Predicting climate and weather changes is crucial for crop planting and emergency preparedness. Deep learning, can effectively predict rainfall, aiding in crop location. A comparative study between Long Short-Term Memory (LSTM) neural networks and Wavelet Neural Network (WNN) models was conducted for spatio-temporal prediction of rainfall and runoff time-series data [243]. In this domain, it is crucial [244] that data solutions are crafted by individuals who possess a deep comprehension of the problems and context, rather than solely by those well-versed in algorithms. This is especially pertinent when addressing the challenges associated with integrating diverse data sources and data streams resulting from emerging IoT and sensing technologies, including the implementation of autonomous data quality checks.

5.2.3. Security and cybersecurity

A set of tools and techniques known as cybersecurity works to protect computer networks, systems, software, and data against threats, illegal access, alteration, and damage. In this field, deep learning is becoming more and more significant since it makes a wide range of applications possible. The following are some popular deep learning applications for cybersecurity: Network intrusion detection, which has seen the development of several DL-based techniques in recent years. More specifically, deep learning approaches have been used to generate models of normal behavior and identify deviations that traditional methods are unable to recognize, thereby improving generalization capabilities for advanced attack detection [245,246]. Malware detection and analysis, or malicious software, is a generic term for any program designed by nefarious individuals to cause harm to networks, devices, and systems. Many machine learning-based solutions that address scalability by automating different phases of malware detection and classification procedures have been developed. Still, their efficiency is limited by high false positive rates, which make them not accurate. Researchers have increasingly concentrated on deep learning-based systems to overcome this issue [247,248]. Deep learning-based algorithms have been shown to categorize malware significantly faster than human analysis while maintaining excellent accuracy rates.

5.2.4. Energy

Stable supply and efficient consumption of energy are essential to cope with rapid climate changes and resources. Deep Learning has broad applications in many fields related to energy systems. Among the most widely used and developing uses of deep learning for energy systems are the following: Energy consumption and demand forecast, in which RNN, LSTM, were employed in regression tasks [249]. Predicting the output power of solar systems is one prominent use, an interesting case study in South Africa examined the global solar radiation prediction [250]. For this analysis, Artificial Neural Networks (ANN) was used, more specifically the General Regression Neural Network (GRNN). A specific hybrid optimization technique was presented in [251] to optimize daily operations in building energy and storage systems. The approach uses a Deep Neural Network (DNN) model to forecast the integrated cooling tower systems' ideal performance. An advanced load profiling framework has been introduced in [252], utilizing embedded deep clustering techniques inspired by DEC, IDEC, and DynAE. This unified end-to-end unsupervised learning framework facilitates the automatic transformation of smart meter data into cluster-friendly representations while preserving essential data characteristics. Notably, it outperforms state-of-the-art methods, delivering superior load profiles, meaning distinct and well-defined load curves for various consumer groups.

6. Issues and challenges

Despite the success and significant advantages of deep clustering, the clustering process is still frequently applied using traditional algorithms in many industries which often leads to degenerate solutions or leaves no space for further improvement. Currently, few studies propose comprehensive solutions based on deep architectures that can be applied straightforwardly. In this section, issues and challenges are discussed in three major points.

6.1. Make the deep clustering more user friendly

Deep learning algorithms for clustering are not really mature. They are often applicable to well-known benchmark datasets only and are not yet considered useful standard machine learning tools. More importantly, these methods require a vast amount of data to train, need extensive computational power, are time-consuming, and can be unstable. This is mainly due to the complexity of CNN architectures themselves and the huge number of parameters that need to be optimized. In addition, most of the deep clustering schemes require additional parameters to be tuned, requiring real expertise and often an ad-hoc selection. They are not user-friendly and generally put off investigators. To have real-world applicability, clustering applications need to have as few hyper-parameters as possible, based on simpler and more automatic schemes. One challenge should consist of providing a parsimonious and accessible clustering processing scheme that incorporates deep learning-style feature extraction but without the complex hyper-parameter tuning procedure. How can one bridge the gap between deep learning-based clustering methods and widely available standard clustering techniques?

6.2. Better clustering schemes

The clustering problem remains ill-defined. However, three basic notions of what a cluster is have led to three main types of algorithms and a myriad of heuristics allowing major advances, especially in low dimensions. If a cluster is defined by its center and a basin of attraction then distance is the central concept. It is also possible to define a cluster as a dense area separated from another cluster by a sparsely populated zone; in this case, density is the key idea. Finally, a third definition is based on a set of connected points, in which case the neighborhood is of prime concern. In deep clustering, the classical

way consists in an integrated approach that performs clustering and representation learning simultaneously. Then, they benefit from each other. The clustering provides feedback to the transformation, and in return, the non-linear transformation can alter the embedded space to improve the clustering. These works have achieved impressive results but they are often based on classic center-based, divergence-based, or hierarchical clustering formulations and thus inherit some limitations from classical methods. In particular, some algorithms require setting the number of clusters a priori. The optimization procedures they employ involve discrete reconfigurations of the objective, such as discrete reassignments of data points to centroids or the merging of putative clusters in an agglomerative procedure. These methods, in general, do not take into account the local information of clusters and do not consider that points with different densities should play different roles in density-based clustering techniques. With non-deep-clustering techniques, Although imperfect, many heuristics have been imagined to face standard clustering issues encountered with unfriendly data organizations, to automatically determine the cluster number... Deep clustering methods are more challenging as they involve a complex procedure that is significantly affected by the choice of a latent representation and the latter has to be suitable for clustering. There is then the need to integrate an optimization procedure that simultaneously makes it possible to generate clusters while handling representation learning to avoid its blindness. Integrating improved clustering schemes with this constraint is not easy, and points out a real challenge for researchers.

6.3. Better exploratory analysis of big and complex data: utopia?

In all clustering schemes, the question of how to measure distance or similarity appropriately is crucial for the performance of clustering methods. Unfortunately, as the classical notions of distances/density are not valid with deep clustering schemes, there is the need to project in a latent space, and this projection can be done only under some assumptions and/or guidance. The goal is to map the original data into a new semantic latent space so that in this latent space, one can determine natural partitions. Clustering is unsupervised-based but unsupervised representation learning is an ill-defined problem if the downstream task can be arbitrary. With "pure" unsupervised methods the learned representations lack discriminability, especially for heterogeneous data distributions, and the performance often encounters a bottleneck due to the lack of supervision information. Hence, most of the current methods use strong inductive biases and modeling assumptions. Implicit or explicit supervision remains a key enabler and, depending on the mechanism for enforcing supervision, different degrees are required. Semi-supervised approaches, pseudo-labeling, and pretext tasks are usually included to transform an unsupervised problem into a supervised one. As an example, state-of-the-art contrastive methods are trained by reducing the distance between representations of different augmented views of the same data patterns and increasing the distance between representations of augmented views of different data patterns. These methods need a careful treatment of negative pairs by either relying on large batch sizes, memory banks, or customized mining strategies to retrieve the negative pairs. Their performance critically depends on the choice of pattern augmentations. In fact, the DNA of these approaches can be roughly assimilated to distance metric learning approaches as they are investigated in order to reflect high-level semantics concepts, such as whether data look alike or differ. In sum, most of the deep clustering approaches have obtained promising results based on pseudo-supervision strategies that rely heavily on human expertise. Obviously, they have shifted from their initial goal of knowledge discovery.

Is it utopian to seek hidden patterns in high-dimensional space without any or little prior information? This is the central question. Learning useful representations with little or no supervision is a key challenge in artificial intelligence. For deep clustering, one can assume a little supervision as is done with non-deep clustering methods but on the condition that is useful for many real-world problems simultaneously, e.g. assume a given level of genericity. Self-supervised learning is promising but at the present time, it is complex and insufficiently self-tuning, e.g., generating appropriate data augmentation to match with the data semantics alone. The challenge is how to guide the transformation between the spaces to highlight different representations that have different semantic contents from which natural partitions can be extracted. As finding an optimal augmentation strategy for the data is non-trivial and there is no guarantee of quality/efficiency, better data augmentation strategies should be obtained by learning or searching augmentation strategies. The challenge should be to automatically customize a set of optimal data augmentation schemes for a given data set. Thus, the next milestone in deep clustering should be a real advance in the knowledge discovery task by reducing the current human role of supervision to be more on the core idea of clustering schemes and improve efficiency.

7. Conclusion

In this paper, a survey of the state of the art of deep clustering techniques was conducted. Without being fully self-contained, the most important baseline techniques and tricks involved in deep clustering were revisited to facilitate the reading for non-experts. Since the pioneer proposals, deep clustering has received increasing attention from the AI community as neural network models have been demonstrated to be powerful tools for learning representations and extracting features. The core reason for deep clustering originates from the curse of dimensionality issues that hinder the overall research advances in clustering. Impressive results were recently achieved in the past years via the generation of novel learning strategies and smart tricks. A part of the success inherits from the effort made by researchers to face issues of supervised learning which requires a huge number of human-annotated labels. Today, deep clustering approaches clearly outperform non-deep clustering ones when handling datasets with high dimensional spaces.

The paper proposed an overview of deep clustering techniques via a simple taxonomy based on architectures, where the reader is first provided with the essential information. A detailed and comprehensive review of each family is then proposed in which the motivations and mathematical representations of different kinds of algorithms are included. Deep clustering methods of the state-of-the-art showing significant classification results are highlighted and evaluated via the most benchmarked data sets. This was done by aggregating the scores of the experimental result sections of different papers. A multicriteria evaluation is suggested to assist investigators in choosing the most suitable algorithm. This evaluation will aid in the selection process by taking into account multiple criteria. Lastly, we synthesize the dissemination of the methods by application domains and tasks while suggesting open challenges for researchers in the field.

Remaining issues. Except for some rare very recent methods that appear very promising, most unsupervised deep clustering methods generate excellent results on simple datasets but encounter more difficulties on complex or unfriendly datasets.

While the field of computer vision has seen significant growth in advanced frameworks for deep clustering, there has been a limited effort to apply these state-of-the-art techniques to the complex domain of time series data. Time series data presents a unique challenge due to its requirement to consider both temporal dependencies and relationships between variables simultaneously.

Despite being successful and having significant advantages, traditional clustering algorithms often result in degenerate solutions, leaving no room for further improvement. The question of unsupervised representation is only partially handled. It remains difficult, although central for the discovery aspect to produce different embeddings where semantically similar data are close, while semantically different data are

far apart. AE family architectures have proven relevant for relatively simple data sets using nonlinear projective approaches. They were often considered until 2020 to be the state of the art against CNN-based approaches. However, these assumptions are based on strong data reconstruction and may not reveal complex semantics.

Despite their inherent complexities, generative approaches also appear promising as they are capable of re-producing training data distribution and are concise in that they seek common causes for different data and share information between them. Data generated by VAE are usually not as accurate as GAN, while GAN lacks the encoder–decoder architecture, and is hard to train.

Perspectives. For the past 2–3 years, innovative CNN-based approaches employing intelligent contrastive schemes coupled with data augmentation have consistently outperformed Autoencoder (AE) schemes, particularly in the realm of image processing, thereby altering the landscape of leadership in this domain. They have demonstrated their ability to capture the strongest semantics of the original data. This is a strong advance and it can be claimed that a milestone has been reached. However, there is a new challenge since the efficiency is highly dependent on the pretext task and the veracity of the data augmentation. The state-of-the-art solutions are data-dependent and the most remarkable results are in the field of image analysis.

Promising advancements in Graph Neural Network (GNN)-based models, incorporating contrastive and adversarial training techniques inspired by image processing, are emerging as particularly effective, albeit intricate. This is particularly evident in the domains of community detection and text clustering.

While all these novel schemes can effectively improve the model, the counterpart is the increase in the complexity of training models as well as the computational cost.

Clustering algorithms are inherently unsupervised. However, the success in deep clustering can ultimately be attributed to self-supervised learning, where algorithms are guided on which parts of the data to focus on. It is, therefore, a form of indirect supervision. Is it illusory to do otherwise?

Can Deep clustering really work without a supervision mechanism when handling complex data structures? There is rarely a one-size-fits-all solution to group the data, and the chance of grouping them as expected by humans is very small. The way to include pseudo-supervision mechanisms is the current trend to capture data semantics and should be investigated in greater depth. However, the clustering approach is not truly exploratory as pseudo supervision is manually oriented. Surprisingly, there is little connection between deep feature selection studies and deep clustering processes even if all aim at facing the issue of the curse of dimensionality differently. The feature selection process is sometimes integrated into deep clustering processes, particularly when employing specific regularization tasks. It can be a virtuous direction for future studies to consider feature selection at different levels of abstraction, and complementary to the core clustering process.

The core challenge concerns a better “game” between learning semantically meaningful features and clustering. How the semantic confidence of clusters can be significantly improved remains an open question. The last successes in deep clustering are due to a “pseudo-supervision” strategy involving extensive use of data augmentation. One strong novel advance should be to automatically customize a set of optimal data augmentation schemes for a given data set. Ultimately, the ideal promise for deep clustering would be to develop end-to-end frameworks able to discover different natural partitions highlighting various hidden structures under very few parameters to tune and reduce the gap between traditional clustering approaches and deep learning ones.

CRediT authorship contribution statement

Frédéric Ros: Writing – review & editing, Writing – original draft, Methodology, Formal analysis, Conceptualization. **Rabia Riad:** Writing – review & editing, Writing – original draft, Visualization, Supervision, Investigation. **Serge Guillaume:** Writing – review & editing, Writing – original draft, Validation, Supervision, Formal analysis.

Declaration of competing interest

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Data availability

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

References

[1] A.E. Ezugwu, A.M. Ikotun, O.O. Oyelade, L. Abualigah, J.O. Agushaka, C.I. Eke, A.A. Akinyelu, A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects, Eng. Appl. Artif. Intell. 110 (2022) 104743.
[2] J.G. Dy, C.E. Brodley, Feature selection for unsupervised learning, J. Mach. Learn. Res. 5 (Aug) (2004) 845–889.
[3] S. Khalid, T. Khalil, S. Nasreen, A survey of feature selection and feature extraction techniques in machine learning, in: 2014 Science and Information Conference, 2014, pp. 372–378.
[4] T. Dokeroglu, A. Deniz, H.E. Kiziloz, A comprehensive survey on recent metaheuristics for feature selection, Neurocomputing 494 (2022) 269–296.
[5] L.K. Saul, S.T. Roweis, An introduction to locally linear embedding, 2000, unpublished. Available at: https://cs.nyu.edu/~roweis/lle/papers/lleintro.pdf.
[6] M. Balasubramanian, E.L. Schwartz, The isomap algorithm and topological stability, Science 295 (5552) (2002) 7.
[7] T. Kohonen, The self-organizing map, Proc. IEEE 78 (9) (1990) 1464–1480.
[8] Y. Munakata, J. Pfaffly, Hebbian learning and development, Dev. Sci. 7 (2) (2004) 141–148.
[9] G.E. Hinton, Deep belief networks, Scholarpedia 4 (5) (2009) 5947.
[10] F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Schölkopf, O. Bachem, Challenging common assumptions in the unsupervised learning of disentangled representations, in: International Conference on Machine Learning, 2019, pp. 4114–4124.
[11] D.E. Rumelhart, J.L. McClelland, A general framework for parallel distributed processing, in: Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations, 1987, pp. 45–76.
[12] J. Xie, R. Girshick, A. Farhadi, Unsupervised deep embedding for clustering analysis, in: International Conference on Machine Learning, 2016, pp. 478–487.
[13] Z. Li, F. Liu, W. Yang, S. Peng, J. Zhou, A survey of convolutional neural networks: Analysis, applications, and prospects, IEEE Trans. Neural Netw. Learn. Syst. 33 (12) (2022) 6999–7019.
[14] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, in: International Conference on Machine Learning, 2020, pp. 1597–1607.
[15] K. Ohri, M. Kumar, Review on self-supervised image recognition using deep neural networks, Knowl.-Based Syst. 224 (2021) 107090.
[16] M. Gori, G. Monfardini, F. Scarselli, A new model for learning in graph domains, in: Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005. Vol. 2, IEEE, 2005, pp. 729–734.
[17] F. Scarselli, M. Gori, A.C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, IEEE Trans. Neural Netw. 20 (1) (2008) 61–80.
[18] C. Gallicchio, A. Micheli, Graph echo state networks, in: The 2010 International Joint Conference on Neural Networks (IJCNN), IEEE, 2010, pp. 1–8.
[19] F. Harary, The determinant of the adjacency matrix of a graph, SIAM Rev. 4 (3) (1962) 202–210.
[20] P. Goyal, E. Ferrara, Graph embedding techniques, applications, and performance: A survey, Knowl.-Based Syst. 151 (2018) 78–94.
[21] S.E. Schaeffer, Graph clustering, Comput. Sci. Rev. 1 (1) (2007) 27–64.
[22] S. Dongen, A Cluster Algorithm for Graphs, CWI (Centre for Mathematics and Computer Science), 2000.
[23] U. Brandes, M. Gaertler, D. Wagner, Experiments on graph clustering algorithms, in: European Symposium on Algorithms, Springer, 2003, pp. 568–579.
[24] M. Xu, Understanding graph embedding methods and their applications, SIAM Rev. 63 (4) (2021) 825–853.

[25] D. Kuang, C. Ding, H. Park, Symmetric nonnegative matrix factorization for graph clustering, in: Proceedings of the 2012 SIAM International Conference on Data Mining, SIAM, 2012, pp. 106–117.

[26] E. Müller, Graph clustering with graph neural networks, J. Mach. Learn. Res. 24 (2023) 1–21.

[27] H. Yang, J. Wang, R. Duan, C. Yan, DCOM-GNN: A deep clustering optimization method for graph neural networks, Knowl.-Based Syst. 279 (2023) 110961.

[28] M. Ciortan, M. Defrance, GNN-based embedding for clustering scRNA-seq data, Bioinformatics 38 (4) (2022) 1037–1044.

[29] X. He, B. Wang, R. Li, J. Gao, Y. Hu, G. Huo, B. Yin, Graph structure learning layer and its graph convolution clustering application, Neural Netw. 165 (2023) 1010–1020.

[30] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial networks, Commun. ACM 63 (11) (2020) 139–144.

[31] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, A.A. Bharath, Generative adversarial networks: An overview, IEEE Signal Process. Mag. 35 (1) (2018) 53–65.

[32] D. Saxena, J. Cao, Generative adversarial networks (GANs) challenges, solutions, and future directions, ACM Comput. Surv. 54 (3) (2021) 1–42.

[33] D.P. Kingma, M. Welling, et al., An introduction to variational autoencoders, Found. Trends® Mach. Learn. 12 (4) (2019) 307–392.

[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).

[35] B. Lafabregue, J. Weber, P. Gançarski, G. Forestier, End-to-end deep representation learning for time series clustering: a comparative study, Data Min. Knowl. Discov. 36 (1) (2022) 29–81.

[36] A. Javed, B.S. Lee, D.M. Rizzo, A benchmark study on time series clustering, Mach. Learn. Appl. 1 (2020) 100001.

[37] F.M. Shiri, T. Perumal, N. Mustapha, R. Mohamed, A comprehensive overview and comparative analysis on deep learning models: CNN, RNN, LSTM, GRU, 2023, arXiv preprint arXiv:2305.17473.

[38] R. Mao, G. Chen, X. Zhang, F. Guerin, E. Cambria, GPTEval: A survey on assessments of ChatGPT and GPT-4, 2023, arXiv preprint arXiv:2308.12488.

[39] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, 2018, arXiv preprint arXiv:1810.04805.

[40] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, 2020, arXiv preprint arXiv:2010.11929.

[41] J. Yang, J. Liu, N. Xu, J. Huang, Tvt: Transferable vision transformer for unsupervised domain adaptation, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2023, pp. 520–530.

[42] P. Mitra, C. Murthy, S.K. Pal, Unsupervised feature selection using feature similarity, IEEE Trans. Pattern Anal. Mach. Intell. 24 (3) (2002) 301–312.

[43] R. Xu, D. Wunsch, Survey of clustering algorithms, IEEE Trans. Neural Netw. 16 (3) (2005) 645–678.

[44] A. Ahmad, S.S. Khan, Survey of state-of-the-art mixed data clustering algorithms, IEEE Access 7 (2019) 31883–31902.

[45] E. Aljalbout, V. Golkov, Y. Siddiqui, M. Strobel, D. Cremers, Clustering with deep learning: Taxonomy and new methods, 2018, arXiv:1801.07648.

[46] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, J. Long, A survey of clustering with deep learning: From the perspective of network architecture, IEEE Access 6 (2018) 39501–39514.

[47] J. Schnellbach, M. Kajo, Clustering with deep neural networks–An overview of recent methods, Network 39 (2) (2020) 39–43.

[48] A. Chefrour, L. Souici-Meslati, Unsupervised deep learning: Taxonomy and algorithms, Informatica 46 (2) (2022) 151–168.

[49] S. Zhou, H. Xu, Z. Zheng, J. Chen, J. Bu, J. Wu, X. Wang, W. Zhu, M. Ester, et al., A comprehensive survey on deep clustering: Taxonomy, challenges, and future directions, 2022, arXiv preprint arXiv:2206.07579.

[50] Y. Ren, J. Pu, Z. Yang, J. Xu, G. Li, X. Pu, P.S. Yu, L. He, Deep clustering: A comprehensive survey, 2022, arXiv:2210.04142.

[51] M.R. Karim, O. Beyan, A. Zappa, I.G. Costa, D. Rebholz-Schuhmann, M. Cochez, S. Decker, Deep learning-based clustering approaches for bioinformatics, Brief. Bioinform. 22 (1) (2021) 393–415.

[52] X. Su, S. Xue, F. Liu, J. Wu, J. Yang, C. Zhou, W. Hu, C. Paris, S. Nepal, D. Jin, Q.Z. Sheng, P.S. Yu, A comprehensive survey on community detection with deep learning, IEEE Trans. Neural Netw. Learn. Syst. (2022) 1–21.

[53] P. Dixit, S. Silakari, Deep learning algorithms for cybersecurity applications: A technological and status review, Comp. Sci. Rev. 39 (2021) 100317.

[54] S. Yu, J. Liu, Z. Han, Y. Li, Y. Tang, C. Wu, Representation learning based on autoencoder and deep adaptive clustering for image clustering, Math. Probl. Eng. 2021 (2021) 3742536.

[55] Q. Xuan, X. Li, Z. Chen, D. Xu, S. Zheng, X. Yang, Deep transfer clustering of radio signals, 2021, arXiv:2107.12237.

[56] J. Gao, P. Li, Z. Chen, J. Zhang, A survey on deep learning for multimodal data fusion, Neural Comput. 32 (5) (2020) 829–864.

[57] Y. Bengio, A. Courville, P. Vincent, Representation learning: A review and new perspectives, IEEE Trans. Pattern Anal. Mach. Intell. 35 (8) (2013) 1798–1828.

[58] M. Wang, W. Deng, Deep visual domain adaptation: A survey, Neurocomputing 312 (2018) 135–153.

[59] D. Zhang, J. Yin, X. Zhu, C. Zhang, Network representation learning: A survey, IEEE Trans. Big Data 6 (1) (2018) 3–28.

[60] J. Wang, C. Lan, C. Liu, Y. Ouyang, T. Qin, W. Lu, Y. Chen, W. Zeng, P. Yu, Generalizing to unseen domains: A survey on domain generalization, IEEE Trans. Knowl. Data Eng. (2022) 1.

[61] A. Achille, S. Soatto, Emergence of invariance and disentanglement in deep representations, J. Mach. Learn. Res. 19 (1) (2018) 1947–1980.

[62] M. Mirza, S. Osindero, Conditional generative adversarial nets, 2014, arXiv:1411.1784.

[63] A. Jaiswal, A.R. Babu, M.Z. Zadeh, D. Banerjee, F. Makedon, A survey on contrastive self-supervised learning, Technologies 9 (1) (2021).

[64] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, G.E. Hinton, Big self-supervised models are strong semi-supervised learners, Adv. Neural Inf. Process. Syst. 33 (2020) 22243–22255.

[65] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, A. Joulin, Unsupervised learning of visual features by contrasting cluster assignments, Adv. Neural Inf. Process. Syst. 33 (2020) 9912–9924.

[66] H. Zhong, C. Chen, Z. Jin, X.-S. Hua, Deep robust clustering by contrastive learning, 2020, arXiv:2008.03030.

[67] D. Zhang, F. Nan, X. Wei, S. Li, H. Zhu, K. McKeown, R. Nallapati, A. Arnold, B. Xiang, Supporting clustering with contrastive learning, 2021, arXiv:2103.12953.

[68] Y. Li, P. Hu, Z. Liu, D. Peng, J.T. Zhou, X. Peng, Contrastive clustering, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 8547–8555.

[69] X. Deng, D. Huang, D.-H. Chen, C.-D. Wang, J.-H. Lai, Strongly augmented contrastive clustering, 2022, arXiv:2206.00380.

[70] W. Van Gansbeke, S. Vandenhende, S. Georgoulis, M. Proesmans, L. Van Gool, Scan: Learning to classify images without labels, in: European Conference on Computer Vision, 2020, pp. 268–285.

[71] C. Shorten, T.M. Khoshgoftaar, A survey on image data augmentation for deep learning, J. Big Data 6 (1) (2019) 1–48.

[72] C. Shorten, T.M. Khoshgoftaar, B. Furht, Text data augmentation for deep learning, J. Big Data 8 (1) (2021) 1–34.

[73] M. Bayer, M.-A. Kaufhold, C. Reuter, A survey on data augmentation for text classification, ACM Comput. Surv. 55 (7) (2022) 1–39.

[74] Y. Bengio, J. Louradour, R. Collobert, J. Weston, Curriculum learning, in: Proceedings of the 26th Annual International Conference on Machine Learning, 2009, pp. 41–48.

[75] X. Wang, Y. Chen, W. Zhu, A survey on curriculum learning, IEEE Trans. Pattern Anal. Mach. Intell. 44 (9) (2022) 4555–4576.

[76] K. Ghasedi, X. Wang, C. Deng, H. Huang, Balanced self-paced learning for generative adversarial clustering network, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4391–4400.

[77] C. Hou, F. Nie, X. Li, D. Yi, Y. Wu, Joint embedding learning and sparse regression: A framework for unsupervised feature selection, IEEE Trans. Cybern. 44 (6) (2013) 793–804.

[78] I. Goodfellow, Y. Bengio, A. Courville, Regularization for deep learning, in: Deep Learning, MIT press, Cambridge, MA, USA, 2016, pp. 216–261.

[79] T. Gale, E. Elsen, S. Hooker, The state of sparsity in deep neural networks, 2019, arXiv:1902.09574.

[80] A. Rahangdale, S. Raut, Deep neural network regularization for feature selection in learning-to-rank, IEEE Access 7 (2019) 53988–54006.

[81] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.

[82] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, 2014, arXiv preprint arXiv:1409.0473.

[83] Q. Guan, Y. Huang, Z. Zhong, Z. Zheng, L. Zheng, Y. Yang, Diagnose like a radiologist: Attention guided convolutional neural network for thorax disease classification, 2018, arXiv preprint arXiv:1801.09927.

[84] C. Wang, S. Pan, R. Hu, G. Long, J. Jiang, C. Zhang, Attributed graph clustering: A deep attentional embedding approach, 2019, arXiv:1906.06532.

[85] Y. Chen, L. Liu, V. Phonevilay, K. Gu, R. Xia, J. Xie, Q. Zhang, K. Yang, Image super-resolution reconstruction based on feature map attention mechanism, Appl. Intell. 51 (2021) 4367–4380.

[86] M. Tschannen, O. Bachem, M. Lucic, Recent advances in autoencoder-based representation learning, 2018, arXiv:1812.05069.

[87] A.R. Sajun, I. Zualkernan, Survey on implementations of generative adversarial networks for semi-supervised learning, Appl. Sci. 12 (3) (2022).

[88] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, P. Abbeel, Infogan: Interpretable representation learning by information maximizing generative adversarial nets, Adv. Neural Inf. Process. Syst. 29 (2016) 2180–2188.

[89] S. Mukherjee, H. Asnani, E. Lin, S. Kannan, ClusterGAN: Latent space clustering in generative adversarial networks, in: Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, 2019, pp. 4610–4617.

[90] Z. Jiang, Y. Zheng, H. Tan, B. Tang, H. Zhou, Variational deep embedding: An unsupervised and generative approach to clustering, 2017, arXiv:1611.05148.

[91] N. Dilokthanakul, P.A.M. Mediano, M. Garnelo, M.C.H. Lee, H. Salimbeni, K. Arulkumaran, M. Shanahan, Deep unsupervised clustering with Gaussian mixture variational autoencoders, 2017, arXiv:1611.02648.

[92] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, et al., Bootstrap your own latent-a new approach to self-supervised learning, Adv. Neural Inf. Process. Syst. 33 (2020) 21271–21284.

[93] P. Huang, Y. Huang, W. Wang, L. Wang, Deep embedding network for clustering, in: 2014 22nd International Conference on Pattern Recognition, 2014, pp. 1532–1537.

[94] B. Yang, X. Fu, N.D. Sidiropoulos, M. Hong, Towards k-means-friendly spaces: Simultaneous deep learning and clustering, in: International Conference on Machine Learning, 2017, pp. 3861–3870.

[95] F. Li, H. Qiao, B. Zhang, Discriminatively boosted image clustering with fully convolutional auto-encoders, Pattern Recognit. 83 (2018) 161–173.

[96] X. Guo, L. Gao, X. Liu, J. Yin, Improved deep embedded clustering with local structure preservation, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI '17, 2017, pp. 1753–1759.

[97] R. McConville, R. Santos-Rodriguez, R.J. Piechocki, I. Craddock, N2d:(not too) deep clustering via clustering the local manifold of an autoencoded embedding, in: 2020 25th International Conference on Pattern Recognition (ICPR), 2021, pp. 5145–5152.

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

[99] X. Guo, X. Liu, E. Zhu, X. Zhu, M. Li, X. Xu, J. Yin, Adaptive self-paced deep clustering with data augmentation, IEEE Trans. Knowl. Data Eng. 32 (9) (2019) 1680–1693.

[100] K. Ghasedi, A. Herandi, C. Deng, W. Cai, H. Huang, Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5736–5745.

[101] W. Guo, K. Lin, W. Ye, Deep embedded K-means clustering, in: 2021 International Conference on Data Mining Workshops (ICDMW), 2021, pp. 686–694.

[102] J. Chang, L. Wang, G. Meng, S. Xiang, C. Pan, Deep adaptive image clustering, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5879–5887.

[103] M. Caron, P. Bojanowski, A. Joulin, M. Douze, Deep clustering for unsupervised learning of visual features, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 132–149.

[104] X. Ji, J.F. Henriques, A. Vedaldi, Invariant information clustering for unsupervised image classification and segmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9865–9874.

[105] W. Hu, T. Miyato, S. Tokui, E. Matsumoto, M. Sugiyama, Learning discrete representations via information maximizing self-augmented training, in: International Conference on Machine Learning, 2017, pp. 1558–1567.

[106] J. Zbontar, L. Jing, I. Misra, Y. LeCun, S. Deny, Barlow twins: Self-supervised learning via redundancy reduction, in: International Conference on Machine Learning, 2021, pp. 12310–12320.

[107] K. He, H. Fan, Y. Wu, S. Xie, R. Girshick, Momentum contrast for unsupervised visual representation learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9729–9738.

[108] W. Cao, Z. Zhang, C. Liu, R. Li, Q. Jiao, Z. Yu, H.-S. Wong, Unsupervised discriminative feature learning via finding a clustering-friendly embedding space, Pattern Recognit. 129 (2022) 108768.

[109] Q. Hu, X. Wang, W. Hu, G.-J. Qi, Adco: Adversarial contrast for efficient learning of unsupervised representations from self-trained negative adversaries, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 1074–1083.

[110] J. Huang, S. Gong, X. Zhu, Deep semantic clustering by partition confidence maximisation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 8849–8858.

[111] Z. Dang, C. Deng, X. Yang, K. Wei, H. Huang, Nearest neighbor matching for deep clustering, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 13693–13702.

[112] C. Xu, R. Lin, J. Cai, S. Wang, Deep image clustering by fusing contrastive learning and neighbor relation mining, Knowl.-Based Syst. 238 (2022) 107967.

[113] C. Niu, H. Shan, G. Wang, Spice: Semantic pseudo-labeling for image clustering, IEEE Trans. Image Process. 31 (2022) 7264–7278.

[114] Z. Huang, J. Chen, J. Zhang, H. Shan, Learning representation for clustering via prototype scattering and positive sampling, IEEE Trans. Pattern Anal. Mach. Intell. 45 (6) (2023) 7509–7524.

[115] Y. Li, M. Yang, D. Peng, T. Li, J. Huang, X. Peng, Twin contrastive learning for online clustering, Int. J. Comput. Vis. 130 (9) (2022) 2205–2221.

[116] P. Bachman, O. Alsharif, D. Precup, Learning with pseudo-ensembles, Adv. Neural Inf. Process. Syst. 27 (2014).

[117] T. Miyato, S.-i. Maeda, M. Koyama, S. Ishii, Virtual adversarial training: a regularization method for supervised and semi-supervised learning, IEEE Trans. Pattern Anal. Mach. Intell. 41 (8) (2018) 1979–1993.

[118] T. Wang, P. Isola, Understanding contrastive representation learning through alignment and uniformity on the hypersphere, in: International Conference on Machine Learning, 2020, pp. 9929–9939.

[119] S. Yang, L. Li, S. Wang, W. Zhang, Q. Huang, A graph regularized deep neural network for unsupervised image representation learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1203–1211.

[120] Y. Li, F. Nie, H. Huang, J. Huang, Large-scale multi-view spectral clustering via bipartite graph, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 29, 2015, http://dx.doi.org/10.1609/aaai.v29i1.9598, URL: https://ojs.aaai.org/index.php/AAAI/article/view/9598.

[121] A. Longa, V. Lachi, G. Santin, M. Bianchini, B. Lepri, P. Lio, F. Scarselli, A. Passerini, Graph Neural Networks for temporal graphs: State of the art, open challenges, and opportunities, 2023, arXiv preprint arXiv:2302.01018.

[122] P. Holme, J. Saramäki, Temporal networks, Phys. Rep. 519 (3) (2012) 97–125.

[123] N. Park, R. Rossi, E. Koh, I.A. Burhanuddin, S. Kim, F. Du, N. Ahmed, C. Faloutsos, Cgc: Contrastive graph clustering forcommunity detection and tracking, in: Proceedings of the ACM Web Conference 2022, 2022, pp. 1115–1126.

[124] M. Liu, Y. Liu, K. Liang, S. Wang, S. Zhou, X. Liu, Deep temporal graph clustering, 2023, arXiv preprint arXiv:2305.10738.

[125] Y. Jia, Z. Gu, Z. Jiang, C. Gao, J. Yang, Persistent graph stream summarization for real-time graph analytics, World Wide Web (2023) 1–21.

[126] M. Rosvall, C.T. Bergstrom, Maps of random walks on complex networks reveal community structure, Proc. Natl. Acad. Sci. 105 (4) (2008) 1118–1123.

[127] D. Blondel Vincent, G. Jean-Loup, L. Renaud, L. Etienne, Fast unfolding of communities in large networks, J. Stat. Mech.: Theory Exp. 10 (2008) (2008) P10008.

[128] F.D. Malliaros, M. Vazirgiannis, Clustering and community detection in directed networks: A survey, Phys. Rep. 533 (4) (2013) 95–142.

[129] P. Bedi, C. Sharma, Community detection in social networks, Wiley Interdiscip. Rev.: Data Min. Knowl. Discov. 6 (3) (2016) 115–135.

[130] S. Xing, X. Shan, L. Fanzhen, W. Jia, Y. Jian, Z. Chuan, H. Wenbin, P. Cecile, N. Surya, J. Di, et al., A comprehensive survey on community detection with deep learning, IEEE Trans. Neural Netw. Learn. Syst. (2022).

[131] X. Zhang, H. Liu, Q. Li, X.-M. Wu, Attributed graph clustering via adaptive graph convolution, 2019, arXiv preprint arXiv:1906.01210.

[132] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, S.Y. Philip, A comprehensive survey on graph neural networks, IEEE Trans. Neural Netw. Learn. Syst. 32 (1) (2020) 4–24.

[133] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, M. Sun, Graph neural networks: A review of methods and applications, AI Open 1 (2020) 57–81.

[134] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, et al., Graph attention networks, stat 1050 (20) (2017) 10–48550.

[135] T. Kipf, M. Welling, Variational graph auto-encoders, in: NIPS Workshop on Bayesian Deep Learning, 2016.

[136] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, C. Zhang, Adversarially regularized graph autoencoder for graph embedding, 2019, arXiv:1802.04407.

[137] G. Cui, J. Zhou, C. Yang, Z. Liu, Adaptive graph encoder for attributed graph embedding, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 976–985.

[138] S. Suresh, P. Li, C. Hao, J. Neville, Adversarial graph augmentation to improve graph contrastive learning, Adv. Neural Inf. Process. Syst. 34 (2021) 15920–15933.

[139] E. Pan, Z. Kang, Multi-view contrastive graph clustering, Adv. Neural Inf. Process. Syst. 34 (2021) 2148–2159.

[140] W. Xia, Q. Wang, Q. Gao, M. Yang, X. Gao, Self-consistent contrastive attributed graph clustering with pseudo-label prompt, IEEE Trans. Multimed. (2022).

[141] Y. Liu, W. Tu, S. Zhou, X. Liu, L. Song, X. Yang, E. Zhu, Deep graph clustering via dual correlation reduction, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36, 2022, pp. 7603–7611, http://dx.doi.org/10.1609/aaai.v36i7.20726, URL: https://ojs.aaai.org/index.php/AAAI/article/view/20726.

[142] L. Gong, S. Zhou, W. Tu, X. Liu, Attributed graph clustering with dual redundancy reduction, in: L.D. Raedt (Ed.), Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22, International Joint Conferences on Artificial Intelligence Organization, 2022, pp. 3015–3021, http://dx.doi.org/10.24963/ijcai.2022/418, Main Track.

[143] Y. Dong, Z. Wang, J. Du, W. Fang, L. Li, Attention-based hierarchical denoised deep clustering network, World Wide Web 26 (1) (2023) 441–459.

[144] B. Fatemi, L. El Asri, S.M. Kazemi, SLAPS: Self-supervision improves structure learning for graph neural networks, Adv. Neural Inf. Process. Syst. 34 (2021) 22667–22681.

[145] S. Pan, R. Hu, S.-f. Fung, G. Long, J. Jiang, C. Zhang, Learning graph embedding with adversarial training methods, IEEE Trans. Cybern. 50 (6) (2019) 2475–2487.

[146] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, M. Guo, GraphGAN: Graph representation learning with generative adversarial nets, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, 2018, http://dx.doi.org/10.1609/aaai.v32i1.11872, URL: https://ojs.aaai.org/index.php/AAAI/article/view/11872.

[147] Z. Tao, H. Liu, J. Li, Z. Wang, Y. Fu, Adversarial graph embedding for ensemble clustering, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, International Joint Conferences on Artificial Intelligence Organization, 2019, pp. 3562–3568, http://dx.doi.org/10.24963/ijcai.2019/494.

[148] L. Yang, Y. Wang, J. Gu, C. Wang, X. Cao, Y. Guo, JANE: Jointly adversarial network embedding, in: IJCAI, 2020, pp. 1381–1387.

[149] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu, P. Cui, Structural deep clustering network, in: Proceedings of the Web Conference 2020, 2020, pp. 1400–1410.

[150] W. Tu, S. Zhou, X. Liu, X. Guo, Z. Cai, E. Zhu, J. Cheng, Deep fusion clustering network, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 9978–9987, http://dx.doi.org/10.1609/aaai.v35i11.17198, URL: https://ojs.aaai.org/index.php/AAAI/article/view/17198.

[151] Y. Liu, X. Yang, S. Zhou, X. Liu, S. Wang, K. Liang, W. Tu, L. Li, Simple contrastive graph clustering, IEEE Trans. Neural Netw. Learn. Syst. (2023).

[152] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, V. Lempitsky, Domain-adversarial training of neural networks, J. Mach. Learn. Res. 17 (1) (2016) 2096–2030.

[153] Y. Wang, B. Yu, L. Wang, C. Zu, D.S. Lalush, W. Lin, X. Wu, J. Zhou, D. Shen, L. Zhou, 3D conditional generative adversarial networks for high-quality PET image estimation at low dose, Neuroimage 174 (2018) 550–562.

[154] J.T. Springenberg, Unsupervised and semi-supervised learning with categorical generative adversarial networks, 2016, arXiv:1511.06390.

[155] G.-J. Qi, Loss-sensitive generative adversarial networks on lipschitz densities, Int. J. Comput. Vis. 128 (5) (2020) 1118–1140.

[156] J. Adler, S. Lunz, Banach wasserstein GAN, in: Proceedings of the 32nd International Conference on Neural Information Processing Systems, 2018, pp. 6755–6764.

[157] T. Dam, S.G. Anavatti, H.A. Abbass, Improving ClusterGAN using self-augmented information maximization of disentangling latent spaces, 2023, arXiv:2107.12706.

[158] C. Ling, G. Cao, W. Cao, H. Wang, H. Ren, IAE-ClusterGAN: A new Inverse autoencoder for Generative Adversarial Attention Clustering network, Neurocomputing 465 (2021) 406–416.

[159] D.P. Kingma, M. Welling, Auto-encoding variational Bayes, 2022, arXiv:1312.6114.

[160] L. Rokach, O. Maimon, Clustering methods, in: Data Mining and Knowledge Discovery Handbook, 2005, pp. 321–352.

[161] C. Zhang, J. Bütepage, H. Kjellström, S. Mandt, Advances in variational inference, IEEE Trans. Pattern Anal. Mach. Intell. 41 (8) (2018) 2008–2026.

[162] D.P. Kingma, T. Salimans, M. Welling, Variational dropout and the local reparameterization trick, Adv. Neural Inf. Process. Syst. 28 (2015) 2575–2583.

[163] K.P. Murphy, Machine Learning: A Probabilistic Perspective, MIT Press, 2012.

[164] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, A. Courville, Adversarially learned inference, 2017, arXiv:1606.00704.

[165] J. Donahue, P. Krähenbühl, T. Darrell, Adversarial feature learning, 2017, arXiv:1605.09782.

[166] W. Hu, C. Chen, F. Ye, Z. Zheng, Y. Du, Learning deep discriminative representations with pseudo supervision for image clustering, Inform. Sci. 568 (2021) 199–215.

[167] L. Mahon, T. Lukasiewicz, Selective pseudo-label clustering, in: German Conference on Artificial Intelligence (Künstliche Intelligenz), Springer, 2021, pp. 158–178.

[168] L. McInnes, J. Healy, J. Melville, UMAP: Uniform manifold approximation and projection for dimension reduction, 2020, arXiv:1802.03426.

[169] X. Jiang, P. Qian, Y. Jiang, Y. Gu, A. Chen, Deep self-supervised clustering with embedding adjacent graph features, Syst. Sci. Control Eng. 10 (1) (2022) 336–346.

[170] J. Yang, D. Parikh, D. Batra, Joint unsupervised learning of deep representations and image clusters, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5147–5156.

[171] S. Huang, Z. Kang, Z. Xu, Q. Liu, Robust deep k-means: An effective and simple method for data clustering, Pattern Recognit. 117 (2021) 107996.

[172] A.V.D. Oord, Y. Li, O. Vinyals, Representation learning with contrastive predictive coding, 2019, arXiv:1807.03748.

[173] J. Li, P. Zhou, C. Xiong, S.C. Hoi, Prototypical contrastive learning of unsupervised representations, 2020, arXiv preprint arXiv:2005.04966.

[174] Y. Tao, K. Takagi, K. Nakata, Clustering-friendly representation learning via instance discrimination and feature decorrelation, 2021, arXiv preprint arXiv:2106.00131.

[175] X. Chen, K. He, Exploring simple siamese representation learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 15750–15758.

[176] D. Wang, P. Cui, W. Zhu, Structural deep network embedding, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 1225–1234.

[177] S. Cao, W. Lu, Q. Xu, Deep neural networks for learning graph representations, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 30, 2016, http://dx.doi.org/10.1609/aaai.v30i1.10179, URL: https://ojs.aaai.org/index.php/AAAI/article/view/10179.

[178] S. Ding, B. Wu, X. Xu, L. Guo, L. Ding, Graph clustering network with structure embedding enhanced, Pattern Recognit. 144 (2023) 109833.

[179] L. Ren, Y. Qin, Y. Chen, C. Lin, R. Huang, Deep document clustering via adaptive hybrid representation learning, Knowl.-Based Syst. (2023) 111058.

[180] J. Donahue, K. Simonyan, Large scale adversarial representation learning, Adv. Neural Inf. Process. Syst. 32 (2019).

[181] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, J.M. Pérez, I. Perona, An extensive comparative study of cluster validity indices, Pattern Recognit. 46 (1) (2013) 243–256.

[182] H. Xiong, Z. Li, Clustering validation measures, in: Data Clustering, Chapman and Hall/CRC, 2018, pp. 571–606.

[183] A.K. Abdalameer, M. Alswaitti, A.A. Alsudani, N.A.M. Isa, A new validity clustering index-based on finding new centroid positions using the mean of clustered data to determine the optimum number of clusters, Expert Syst. Appl. 191 (2022) 116329.

[184] F. Ros, R. Riad, S. Guillaume, PDBI: A partitioning Davies-Bouldin index for clustering evaluation, Neurocomputing 528 (2023) 178–199.

[185] P. Favati, O. Menchi, An internal validity index for arbitrarily shaped clusters, Expert Syst. Appl. 235 (2024) 121124.

[186] E. Rendón, I. Abundez, A. Arizmendi, E.M. Quiroz, Internal versus external cluster validation indexes, Int. J. Comput. Commun. 5 (1) (2011) 27–34.

[187] P.A. Estévez, M. Tesmer, C.A. Perez, J.M. Zurada, Normalized mutual information feature selection, IEEE Trans. Neural Netw. 20 (2) (2009) 189–201.

[188] J.M. Santos, M. Embrechts, On the use of the adjusted rand index as a metric for evaluating supervised classification, in: International Conference on Artificial Neural Networks, 2009, pp. 175–184.

[189] H.W. Kuhn, The Hungarian method for the assignment problem, Nav. Res. Logist. Q. 2 (1–2) (1955) 83–97.

[190] C.C. Aggarwal, C. Zhai, A survey of text clustering algorithms, in: Mining Text Data, Springer, 2012, pp. 77–128.

[191] S. Kumar, A.K. Kar, P.V. Ilavarasan, Applications of text mining in services management: A systematic literature review, Int. J. Inf. Manag. Data Insights 1 (1) (2021) 100008.

[192] A. Karami, M. Lundy, F. Webb, Y.K. Dwivedi, Twitter and research: A systematic literature review through text mining, IEEE Access 8 (2020) 67698–67717.

[193] C. Wang, P. Nulty, D. Lillis, A comparative study on word embeddings in deep learning for text classification, in: Proceedings of the 4th International Conference on Natural Language Processing and Information Retrieval, 2020, pp. 37–46.

[194] V. Dogra, S. Verma, P. Chatterjee, J. Shafi, J. Choi, M.F. Ijaz, et al., A complete process of text classification system using state-of-the-art NLP models, Comput. Intell. Neurosci. 2022 (2022).

[195] H. Shi, T. Sakai, Self-supervised and few-shot contrastive learning frameworks for text clustering, IEEE Access (2023).

[196] M.E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, in: M. Walker, H. Ji, A. Stent (Eds.), Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 2227–2237, http://dx.doi.org/10.18653/v1/N18-1202, URL: https://aclanthology.org/N18-1202.

[197] R. Chalapathy, S. Chawla, Deep learning for anomaly detection: A survey, 2019, arXiv:1901.03407.

[198] S. Basumallik, R. Ma, S. Eftekharnejad, Packet-data anomaly detection in PMU-based state estimator using convolutional neural network, Int. J. Electr. Power Energy Syst. 107 (2019) 690–702.

[199] X. Ma, J. Wu, S. Xue, J. Yang, C. Zhou, Q.Z. Sheng, H. Xiong, L. Akoglu, A comprehensive survey on graph anomaly detection with deep learning, IEEE Trans. Knowl. Data Eng. (2021) 1, http://dx.doi.org/10.1109/TKDE.2021.3118815.

[200] H. Liu, J. Li, Y. Wu, Y. Fu, Clustering with outlier removal, IEEE Trans. Knowl. Data Eng. 33 (6) (2021) 2369–2379, http://dx.doi.org/10.1109/TKDE.2019.2954317.

[201] G. Pang, C. Shen, L. Cao, A.V.D. Hengel, Deep learning for anomaly detection: A review, ACM Comput. Surv. (CSUR) 54 (2) (2021) 1–38.

[202] X. Ma, J. Wu, S. Xue, J. Yang, C. Zhou, Q.Z. Sheng, H. Xiong, L. Akoglu, A comprehensive survey on graph anomaly detection with deep learning, IEEE Trans. Knowl. Data Eng. (2021).

[203] K. Choi, J. Yi, C. Park, S. Yoon, Deep learning for anomaly detection in time-series data: review, analysis, and guidelines, IEEE Access 9 (2021) 120043–120065.

[204] C. Wu, S. Shao, C. Tunc, P. Satam, S. Hariri, An explainable and efficient deep learning framework for video anomaly detection, Clust. Comput. (2021) 1–23.

[205] S. Kim, K. Choi, H.-S. Choi, B. Lee, S. Yoon, Towards a rigorous evaluation of time-series anomaly detection, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36, 2022, pp. 7194–7201, http://dx.doi.org/10.1609/aaai.v36i7.20680, URL: https://ojs.aaai.org/index.php/AAAI/article/view/20680.

[206] R. Riad, F. Ros, M.E. hajji, R. Harba, An industrial portrait background removal solution based on knowledge infusion, Appl. Intell. 52 (10) (2022) 11592–11605, http://dx.doi.org/10.1007/s10489-021-03099-3.

[207] J.R. Hershey, Z. Chen, J. Le Roux, S. Watanabe, Deep clustering: Discriminative embeddings for segmentation and separation, in: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2016, pp. 31–35, http://dx.doi.org/10.1109/ICASSP.2016.7471631.

[208] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, D. Terzopoulos, Image segmentation using deep learning: A survey, IEEE Trans. Pattern Anal. Mach. Intell. 44 (7) (2022) 3523–3542, http://dx.doi.org/10.1109/TPAMI.2021.3059968.

[209] X. Wang, H. Ma, S. You, Deep clustering for weakly-supervised semantic segmentation in autonomous driving scenes, Neurocomputing 381 (2020) 20–28, http://dx.doi.org/10.1016/j.neucom.2019.11.019, URL: https://www.sciencedirect.com/science/article/pii/S0925231219315784.

[210] X. Xia, B. Kulis, W-Net: A deep model for fully unsupervised image segmentation, 2017, arXiv:1711.08506.

[211] L. Zhou, W. Wei, DIC: Deep image clustering for unsupervised image segmentation, IEEE Access 8 (2020) 34481–34491, http://dx.doi.org/10.1109/ACCESS.2020.2974496.

[212] Y. He, H. Yu, X. Liu, Z. Yang, W. Sun, A. Mian, Deep learning based 3D segmentation: A survey, 2023, arXiv:2103.05423.

[213] T. Moriya, H.R. Roth, S. Nakamura, H. Oda, K. Nagara, M. Oda, K. Mori, Unsupervised segmentation of 3D medical images based on clustering and deep representation learning, in: B. Gimi, A. Krol (Eds.), Medical Imaging 2018: Biomedical Applications in Molecular, Structural, and Functional Imaging, SPIE, 2018, 1057820, http://dx.doi.org/10.1117/12.2293414.

[214] M. Gao, F. Zheng, J.J. Yu, C. Shan, G. Ding, J. Han, Deep learning for video object segmentation: a review, Artif. Intell. Rev. 56 (1) (2023) 457–531.

[215] T. Zhou, F. Porikli, D.J. Crandall, L. Van Gool, W. Wang, A survey on deep learning technique for video segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 45 (6) (2022) 7099–7122.

[216] Y. Zhu, H. Shuai, G. Liu, Q. Liu, Self-supervised video representation learning using improved instance-wise contrastive learning and deep clustering, IEEE Trans. Circuits Syst. Video Technol. 32 (10) (2022) 6741–6752, http://dx.doi.org/10.1109/TCSVT.2022.3169469.

[217] X. Wang, R.J. Hyndman, F. Li, Y. Kang, Forecast combinations: an over 50-year review, Int. J. Forecast. (2022).

[218] M. Jain, S. A.V., S. Denman, S. Sridharan, C. Fookes, LSTM guided ensemble correlation filter tracking with appearance model pool, Comput. Vis. Image Underst. 195 (2020) 102935, http://dx.doi.org/10.1016/j.cviu.2020.102935, URL: https://www.sciencedirect.com/science/article/pii/S1077314220300229.

[219] Z.-Q. Zhao, P. Zheng, S.-T. Xu, X. Wu, Object detection with deep learning: A review, IEEE Trans. Neural Netw. Learn. Syst. 30 (11) (2019) 3212–3232, http://dx.doi.org/10.1109/TNNLS.2018.2876865.

[220] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, Los Alamitos, CA, USA, 2016, pp. 779–788, http://dx.doi.org/10.1109/CVPR.2016.91, URL: https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.91.

[221] W. Zhu, Y. Shen, M. Liu, L.P. Aguirre Sanchez, L. Sun, GMAIR: Unsupervised object detection based on spatial attention and Gaussian mixture model, Intell. Neurosci. 2022 (2022) http://dx.doi.org/10.1155/2022/7254462.

[222] R. Charakorn, Y. Thawornwattana, S. Itthipuripat, N. Pawlowski, P. Manoonpong, N. Dilokthanakul, An explicit local and global representation disentanglement framework with applications in deep clustering and unsupervised object detection, 2020, arXiv:2001.08957.

[223] X. Wang, R. Girdhar, S.X. Yu, I. Misra, Cut and learn for unsupervised object detection and instance segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 3124–3134.

[224] S. Xiong, B. Li, S. Zhu, DCGNN: A single-stage 3D object detection network based on density clustering and graph neural network, Complex Intell. Syst. 9 (3) (2023) 3399–3408.

[225] Y. Yu, Z. Huang, F. Li, H. Zhang, X. Le, Point Encoder GAN: A deep learning model for 3D point cloud inpainting, Neurocomputing 384 (2020) 192–199.

[226] Y. Zhang, D. Huang, Y. Wang, PC-RGNN: Point cloud completion and graph neural network for 3D object detection, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 3430–3437, http://dx.doi.org/10.1609/aaai.v35i4.16456, URL: https://ojs.aaai.org/index.php/AAAI/article/view/16456.

[227] W. Shi, R. Rajkumar, Point-gnn: Graph neural network for 3d object detection in a point cloud, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1711–1719.

[228] B. Sun, P. Zhou, L. Du, X. Li, Active deep image clustering, Knowl.-Based Syst. (2022) 109346.

[229] B. Chiu, S.K. Sahu, D. Thomas, N. Sengupta, M. Mahdy, Autoencoding keyword correlation graph for document clustering, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 3974–3981.

[230] S. Wang, M. Gong, W. Liu, Y. Wu, Preventing epidemic spreading in networks by community detection and memetic algorithm, Appl. Soft Comput. 89 (2020) 106118, http://dx.doi.org/10.1016/j.asoc.2020.106118, URL: https://www.sciencedirect.com/science/article/pii/S1568494620300582.

[231] F. Karimi, S. Lotfi, H. Izadkhah, Community-guided link prediction in multiplex networks, J. Informetr. 15 (4) (2021) 101178, http://dx.doi.org/10.1016/j.joi.2021.101178, URL: https://www.sciencedirect.com/science/article/pii/S1751157721000493.

[232] H.A. Deylami, M. Asadpour, Link prediction in social networks using hierarchical community detection, in: 2015 7th Conference on Information and Knowledge Technology (IKT), 2015, pp. 1–5, http://dx.doi.org/10.1109/IKT.2015.7288742.

[233] X. Su, S. Xue, F. Liu, J. Wu, J. Yang, C. Zhou, W. Hu, C. Paris, S. Nepal, D. Jin, Q.Z. Sheng, P.S. Yu, A comprehensive survey on community detection with deep learning, IEEE Trans. Neural Netw. Learn. Syst. (2022) 1–21, http://dx.doi.org/10.1109/TNNLS.2021.3137396.

[234] F. Liu, S. Xue, J. Wu, C. Zhou, W. Hu, C. Paris, S. Nepal, J. Yang, P.S. Yu, Deep learning for community detection: progress, challenges and opportunities, 2020, arXiv preprint arXiv:2005.08225.

[235] D. Jin, Z. Yu, P. Jiao, S. Pan, D. He, J. Wu, S.Y. Philip, W. Zhang, A survey of community detection approaches: From statistical modeling to deep learning, IEEE Trans. Knowl. Data Eng. 35 (2) (2021) 1149–1170.

[236] S. Suganyadevi, V. Seethalakshmi, K. Balasamy, A review on deep learning in medical image analysis, Int. J. Multimed. Inf. Retr. 11 (1) (2022) 19–38.

[237] T. Kart, W. Bai, B. Glocker, D. Rueckert, DeepMCAT: Large-scale deep clustering for medical image categorization, in: Deep Generative Models, and Data Augmentation, Labelling, and Imperfections: First Workshop, DGM4MICCAI 2021, and First Workshop, DALI 2021, Held in Conjunction with MICCAI 2021, Strasbourg, France, October 1, 2021, Proceedings, Springer-Verlag, Berlin, Heidelberg, 2021, pp. 259–267, http://dx.doi.org/10.1007/978-3-030-88210-5_26.

[238] J. Yan, H. Chen, X. Li, J. Yao, Deep contrastive learning based tissue clustering for annotation-free histopathology image analysis, Comput. Med. Imaging Graph. 97 (2022) 102053, http://dx.doi.org/10.1016/j.compmedimag.2022.102053, URL: https://www.sciencedirect.com/science/article/pii/S089561112200026X.

[239] G. Eraslan, L.M. Simon, M. Mircea, N.S. Mueller, F.J. Theis, Single-cell RNA-seq denoising using a deep count autoencoder, Nature Commun. 10 (1) (2019) 390, http://dx.doi.org/10.1038/s41467-018-07931-2.

[240] C. Bhatt, I. Kumar, V. Vijayakumar, K.U. Singh, A. Kumar, The state of the art of deep learning models in medical science and their challenges, Multimedia Syst. 27 (4) (2021) 599–613.

[241] S. Zhong, K. Zhang, M. Bagheri, J.G. Burken, A. Gu, B. Li, X. Ma, B.L. Marrone, Z.J. Ren, J. Schrier, et al., Machine learning: new ideas and tools in environmental science and engineering, Environ. Sci. Technol. 55 (19) (2021) 12741–12754.

[242] T. Perol, M. Gharbi, M. Denolle, Convolutional neural network for earthquake detection and location, Sci. Adv. 4 (2) (2018) e1700578, http://dx.doi.org/10.1126/sciadv.1700578, URL: https://www.science.org/doi/abs/10.1126/sciadv.1700578. arXiv:https://www.science.org/doi/pdf/10.1126/sciadv.1700578.

[243] Y.O. Ouma, R. Cheruyot, A.N. Wachera, Rainfall and runoff time-series trend analysis using LSTM recurrent neural network and wavelet neural network with satellite-based meteorological data: case study of Nzoia hydrologic basin, Complex Intell. Syst. 8 (1) (2022) 213–236, http://dx.doi.org/10.1007/s40747-021-00365-2.

[244] A.Y. Sun, B.R. Scanlon, How can Big Data and machine learning benefit environment and water management: a survey of methods, applications, and future directions, Environ. Res. Lett. 14 (7) (2019) 073001.

[245] D. Srilatha, N. Thillaiarasu, DDoSNet: A deep learning model for detecting network attacks in cloud computing, in: 2022 4th International Conference on Inventive Research in Computing Applications (ICIRCA), 2022, pp. 576–581, http://dx.doi.org/10.1109/ICIRCA54612.2022.9985524.

[246] M.A. Ferrag, L. Maglaras, S. Moschoyiannis, H. Janicke, Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study, J. Inf. Secur. Appl. 50 (2020) 102419, http://dx.doi.org/10.1016/j.jisa.2019.102419, URL: https://www.sciencedirect.com/science/article/pii/S2214212619305046.

[247] L. Vu, Q.U. Nguyen, D.N. Nguyen, D.T. Hoang, E. Dutkiewicz, Deep transfer learning for IoT attack detection, IEEE Access 8 (2020) 107335–107344, http://dx.doi.org/10.1109/ACCESS.2020.3000476.

[248] M. Dib, S. Torabi, E. Bou-Harb, C. Assi, A multi-dimensional deep learning framework for IoT malware classification and family attribution, IEEE Trans. Netw. Serv. Manag. 18 (2) (2021) 1165–1177, http://dx.doi.org/10.1109/TNSM.2021.3075315.

[249] P.V.B. Ramos, S.M. Villela, W.N. Silva, B.H. Dias, Residential energy consumption forecasting using deep learning models, Appl. Energy 350 (2023) 121705, http://dx.doi.org/10.1016/j.apenergy.2023.121705, URL: https://www.sciencedirect.com/science/article/pii/S0306261923010693.

[250] T.R. Govindasamy, N. Chetty, Machine learning models to quantify the influence of PM10 aerosol concentration on global solar radiation prediction in South Africa, Clean. Eng. Technol. 2 (2021) 100042, http://dx.doi.org/10.1016/j.clet.2021.100042, URL: https://www.sciencedirect.com/science/article/pii/S2666790821000021.

[251] S. Ikeda, T. Nagai, A novel optimization method combining metaheuristics and machine learning for daily optimal operations in building energy and storage systems, Appl. Energy 289 (2021) 116716, http://dx.doi.org/10.1016/j.apenergy.2021.116716, URL: https://www.sciencedirect.com/science/article/pii/S0306261921002361.

[252] R.Z. Homod, H. Togun, A.K. Hussein, F.N. Al-Mousawi, Z.M. Yaseen, W. Al-Kouz, H.J. Abd, O.A. Alawi, M. Goodarzi, O.A. Hussein, Dynamics analysis of a novel hybrid deep clustering for unsupervised learning by reinforcement of multi-agent to energy saving in intelligent buildings, Appl. Energy 313 (2022) 118863.