12-2019

Salience-aware adaptive resonance theory for large-scale sparse data clustering

Lei MENG

Ah-hwee TAN
Singapore Management University, ahtan@smu.edu.sg

Chunyan MIAO

Citation

MENG, Lei; TAN, Ah-hwee; and MIAO, Chunyan. Salience-aware adaptive resonance theory for large-scale sparse data clustering. (2019). *Neural Networks*. 120, 143-157.
Available at: https://ink.library.smu.edu.sg/sis_research/5240

Neural Networks 120 (2019) 143–157

Contents lists available at ScienceDirect

Neural Networks

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

2019 Special Issue

Salience-aware adaptive resonance theory for large-scale sparse data clustering

Lei Meng [a,*], Ah-Hwee Tan [b], Chunyan Miao [b,c]

[a] NExT++, National University of Singapore, Singapore
[b] School of Computer Science and Engineering, Nanyang Technological University, Singapore
[c] NTU-UBC Research Center of Excellence in Active Living for the Elderly (LILY), Nanyang Technological University, Singapore

A R T I C L E   I N F O

Article history:
Available online 21 September 2019

Keywords:
Adaptive resonance theory
Clustering
Sparse data
Subspace learning
Feature weighting
Parameter adaptation

A B S T R A C T

Sparse data is known to pose challenges to cluster analysis, as the similarity between data tends to be ill-posed in the high-dimensional Hilbert space. Solutions in the literature typically extend either k-means or spectral clustering with additional steps on representation learning and/or feature weighting. However, adding these usually introduces new parameters and increases computational cost, thus inevitably lowering the robustness of these algorithms when handling massive ill-represented data. To alleviate these issues, this paper presents a class of self-organizing neural networks, called the salience-aware adaptive resonance theory (SA-ART) model. SA-ART extends Fuzzy ART with measures for cluster-wise salient feature modeling. Specifically, two strategies, i.e. cluster space matching and salience feature weighting, are incorporated to alleviate the side-effect of noisy features incurred by high dimensionality. Additionally, cluster weights are bounded by the statistical means and minimums of the samples therein, making the learning rate also self-adaptable. Notably, SA-ART allows clusters to have their own sets of self-adaptable parameters. It has the same time complexity of Fuzzy ART and does not introduce additional hyperparameters that profile cluster properties. Comparative experiments have been conducted on the ImageNet and BlogCatalog datasets, which are large-scale and include sparsely-represented data. The results show that, SA-ART achieves 51.8% and 18.2% improvement over Fuzzy ART, respectively. While both have a similar time cost, SA-ART converges faster and can reach a better local minimum. In addition, SA-ART consistently outperforms six other state-of-the-art algorithms in terms of precision and F1 score. More importantly, it is much faster and exhibits stronger robustness to large and complex data.

1. Introduction

Clustering high-dimensional sparse data is commonly required in real-world big data applications, especially for web data, such as web photo organization and retrieval based on user-contributed tags (Meng & Tan, 2012; Meng et al., 2015), social community detection using user relational networks (Meng & Tan, 2014), and personalized e-commerce product recommendation using users' online behavior and posts (Feng & Qian, 2014; West, Wesley-Smith, & Bergstrom, 2016). In such studies, a data object may be represented by a vector of 10k entries of which only 100 entries have non-zero values. For example, in personalized product recommendation, the length of a user's feature vector is equal to the number of products in total, while only the user-purchased entries are non-zero. It poses a great challenge to clustering algorithms since the "true" similarity is easily buried in the massive number of mismatched noisy features, resulting in a downgraded clustering performance (Muja & Lowe, 2014; Tomasev, Radovanovic, Mladenic, & Ivanovic, 2014). Fig. 1 presents a real-world example on sparse data clustering, and this motivates the development of novel clustering algorithms that can accurately identify the salient features of different data clusters.

To handle the aforementioned problems in clustering high-dimensional sparse data, existing methods typically follow three main directions, including subspace clustering, soft subspace clustering, and co-clustering:

- **Subspace clustering** is also named "hard weighting" (Jing, Ng, & Huang, 2007). It identifies a set of subspaces that have much lower dimensionality but better distinguishing power for identifying data clusters (Kriegel, Kröger, & Zimek, 2009). Subsequently, a traditional clustering algorithm is used to partition data on the new representations. Such subspaces may be obtained from the original feature space (Agrawal, Gehrke, Gunopulos, & Raghavan, 2005) or some new spaces

[*] Corresponding author.
E-mail address: lmeng@nus.edu.sg (L. Meng).

https://doi.org/10.1016/j.neunet.2019.09.014
0893-6080/© 2019 Elsevier Ltd. All rights reserved.

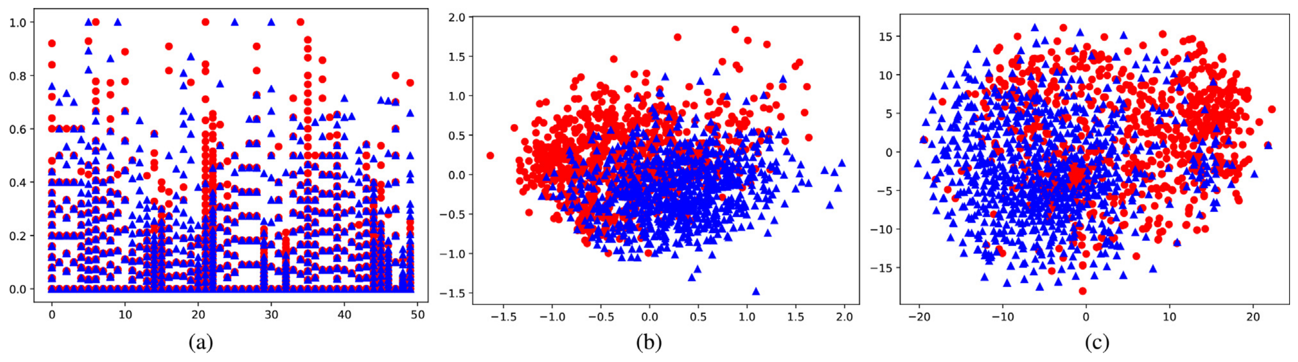

Fig. 1. Illustration for the challenges in sparse data clustering using data visualization of two classes from the ImageNet dataset. (a) Feature distribution of the first 50 out of 1000 features (for best view). 2D projection of 1000 features of the same data using (b) PCA and (c) t-SNE. It is difficult to find a good partitioning for the two classes (in red dot and blue triangle) based on pure statistical distributions in the geometrical feature space due to a significant overlap in non-zero entries (the other 950 dimensions show a similar or even worse situation). Traditional feature dimension reduction techniques, such as a linear mapping with preserved reconstruction energy (PCA) and a non-linear mapping with preserved local neighboring structures (t-SNE), may not work well. However, the difference in bursting features makes it possible to distinguish the two classes by exploring the associations between their representative features and distributions. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

derived by estimating data associations in the original space using local neighboring structures (Li, Liu, Tang, & Lu, 2015), linear correlations (Elhamifar & Vidal, 2013; Vidal & Favaro, 2014), feature-wise distribution densities (Zhu, Ting, & Carman, 2018), and deep representation learning (Zhou, Hou, & Feng, 2018).

- **Soft subspace clustering** refers to feature weighting methods (Jing et al., 2007). It learns to weight the features in the original space for individual clusters. As a result, more weights will be given to the salient features of a cluster, when computing the similarity between a data object and this cluster. This approach typically incorporates feature weighting strategies into traditional clustering algorithms, such as k-means (Chen, Ye, Xu, & Huang, 2012; De Amorim & Mirkin, 2012; Huang, Ng, Rong, & Li, 2005; Jing et al., 2007), Fuzzy c-means (Zhou, Chen, Chen, Zhang, & Li, 2016), and DBSCAN (Bohm, Railing, Kriegel, & Kroger, 2004).
- **Co-clustering** is also called "biclustering" (Xu & Wunsch II, 2011). It was first introduced to simultaneously cluster the rows and columns of a matrix. Later, it was used in data mining scenarios for clustering both data and features (Mirkin, 2013). This makes the data in the same cluster share a set of highly-correlated features. Existing algorithms applicable to sparse data typically extend traditional clustering algorithms, such as self-organizing neural networks (Xu & Wunsch II, 2011) and generative mixture models (Salah, Rogovschi, & Nadif, 2016).

Although the aforementioned clustering algorithms have shown improved performance for clustering high-dimensional sparse data, their performance may be downgraded in terms of speed and robustness, when applied to large-scale data. This is mainly due to the increased data volume and complexity. The large volume of data significantly increases their computational cost on the selection of salient features, i.e. subspaces. While the increased complexity of data, such as the number and densities of clusters, may result in the difficulties in hyperparameter settings or even failures in the assumptions for data distributions in the original feature space. This raises the need for novel clustering algorithms that scale well for big data and are able to self-adapt their hyperparameters to increase model robustness.

To address these issues, this paper presents a salience-aware adaptive resonance theory (SA-ART) for clustering large-scale sparse data. It extends from Fuzzy ART (Carpenter, Grossberg, & Rosen, 1991b), a competitive neural network that has a linear time complexity, converges fast, and does not need to predefine the number of clusters. SA-ART inherits these nice properties and seamlessly integrates three statistical measures into its functions of similarity measure and cluster weight modeling. These additional measures make it possible for SA-ART to incrementally discover the cluster-wise salient features and self-adapt the learning rates for cluster weights of individual clusters. Notably, by introducing a heuristic method (Meng, Tan, & Wunsch, 2016) to self-adapt the cluster-wise vigilance parameter (the only parameter in Fuzzy ART defining the threshold for intra-cluster similarity), SA-ART allows all the important parameters, including the learning rate and the vigilance parameter, to be self-adaptable. This alleviates the sensitivity of SA-ART to its hyperparameters, thus increasing its robustness to large-scale and complex data.

Experiments were conducted on two large-scale real-world datasets, including a subset of ImageNet dataset of 74k images belonging to 50 classes and the BlogCatalog dataset of 66k blog writers belonging to 147 classes. Experimental results, including parameter sensitivity analysis, clustering performance comparison, robustness to noise, and efficiency analysis, showed superior performance of SA-ART in terms of computational efficiency, intra-cluster purity, and the overall quality of the cluster structures generated.

2. Related work

This section introduces existing studies on sparse data clustering, which mainly follow three directions, including subspace clustering, soft subspace clustering, and co-clustering. Each of the directions, including different types of algorithms, is detailed in the following sections.

2.1. Subspace clustering

Subspace clustering aims to identify different subspaces, i.e. features sets, where data of the same category in their subspace are close to each other, but all are far from those outside the category. Considering different ways of finding such subspaces, existing methods can be categorized into four types, as illustrated below.

2.1.1. Subspace selection in original space

This line of algorithms finds valid subspaces by estimating the data distributions under the enumerated combinations of features from the original feature space. Since it is an NP-hard problem, approximation strategies are usually adopted to reduce the computation.

One example (Fern & Brodley, 2003) is to first perform clustering on a set of randomly selected subspaces and then average the results to determine whether two data objects are in the same cluster. The famous CLIQUE (Agrawal et al., 2005) combines grid and density-based clustering. It is based on a theory that a grid in a k-D space is dense if and only if all of its projections in the $(k-1)$-D are dense. CLIQUE starts from every single feature to detect dense grids, and it further increases the dimensions to find the maximum size of a subspace that makes the grid remain dense. Dimensionality unbiased subspace clustering (DUSC) (Assent, Krieger, Müller, & Seidl, 2007) extends traditional density-based clustering with an expected density estimator to search for subspace clusters. A recently-proposed algorithm, i.e. Clustering by Shared Subspaces (CSSub) (Zhu et al., 2018), measures data similarity using the counts of shared subspaces. It first enumerates subspaces to find the core points with sufficient neighbors in a subspace. Subsequently, the K-medoids algorithm is used to cluster the core points using the number of shared subspaces as a similarity measure.

2.1.2. *Linear correlation analysis*

Besides enumerating subspaces to find qualified data clusters, an elegant line of approach follows the theory of linear correlation analysis, which assumes that data in the same subspace should be able to be represented using a linear combination of all the other data. Sparse subspace clustering (SSC) (Elhamifar & Vidal, 2013) is a popular example. First, the coefficients of such linear combinations for each data object are computed, which estimate the linear correlation between this data object and the others. Subsequently, these coefficients are used as the new representations of data, since data in the same subspace should have similar coefficients. The final clusters are obtained using spectral clustering algorithms.

This seeding method has several variants with theoretical analysis (Wang, Wang, & Singh, 2019; Wang & Xu, 2016). It also leads to a series of algorithms that follow a similar pipeline but put different constraints on the coefficients, such as the low-rank property (Vidal & Favaro, 2014), the data correlation threshold (Heckel & Bölcskei, 2015; Peng, Yi, & Tang, 2015), and the block diagonal property (Lu, Feng, Lin, Mei, & Yan, 2019).

2.1.3. *Deep representation learning*

Deep neural networks have shown superior power in representation learning, usually called "deep learning", for major pattern recognition and computer vision tasks under a supervised learning paradigm. Recently, two approaches have been investigated for unsupervised representation learning and clustering, including autoencoder and generative adversarial networks (GAN).

Autoencoder (Song, Liu, Huang, Wang, & Tan, 2013) uses an encoding–decoding framework, where the encoder, typically a widely-used network such as ResNet, learns to map the input data to a feature vector and the decoder learns to recover the input data from this vector. This intermediate feature vector is therefore deemed to compress the input data with minimum information loss. However, a simple joint algorithm connecting autoencoder with traditional clustering algorithm, such as k-means, does not consider the sparsity of data. A recent study (Kang et al., 2018) addresses this problem by introducing an "inter-feature graph" that reinforces learning from correlated features. However, building this graph requires domain knowledge. From another perspective, deep subspace clustering networks (DSC-Nets) (Ji, Zhang, Li, Salzmann, & Reid, 2017) follows the theory of linear correlation analysis, as discussed in Section 2.1.2. It takes advantage of deep learning, rather than matrix factorization, to learn the feature vector that is "self-expressive", i.e. the feature vector of a data object can be approximated using a linear combination of those of the other data. Similarly, a spectral clustering is used to perform clustering on the new representations.

GAN (Goodfellow et al., 2014) uses two networks, i.e. a generator and a discriminator. Both iteratively regularize the data distribution in a latent space. The generator maps an input data to a latent representation while the discriminator classifies whether this representation follows a specific distribution or not. Adversarial autoencoder (Makhzani, Shlens, Jaitly, Goodfellow, & Frey, 2016) is the first work that uses an autoencoder with two GANs for clustering MINIST handwritten digit images. The GANs condition the encoder output, i.e. a cluster indicator and a style vector to be in one-hot and Gaussian distributions, respectively. However, adversarial autoencoder is a general algorithm, and it does not consider data sparsity. Gaussian mixture GANs (GM-GANs) (Ben-Yosef & Weinshall, 2018) follows the theory of the Gaussian mixture model (GMM). It assumes that complex data can be represented by a mixture of k Gaussian distributions and uses GANs to condition the latent representations produced by the encoder to fit GMM. Testing samples are therefore classified to one of the learned GMM distributions. Deep adversarial subspace clustering (DASC) (Zhou et al., 2018) follows the "self-expression" learning and spectral clustering procedures of DSC-Nets (Ji et al., 2017). Beyond that, a GAN for each cluster is used to sample fake samples from the learned cluster weights and optimize the weights until the discriminator cannot distinguish between the fake samples and the real data in this cluster.

2.2. *Soft subspace clustering*

Soft subspace clustering (Deng, Choi, Jiang, Wang, & Wang, 2016) focuses on the problem of adaptively evaluating the importance of features for different clusters to improve the similarity measures and alleviate the side-effects of ill-posed features.

To this end, different explorations on the metrics for computing the feature importance scores have been investigated. Among many, the weighted k-means algorithm has been extensively explored. One of the earliest studies (Huang et al., 2005) presents a weighted k-means where the importance of a feature to a cluster is measured by the intra-cluster sum-of-distance along that feature. However, a feature is equally weighted in all clusters. Therefore, an information entropy-based strategy (Jing et al., 2007) further improves this algorithm by placing a constraint on the feature weights to avoid the problem of identifying few salient features for a cluster. Several follow-up studies investigate new feature distance metrics or constraints, such as using a Minkowski β-metric (De Amorim & Mirkin, 2012) or a kernel (Wang et al., 2016) as distance measures; adding between-cluster penalties (Deng, Choi, Chung, & Wang, 2010); introducing feature group weights to award data points in the same cluster when sharing more common features (Chen et al., 2012; Gan & Ng, 2015); and extensions to the Fuzzy c-means algorithm (Chitsaz & Jahromi, 2016; Zhou et al., 2016), online clustering (Zhu, Cao, Yang, & Lei, 2014), and fuzzy rule generation (Xu et al., 2019).

Besides the k-means embedded algorithms, DBSCAN (Bohm et al., 2004) has been extended to assign higher weights to the features that have lower variance. A filtering approach (Boongoen, Shang, Iam-On, & Shen, 2011) incorporates a reliability measure that enumerates the k-nearest neighbors along each feature for each data point to compute the importance score of a feature to a potential cluster. This scoring function can be embedded in k-means, agglomerative hierarchical clustering, and spectral clustering. An evolutionary algorithm (Xia, Zhuang, & Yu, 2013) uses a multi-objective evolutionary method with a projection similarity validity index for subspace discovery.

2.3. Co-clustering

Co-clustering (Dhillon, 2001) aims to find the data clusters where samples in the same cluster have a similar distribution over a subset of features. It works in sparse data clustering by enforcing the correlation of features in the same cluster.

Among many co-clustering algorithms, two algorithms are potentially applicable for high-dimensional sparse data. Biclustering ARTMAP (BARTMAP) (Xu & Wunsch II, 2011) extends from Fuzzy ARTMAP (Carpenter, Grossberg, Markuzon, Reynolds, Rosen, et al., 1992). It uses two Fuzzy ARTs (Carpenter et al., 1991b), an ART$_a$ for clustering data and an ART$_b$ for clustering features. Specifically, BARTMAP follows two criteria to assign a data object to a cluster: (1) the data object is similar to the generalized distribution of data, i.e. cluster weights, of the selected cluster (measured by ART$_a$) and (2) the data object shares feature distribution in a subset of features (measured by ART$_b$). BARTMAP therefore has a flexibility to control the degree of match at both pattern and feature levels. Another algorithm (Salah et al., 2016) is a generative model. It uses a mixture of von Mises–Fisher distributions, instead of Gaussian distribution, for co-clustering. The proposed algorithm iteratively clusters data and features, and it can be used for either hard or soft subspace clustering.

3. Problem statement

Given a dataset $\mathcal{I} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, where $\mathbf{x}_n = [x_{n,1}, \dots, x_{n,M}]$ ($x_{n,M} \in [0, 1]$) is the feature vector of the nth data object, N is the size of the dataset $\mathcal{I}$, and M is the number of features. This paper proposes a clustering algorithm that partitions the N data objects into J clusters, each of which has a weight vector $\mathbf{w}_j = [w_{j,1}, \dots, w_{j,M}]$ to represent the generalized distribution of the data therein and a salience scoring vector $\mathbf{s}_j = [s_{j,1}, \dots, s_{j,M}]$ to reveal the importance of each feature to this cluster, where $0 < s_{j,m} < 1$ and $\sum_m s_{j,m} = 1$. In the context of sparse data clustering, there exists a small subset of features $p \subset \{1, \dots, M\}$ such that $\sum_{i\in p} |s_{j,i}| \approx 1$ and $\sum_{i\in p} w_{j,i} \approx |\mathbf{w}_j|$, where $w_{j,i} \in [0, 1]$ and $|.|$ is ℓ_1 norm.

This problem is challenging for two reasons:

1. **Noisy features:** The high dimensionality allows data to have diverse representations while the sparsity implicitly increases the impact of each feature when computing data similarity. Combined, these factors enhance the side-effect of noisy features that contribute very little to, or even harm, the representation of data clusters.
2. **Complexity in data distribution:** The increase in dimensionality may significantly increase the time complexity of clustering algorithms and break their assumptions on data distribution in the feature space. These will make the algorithms not scalable to big data and downgrade their robustness to hyperparameters.

Existing studies, as introduced in Section 2, typically focus on learning either data representation or distance metrics. Doing this usually introduces expensive computations and additional parameters. This motivates the study in this paper to seek out a clustering algorithm for sparse data clustering that has a low time cost and the ability to self-adapt its hyperparameters to fit data characteristics.

4. Theoretical analysis on Fuzzy ART

This section provides a theoretical analysis on the characteristics and clustering behaviors of Fuzzy ART, i.e. the base model of the proposed SA-ART. Following this analysis, the benefits and potential problems of Fuzzy ART are revealed.

4.1. ART and Fuzzy ART

Adaptive resonance theory (ART) (Carpenter & Grossberg, 1987b, 2017) is both a cognitive and neural theory of how the brain quickly learns to categorize, recognize, and predict objects and events in a changing world. It results in a family of unsupervised learning frameworks, such as ART 1 (Carpenter & Grossberg, 1987b), ART 2 (Carpenter & Grossberg, 1987a), ART 2-A (Carpenter, Grossberg, & Rosen, 1991a), ART 3 (Carpenter & Grossberg, 1990), and Fuzzy ART (Carpenter et al., 1991b). These ART-embodied competitive neural networks usually have a linear time complexity, converge fast, and do not need to predefine the number of clusters. Additionally, their variants have shown promising performance in real-world clustering tasks (Damelin, Gu, Wunsch, & Xu, 2015; Meng et al., 2016; Meng, Tan, & Xu, 2014; da Silva, Elnabarawy, & Wunsch II, 2019; da Silva & Wunsch, 2016).

Fuzzy ART is a popular choice among many other ART variants for clustering, since it uses fuzzy operators and introduces a "complement coding" to address the problem of "category proliferation". This problem is caused by a continuous learning from noisy/inconsistent features, making a cluster's weight values quickly become small and flat. As such, a new cluster must be generated to represent this cluster. Fuzzy ART performs clustering in an incremental manner, following the four main steps below:

1. **Complement coding:** Let $\mathbf{I} = \mathbf{x}$ denote an input data object, where $\mathbf{x} = [x_1, \dots, x_M]$ and $x_m \in [0, 1]$. Complement coding concatenates $\mathbf{x}$ with its complement vector $\bar{\mathbf{x}} = \mathbf{1} - \mathbf{x}$. Therefore, the final input feature vector is $\mathbf{I} = [\mathbf{x}, \bar{\mathbf{x}}]$.
2. **Category choice:** Given $\mathbf{I}$, this step uses a choice function to compute an asymmetric similarity score for each cluster c_j. It evaluates to which degree its weight vector $\mathbf{w}_j$ is a fuzzy subset of $\mathbf{I}$, defined by
$$T(c_j, \mathbf{I}) = \frac{|\mathbf{I} \wedge \mathbf{w}_j|}{\alpha + |\mathbf{w}_j|}, \tag{1}$$
where $\alpha \approx 0$ is a positive value to give priority to the clusters of dense features, the fuzzy AND operation $\wedge$ is defined by $(\mathbf{p} \wedge \mathbf{q})_i \equiv \min(p_i, q_i)$, and the norm $|.|$ is equivalent to the ℓ_1 norm, defined by $|\mathbf{p}| \equiv \sum_i p_i$.
If no cluster exists, a new cluster c_1 with $\mathbf{w}_j = \mathbf{I}$ is created to encode $\mathbf{I}$. Otherwise, the winner c_{j^*} is selected, where $j^* = \arg\max_j T(c_j, \mathbf{I})$.
3. **Template matching:** If a winner c_{j^*} exists, this step uses a match function to compute an asymmetric match score, which evaluates to which degree $\mathbf{I}$ is a fuzzy subset of c_{j^*}, defined by
$$M(c_{j^*}, \mathbf{I}) = \frac{|\mathbf{I} \wedge \mathbf{w}_{j^*}|}{|\mathbf{I}|}. \tag{2}$$
If the match score satisfies a "vigilance criteria" such that $M(c_{j^*}, \mathbf{I}) \geq \rho$, a resonance occurs. This leads to the assignment of $\mathbf{I}$ to c_{j^*}. Note that $\rho \in (0, 1]$ is "vigilance parameter", and it constrains the minimum intra-cluster similarity.
Otherwise, if $M(c_{j^*}, \mathbf{I}) < \rho$, the category choice step is revisited to select a winner from the remaining clusters. If no winner satisfies the "vigilance criteria", a new clusterc_{J+1} with $\mathbf{w}_{J+1} = \mathbf{I}$ is created.
4. **Prototype learning:** A resonance incurs the update of cluster weights $\mathbf{w}_{j^*}$ of c_{j^*} via a learning function, defined by
$$\hat{\mathbf{w}}_{j^*} = \beta(\mathbf{I} \wedge \mathbf{w}_{j^*}) + (1 - \beta)\mathbf{w}_{j^*}, \tag{3}$$
where $\beta \in (0, 1]$ is the learning parameter.

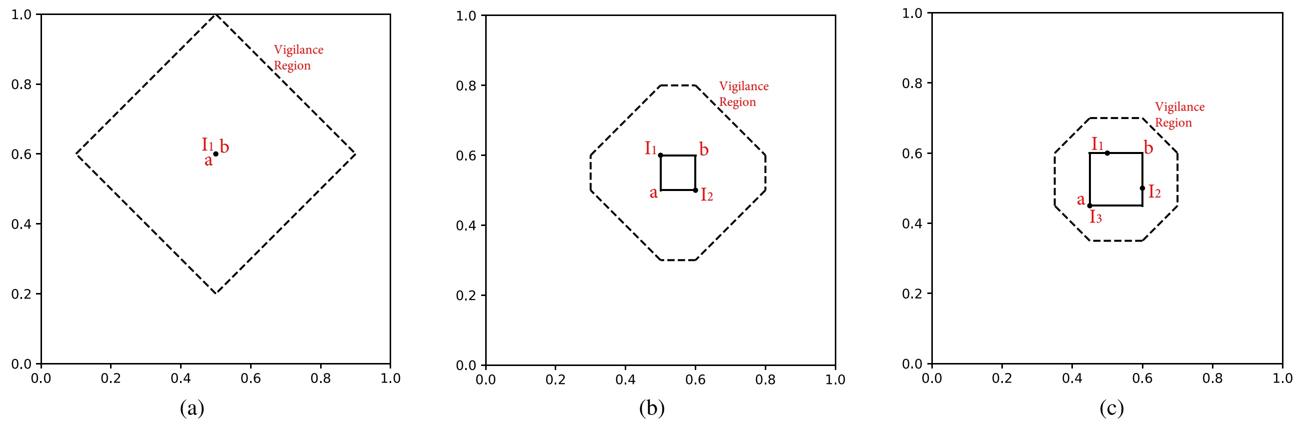

Fig. 2. Illustration of the clustering behavior of Fuzzy ART in 2D space under learning rate $\beta = 1$ and vigilance parameter $\rho = 0.8$. (a) The first complement coded input pattern $\mathbf{I}_1 = [0.5, 0.6, 0.5, 0.4]$ leads to the creation of the first cluster c_1 with $\mathbf{w}_1 = \mathbf{I}_1$. Denoting $\mathbf{w}_1 = [\mathbf{a}, 1 - \mathbf{b}]$ such that $\mathbf{a} = [0.5, 0.6]$ and $\mathbf{b} = [0.5, 0.6]$, it is observed that the rectangle formed by $\mathbf{w}_1$ is a single point overlapped with $\mathbf{I}_1$. Its vigilance region (VR) in dashed lines is a 90-degree clock-wise rotated square centered at $\mathbf{I}_1$. (b) Encoding $\mathbf{I}_2 = [0.6, 0.5, 0.4, 0.5]$ results in $\mathbf{w}_2 = [0.5, 0.5, 0.4, 0.4]$ represented by a rectangle with a bottom-left vertex $\mathbf{a} = [0.5, 0.5]$ and a top-right vertex $\mathbf{b} = [0.6, 0.6]$. The corresponding VR becomes an octagon centered at $\mathbf{w}_2$ and shrinks significantly. (c) After encoding $\mathbf{I}_3 = [0.45, 0.45, 0.55, 0.55]$, it was observed that the rectangle $\mathbf{w}_3 = [0.45, 0.45, 0.4, 0.4]$ expands towards $\mathbf{I}_3$ while the VR shrinks.

4.2. Geometric properties of Fuzzy ART

As observed from the key functions of Fuzzy ART in the above section, Fuzzy ART uses the choice function in Eq. (1) to select the best-matching cluster c_{j*} that shares the key feature distribution with $\mathbf{I}$. Subsequently, the match function in Eq. (2) limits the intra-cluster similarity of c_{j*} using the vigilance parameter ρ. The learning function in Eq. (3) monotonously decreases the cluster weights, so the features with inconsistent values will be suppressed.

It is important to consider how Fuzzy ART behaves in the feature space and how complement coding works to prevent “category proliferation”. As demonstrated in the Fuzzy ART paper (Carpenter et al., 1991b), by denoting the feature and complement parts of $\mathbf{w}_j$ using $\mathbf{a}$ and $\bar{\mathbf{b}} = 1 - \mathbf{b}$, respectively, $\mathbf{w}_j$ can be represented by a hyper-rectangle with size $R \leqslant M(1 - \rho)$, as shown in Fig. 2(b) and (c). Through a geometrical analysis of Fuzzy ART’s behaviors in the feature space (Meng et al., 2016), it is further demonstrated that the match function in Eq. (2) with the vigilance parameter ρ form a hyper-octagon. It, referred to as “vigilance region” (VR), is centered at the cluster weight hyper-rectangle $\mathbf{w}_j$. More importantly, it has been demonstrated that the behavior that causes Fuzzy ART to form clusters is the key to resolving “cluster proliferation”.

As observed in Fig. 2(a), a newly created cluster has a big VR, so that any incoming data object falling in this region will be accepted. A cluster with multiple data objects will have its weight vector $\mathbf{w}_j$ represented in a hyper-rectangle, and there will be a significant shrinking in the VR, as shown in see Fig. 2(b). The hyper-rectangle will further expand until it and the VR are similar in size (see Fig. 2(c)). Note that the learning rate $\beta \in (0, 1]$ controls the degree that the hyper-rectangle $\mathbf{w}_j$ expands towards the input data object $\mathbf{I}$. Besides, data objects falling in the hyper-rectangle will have the match score $M(c_{j*}, \mathbf{I}) = 1$. Encoding them will incur no expansion of the hyper-rectangle $\mathbf{w}_j$.

4.3. Characteristics of Fuzzy ART in clustering

The clustering behavior illustrated in Fig. 2 reveals several distinct characteristics of Fuzzy ART:

1. **Lazy cluster centers:** The geometric location of a cluster, i.e. the weight rectangle $\mathbf{w}_j$, is largely determined by the first data object, and the size of $\mathbf{w}_j$ is determined by the vigilance parameter ρ, which controls the intra-cluster similarity.
2. **Learning stability:** The learning function in Eq. (3) decreases the weights of clusters $\mathbf{w}_j$ in every resonance. Therefore, non-salient features should have low weight values. In the feature space, learning corresponds to the expansion of cluster hyper-rectangles as shown in Fig. 2, and a cluster will stop learning when its hyper-rectangle reaches the maximum size. This learning strategy ensures Fuzzy ART, equipped with a suitable ρ, will stably discover representative distributions of data clusters in the feature space.
3. **Clustering by creating hyper-octagons:** With the property of lazy cluster centers, Fuzzy ART creates hyper-octagons to cover the regions in the feature space where data objects lie. This is different from the typical way of existing clustering algorithms where the clusters are of open regions.
4. **Low intra-cluster scatters:** The above properties of the close-form hyper-octagons make it possible for Fuzzy ART to generate dense clusters when using a high value of ρ to control the minimum intra-cluster similarity.
5. **Fast converging:** Since Fuzzy ART creates hyper-octagons to fill the entire feature space, it may obtain reasonable results even in the first epoch. The following epochs will improve the cluster assignment of data caused by the randomness in data orders. This will refine the distributions of the hyper-octagons in the feature space and generate a few new clusters to fill the blank spaces caused by the shrinking of hyper-octagons.
6. **Denoising:** Fuzzy ART allows new clusters to be used for encoding the data that are sparsely distributed in the feature space. This protects the dense clusters against the ill-generalization of cluster weights caused by noises. Moreover, it handles the intra-class diversity where data of the same category but diversely distributed in the nearby feature spaces will be encoded using multiple clusters.

4.4. Problems of Fuzzy ART in clustering

While Fuzzy ART exhibits positive properties in data clustering, there are some potential problems that may result in a downgraded performance, as observed from Fig. 2:

1. **Inconsistent VR:** Fig. 2(a) shows that the VR of a new cluster covers a large region (36%) of the entire feature

space even under $\rho = 0.8$. This raises a risk of clustering a dissimilar data object on the edge of a VR, resulting in a long-and-narrow hyper-rectangle as cluster weights $\mathbf{w}_j$ across the feature space. While using a high vigilance value, such as $\rho = 0.9$, will result in an over-generation of clusters, since the VR of a cluster shrinks significantly after encoding the second data object, as shown in Fig. 2(b). Therefore, it is important to have a self-adaptation mechanism that can adaptively tune the vigilance parameter ρ to generate the hyper-octagons of VRs that fit the data distributions better.

2. **Over-split in data clusters:** The properties of lazy cluster centers and rapidly-shrinking VRs introduce a problem where, given a group of randomly presented data, the hyper-octagon of a cluster may converge to one direction and require the creation of a new cluster to fill the empty space. Things become worse if the input data always falls just outside of the VR of a cluster.
3. **Noise sensitivity:** Since Fuzzy ART's learning function monotonously decreases cluster weights, a newly-generated cluster is vulnerable to cluster weight erosion by the wrongly-assigned data objects. That is, by continuously learning from the data objects that have low values in the salient features for this cluster, the distribution of cluster weights may quickly become flat and cannot match the feature distributions of any categories any longer.

5. Salience-aware adaptive resonance theory for sparse data clustering

This section presents the salience-aware adaptive resonance theory (SA-ART), which aims to address the aforementioned problems of Fuzzy ART and make it conducive to scalable and robust sparse data clustering.

SA-ART follows the soft subspace clustering approach to model the importance of features for individual clusters. However, instead of using randomly initialized scores that are optimized to minimize clustering errors, this paper uses three simple statistical measures, i.e. frequency, mean, and variance, to estimate the importance and distributions of features for each cluster. Using these measures will incrementally reveal the frequent features that have high and stable values in clusters, i.e. salient features.

New functions for category choice, template matching, and prototype learning are developed, so that SA-ART can seamlessly incorporate the aforementioned cluster-wise feature statistics into the same procedures of Fuzzy ART while keeping the same linear time complexity $O(n)$. To summarize, SA-ART introduces the following new strategies to enhance learning from sparse data:

1. **Similarity measure in shared cluster space:** Instead of using the entire feature vector for similarity measure (see Eqs. (1) and (2)), SA-ART uses only the shared features between $\mathbf{I}$ and the $\mathbf{w}_{j^*}$'s salient features. As illustrated in Section 4.3, the hyper-octagon, i.e. VR, of a cluster is around the first data point it encoded. Therefore, this strategy allows SA-ART to gather the data with a similar distribution with cluster weights. It also avoids considering noisy features in the similarity measure process.
2. **Learning rate self-adaptation:** SA-ART adopts a new learning strategy that allows the features of each cluster to have their own learning rates in the prototype learning step (see Eq. (3)). It is also based on the cluster-wise feature statistics, which depresses learning from anomaly values that do not follow the features' Gaussian likelihoods.
3. **Cluster weight recovery:** To alleviate the possible erosion of cluster weights, SA-ART allows an increase in weight values. However, this recovery is also achieved in the prototype learning stage. Besides, the recovered values are bounded by the feature statistics.

These strategies on cluster-wise salient feature modeling enable SA-ART to stably learn the patterns of data clusters involving frequent noisy features. Additionally, SA-ART incorporates the activation maximization rule (AMR) (Meng et al., 2016) to make the vigilance parameter ρ self-adaptable. In this way, SA-ART not only inherits the advantages of Fuzzy ART, but also has all important hyperparameters self-adaptable.

The following sections will introduce the proposed statistical measures for feature salience estimation and learning rate self-adaptation, illustrate the new similarity measure and learning functions of SA-ART with theoretical proofs, brief AMR for the self-adaptation of the vigilance parameter ρ, present the entire SA-ART algorithm, and analyze its time complexity.

5.1. Statistical measures for feature salience estimation and distribution modeling

SA-ART uses three statistical measures, including frequency $f_{j,m}$, mean $\mu_{j,m}$, and variance $\sigma^2_{j,m}$, to profile a feature $w_{j,m}$ of a cluster c_j. The sections below present how these measures are used to evaluate the importance of features and guide the prototype learning step.

5.1.1. Feature salience estimation

Given a cluster c_j with its weight vector $\mathbf{w}_j = [w_{j,1}, \ldots, w_{j,2M}]$ and the profile of each feature $w_{j,m}$, including frequency $f_{j,m}$, mean $\mu_{j,m}$, and variance $\sigma^2_{j,m}$, the importance of a feature $w_{j,m}$ is computed by using a salience score, defined by

$$s_{j,m} = \begin{cases} 0, & \text{if } f_{j,m} = 0, \\ \lambda \cdot f_{j,m} + (1-\lambda) \cdot e^{-\sigma_{j,m}}, & \text{otherwise,} \end{cases} \tag{4}$$

where $\lambda \in [0, 1]$ balances the salience estimation between frequency and stability.

It is observed that $s_{j,m} \in [0, 1]$, given that $f_{j,m} \in [0, 1]$ and $\sigma_{j,m} \in (0, 1]$. $s_{j,m} = 1$ only when $f_{j,m} = 1$ and $\sigma_{j,m} = 0$, indicating that $w_{j,m}$ is of the highest importance if all data objects $\mathbf{x}_n$ in c_j share a single value of $w_{j,m}$. Therefore, a newly-created cluster c_j will have $s_{j,m} = 1$ for all the non-zero entries $w_{j,m} > 0$. After a series of resonances, the features that do not frequently appear or have diverse values will have lower scores.

5.1.2. Learning rate self-adaptation

Given the input data $\mathbf{I}_n = [\mathbf{x}_n, \mathbf{1} - \mathbf{x}_n] = [x_{n,1}, \ldots, x_{n,2M}]$, upon the notations used in Section 5.1.1, $\theta_{j,m}$ is introduced as the learning rate that determines how much the corresponding weight value of $w_{j,m}$ moves towards the newly-encoded data object $\mathbf{I}_n$, defined by

$$\theta_{j,m} = \begin{cases} e^{-\frac{9\cdot(x_{n,m}-\mu_{j,m})^2}{2\cdot\min(\mu_{j,m}+0.01,\,1-\mu_{j,m})^2}}, & \text{if } \sigma^2_{j,m} = 0, \\ e^{-\frac{(x_{n,m}-\mu_{j,m})^2}{2\cdot\sigma^2_{j,m}}}, & \text{otherwise.} \end{cases} \tag{5}$$

As observed, $\theta_{j,m} \in (0, 1]$ takes advantage of the Gaussian likelihood function. It essentially measures to which degree the input feature value $x_{n,m}$ matches with the value distribution of $w_{j,m}$. This strategy allows the value of $w_{j,m}$ to be not far from its statistical mean value $\mu_{j,m}$. In this way, Eq. (5) prevents unstable learning from the $x_{n,m}$ of too high or low values, compared to $\mu_{j,m}$.

Note that for features with zero variance, the value of $\frac{\min(\mu_{j,m}+0.01,1-\mu_{j,m})^2}{9}$ is used. This is based on the notion that given a Gaussian distribution function $f(x|\mu,\sigma^2)$, 99.7% of the values are located in the range of $[\mu-3\sigma,\mu+3\sigma]$. Viewing that $w_{j,m}\in[0,1]$, the aforementioned value is obtained according to $\mu_{j,m}-3\sigma_{j,m}\geq 0$ and $\mu_{j,m}+3\sigma_{j,m}\leq 1$. Besides, $\mu_{j,m}+0.01$ is used to avoid the case when $\mu_{j,m}=0$.

5.2. Modified Fuzzy ART functions with feature salience measures

This section illustrates the proposed similarity measure and learning functions of SA-ART that integrate the three statistical feature salience measures into the three main functions of fuzzy ART, i.e. category choice, template matching, and prototype learning.

5.2.1. Category choice with cluster space matching

Given a complement-coded input data object $\mathbf{I}_n=[\mathbf{x}_n,\mathbf{1}-\mathbf{x}_n]$ where $\mathbf{x}_n=[x_{n,1},\ldots,x_{n,M}]\in\mathcal{I}=\{\mathbf{x}_n|_{n=1}^N\}$ and $x_{n,m}\in[0,1]$, and a cluster $c_j\in\mathcal{C}=\{c_j|_{j=1}^J\}$ with its weight vector $\mathbf{w}_j=[w_{j,1},\ldots,w_{j,2M}]\in\mathcal{W}=\{\mathbf{w}_j|_{j=1}^J\}$ and salience scoring vector $\mathbf{s}_j=[s_{j,1},\ldots,s_{j,2M}]$ where $\mathbf{s}_j\in\mathcal{S}=\{\mathbf{s}_j|_{j=1}^J\}$, the function for category choice is defined by:

$$T(c_j,\mathbf{I}_n)=\begin{cases}0, & \text{if } p=\varnothing,\\ \frac{|\mathbf{u}_p\circ\mathbf{s}_p|}{\alpha+|\mathbf{w}_j\circ\mathbf{s}|}, & \text{otherwise,}\end{cases} \tag{6}$$

where $\alpha=0.01$ is used by default for the same purpose as in Eq. (1). $\mathbf{u}=\mathbf{I}_n\wedge\mathbf{w}_j=[u_1,\ldots,u_{2M}]$. $p=\{i|w_{j,i}>0\}$ includes the indexes of the non-zero entries of $\mathbf{w}_j$. $s_{j,m}\in\mathbf{s}_p$ is defined in Eq. (4) and the cluster index "j" is omitted for simplicity. "$\circ$" is the element-wise product, and $|.|$ is defined in Eq. (1).

Compared to the original choice function as defined by Eq. (1), the key differences include (1) all feature values are weighted and (2) only the intersection of salient features is considered for similarity measure. This new function operates under the assumption that even in high noise circumstances, data objects of the same class should have more shared features where the salient features are hidden. Its properties for robust similarity measure are proven below.

Property 1. *The category choice function $T(c_j,\mathbf{I}_n)$ selects the cluster c_j, where the salient feature distribution best matches with the input pattern $\mathbf{I}_n$.*

Proof. As observed from Eq. (6), $T(c_j,\mathbf{I}_n)$ measures to which degree $\mathbf{w}_j$ is a fuzzy subset of $\mathbf{I}_n$ in the weighted shared space. Therefore, $T(c_j,\mathbf{I}_n)=1$ only in the condition where I_n has higher values than $\mathbf{w}_j$ in terms of all the features of $\mathbf{w}_j$ that have non-zero values. When the shared features are less than the non-zero features of c_j, $T(c_j,\mathbf{I}_n)<1$ consistently holds. The salience scoring vector $\mathbf{s}_j$ further gives priority to the cluster where the distribution of salient features is better matched. □

Property 2. *The category choice function $T(c_j,\mathbf{I}_n)$ essentially computes the similarity between $\mathbf{I}_n$ and $\mathbf{w}_j$ in the entire feature space.*

Proof. Let t be the indexes for the zero entries of $\mathbf{u}$. Since $\mathbf{u}_t=\mathbf{0}$, the following may be easily attained:

$$\begin{aligned}\frac{|\mathbf{u}_p\circ\mathbf{s}_p|}{|\mathbf{w}_j\circ\mathbf{s}|}&=\frac{\sum_{i\in p}u_is_i}{\sum_m w_{j,m}s_m}+0,\\ &=\frac{\sum_{i\in p}u_is_i}{\sum_m w_{j,m}s_m}+\frac{\sum_{i\in t}u_is_i}{\sum_m w_{j,m}s_m},\\ &=\frac{\sum_m u_ms_m}{\sum_m w_{j,m}s_m},\\ &=\frac{|\mathbf{u}\circ\mathbf{s}|}{|\mathbf{w}_j\circ\mathbf{s}|}.\end{aligned} \tag{7}$$

Note that since the non-zero entry of c_j is known a priori, this property eliminates substantial computation when the dimensionality is high. □

5.2.2. Template matching with a weighted intra-cluster similarity measure

As demonstrated in Properties 1 and 2, the new category choice function in Eq. (6), selects the cluster c_{j^*} where the salient feature distribution of $\mathbf{w}_{j^*}$ is best satisfied by $\mathbf{I}_n$. Using the same notations, template matching in a similar way evaluates to which degree $\mathbf{I}_n$ is similar to $\mathbf{w}_{j^*}$, defined by

$$M(c_{j^*},\mathbf{I}_n)=\frac{|\mathbf{u}_p\circ\mathbf{s}_p|}{|\mathbf{I}_n\circ\mathbf{s}|}\geq\rho. \tag{8}$$

$M(c_{j^*},\mathbf{I}_n)$ measures to which degree $\mathbf{I}_n$ is a fuzzy subset of $\mathbf{w}_{j^*}$ in the weighted feature space. Note that $p=\varnothing$ is not considered here since this situation will be filtered in the category choice step. Property 3 demonstrates that a combination of the two asymmetric similarity measures $T(c_j,\mathbf{I}_n)$ and $M(c_j,\mathbf{I}_n)$ will form a symmetric similarity evaluation between $\mathbf{w}_j$ and $\mathbf{I}_n$.

Property 3. *The combination of $T(c_j,\mathbf{I}_n)$ and $M(c_j,\mathbf{I}_n)$ symmetrically evaluates the similarity between $\mathbf{w}_j$ and $\mathbf{I}_n$ considering both the distribution of shared features and the match of salient features.*

Proof. As demonstrated by a prior study (Meng et al., 2014) (see Property 1 under Section 5.2 for details), a combination of $T(c_j,\mathbf{I}_n)$ and $M(c_j,\mathbf{I}_n)$ will assign $\mathbf{I}_n$ to the cluster with the closest matching in feature distribution. Beyond this conclusion, the similarity measure using shared features is considered under two settings $p=q$ and $p<q$, where $q=\{i|w_{j,i}>0\}$ is the indexes of the non-zero entries of $\mathbf{w}_j$:

1. **When** $p=q$: Two extreme cases (1) $\mathbf{u}_p=\mathbf{w}_p$ and (2) $\mathbf{u}_p=\mathbf{I}_p$ are considered ("j" in $\mathbf{I}_j$ is omitted for simplicity):
   - **In Case (1):** $T(c_j,\mathbf{I}_n)=1$ and $M(c_j,\mathbf{I}_n)=\frac{\sum_{i\in p}u_is_i}{\sum_m I_{n,m}s_m}<1$. This indicates that c_j is given priority in category choice for $\mathbf{I}_n$, since for $\forall i\in p$, there is $w_{j,i}\leq x_{n,i}$. However, a resonance occurs only when $M(c_j,\mathbf{I}_n)\geq\rho$, thus limiting its lower bound.
   - **In Case (2):** $T(c_j,\mathbf{I}_n)=\frac{\sum_{i\in p}u_is_i}{\sum_m w_{j,m}s_m}<1$ and $M(c_j,\mathbf{I}_n)=1$. It indicates that for $\forall i\in p$, there is $w_{j,i}\geq x_{n,i}$, therefore c_j definitely passes the vigilance criteria. However, priority is given to the clusters with lower weight values.
     One concern regarding this case is the unlimited upper bound, which may incur a mismatch in the early clustering stage when p includes a set of noisy features. However, Section 5.2.3 will demonstrate that the side-effect of such wrong categorization in prototype learning can be minimized using the proposed learning function.
2. **When** $p<q$: Beyond the analysis under the setting $p=q$, the denominators of $T(c_j,\mathbf{I}_n)$ and $M(c_j,\mathbf{I}_n)$ will further penalize for missing salient features. Therefore, higher scores will be given to the cluster c_j that shares more shared salient features with $\mathbf{I}_n$. □

5.2.3. Prototype learning with noise depression and salience recovery

When the input pattern $\mathbf{I}_n$ passes the vigilance criteria, i.e. Eq. (8), of the selected cluster c_j, prototype learning is triggered to update the weight vector $\mathbf{w}_j$ of cluster c_j. Following the notations defined in Section 5.1.2 and denoting $\mathbf{v}_j = [\min(x_1, \mu_1), \ldots, \min(x_{2M}, \mu_{2M})]$, the learning function is defined by

$$\hat{\mathbf{w}} = \mathbf{v}_j \circ \boldsymbol{\theta}_j + \mathbf{w}_j \circ (\mathbf{1} - \boldsymbol{\theta}_j). \tag{9}$$

where $\boldsymbol{\theta}_j = [\theta_{j,1}, \ldots, \theta_{j,2M}]$ includes the learning rates for each feature $\mathbf{w}_{j,m}$, computed using Eq. (5). The following properties demonstrate this learning function's ability in modeling the pattern distribution of sparse data.

Property 4. *The proposed prototype learning function, i.e. Eq. (9), stably updates the weight values of $\mathbf{w}_j$, which are bounded by the statistical means and the lowest values.*

Proof. Eq. (9) illustrates that for $\forall m$, $w_{j,m}$ moves towards $\min(x_{n,m}, \mu_{j,m})$, of which the degree depends on $\theta_{j,m}$. Therefore, if $x_{n,m} \gg \mu_{j,m}$, $w_{j,m}$ moves towards $\mu_{j,m}$ with a small increase; similarly, $w_{j,m}$ moves towards $x_{n,m}$ with a small decrease if $x_{n,m} \ll \mathbf{w}_{j,m}$. As such, Eq. (9) achieves stable learning and sets the upper and lower bounds to weight values $\mathbf{w}_{j,m}$. □

Property 5. *The prototype learning function, i.e. Eq. (9), can identify and depresses noisy feature values.*

Proof. Noisy features for a cluster are defined as those that are less likely to have non-zero or stable values. Considering a noisy feature $w_{j,m}$, it is likely to have a small mean value $\mu_{j,m}$ and a large variance $\sigma^2_{j,m}$. During the learning process, as demonstrated in Property 4, Eq. (9) will gradually decrease the upper bound for $w_{j,m}$, i.e. $\mu_{j,m}$, and the large $\sigma^2_{j,m}$ facilitates a rapid decrease in $w_{j,m}$ to its lower bound. □

Property 6. *The prototype learning function, i.e. Eq. (9), can recover salient features during iterative learning.*

Proof. The intra-class diversity leads to the situation that the data objects of the same category follow a general feature distribution, but have low or even zero values for some salient features. Considering $w_{j,m} = 0$ a salient feature of c_j and $x_{n,m} > 0$ the feature of a correctly-assigned data object $\mathbf{I}_n$ in c_j, Eq. (5) ensures that $\theta_{j,m} > 0$ so that $\hat{w}_{j,m} > 0$. As such, the values of salient features of a cluster can gradually be recovered to their true distributions. □

5.3. Activation maximization rule for vigilance adaptation

SA-ART incorporates the activation maximization rule (AMR) (Meng et al., 2016) to allow each cluster to have a self-adaptable vigilance parameter ρ. AMR comes from a simple observation that a cluster with a small value of ρ happens to incur continuous resonances. While that of a large value is likely to cause a reset even it is selected as the winner. Therefore, AMR is triggered after each resonance or reset to restrain the continuous activation of the same cluster and promote the activation of clusters that usually incur resets, as defined by

$$\hat{\rho}_{j^*} = \begin{cases} (1+\delta)\rho_{j^*}, & \text{if resonance occurs,} \\ (1-\delta)\rho_{j^*}, & \text{if reset occurs.} \end{cases} \tag{10}$$

Although simple and heuristic, AMR is favored by its lightweight computation (only two lines 18 and 30 in Algorithm 1), and it has shown promising performance in alleviating ART's sensitivity to a fixed ρ (Meng et al., 2016). When ρ is small, AMR can generate more clusters, rather than a few big clusters of mixed data objects from many categories; when ρ is large, AMR can significantly reduce the number of small clusters generated and achieve a better performance.

Algorithm 1 Salience Aware Adaptive Resonance Theory

Input: Input data $\mathcal{I} = \{\mathbf{I}_1, \ldots, \mathbf{I}_N\}$, model parameters ρ_0, λ, $\alpha = 0.01$, and $\delta = 0.1$, the maximum iteration t_{max} or the terminating threshold ε.

```
1: repeat
2:     Set $n = 1$.
3:     repeat
4:         If the cluster set $\mathcal{C} = \varnothing$:
5:             Create a cluster $c_1$ with $\mathbf{w}_1 = \mathbf{I}_n$.
6:             Set $c_1$'s mean $\boldsymbol{\mu}_1$, variance $\boldsymbol{\sigma}_1^2$, and frequency $\mathbf{f}$.
7:             Compute $\mathbf{s}_1$ using Equation (4).
8:             Set $\rho_1 = \rho_0$
9:             Set $n = n + 1$.
10:            Return
11:        For $\forall c_j \in \mathcal{C} = \{c_j|_{j=1}^J\}$:
12:            Compute $T(c_j, \mathbf{I}_n)$ using Equation (6).
13:        repeat
14:            Pick the winner $c_{j^*} = \arg\max_{c_j \in \mathcal{C}} T(c_j, \mathbf{I}_n)$.
15:            Compute $M(c_{j^*}, \mathbf{I}_n)$ using Equation (8).
16:            If $M(c_{j^*}, \mathbf{I}_n) < \rho_{j^*}$:
17:                Set $T(c_{j^*}, \mathbf{I}_n) = -1$.
18:                Set $\rho_{j^*} = (1 - \delta)\rho_{j^*}$.
19:        until $M(c_{j^*}, \mathbf{I}_n) \geq \rho_{j^*}$, or for $\forall c_j \in \mathcal{C}$, $M(c_j, \mathbf{I}_n) < \rho_j$.
20:        If for $\forall c_j \in \mathcal{C}$, $T(c_j, \mathbf{I}_n) = -1$:
21:            Create a new cluster $c_{J+1}$ with $\mathbf{w}_{J+1} = \mathbf{I}_n$.
22:            Set its feature means $\boldsymbol{\mu}_1$, variances $\boldsymbol{\sigma}_1^2$, and frequencies $\mathbf{f}$.
23:            Compute $\mathbf{s}_{J+1}$ using Equation (4).
24:            Set $\rho_{J+1} = \rho_0$.
25:        Else:
26:            Update $\mathbf{w}_{j^*}$ using Equation (9).
27:            Update $\mathbf{f}_{j^*}$ using Equation (11).
28:            Update $\boldsymbol{\mu}_{j^*}$ and $\boldsymbol{\sigma}_{j^*}^2$ using Equations (12) and (14).
29:            Compute $\mathbf{s}_{j^*}$ using Equation (4).
30:            Set $\rho_{j^*} = (1 + \delta)\rho_{j^*}$.
31:        Set $n = n + 1$.
32:    until $n > N$.
33: until $t = t_{max}$, or $|D(\hat{\mathbf{w}}) - D(\mathbf{w})|_1 < \varepsilon$.
```

Output: Data clusters $\{c_j\}_{j=1}^J$, cluster weight vectors $\{\mathbf{w}_j|_{j=1}^J\}$

5.4. Summary of SA-ART algorithm

The pseudo code of SA-ART is given in Algorithm 1. SA-ART requires four hyper-parameters as model input:

1. Vigilance parameter ρ as defined in Eqs. (8) and (10),
2. Choice parameter α as defined in Eq. (6),
3. Balancing parameter λ as defined in Eq. (4),
4. Restraint parameter δ as defined in Eq. (10).

Note that $\alpha = 0.01$ is used by default. $\delta = 0.1$ is consistently used for AMR. Moreover, the experiments reveal that SA-ART usually achieves the best performance with $\lambda = 0.9$, and a similar performance is typically achieved when $\lambda \in [0.6, 0.9]$.

Based on the above observations, SA-ART only requires the initial value of the vigilance parameter ρ_0 to be tuned to constrain the intra-cluster similarity. A ρ_0 that is too small results in the generation of a few clusters with flat feature distributions; while one that is too large results in the excessive splitting of data groups. AMR helps in both cases by self-adapting the vigilance values of all clusters during the clustering process.

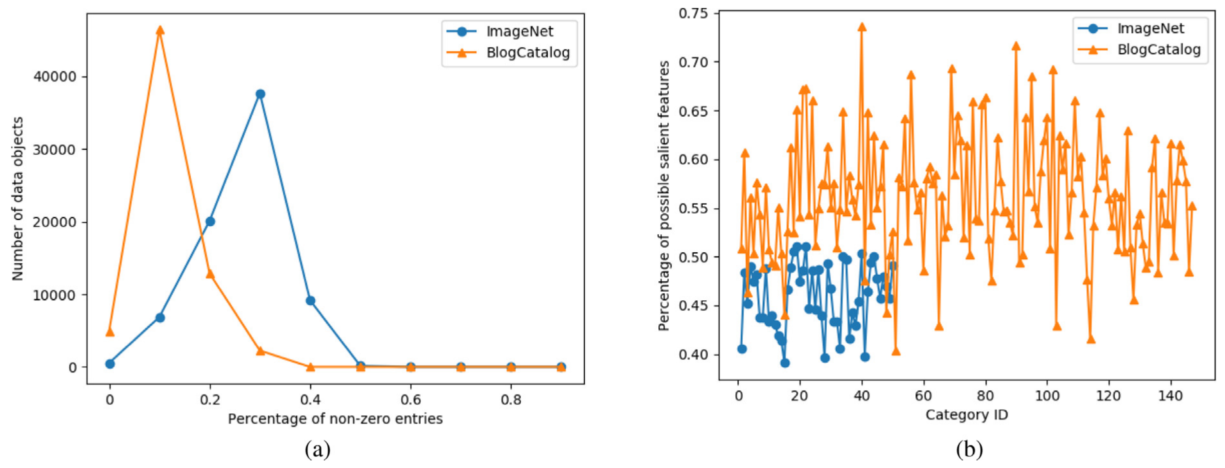

Fig. 3. The (a) sparsity and (b) noise levels of ImageNet and BlogCatalog datasets.

Lines 3–32 in the pseudo code illustrate the main steps of SA-ART for one input data $\mathbf{I}_n$. It is notable that since SA-ART performs clustering in an incremental manner, given an input data object $\mathbf{I}$ and a cluster c_j, the Welford's online algorithm (Welford, 1962) was used to update the feature mean vector $\boldsymbol{\mu}$ and variance vector $\boldsymbol{\sigma}^2$, which can effectively prevent computing loss. The incremental update equations for the feature frequency $\mathbf{f}$, $\boldsymbol{\mu}$, and $\boldsymbol{\sigma}^2$ (the cluster index j is omitted for simplicity) are defined by

$$\hat{f}_i = \begin{cases} \frac{L}{L+1} f_i, & \text{if } x_i = 0, \\ \frac{L}{L+1} f_i + \frac{1}{L+1}, & \text{otherwise}, \end{cases} \tag{11}$$

$$\hat{\boldsymbol{\mu}} = \boldsymbol{\mu} + (\mathbf{I} - \boldsymbol{\mu})(L+1)^{-1}, \tag{12}$$

$$\hat{\mathbf{Z}} = \mathbf{Z} + (\mathbf{I} - \boldsymbol{\mu}) \circ (\mathbf{I} - \hat{\boldsymbol{\mu}}), \tag{13}$$

$$\hat{\boldsymbol{\sigma}}^2 = \hat{\mathbf{Z}} L^{-1}, \tag{14}$$

where f_i is the frequency of the ith feature of the cluster weight $\mathbf{w}_j$, x_i is the value of the ith feature of the input data object $\mathbf{I}$, $L+1$ is the cluster size after data assignment. Notably, when $L = 1$, Eq. (11) still holds; $\mathbf{Z} = \mathbf{0}$ given $\mathbf{Z} = \sum_{\mathbf{I} \in c} (\mathbf{I} - \boldsymbol{\mu}) \circ (\mathbf{I} - \boldsymbol{\mu})$.

SA-ART iteratively performs data clustering in an incremental manner. In addition to conventional Fuzzy ART, it computes the cluster-wise feature statistics to effectively model the cluster-wise pattern distributions and simultaneously discover the salient features therein. The clustering process stops when it reaches a maximum number of iterations or when one more iteration incurs with no significant changes in the overall intra-cluster distance, defined by

$$D(\mathbf{W}) = \sum_{c \in \mathcal{C}} \sum_{\mathbf{I}, \boldsymbol{\mu} \in c} |(\mathbf{I} - \boldsymbol{\mu}) \circ (\mathbf{I} - \boldsymbol{\mu})|. \tag{15}$$

5.5. *Time complexity analysis*

SA-ART incurs computations mainly in the process of category choice, template matching, prototype learning, and the update of cluster-level feature statistics and salience scoring vectors. As shown in Eq. (6), the computational cost of the category choice function $T(c_j, \mathbf{I}_n)$ for all clusters has a time complexity of $O(MJ)$, where J is the number of current clusters and $2M$ is the number of features of $\mathbf{I}_n$. Template match using Eq. (8) mainly operates on the feature vectors $\mathbf{w}_j$ and $\mathbf{I}_n$, of which the time complexity is $O(M)$. Similarly, the update of cluster weight vectors using Eqs. (5) and (9) also has a time complexity of $O(M)$. The time complexity for computing vectors $\boldsymbol{\mu}$, $\boldsymbol{\sigma}^2$, $\mathbf{Z}$, $\mathbf{f}$, and $\mathbf{s}$, using Eqs. (4) and ((12)–(14)), is $O(M)$. Therefore, given a dataset of N objects, the overall time complexity of SA-ART with t epochs is $O(tNMJ)$ at the worst case. However, since SA-ART usually converges in $t < 20$ and J is gradually increased, it is typically faster than conventional clustering algorithms of a linear time complexity, such as k-means.

6. Experiments

This section presents the experimental analysis of SA-ART on two real-world datasets, i.e. the ImageNet and BlogCatalog datasets. Experiments were conducted in terms of five aspects, including parameter sensitivity analysis, ablation study, clustering performance comparison, sensitivity to data complexity, and time cost comparison.

6.1. *Datasets*

To evaluate the performance of SA-ART on clustering large-scale high-dimensional data, experiments were conducted on the following two datasets:

- **ImageNet dataset**: It is renowned for computer vision challenges, having over 1m images of 1000 categories for large-scale visual recognition tasks (Russakovsky et al., 2015). These images are collected from web search engines and social platforms, such as Google search and Flickr. This study used images from the first 50 categories,[1] which are represented in bag-of-words (BOW) SIFT features and normalized using $min--max$ normalization. In total, 74,506 images are used, each of which is represented with a 1000-D feature vector.
- **BlogCatalog dataset**: It includes the blogs and friendship network of 88k online users, originally created for detecting the overlapping social user groups (Wang, Tang, Gao, & Liu, 2010). We use the cleaned dataset from our prior work on social community detection (Meng & Tan, 2014). It includes 17,824 words extracted from the blogs of 66,418 users in 147 social groups. Each user is represented using a 17,824-D multi-hot feature vector.

Fig. 3 generally reveals the sparsity and noise levels of the two datasets. Fig. 3(a) illustrates the number of data objects in terms of the percentage of non-zero entries. The numbers are divided into ten bins, e.g. bin value 0.2 in $x - axis$ indicates having the percentage value $\in [0.2, 0.3)$. As observed, the data

[1] Image features are available at http://image-net.org/download-features.

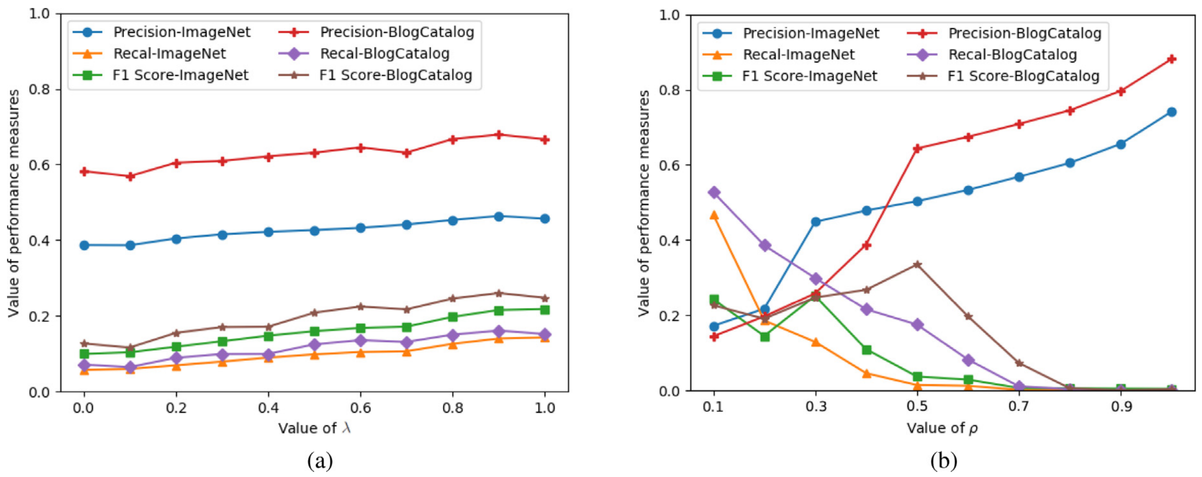

Fig. 4. The influence of parameters (a) λ and (b) ρ on the performance of SA-ART on ImageNet and BlogCatalog datasets.

objects in both datasets have zero values in at least 50% of the features. Most of the ImageNet data have only 30% non-zero features while that of the BlogCatalog dataset is just 10%. Fig. 3(b) shows the percentages of the potential salient features for each category. Such salient features are the high-frequency features identified using the Otsu's method (Sezgin & Sankur, 2004). For the ImageNet dataset, it was observed that, on average, around 45% of the features frequently appear in a category, i.e. only 15% of the total features are potentially useful for modeling cluster pattern. The BlogCatalog dataset has a higher level of proportion, i.e. around 55% on average, but with a large variance. This statistical information shows the sufficient sparsity and noise levels of the experimental data.

6.2. Performance measures

The performance of all the clustering algorithms was evaluated using three commonly-used metrics (Xu & Wunsch, 2008), including the weighted average precision $P = \sum_c \frac{L_c}{\sum_c L_c} p_c$, the weighted average recall $R = \sum_c \frac{L_c}{\sum_c L_c} r_c$, and the weighted average F1 score $F = \sum_c \frac{L_c}{\sum_c L_c} \frac{2p_c r_c}{p_c + r_c}$, where L_c is the size of cluster c, and p_c, r_c are the precision and recall of c, respectively. Specifically, $p_c = \frac{L_{c,t}}{L_c}$ and $r_c = \frac{L_{c,t}}{\sum_c L_{c,t}}$, where t is the true class of cluster c determined by the majority of samples in c, and $L_{c,t}$ is the number of samples of class t in c. To alleviate the initialization factor, the performance of SA-ART and the algorithms in comparison was an average of five runs on either randomly shuffled data or the initial cluster seeds.

6.3. Parameter sensitivity evaluation

This section provides the analysis on the sensitivity of SA-ART to the input parameters, i.e. λ and ρ, which control the cluster-level feature salience weighting and intra-cluster similarity, respectively. To this end, Fig. 4 reports the clustering performance of SA-ART with varied values of λ and ρ in 20 iterations on the two datasets.

Fig. 4(a) illustrates the performance of SA-ART with the best setting of ρ, i.e. $\rho = 0.7$ and 0.75 for ImageNet and BlogCatalog datasets, respectively. As observed, the consistent flat curves show that SA-ART is generally robust to λ. Referring to Eq. (4), the curves indicate that "frequency as salient feature" is much more discriminative than "stability in feature value". Besides, introducing a minor consideration of the latter improves the overall salient feature estimation. By setting $\lambda = 0.9$, the performance curves of varied ρ values were obtained, as shown in Fig. 4(b). It was observed that, for both datasets, the increase in the value of ρ leads to the increase of P and the decrease of R; F reaches the largest value at the elbow where P increases sharply. Notably, SA-ART keeps improving P with respect to the increase in the value (up to 0.99) of ρ. This demonstrates the effectiveness of using a vigilance parameter to constrain the minimum intra-cluster similarity. It enables SA-ART, when equipped with a large value of ρ, to handle the diversity and noise of sparse data. However, it is at the cost of the generalization power, i.e. the over-generation of small clusters. This is reflected from the sharp decrease of F after reaching its maximum. In contrast, a lower ρ helps build a more compact cluster network, but it may easily suffer from noise.

6.4. Ablation study

SA-ART incorporates several strategies to improve Fuzzy ART's performance on sparse data, including (1) salient feature weighting (SFW), (2) similarity measure with shared feature matching (SFM), (3) learning rate self-adaptation (LRS), and (4) activation maximization rule (AMR). This section evaluates the effects of individual strategies on the clustering performance of SA-ART.

With $\beta = 0.6$ and $\rho = 0.6$, the clustering results produced by different combinations of Fuzzy ART, i.e. the base model, and the aforementioned strategies are investigated. The quality of the generated cluster structure is evaluated using average precision, the number of clusters, and the average intra-cluster distance (computed using the averaged ℓ_2 norm) between the original input data $\mathbf{x}$ and the weight hyper-rectangle $\mathbf{a}$ and $\mathbf{b}$, as shown in Fig. 2. Precision and the intra-cluster distance reflect the quality of individual clusters with and without category labels, respectively. Therefore, the intra-cluster distance essentially measures how well SA-ART can identify all dense regions in the feature space. The number of clusters indicates both the cluster structure complexity and the recall aspects.

As reported in Table 1, using SFW leads to a significantly improvement over Fuzzy ART with higher precision, fewer clusters, and more compact clusters. Both LRS and AMR also can enhance the performance of Fuzzy ART, demonstrating the effectiveness of the two self-adaptation methods for the learning rate β and the vigilance parameter ρ. Solely SFM does not work well due to the possible inclusion of noisy features. However, using both LRS and SFM boosts the performance thanks to a better modeling of cluster weights. Integrating all the three strategies proposed in this paper makes Fuzzy ART achieve an absolute increase of 10%

Table 1
Ablation study on the clustering performance of SA-ART in terms of weighted average precision (P), number of clusters (#cluster), and intra-cluster distance (Dist), on the ImageNet dataset with different combinations of strategies: (1) salient feature weighting (SFW), (2) similarity measure with shared feature matching (SFM), (3) learning rate self-adaptation (LRS), and (4) activation maximization rule (AMR).

Algorithms	P	#cluster	Dist
Base (Fuzzy ART)	0.291	363	3.58
Base + SFW	0.368	325	3.24
Base + SFM	0.284	370	3.71
Base + LRS	0.322	351	3.44
Base + AMR	0.306	334	3.51
Base + SFM + LRS	0.343	349	3.36
Base + SFM + LRS + SFW	0.403	314	3.25
All (SA-ART)	0.418	273	3.16

in average precision. By further incorporating AMR, SA-ART can self-adapt all the important model parameters, and it is also able to use fewer clusters to better partition data clusters.

6.5. Clustering performance comparison

6.5.1. Overview of algorithms in comparison

The performance of SA-ART is compared with Fuzzy ART and the six state-of-the-art clustering algorithms that are applicable to large-scale sparse data, including:

- **Sparse subspace clustering (SSC)** (Elhamifar & Vidal, 2013): Since SSC is a two-step algorithm, we use the authors' Matlab code[2] to compute the new embedding vectors of data, and use the Python package "sklearn.cluster. SpectralClustering" for the latter step of spectral clustering. It is for a fair comparison with TRR and DSC-Nets, in terms of clustering performance and time cost. In experiments, we manually tuned the number of clusters J, and the subspaces' number n and dimensionality d.
- **Thresholding ridge regression (TRR)** (Peng et al., 2015): TRR is a variant of SSC. For the same reason, we also use the authors' Matlab code[3] to compute the new embedding vectors of data, and use the Python package "sklearn.cluster. SpectralClustering" for spectral clustering. In experiments, the number of clusters J and the balancing parameter λ_{TRR} were manually tuned.
- **Entropy weighting k-means (EWKM)** (Jing et al., 2007): EWKM was implemented in python 3.6. In experiments, the number of clusters J and the entropy weighting parameter γ were manually tuned.
- **Feature grouping k-means (FGKM)** (Chen et al., 2012): FGKM was implemented in python 3.6. In experiments, the number of clusters J, the group weighting parameter λ_{FGKM}, and the feature weighting parameter η were manually tuned.
- **Biclustering ARTMAP (BARTMAP)** (Xu & Wunsch II, 2011): BARTMAP was implemented in python 3.6. In experiments, the threshold for gene cluster matching η, the learning rates β_a and β_b and the vigilance parameters ρ_a and ρ_b for ART$_a$ and ART$_b$, respectively, were manually tuned.
- **Deep Subspace Clustering Networks (DSC-Nets)** (Ji et al., 2017): The autoencoder framework was implemented in Python 3.6 using Pytorch 0.41 with Cuda 9.0. Following the original paper, both the encoder and decoder have three layers. However, due to the difference in problem domains, we used three fully-connected ones. The encoder is of 4096–4096–2048 neurons, each of which, except the last one, is followed by a batch normalization and a ReLU layer. The self-expressive layer has 512 neurons. The decoder reverses the operations. It has 2048–4096–4096 neurons, each of which, except the first one, is followed by a batch normalization and a ReLU layer. Adam is used as optimizer for the autoencoder. In experiments, we manually tuned batch size bz and learning rate lr with decay rate dc. The following spectral clustering was also implemented using the Python package "sklearn.cluster.SpectralClustering", and the number of clusters J was manually tuned.

[2] Matlab code for SSC is available at http://www.vision.jhu.edu/code/.

[3] Matlab code for TRR is available at http://pengxi.me/publications/.

6.5.2. Algorithm parameter selection

This section details our procedures of parameter selection for all algorithms. It is notable that the eight algorithms either use Fuzzy ART or k-means to obtain to final clustering results. Specifically, BARTMAP and SA-ART are based on Fuzzy ART; EWKM and FGKM are based on k-means; SSC, TRR, and DSC-Nets extend from spectral clustering, which also uses k-means to cluster the spectral embeddings. Therefore, identifying the suitable hyper-parameters of k-means and Fuzzy ART is the first and the most important step.

To achieve this, we first tuned the parameters of Fuzzy ART, i.e. the learning rate β and the vigilance parameter ρ. It is mainly because Fuzzy ART is fast and does not require to specify the number of clusters. Besides, Fuzzy ART is easy to tune since it uses solely a ratio value of ρ to control the minimum intra-cluster similarity. We followed a typical approach (Meng & Tan, 2014) to use a moderate value for learning rate, say $\beta = 0.6$, and tune the value of ρ until Fuzzy ART generated a small amount of small clusters in the first epoch. For example, on the ImageNet dataset, we started from an empirical value of $\rho = 0.3$ and one epoch of running cost only about 20 s. Subsequently, we increased the value of ρ by 0.05 to find whether this resulted in a significant increase in the number of small clusters generated. Since each category of the ImageNet dataset has 1000 data objects on average, we empirically define a small cluster as that has less than 100 data objects. After several rounds of running, $\rho = 0.6$ was identified to be the most suitable setting, since it led to the generation of 13 small clusters while this number increased to 38 at $\rho = 0.65$. The last step is to select the best setting by fine-tuning the value of ρ and β and evaluating the clustering performance. Finally, $\rho = 0.6$ and $\beta = 0.6$ were selected. The performance of Fuzzy ART under this setting is reported in Table 2.

The setting of Fuzzy ART is directly applicable to ART$_a$ of BARTMAP and SA-ART as a starting point, since both aim to identify data clusters. Regarding the parameter turning for ART$_b$ of BARTMAP, we repeated the method for Fuzzy ART as described in the paragraph above, i.e. using $\beta = 0.6$ and tuning the value of ρ, since no prior knowledge and labels were available for clustering features. Having a reasonable setting of ART$_b$, we tuned the value of the matching threshold η. This also requires a further fine-tuning for ART$_a$ and ART$_b$. The parameter setting for SA-ART requires a further selection from scratch for the balancing parameter λ and the restraint parameter δ in AMR. Luckily, SA-ART is generally robust to these two parameters, since both are weighting parameters, rather than those that define the properties of cluster, such as ρ (see Section 5.4 for a discussion on parameter selection and Fig. 4(a) for the sensitivity of SA-ART to λ). Therefore, SA-ART solely requires a fine-tuning for ρ following the method for Fuzzy ART.

The parameter selection for ART variants makes that for EWKM and FGKM easier, since the range for a suitable number of clusters is identified. As such, we started with moderate settings of the

Table 2
The clustering performance (*mean* $\pm$ *std*) of algorithms under the best settings on the ImageNet and BlogCatalog datasets in terms of average precision (P), average recall (R), and F1 score (F). **Model settings (Using two values for one parameter means different values for the two datasets were used):** SSC ($J = 300, 400$; $n = 300, 400$; $d = 200, 1000$), TRR ($J = 350, 400$; $\lambda_{TRR} = 1$), EWKM ($J = 250, 400$; $\gamma = 1$), FGKM ($J = 300, 400$; $\lambda_{FGKM} = 1$, $\eta = 4$), BARTMAP ($\eta = 0.2$; $\beta_a = \beta_b = 0.6$; $\rho_b = 0.8$; $\rho_a = 0.6, 0.7$ ($J = 362, 474$)), DSC-Nets ($J = 400, 450$; 4096–4096–512; $bz = 128$; lr = 1e-5; $dc = 0.1$), Fuzzy ART ($\beta = 0.6$; $\rho = 0.6, 0.7$ ($J = 358, 483$)), and SA-ART ($\rho = 0.7, 0.75$ ($J = 349, 460$)).

		SSC	TRR	EWKM	FGKM
ImageNet	P	0.347 ± 0.049	0.382 ± 0.041	0.408 ± 0.035	0.385 ± 0.056
	R	0.145 ± 0.022	0.156 ± 0.032	0.185 ± 0.022	0.147 ± 0.041
	F	0.181 ± 0.032	0.196 ± 0.028	0.226 ± 0.029	0.192 ± 0.033
BlogCatalog	P	0.573 ± 0.028	0.607 ± 0.036	0.590 ± 0.026	0.612 ± 0.035
	R	0.221 ± 0.025	0.235 ± 0.031	$\mathbf{0.238 \pm 0.031}$	0.224 ± 0.022
	F	0.300 ± 0.021	0.311 ± 0.027	0.309 ± 0.019	0.327 ± 0.028
		BARTMAP	DSC-Nets	Fuzzy ART	SA-ART
ImageNet	P	0.352 ± 0.053	0.413 ± 0.039	0.285 ± 0.047	$\mathbf{0.449 \pm 0.033}$
	R	0.119 ± 0.023	$\mathbf{0.193 \pm 0.028}$	0.102 ± 0.021	0.129 ± 0.036
	F	0.184 ± 0.047	0.249 ± 0.029	0.166 ± 0.026	$\mathbf{0.252 \pm 0.037}$
BlogCatalog	P	0.616 ± 0.019	0.628 ± 0.026	0.588 ± 0.031	$\mathbf{0.645 \pm 0.026}$
	R	0.184 ± 0.027	0.223 ± 0.024	0.158 ± 0.024	0.176 ± 0.028
	F	0.292 ± 0.025	0.325 ± 0.021	0.285 ± 0.031	$\mathbf{0.337 \pm 0.022}$

entropy weighting parameter $\gamma = 1$ for EWKM and the group and feature weighting parameters $\lambda_{FGKM} = 1$ and $\eta = 1$, respectively, for FGKM. Subsequently, their performance was scanned by varying the number of clusters J from 200 to 500, with an interval of 50. Having a suitable value range of J, all parameters were fine-tuned to obtain the best setting. Notably, unlike ART variants that need only one epoch of running for parameter selection, EWKM and FGKM should run 20 epochs since the k-means algorithm requires a number of iterations to achieve a reasonable clustering result. Luckily, their computation costs are dominated by k-means, which are around 30 s to complete one epoch.

SSC, TRR, and DSC-Nets follow a similar procedure of (1) deriving new data representations and (2) using spectral clustering to obtain clustering results. The latter requires only the number of clusters J, which we already have empirical priors. In the first step, SSC needs to specify the number of subspaces n and their dimensionalities d. Both should be enumerated to find the proper settings according to the final clustering performance. Therefore, the time required for assessing a setting is about one hour (see Fig. 6(a) for an instance). For the first trial, we empirically set n to be the number of categories, i.e. 50 for ImageNet and 147 for BlogCatalog, and d to be 1/10 of the original features, i.e. 100 for ImageNet and 1782 for BlogCatalog. The number of clusters $J = 300$ is used, which works well for k-means-based algorithms. Subsequently, by fixing J and d, we tuned the number of subspaces n, since it profiles the basis vectors available for the representation of data objects. With a list of proper values of n, d was tuned, followed by the fine-tuning of all the three parameters. TRR is much easier since it has only a balancing parameter λ_{TRR}. We followed the experimental analysis in the original paper to evaluate the performance of TRR with $\lambda_{TRR} \in \{1e-3, 0.01, 0.1, 1, 10\}$ and $J = 300$, followed by a fine-tuning of J.

Regarding DSC-Nets, we first used the three-layer autoencoder as described in the last section. The learning rate was chosen from lr = {1e-5, 1e-4, 1e-3}, with $dc = 0.1$ for every four epochs (This is a commonly-used strategy for Adam optimization). The batch size was chosen from bz ={64, 128, 256, 512}. With a fixed $J = 300$, the final clustering performance was used for parameter selection. We also investigated to use the self-expressive layer of 1024 neurons and a more compact autoencoder network of 2048–2048–1024 neurons. Difference from the other algorithms, training the autoencoder network requires GPU acceleration. The entire training process requires 12 epochs to converge, each of which costs 20 s using a single Tesla V100 GPU of 16 GB. However, using CPU can be 20–40 times slower. Notably, a pretrain-and-finetune strategy is used in the original paper. That is, two models should be trained. The one without the self-expressive loss is first trained to learn the proper encoding–decoding mappings. Subsequently, the second model loads the parameters of the first model, and finetunes them under the self-expressive loss. In this case, the learning rate for the second model should be much lower, say 100 times, than the first model.

6.5.3. *Clustering result analysis*

Table 2 reports the performance of each algorithm under the best parameter settings, determined by the F1 score. The performance averages ten runs of each algorithm with different initializations. Specifically, the ART variants, i.e. BARTMAP, Fuzzy ART, and SA-ART, are initialized with randomly-shuffled data, and the others use randomly-selected cluster seeds for k-means. Besides, BARTMAP, Fuzzy ART, and SA-ART run 20 epochs to obtain the final results, while the other algorithms run 100 epochs since the k-means algorithm is used by them and it converges slow for sparse data. Note that SSC, TRR, and DSC-Nets require a first-stage for computing new data representations.

As observed, SA-ART consistently outperforms the others in terms of precision and F1 score while having a lower but reasonable performance in recall. Since SA-ART models local salient distributions for clusters, the higher precision is achieved at the sacrifice of its generalization power. However, F1 score demonstrates the superior overall performance of SA-ART. Specifically, SA-ART is a significantly improvement from its base model, i.e. Fuzzy ART. It significantly outperforms Fuzzy ART on both datasets with a statistical significance of $p < 0.01$ for precision and F1 score. It also achieves a marginal $p < 0.05$ for recall.

Considering the performance on precision, SA-ART significantly outperforms the second best, i.e. DSC-Nets, with $p < 0.05$ on the ImageNet dataset. Regarding the BlogCatalog dataset, its performance is not significantly different from the second best, i.e. DSC-Nets, with $p = 0.12$. However, it is significantly different from the third best, i.e. BARTMAP, with $p < 0.01$. Although SA-ART's performance in recall is statistically significantly lower than the best performance, it is not significantly different from the third best, i.e. TRR, with $p = 0.13$ on the ImageNet dataset. In terms of the overall performance, SA-ART's performance of F1 score on the ImageNet dataset is not significantly different from that of DSC-Nets ($p = 0.75$), but it significantly outperforms the third best, i.e. EWKM, with $p = 0.016$. In terms of the BlogCatalog

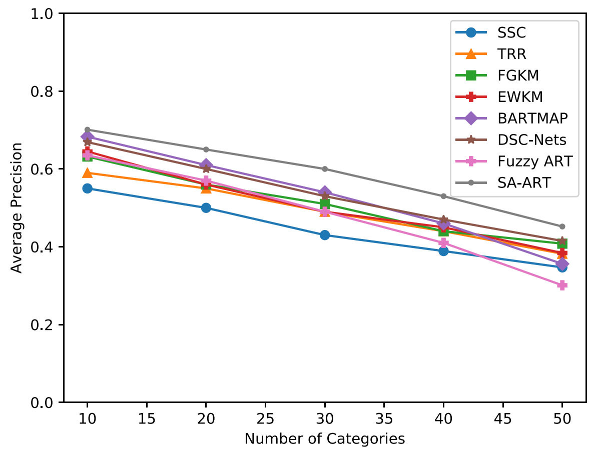

Fig. 5. Curves on performance drop of clustering algorithms when applied to the data with an increasing number of categories on ImageNet dataset.

dataset, SA-ART significantly outperforms TRR with $p = 0.023$ and SSC, EWKM, BARTMAP, and Fuzzy ART with $p < 0.01$.

All algorithms perform worse on the ImageNet dataset. This is expected since the discriminative power of visual features is typically much lower than that of text. Interestingly, Fuzzy ART does not perform well on the ImageNet dataset, but it achieves a performance comparable to SSC, TRR, and EWKM, on the BlogCatalog dataset. This reveals that Fuzzy ART is vulnerable to the sparse and noisy visual features. BARTMAP improves this problem by introducing a constraint on feature associations, while SA-ART further enhances Fuzzy ART's robustness by incorporating salience-aware methods for sparse data.

The algorithms based on spectral clustering, i.e. SSC and TRR, perform similarly but slightly worse that EWKM and FGKM. However, DSC-Nets significantly outperforms them. This shows that sparse data may have a substantial side-effect on the subspace derivation using linear correlation analysis, as discussed in Section 2.1.2. In contrast, it also demonstrates the strength of using non-linear mappings to find the linear correlation coefficients for input data. It is notable that deep learning of feature representation requires sufficient data for training, and it is computationally expensive for large-scale data. Besides, DSC-Nets are still a two-step algorithm. Therefore, light-weight clustering algorithms that directly pass the cluster-aware signals as supervision for deep neural networks should be a new direction for clustering large-scale sparse data.

6.6. Robustness to data complexity

This section investigates the robustness of SA-ART to the increase in the size and the complexity of data. To this end, data of increasing size and the number of categories are fed to SA-ART and the other algorithms in comparison, to assess their performances. All algorithms are under their own best settings as reported in Section 6.5.

As shown in Fig. 5, SA-ART consistently achieves the best performance and exhibits a smoother drop in performance with respect to the increase in the size of data. Fuzzy ART shows a fast drop in performance since it does not have strategies to properly handle noisy data and features. The soft subspace clustering algorithms, i.e. FGKM and EWKM, and the "hard" subspace clustering algorithms, i.e. TRR and DSC-Nets, achieve comparable performance. DSC-Nets shows a relatively stable performance, which indicates the promising of deep learning on unsupervised representation learning for sparse data.

6.7. Time cost comparison

The time cost of SA-ART is compared with its base model, i.e. Fuzzy ART, the spectral-based algorithms (Spectral clustering, SSC, and TRR), and the k-means-based algorithms (k-means, EWKM, and FGKM), on the ImageNet dataset in terms of running time and convergence speed. The time cost of the spectral-based algorithms includes the time for coefficient computing and spectral clustering (We use "sklearn.cluster.SpectralClustering" to alleviate the efficiency difference using Matlab and Python). DCS-Nets involves GPU accelerations, without which the training of the autoencoder is very slow. Therefore, DCS-Nets is not suitable for time cost comparison. BARTMAP has a similar computational cost to Fuzzy ART given a pretrained ART_b. For a fair comparison, each algorithm is forced to run 20 iterations and generate 200 clusters. All algorithms are running on the server with 64 GB memory and 11 processors of Intel(R) Xeon(R) CPU E5-2603 v3 @ 1.60 GHz.

Fig. 6(a) reports the running time of the algorithms on data with an increasing number of categories. As observed, SA-ART and Fuzzy ART have nearly the same time cost, which is much lower than other algorithms. Besides, the feature weighting approach is far more computationally efficient than the subspace learning one for large-scale data when high-performance computing is not accessible. Fig. 6(b) reports the convergence speed of algorithms running on the whole ImageNet dataset. It is observed that SA-ART inherits the fast learning capability of Fuzzy ART and stabilizes with a shorter intra-cluster distance after 16 iterations.

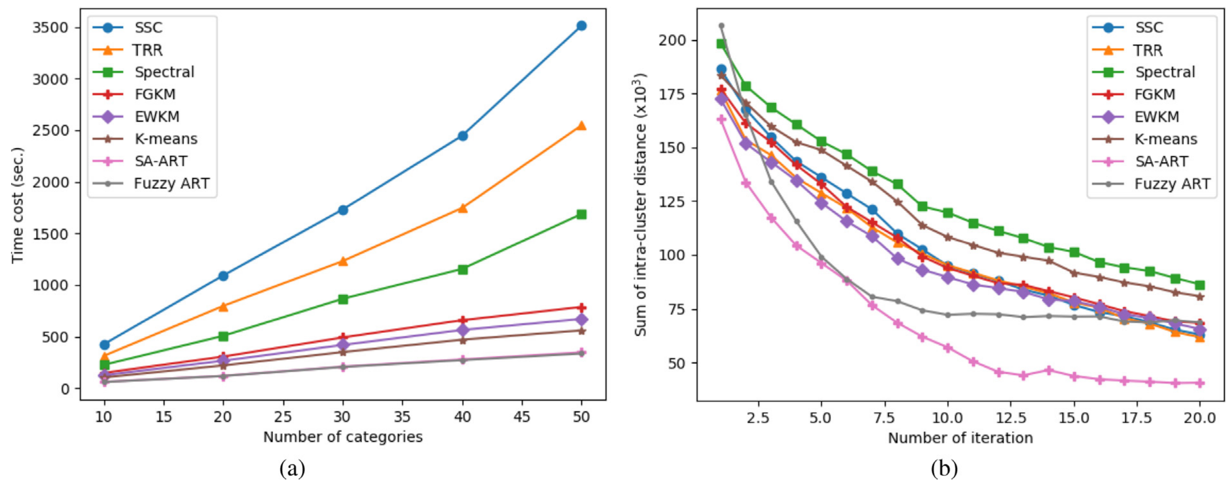

Fig. 6. The (a) time cost and (b) convergence speed of clustering algorithms on ImageNet dataset.

7. Conclusion

This paper presents a soft subspace learning algorithm, i.e. the salience-aware adaptive resonance theory (SA-ART), for clustering large-scale sparse data. SA-ART inherits the advantages of its base model, i.e. Fuzzy ART, including linear time complexity, fast converging ability, and no need for a predefined number of clusters. To discover the cluster-wise salient features, SA-ART adopts three statistical measures and seamlessly incorporates them into the Fuzzy ART's typical procedures. Comparing to Fuzzy ART, SA-ART has two major differences:

1. The shared feature matching strategy integrated in the choice and match functions, as defined in Eqs. (6) and (8). It enables SA-ART to effectively handle noisy features and the computation incurred by high dimensionality.
2. The learning function as defined in Eq. (9). It removes the need for the learning rate β as used in Fuzzy ART. Instead, by modeling the mean values and the upper and lower bounds for cluster weights, the rate of learning is determined by how well the feature value matches the statistics of the corresponding weight. More importantly, rather than monotonous suppression, this adaptation method is able to recover, i.e. increase, the weight values to their statistical means to better handle mismatches.

Experimental results show that SA-ART is promising in terms of its sensitivity to the sole vigilance parameter ρ and its ability to produce clusters with high precision and reasonably complex structures. Besides the progress achieved so far, SA-ART still has a good extensibility to incorporate new theories and approaches. First, the cluster space matching strategy allows SA-ART to model the cluster-wise salient features. However, such robustness to noise is achieved at the cost of sacrificing the generalization power. Therefore, adaptive merging strategies for clusters will significantly improve SA-ART's performance. Secondly, a smart tuning mechanism for vigilance parameter ρ during the initial epochs will help SA-ART in both salient feature modeling and convergence.

Acknowledgments

This research is supported, in part, by the National Research Foundation, Prime Minister's Office, Singapore under its IDM Futures Funding Initiative; in part, by the Ministry of Education Academic Research Fund (MOE AcRF), Singapore; and in part, by the National Research Foundation, Prime Minister's Office, Singapore under its IRC@Singapore Funding Initiative.

References

Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (2005). Automatic subspace clustering of high dimensional data. *Data Mining and Knowledge Discovery*, *11*(1), 5–33.

Assent, I., Krieger, R., Müller, E., & Seidl, T. (2007). Dusc: Dimensionality unbiased subspace clustering. In *IEEE International Conference on Data Mining (ICDM)* (pp. 409–414). IEEE.

Ben-Yosef, M., & Weinshall, D. (2018). Gaussian mixture generative adversarial networks for diverse datasets, and the unsupervised clustering of images, arXiv preprint arXiv:1808.10356.

Bohm, C., Railing, K., Kriegel, H.-P., & Kroger, P. (2004). Density connected clustering with local subspace preferences. In *IEEE international conference on data mining (ICDM)* (pp. 27–34). IEEE.

Boongoen, T., Shang, C., Iam-On, N., & Shen, Q. (2011). Extending data reliability measure to a filter approach for soft subspace clustering. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, *41*(6), 1705–1714.

Carpenter, G. A., & Grossberg, S. (1987a). ART 2: Self-organization of stable category recognition codes for analog input patterns. *Applied Optics*, *26*(23), 4919–4930.

Carpenter, G. A., & Grossberg, S. (1987b). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, *37*(1), 54–115.

Carpenter, G. A., & Grossberg, S. (1990). ART 3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. *Neural Networks*, *3*(2), 129–152.

Carpenter, G. A., & Grossberg, S. (2017). Adaptive resonance theory. In C. Sammut, & G. I. Webb (Eds.), *Encyclopedia of machine learning and data mining* (pp. 24–40). Boston, MA: Springer.

Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H., Rosen, D. B., et al. (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, *3*(5), 698–713.

Carpenter, G. A., Grossberg, S., & Rosen, D. B. (1991a). ART 2-a: An adaptive resonance algorithm for rapid category learning and recognition. *Neural Networks*, *4*(4), 493–504.

Carpenter, G. A., Grossberg, S., & Rosen, D. B. (1991b). Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, *4*(6), 759–771.

Chen, X., Ye, Y., Xu, X., & Huang, J. Z. (2012). A feature group weighting method for subspace clustering of high-dimensional data. *Pattern Recognition*, *45*(1), 434–446.

Chitsaz, E., & Jahromi, M. Z. (2016). A novel soft subspace clustering algorithm with noise detection for high dimensional datasets. *Soft Computing*, *20*(11), 4463–4472.

Damelin, S. B., Gu, Y., Wunsch, D. C., & Xu, R. (2015). Fuzzy adaptive resonance theory, diffusion maps and their applications to clustering and biclustering. *Mathematical Modelling of Natural Phenomena*, *10*(3), 206–211.

De Amorim, R. C., & Mirkin, B. (2012). Minkowski metric, feature weighting and anomalous cluster initializing in k-means clustering. *Pattern Recognition*, *45*(3), 1061–1075.

Deng, Z., Choi, K.-S., Chung, F.-L., & Wang, S. (2010). Enhanced soft subspace clustering integrating within-cluster and between-cluster information. *Pattern Recognition*, *43*(3), 767–781.

Deng, Z., Choi, K.-S., Jiang, Y., Wang, J., & Wang, S. (2016). A survey on soft subspace clustering. *Information Sciences*, *348*, 84–106.

Dhillon, I. S. (2001). Co-clustering documents and words using bipartite spectral graph partitioning. In *ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 269–274). ACM.

Elhamifar, E., & Vidal, R. (2013). Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(11), 2765–2781.

Feng, H., & Qian, X. (2014). Mining user-contributed photos for personalized product recommendation. *Neurocomputing*, *129*, 409–420.

Fern, X. Z., & Brodley, C. E. (2003). Random projection for high dimensional data clustering: A cluster ensemble approach. In *International conference on machine learning (ICML)* (pp. 186–193).

Gan, G., & Ng, M. K.-P. (2015). Subspace clustering with automatic feature grouping. *Pattern Recognition*, *48*(11), 3703–3713.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al. (2014). Generative adversarial nets. In *Advances in neural information processing systems (NIPS)* (pp. 2672–2680).

Heckel, R., & Bölcskei, H. (2015). Robust subspace clustering via thresholding. *IEEE Transactions on Information Theory*, *61*(11), 6320–6342.

Huang, J. Z., Ng, M. K., Rong, H., & Li, Z. (2005). Automated variable weighting in k-means type clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(5), 657–668.

Ji, P., Zhang, T., Li, H., Salzmann, M., & Reid, I. (2017). Deep subspace clustering networks. In *Advances in neural information processing systems (NIPS)* (pp. 24–33).

Jing, L., Ng, M. K., & Huang, J. Z. (2007). An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *IEEE Transactions on Knowledge and Data Engineering*, *19*(8).

Kang, T., Zarringhalam, K., Kuijjer, M., Chen, P., Quackenbush, J., & Ding, W. (2018). Clustering on sparse data in non-overlapping feature space with applications to cancer subtyping. In *IEEE international conference on data mining (ICDM)* (pp. 1079–1084). IEEE.

Kriegel, H.-P., Kröger, P., & Zimek, A. (2009). Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *3*(1), 1.

Li, Z., Liu, J., Tang, J., & Lu, H. (2015). Robust structured subspace learning for data representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *37*(10), 2085–2098.

Lu, C., Feng, J., Lin, Z., Mei, T., & Yan, S. (2019). Subspace clustering by block diagonal representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *41*(2), 487–501.

Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., & Frey, B. (2016). Adversarial autoencoders. In *ICLR 2016 workshop, international conference on learning representations*.

Meng, L., & Tan, A.-H. (2012). Semi-supervised hierarchical clustering for personalized web image organization. In *International joint conference on neural networks (IJCNN)* (pp. 1–8). IEEE.

Meng, L., & Tan, A.-H. (2014). Community discovery in social networks via heterogeneous link association and fusion. In *SIAM International Conference on Data Mining (SDM)* (pp. 803–811). SIAM.

Meng, L., Tan, A.-H., Leung, C., Nie, L., Chua, T.-S., & Miao, C. (2015). Online multimodal co-indexing and retrieval of weakly labeled web image collections. In *International Conference on Multimedia Retrieval* (pp. 219–226). ACM.

Meng, L., Tan, A.-H., & Wunsch, D. C. (2016). Adaptive scaling of cluster boundaries for large-scale social media data clustering. *IEEE Transactions on Neural Networks and Learning Systems*, *27*(12), 2656–2669.

Meng, L., Tan, A.-H., & Xu, D. (2014). Semi-supervised heterogeneous fusion for multimedia data co-clustering. *IEEE Transactions on Knowledge and Data Engineering*, *26*(9), 2293–2306.

Mirkin, B. (2013). *Mathematical classification and clustering, Vol. 11*. Springer Science & Business Media.

Muja, M., & Lowe, D. G. (2014). Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(11), 2227–2240.

Peng, X., Yi, Z., & Tang, H. (2015). Robust subspace clustering via thresholding ridge regression.. In *In AAAI* (pp. 3827–3833).

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). Imagenet: large scale visual recognition challenge. *International Journal of Computer Vision*, *115*(3), 211–252.

Salah, A., Rogovschi, N., & Nadif, M. (2016). Model-based co-clustering for high dimensional sparse data. In *Artificial intelligence and statistics* (pp. 866–874).

Sezgin, M., & Sankur, B. (2004). Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging*, *13*(1), 146–166.

da Silva, L. E. B., Elnabarawy, I., & Wunsch II, D. C. (2019). Dual vigilance fuzzy adaptive resonance theory. *Neural Networks*, *109*, 1–5.

da Silva, L. E. B., & Wunsch, D. C. (2016). An information theoretic ART for robust unsupervised learning. In *International joint conference on neural networks (IJCNN)* (pp. 3023–3029). IEEE.

Song, C., Liu, F., Huang, Y., Wang, L., & Tan, T. (2013). Auto-encoder based data clustering. In *Iberoamerican congress on pattern recognition* (pp. 117–124). Springer.

Tomasev, N., Radovanovic, M., Mladenic, D., & Ivanovic, M. (2014). The role of hubness in clustering high-dimensional data. *IEEE Transactions on Knowledge and Data Engineering*, *26*(3), 739–751.

Vidal, R., & Favaro, P. (2014). Low rank subspace clustering (lrsc). *Pattern Recognition Letters*, *43*, 47–61.

Wang, J., Deng, Z., Choi, K.-S., Jiang, Y., Luo, X., Chung, F.-L., et al. (2016). Distance metric learning for soft subspace clustering in composite kernel space. *Pattern Recognition*, *52*, 113–134.

Wang, X., Tang, L., Gao, H., & Liu, H. (2010). Discovering overlapping groups in social media. In *International conference on data mining (ICDM)* (pp. 569–578). IEEE.

Wang, Y., Wang, Y.-X., & Singh, A. (2019). A theoretical analysis of noisy sparse subspace clustering on dimensionality-reduced data. *IEEE Transactions on Information Theory*, *65*(2), 685–706.

Wang, Y.-X., & Xu, H. (2016). Noisy sparse subspace clustering. *Journal of Machine Learning Research (JMLR)*, *17*(12), 1–41.

Welford, B. (1962). Note on a method for calculating corrected sums of squares and products. *Technometrics*, *4*(3), 419–420.

West, J. D., Wesley-Smith, I., & Bergstrom, C. T. (2016). A recommendation system based on hierarchical clustering of an article-level citation network. *IEEE Transactions on Big Data*, *2*(2), 113–123.

Xia, H., Zhuang, J., & Yu, D. (2013). Novel soft subspace clustering with multi-objective evolutionary approach for high-dimensional data. *Pattern Recognition*, *46*(9), 2562–2575.

Xu, P., Deng, Z., Cui, C., Zhang, T., Choi, K.-S., Suhang, G., et al. (2019). Concise fuzzy system modeling integrating soft subspace clustering and sparse learning. *IEEE Transactions on Fuzzy Systems*.

Xu, R., & Wunsch, D. (2008). *Clustering, Vol. 10*. John Wiley & Sons.

Xu, R., & Wunsch II, D. C. (2011). Bartmap: A viable structure for biclustering. *Neural Networks*, *24*(7), 709–716.

Zhou, J., Chen, L., Chen, C. P., Zhang, Y., & Li, H.-X. (2016). Fuzzy clustering with the entropy of attribute weights. *Neurocomputing*, *198*, 125–134.

Zhou, P., Hou, Y., & Feng, J. (2018). Deep adversarial subspace clustering. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 1596–1604).

Zhu, L., Cao, L., Yang, J., & Lei, J. (2014). Evolving soft subspace clustering. *Applied Soft Computing*, *14*, 210–228.

Zhu, Y., Ting, K. M., & Carman, M. J. (2018). Grouping points by shared subspaces for effective subspace clustering. *Pattern Recognition*, *83*, 230–244.