Deep Multi-Representation Learning for Data Clustering

Mohammadreza Sadeghi*, and Narges Armanfard

***Abstract*—Deep clustering incorporates embedding into clustering in order to find a lower-dimensional space suitable for clustering tasks. Conventional deep clustering methods aim to obtain a single global embedding subspace (aka latent space) for all the data clusters. In contrast, in this paper, we propose a deep multi-representation learning (DML) framework for data clustering whereby each difficult-to-cluster data group is associated with its own distinct optimized latent space and all the easy-to-cluster data groups are associated with a general common latent space. Autoencoders are employed for generating cluster-specific and general latent spaces. To specialize each autoencoder in its associated data cluster(s), we propose a novel and effective loss function which consists of weighted reconstruction and clustering losses of the data points, where higher weights are assigned to the samples more probable to belong to the corresponding cluster(s). Experimental results on benchmark datasets demonstrate that the proposed DML framework and loss function outperform state-of-the-art clustering approaches. In addition, the results show that the DML method significantly outperforms the SOTA on imbalanced datasets as a result of assigning an individual latent space to the difficult clusters.**

***Index Terms*—Data clustering, autoencoder, multiple representation learning, cluster-specific autoencoders.**

I. Introduction

In many science and real-world applications, obtaining useful information about the class (aka label) of data points is hard or expensive. There are algorithms with the goal of handling the sparse labeled data issue, such as weakly supervised [1, 2] and semi-supervised methods [3, 4] that use only a few labeled data during the training phase. Unsupervised clustering methods endeavor to extract valuable knowledge only from unlabeled data samples in a fully unsupervised manner. Clustering addresses many difficulties in practical applications such as astronomical information extraction [5], medical data analysis [6], gene sequencing [7], and information retrieval [8, 9, 10, 11]. The ultimate goal of clustering algorithms is to find similar and dissimilar groups of data samples based on a similarity metric.

Although extensive varieties of clustering methods have been recently proposed, the two conventional algorithms k-means [12] and fuzzy c-means [13] are still utilized in many practical applications [14, 15, 16] because of their simplicity. However, these algorithms do not show promising clustering performance when the data samples are not evenly scattered around cluster centroids. Furthermore, they suffer from the curse of dimensionality [17] when dealing with high-dimensional data; this causes their failure in many recent applications where data samples have numerous features [18].

Recently, deep learning-based clustering algorithms have been significantly investigated in various applications such as image segmentation [19], network embedding [20], face recognition [21], and machine vision [22]. The goal of these algorithms is to find an optimal lower-dimensional representation (aka latent representation) for the input data points, in which data clusters can be separated by performing a traditional clustering algorithm such as k-means or fuzzy c-means.

To obtain the optimal latent space in an unsupervised manner, researchers have proposed different autoencoder (AE) structures [23, 24, 25]. In general, an AE comprises two networks: an encoder network that projects the original input space onto a lower-dimensional space (aka latent space) and a decoder network that aims to reconstruct the original input space using the latent representation of the data points generated by the encoder network. Encoder and decoder networks are trained to minimize data reconstruction loss. More recent algorithms try to make the AE's latent space more suitable for data clustering by minimizing data clustering loss besides the data reconstruction loss [26, 27, 28]. However, all suffer from the following drawbacks. 1) at the beginning of each training epoch, data points are assigned to the most probable cluster – i.e., a crisp cluster assignment is performed at the beginning of each epoch. They then assume that the obtained crisp assignments are correct and compute the clustering loss accordingly, e.g. see [26, 27, 28]. This assumption may mislead the AE's training phase due to the unsupervised nature of the clustering task, where the true cluster assignments of the data samples are unknown. This issue would be more crucial when the non-crisp (aka soft) K-dimensional cluster assignment vector (acquired before converting it to the crisp K-dimensional one-hot vector) is far from the K-dim one-hot vector obtained by the crisp assignment. K is the number of clusters. 2) All the traditional deep-learning-based methods ignore the different characteristics of the data clusters and assign a single common latent space, obtained from the trained AE, to all the data clusters. In fact, they assume that a single latent space can optimally characterize variations over clusters.

In this paper, we offer an alternative to the conventional deep learning-based clustering approaches by introducing the novel concept of deep multi-representation learning (DML) framework where the representation varies across clusters in a

M. Sadeghi and N. Armanfard are with the Department of Electrical and Computer Engineering, McGill University, Montreal, QC, Canada.
E-mail: {mohammadreza.sadeghi, narges.armanfard}@ mcgill.ca
M. Sadeghi and N. Armanfard are also with Mila-Quebec AI Institute, Montreal, QC, Canada.

* Corresponding author.

manner that optimally adapts to the variations within a cluster. We propose that an enhanced description of the original sample space could be obtained by allowing the "difficult" clusters to be associated with their own distinct representation space, which is optimized for that specific cluster. In this study, a cluster is considered difficult if the Euclidean distance of its corresponding center is too close to at least one of the other cluster centers, and a cluster is considered an "easy" one if its corresponding center is far from all the other cluster centers. DML trains a common autoencoder, called General AE, for the easy clusters and cluster-specific autoencoders for the difficult ones. More specifically, the DML's mission is to create distinct latent spaces in which samples of the corresponding cluster(s) are closely packed around their associated center(s). DML is a general framework capable of improving the performance of any existing reliable AE-based clustering algorithm $\mathcal{M}$ that aims at gathering the data points around their corresponding cluster centers, e.g., [29, 28, 27]. To this end, all the DML autoencoders are initialized by the $\mathcal{M}$'s autoencoder network. The DML's autoencoders are then updated to improve the compactness of their corresponding cluster(s) through minimizing a novel loss consisting of *weighted* reconstruction and clustering losses of the data points. For each autoencoder, higher weights are assigned to the samples that are more probable to belong to the corresponding cluster(s). Minimizing the reconstruction losses is to maintain the local structure of the data points [26, 30] and to avoid too much manipulation of the latent spaces by the clustering loss. Minimizing the clustering loss encourages the data points to sit close to their corresponding cluster centers hence helping in providing AE latent spaces that are more appropriate for clustering.

To summarize, the main contributions of this work are:

- We propose the novel concept of deep multi-representation learning, which performs the data clustering task by employing multiple autoencoders: a general autoencoder for the easy clusters and cluster-specific autoencoders for the difficult ones.
- We develop the training procedure of the cluster-specific AEs to discover the appropriate latent representation for the tangled clusters.
- We devise a general AE to obtain representations for the easy data clusters that are well separated from the other clusters.
- We propose to train each of the DML's autoencoders with a novel loss function consisting of weighted reconstruction and clustering losses, where weights are assigned based on the similarity between the data points and the corresponding cluster centers.
- Our extensive set of experiments on nine benchmark datasets demonstrate the effectiveness of the proposed DML framework.

II. Related works

The scope of this paper does not allow for a comprehensive review of previous works. However, we recommend referring to [31, 32, 33] for an in-depth analysis of non-deep learning-based clustering and subspace learning methods. The subsequent discussion will mainly focus on reviewing some relevant deep clustering methods.

A. DNN-based methods

Deep neural networks (DNN) have been broadly investigated to handle clustering tasks. These algorithms aim to train a DNN model in an unsupervised manner. For example, [34] finds a new latent space in which the predicted representation of augmented data samples is as close as possible to those of original data points by maximizing information-theoretic dependency between data points and their predicted representations. RUC [35] presents a clustering framework that could enhance the clustering performance of other algorithms. RUC consists of two steps. In the first step, it purifies datasets by assigning labels to the most confident samples. Then in the second phase, it retrains a neural network using the refined dataset. [36] obtains indicator features for each data sample by training a DNN model in an unsupervised manner. Then, the data are assigned to different clusters based on the extracted feature vectors. Recently, unsupervised contrastive learning algorithms have been widely studied [37, 38]. These algorithms define negative and positive pairs by applying data augmentation. They then project data samples onto a new feature space and minimize (maximize) the distance between positive (negative) pairs in the new space. A thorough review of DNN-based methods can be found in [39]. Among the DNN-based algorithms, the AE-based and generative-based clustering algorithms are widely used in practical applications.

B. AE-based methods

AE-based algorithms use a deep AE to map the original data space to a lower-dimensional latent space. In some conventional algorithms [40, 41], the clustering task is separated from learning the lower-dimensional feature space. [40] trains an AE by imposing a locality-preserving constraint to obtain the lower-dimensional data space. It then applies k-means to define clusters. Graph clustering [42, 43] is an essential branch of clustering, which aims to separate nodes of a graph based on the relationship between their connections (aka edges). [41] utilizes a deep AE to find lower-dimensional features for a graph. It then uses k-means to assign graph nodes to different clusters.

More recent AE-based algorithms simultaneously map the data points onto a lower dimensional space and perform clustering in the lower dimensional space. This is to further enhance the final clustering performance. For instance, deep embedding clustering (DEC) [29] trains a stacked AE layer by layer, where each layer is a denoising AE that aims to reconstruct the previous layer's output after a random corruption. It then discards the decoder part and fine-tunes the encoder part by minimizing a Kullback–Leibler (KL) divergence between the distribution of soft assignments and a pre-determined target distribution. DEC utilizes Students' t-distribution for finding soft assignments to measure the similarity between data points and cluster centers. Because of the unsupervised nature of the clustering problem, the true target distribution of the data points is unknown. DEC suggests

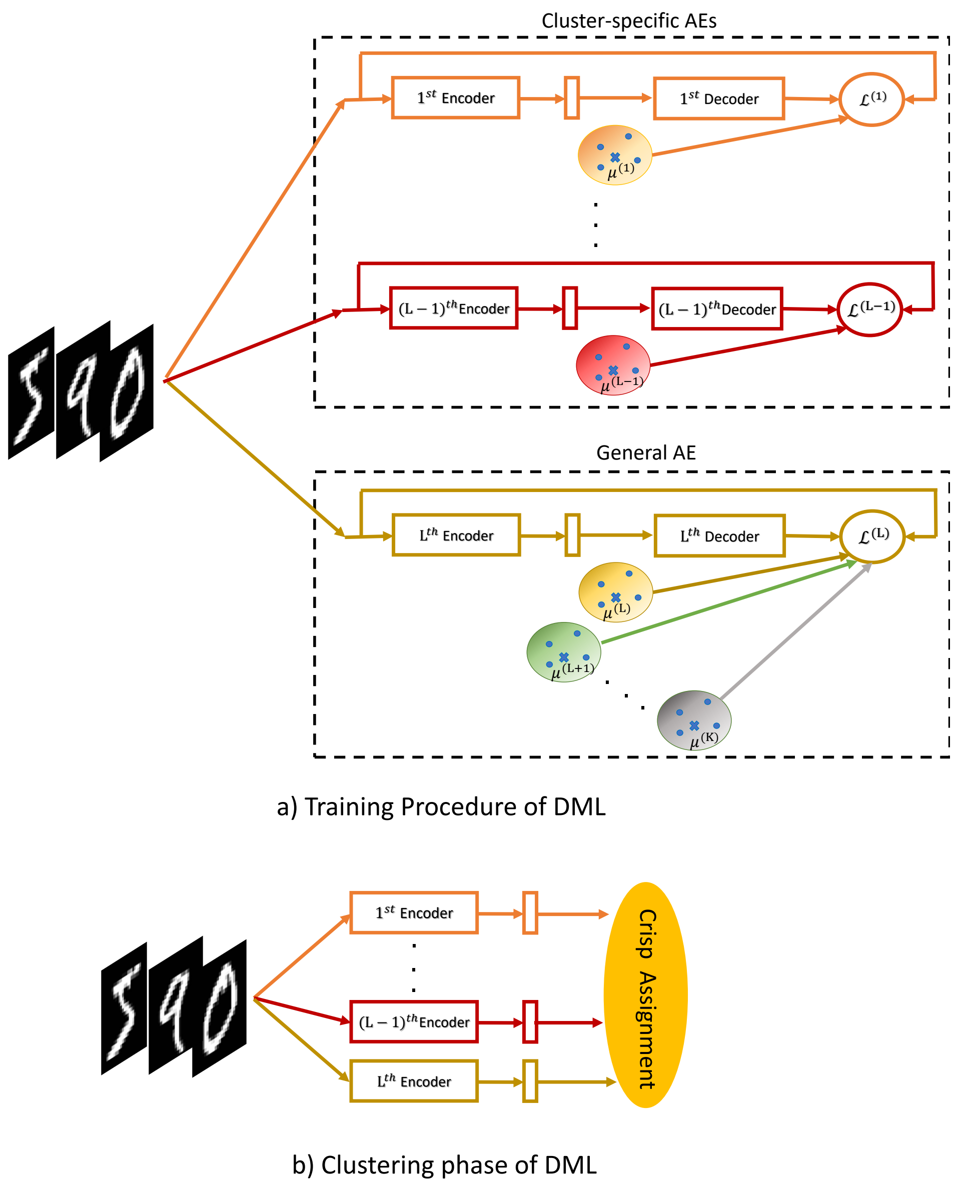


Fig. 1. Block diagram of the proposed DML framework. (a) Training scheme of DML. (b) Final crisp cluster assignment phase.

an arbitrary target distribution defined based on the squared of the soft assignments. Despite DEC, a few recently proposed algorithms, i.e., [28, 27, 26, 30], take advantage of the decoder part in addition to the encoder to retain the data locality structure. For example, improved deep embedding clustering (IDEC) [26] enhances the clustering performance of DEC by minimizing the reconstruction loss of an AE besides the KL divergence of DEC. Improved deep embedding clustering with fuzzy supervision (IDECF) [30] improves DEC by estimating target distribution using a fully connected network called deep fuzzy c-means network. It also considers the reconstruction loss besides the KL divergence loss in its optimization problem. Deep clustering network (DCN) [28] simultaneously embeds data points in a lower-dimensional space and performs clustering. DCN minimizes a weighted combination of the reconstruction loss and the objective function of k-means to find a k-means-friendly latent space where clusters are located around the cluster centers. DCN separately updates the network's weights and cluster centers. The latter is by finding an optimal solution to a discrete optimization problem. Deep k-means (DKM) [27] has the same objective function as DCN. However, it considers a continuous optimization problem for updating the AE's parameters and cluster centers.

C. Deep generative-based algorithms

Variational autoencoders (VAEs) [44] and Generative adversarial networks (GANs) [45] are the two popular deep generative models which could be effective for finding data distribution and performing the data clustering task. For example, variational deep embedding (VaDE) [46] proposes a data generative procedure using Gaussian Mixture Model (GMM) and a deep neural network (DNN). VaDE aims to find the distribution of data points by maximizing the likelihood of a given data sample. It then assigns data points to the most probable cluster. Deep adversarial clustering (DAC) [47] is another generative model that employs an adversarial autoencoder (AAE)[48] for the clustering task. AAE's generator generates a latent code and attempts to deceive the discriminator into believing that the latent code is drawn from the specified distribution. The discriminator, on the other hand, predicts whether a given latent code was generated by the autoencoder or a random vector drawn from the normal distribution. DAC's loss function consists of a reconstruction term, Gaussian mixture model likelihood, and the adversarial objective.

GAN is a training method for generative models that frames the problem as a supervised learning task with two sub-models: the generator model, which is trained to generate new samples, and the discriminator model, which attempts to categorize examples as real or fake (generated). Many GAN-based clustering algorithms [39] have been proposed. [49] presents an unsupervised GAN-based algorithm for learning disentangled representations through maximizing the mutual information between a subset of latent variables and the observation. GAN mixture model (GANMM) [50] extends the concept of GMM by developing a GAN model for each cluster. Vanishing gradients and mode collapse plague the GAN-based clustering algorithms.

III. Proposed method

Consider a K-clustering problem that aims to divide the given dataset $X = \{x_1, x_2, ..., x_N\}$ with N samples into K disjoint groups (aka clusters), where the ith data sample is denoted by x_i, and K is the predefined number of groups.

The DML aims at improving the clustering performance of the existing AE-based clustering method $\mathcal{M}$ that tries to gather data samples around their corresponding cluster center in the single common latent space of the $\mathcal{M}$'s AE. Generally, the clustering performance of $\mathcal{M}$ drops when there are tangled clusters in the AE's latent space. To improve the performance of $\mathcal{M}$, DML assigns distinct autoencoders to such difficult clusters and a common general autoencoder to the well-separated clusters. A cluster is considered as difficult if the Euclidean distance (in the $\mathcal{M}$'s latent space) between its center and the closest cluster center is less than threshold τ which is a user-settable parameter. Non-difficult clusters are considered easy.

A. Notation clarification and initialization

Assume there are L-1 difficult clusters (hence there are K-L-1 easy clusters), where L $<$ K. DML trains an individual AE for each of the difficult clusters and a single common AE for the easy clusters. We refer to the l^{th} AE of DML as AE$^{(l)}, l = 1, \dots, \mathrm{L}$; where AE$^{(\mathrm{L})}$ denotes the general AE assigned to the easy clusters and the remaining AEs, i.e., AE$^{(l)}$ $l = 1, \dots, \mathrm{L}-1$, are assigned to the L$-1$ difficult clusters.

The cluster centers, i.e. $\mu^{(k)}$ $k = 1, \dots, \mathrm{K}$, are initialized to the centers obtained by $\mathcal{M}$. Consequently, all the DML's autoencoders are initialized with the $\mathcal{M}$'s autoencoder.

The AE of $\mathcal{M}$ is denoted by $\tilde{\mathrm{AE}}$. The encoder and decoder of $\tilde{\mathrm{AE}}$ are respectively denoted by $\tilde{f}(.)$ and $\tilde{g}(.)$. Representation of X in the latent space of $\tilde{\mathrm{AE}}$ is denoted by $\tilde{U} = \{\tilde{u}_1, ..., \tilde{u}_N\}$, where $\tilde{u}_i = \tilde{f}(x_i; \tilde{\theta}_e) \in \mathbb{R}^d$, d indicates dimension of the latent space, and $\tilde{\theta}_e$ represents parameters of the encoder network. The reconstructed output of $\tilde{\mathrm{AE}}$ is shown by $\tilde{x}_i = g(\tilde{u}_i; \tilde{\theta}_d)$, where $\tilde{\theta}_d$ denotes the decoder parameters of $\tilde{\mathrm{AE}}$.

The encoder and decoder parameters of the l^{th} autoencoder of DML, i.e. AE$^{(l)}$, are respectively shown by $f^{(l)}(.)$ and $g^{(l)}(.)$. $U^{(l)} = \{u_1^{(l)}, ..., u_N^{(l)}\}$ denotes the representation of X in the latent space of AE$^{(l)}$, where $u_i^{(l)} = f^{(l)}(x_i; \theta_e^{(l)}) \in \mathbb{R}^d$ and $\theta_e^{(l)}$ shows AE$^{(l)}$'s encoder parameters. Also, the reconstructed output of AE$^{(l)}$ is denoted by $\hat{x}_i^{(l)} = g^{(l)}(u_i^{(l)}; \theta_d^{(l)})$, where $\theta_d^{(l)}$ denotes the decoder parameters of AE$^{(l)}$.

In the following, for simplicity, we use the notation $(.)^{(l_k)}$ to refer to the DML's latent space associated with the k^{th} data cluster. In general, if $\mathcal{D} = \{d_1, ..., d_{\mathrm{L}-1}\}$ is the set of difficult clusters and and $\mathcal{E} = \{e_1, ..., e_{\mathrm{K}-\mathrm{L}+1}\}$ is the set of easy clusters, then l_k can be obtained as below:

$$l_k = \sum_{i=1}^{\mathrm{L}-1} i\mathbb{1}\{k = d_i\} + \mathrm{L}\mathbb{1}\{k \in \mathcal{E}\}. \tag{1}$$

As an instance, assume there are four data clusters, i.e., $k = \{1, 2, 3, 4\}$ and K $= 4$, of which the second and third ones (i.e., $k = \{2, 3\}$) are difficult, and the first and fourth clusters (i.e., $k = \{1, 4\}$) are easy. Therefore, in this example, L $= 3$, $\mathcal{D} = \{2, 3\}$, $\mathcal{E} = \{1, 4\}$, and l_k for k = 1, 2, 3 and 4 are respectively equal to 3, 1, 2 and 3 – e.g., the DML's AE associated to $k = 2$ is AE$^{(1)}$, and the representation of x_i in the latent space corresponding to the data cluster $k = 2$ is $u_i^{(1)}$ because $l_2 = 1$. Note that $l_k \in \{1, 2, .., \mathrm{L}\}$.

B. Soft assignment

To focus AE$^{(l_k)}$ on creating a latent space specialized in the k^{th} cluster, we devise a novel loss function that navigates the training process of the AE$^{(l_k)}$'s encoder and decoder networks to pay more attention to the data points similar to $\mu^{(k)}$. To measure the similarity between a data point x_i and the cluster center $\mu^{(k)}$, denoted by p_{ik} where $\mathbf{p}_i = [p_{i1}, \dots, p_{iK}]$, we propose to solve the optimization problem shown in (2a) where $m \geq 1$ is the level of fuzziness [1]. As is shown in (2b), we use the Lagrangian multiplier method [51] to solve (2a) where the Lagrange multiplier γ is computed by substituting p_{ik} from (2b) in the constraint of (2a), as is shown in (2f). The final value for the similarity of data point x_i to the k^{th} cluster, i.e., p_{ik}, is obtained by substituting γ from (2f) in (2b), as is shown in (2g). It can be seen that samples that are closer to $\mu^{(k)}$, in the corresponding latent space $U^{(l_k)}$, take higher

[1](2a) is inspired by the fuzzy c-means clustering method; the main difference is that in (2a) there are multiple representations for a single data point while in fuzzy c-means each data point has a single representation.

similarity values.

$$\min_{p_{ik}} \sum_{k=1}^{K} p_{ik}^{m}||u_i^{(l_k)} - \mu^{(k)}||_2^2 \quad \textbf{s.t.} \quad \sum_{k=1}^{K} p_{ik} = 1 \tag{2a}$$

$$S = \sum_{k=1}^{K} p_{ik}^{m}||u_i^{(l_k)} - \mu^{(k)}||_2^2 - \gamma(\sum_{k=1}^{K} p_{ik} - 1)$$
$$\xrightarrow{\frac{\partial S}{\partial p_{ik}}=0} \quad p_{ik} = \left(\frac{\gamma}{m||u_i^{(l_k)}-\mu^{(k)}||_2^2}\right)^{\frac{1}{m-1}} \tag{2b}$$

since we know $\sum_{k=1}^{K} p_{ik} = 1$, we substitute the value of p_{ik} by the value we obtained in 2b; hence:

$$\sum_{k=1}^{K} p_{ik} = \quad 1 \xrightarrow{2b} \sum_{k=1}^{K}\left(\frac{\gamma}{m||u_i^{(l_k)}-\mu^{(k)}||_2^2}\right)^{\frac{1}{m-1}} = 1 \tag{2c}$$

$$\rightarrow \quad \sum_{k=1}^{K}\left(\frac{\gamma}{m}\right)^{\frac{1}{m-1}}\left(\frac{1}{||u_i^{(l_k)}-\mu^{(k)}||_2^2}\right)^{\frac{1}{m-1}} = 1 \tag{2d}$$

$$\rightarrow \quad \left(\frac{\gamma}{m}\right)^{\frac{1}{m-1}}\left(\sum_{k=1}^{K} \frac{1}{||u_i^{(l_k)}-\mu^{(k)}||_2^2}\right)^{\frac{1}{m-1}} = 1 \tag{2e}$$

$$\rightarrow \quad \gamma = m\left(\frac{1}{\sum_{k=1}^{K}\frac{1}{||u_i^{(l_k)}-\mu^{(k)}||_2^{2/(m-1)}}}\right)^{m-1} \tag{2f}$$

if we substitute the value we obtained for γ in (2b), we have:

$$\xrightarrow{2f} \quad p_{ik} = \frac{\frac{1}{||u_i^{(l_k)}-\mu^{(k)}||_2^{2/(m-1)}}}{\sum_{j=1}^{K}\frac{1}{||u_i^{(l_j)}-\mu^{(j)}||_2^{2/(m-1)}}} \tag{2g}$$

Note that, since $\sum_{k=1}^{K} p_{ik} = 1$, one can also think of p_{ik} as a soft assignment of data point x_i to the k^{th} cluster; hence, the two terms "similarity" and "soft assignment" are interchangeable throughout this paper.

C. Training Procedure

As discussed before, we propose to use weighted samples when training the autoencoder corresponding to the k^{th} cluster, i.e., AE$^{(l_k)}$. We consider the soft assignment p_{ik} (defined in Section III-B) as the weight of sample x_i when training AE$^{(l_k)}$. This is to realize the idea of assigning higher weights to the samples closer to the target cluster center $\mu^{(k)}$, in the corresponding latent space $U^{(l_k)}$.

We define the loss function $\mathcal{L}^{(l_k)}$, shown in (3), to be minimized when training AE$^{(l_k)}$, where $\mathcal{L}_r^{(l_k)}$ and $\mathcal{L}_c^{(l_k)}$ respectively denote the weighted reconstruction and clustering losses, and λ is a hyperparameter that indicates the effect of the clustering loss in the networks' training. AE$^{(l_k)}$'s parameters, i.e. $\theta_e^{(l_k)}$and $\theta_d^{(l_k)}$for $k = 1, \ldots,$ K, are optimized in an end-to-end manner using the back-propagation algorithm while minimizing $\mathcal{L}^{(l_k)}, k = 1, \ldots,$ K.

$$\mathcal{L}^{(l_k)} = \mathcal{L}_r^{(l_k)} + \lambda\mathcal{L}_c^{(l_k)} \tag{3a}$$
$$\mathcal{L}_r^{(l_k)} = \sum_{x_i\in\mathfrak{B}} p_{ik}^{m}||x_i - \hat{x}_i^{(l_k)}||_2^2 \tag{3b}$$
$$\mathcal{L}_c^{(l_k)} = \sum_{x_i\in\mathfrak{B}} p_{ik}^{m}||u_i^{(l_k)} - \mu^{(k)}||_2^2 \tag{3c}$$

Most of the existing deep-clustering methods (see Section II) only consider the reconstruction loss (to be minimized) in the hope of making the latent space more discriminative for data clustering, while the reconstruction loss has no substantial connection with the clustering performance. The proposed loss function shown in (3) considers both of the influential factors reconstruction and clustering performances through incorporating their corresponding losses. This allows AE$^{(l_k)}$ to simultaneously learn a feature representation and compact data points with similar latent representations around their corresponding cluster center.

By incorporating the soft assignment p_{ik} in the reconstruction loss $\mathcal{L}_r^{(l_k)}$, we direct the encoder and decoder networks to be specialized in reconstructing samples that are more probable to belong to the k^{th} cluster. Similarly, including p_{ik} in the clustering loss $\mathcal{L}_c^{(l_k)}$ encourages data points that are more probable to belong to the k^{th} cluster to sit close to their corresponding cluster center $\mu^{(k)}$, in the latent space $U^{(l_k)}$. Thus, by minimizing $\mathcal{L}_c^{(l_k)}$, we implicitly minimize the intra-cluster distances between samples of the k^{th} cluster.

Finally, to update the cluster center $\mu^{(k)}$, we define total loss function $\mathcal{L}_t^{(l_k)}$, $k = 1, ...,$ K, and set its derivative to 0, as is shown in (4).

$$\mathcal{L}_t^{(l_k)} = \sum_{x_i\in X} p_{ik}^{m}\Big(||x_i - \hat{x}_i^{(l_k)}||_2^2 + \lambda||u_i^{(l_k)} - \mu^{(k)}||_2^2\Big)$$
$$\xrightarrow{\frac{\partial\mathcal{L}_t^{(k)}}{\partial\mu^{(k)}}=0} \mu^{(k)} = \frac{\sum_{x_i\in X} p_{ik}^{m}u_i^{(l_k)}}{\sum_{x_i\in X} p_{ik}^{m}} \tag{4}$$

Algorithm 1 DML Algorithm

```
    Input: Data points $X$, $\tilde{\theta}_e$, $\tilde{\theta}_d$, $\mu^{(k)}$ for $k = 1, \ldots,$ K, $\tau$, and MaxIter.
    Output: $\theta_e^{(l)}$, $\theta_d^{(l)}$ for $l = 1, \ldots,$ L.
1:  Find sets $\mathcal{D}$ and $\mathcal{E}$ based on Section III-A.
2:  Initialize $\theta_e^{(l)}$ and $\theta_d^{(l)}$ for $l = 1, \ldots,$ L with $\tilde{\theta}_e$ and $\tilde{\theta}_d$ respectively.
3:  for $iter \in \{1, 2, ..., \text{MaxIter}\}$ do
4:      for $k \in \{1, 2, ..., \text{K}\}$ do
5:          Compute soft assignments $p_{ik}$ using (2g), for $i \in \mathfrak{B}$
6:          Update AE$^{(l_k)}$'s parameters employing loss function (3)
7:      end for
8:  end for
9:  Use crisp assignment based on Section III-D to assign data point to clusters
```

D. Crisp Assignment

We utilize the trained encoders and cluster centers to compute the final degree of membership, p_{ik}, based on (2g). Each data point is assigned to the most probable cluster.

The pseudo-code of DML is presented in Algorithm 1. The block diagram of DML is shown in Fig. 1.

IV. Experiments

In this section, we investigate the effectiveness of our proposed DML framework on nine benchmark datasets by running an extensive set of experiments. Both large-scale and small-scale datasets are considered. The clustering performance of the DML framework is compared against thirteen traditional and state-of-the-art clustering methods. The code of the proposed DML framework is available at https://github.com/Armanfard-Lab/DML.

Demonstration of the DML's performance on the large-scale datasets is shown in Sections IV.C, IV.D, IV.E, IV.F, IV.G, and IV.H. Demonstration of the performance of the DML on the small-scale datasets is shown in Section IV.I.

A. Evaluation Metrics

To evaluate clustering performance, we utilize two widely used metrics, including clustering accuracy (ACC) [52] and normalized mutual information (NMI) [53]. ACC finds the best match between the ground truth and predicted cluster labels. NMI calculates the normalized reduction in entropy of a class when cluster labels are assigned. The formula for ACC and NMI are shown below:

$$ACC = \max_m \frac{\sum_{i=1}^{N} \mathbb{1}\{l_i = map(c_i)\}}{N} \tag{5a}$$

$$NMI = \frac{I(l;c)}{max\{H(l), H(c)\}}, \tag{5b}$$

where l_i and c_i respectively denote the ground truth and predicted labels for a given data point x_i. $map(.)$ represents the best mapping between true and predicted labels. Moreover, $H(.)$ denotes entropy function and $I(l;c)$ presents the mutual information between ground truth label $l = \{l_1, l_2, ..., l_N\}$ and predicted cluster assignments $c = \{c_1, c_2, ..., c_N\}$. ACC and NMI vary in the interval of [0,1], where higher scores specify better clustering performance.

B. Datasets

The effectiveness of our proposed DML framework is evaluated on nine commonly used large- and small-scale datasets. Since clustering is a fully unsupervised task, we merge train and test sets for all datasets. Concatenating train and test sets is a standard practice in the clustering research studies [26, 30, 29, 54].

The large-scale datasets are:
(1) MNIST [55] comprises 60,000 training and 10,000 test gray-scale handwritten digits with size 28×28. This dataset has ten classes.
(2) Fashion MNIST [56] contains various types of fashion items. The number of samples and the image size are the same as those of the MNIST dataset.
(3) 2MNIST is a more challenging dataset that is created by concatenation of the two MNIST and Fashion MNIST datasets. Hence, it consists of 140,000 gray-scale images from 20 classes.
(4) CIFAR-10 [57] contains 60,000 RGB images from 10 different categories, where the size of each image is 32×32.
(5) STL-10 [58] comprises of 13,000 96×96 RGB images from different objects.
(6) CIFAR-100 [57] is similar to the CIFAR-10 in terms of number of samples and image size. However, it has 20 supergroups based on the similarity between images.

The small-scale datasets are:
(7) Coil-20 [59] comprises of 1,440 32×32 images of 20 different objects.
(8) UMIST [60] is contains of 575 112×92 gray-scale face images of 20 individuals.
(9) USPS [61] comprises of 9,298 16×16 handwritten digits 0 to 9.

C. Clustering Performance

The effectiveness of our proposed DML framework is compared with thirteen popular conventional and state-of-the-art clustering algorithms.

k-means [12], large-scale spectral clustering (LSSC) [62], and locality preserving non-negative matrix factorization (LPMF) [63] are among the most commonly used conventional clustering algorithms. Deep-learning-based algorithms include deep embedding clustering (DEC) [29], improved deep embedding clustering (IDEC) [26], deep clustering network (DCN) [28], deep k-means (DKM) [27], variational deep embedding (VaDE) [64], GAN mixture model for clustering (GANMM) [65], [66], and the very recent methods contrastive clustering (CC) [37] and deep successive learning (DSL) [67]. In addition to these algorithms, we compare our results with scalable deep k-subspace clustering (SDkC), which offers a scalable and efficient approach for subspace clustering through the application of deep learning. SDkC is one of the few deep subspace clustering (DSC) methods capable of handling large-scale datasets that feature high-dimensional features and multiple subspaces. Furthermore, we report the clustering performance of the baseline approach AE + k-means, in which k-means is simply applied to the latent representation of an AE that has a similar architecture to the AE's used in the DML method; the AE in AE + k-means is trained to minimize the data reconstruction loss. In Section II, you may find more information about the comparison algorithms. Note that, in all the below Tables and Figures, for the comparison methods, we executed the code released by the authors with the same hyper-parameters specified in the original papers if the results of interest are not reported in the corresponding original paper. When the code is not publicly available or not applicable to the dataset, we put dash marks (-) instead of the corresponding results.

The effectiveness of DML in boosting the performance of the state-of-the-art AE-based clustering methods is shown in Table I. Note that in general, DML would be effective for the AE-based clustering algorithms which aim at creating hyperspheres of data clusters in a lower dimensional space, such as DCN [28], DKM [27] and DSL[67]. In Table I, DML-$\mathcal{M}$ refers to the performance of DML when the AE of algorithm $\mathcal{M}$ is used as the DML's autoencoders (see Section III). As it can be seen from the table, DML significantly improves the clustering performance of the base method $\mathcal{M}$ mainly due to assigning cluster-specific AEs to the difficult data clusters. On average, DML improves ACC (NMI) of DCN, DKM, and DSL, respectively, by 2.50% (2.48%), 2.84% (2.75%), and 0.42% (0.55%). Less significant improvement in DSL compared to that of DCN and DKM could be associated with the fact that DSL implicitly has some sort of cluster-specific training procedure when training its AE, but note that

TABLE I
ACC AND NMI (IN PARENTHESIS) ON THE BENCHMARK DATASETS FOR METHOD $\mathcal{M}$ AND DML-$\mathcal{M}$ WHERE $\mathcal{M} \in \{$DCN, DKM, DSL$\}$. THE MEAN AND THE VARIANCE ARE OBTAINED BY RUNNING TEN INDEPENDENT EXPERIMENTS.

Method \ Datasets	MNIST	Fashion MNIST	2MNIST	CIFAR10	STL10	CIFAR100
DCN	83.00 (81.00)	51.22 (55.47)	41.35 (46.89)	30.47 (24.58)	33.84 (24.12)	20.17 (12.54)
DKM	84.00 (81.54)	51.31 (55.57)	41.75 (46.58)	35.26 (26.12)	32.61 (29.12)	18.14 (12.30)
DSL	96.22 (90.66)	62.90 (63.58)	45.31 (63.00)	83.40 (71.32)	96.02 (91.90)	50.30 (49.80)
DML-DCN	**87.48±0.68 (81.63±0.31)**	**55.57±0.45 (55.90±0.71)**	**44.79±0.34 (58.23±0.29)**	**31.51±0.15 (25.00±0.23)**	**34.05±0.19 (25.14±0.21)**	**21.56±0.31 (13.58±0.26)**
DML-DKM	**91.18±0.78 (82.58±0.34)**	**55.36±0.16 (56.00±0.12)**	**44.66±0.84 (58.30±0.72)**	**36.28±0.21 (27.00±0.18)**	**34.10±0.29 (31.21±0.34)**	**18.56±0.22 (12.69±0.24)**
DML-DSL	**96.36±0.15(91.24±0.22)**	**63.20±0.21 (64.80±0.25)**	**45.83±0.34 (63.40±0.23)**	**84.15±0.36 (71.70±0.29)**	**96.45±0.13 (92.11±0.18)**	**50.68±0.22 (50.19±0.15)**

TABLE II
ACC AND NMI (IN PARENTHESIS) ON THE BENCHMARK DATASETS FOR DIFFERENT CLUSTERING METHODS. THE MEAN AND THE VARIANCE ARE OBTAINED BY RUNNING TEN INDEPENDENT EXPERIMENTS.

Method \ Datasets	MNIST	Fashion MNIST	2MNIST	CIFAR10	STL10	CIFAR100
k-means	53.20 (50.00)	47.40 (51.20)	32.31 (44.00)	22.90 (8.70)	19.20 (12.50)	13.00 (8.40)
LSSC	71.40 (70.60)	49.60 (49.70)	39.77 (51.22)	21.14 (10.89)	18.75 (11.68)	14.60 (7.92)
LPMF	47.10 (45.20)	43.40 (42.50)	34.68 (38.69)	19.10 (8.10)	18.00 (9.60)	11.80 (7.90)
DEC	84.30 (83.72)	51.80 (54.63)	41.20 (53.12)	30.10 (25.70)	35.90 (27.60)	18.50 (13.60)
IDEC	88.13 (83.81)	52.90 (55.70)	40.42 (53.56)	36.99 (32.53)	32.53 (18.85)	19.61 (14.58)
DCN	83.00 (81.00)	51.22 (55.47)	41.35 (46.89)	30.47 (24.58)	33.84 (24.12)	20.17 (12.54)
DKM	84.00 (81.54)	51.31 (55.57)	41.75 (46.58)	35.26 (26.12)	32.61 (29.12)	18.14 (12.30)
AE + k-means	86.03 (80.25)	57.94 (57.15)	44.01 (62.80)	80.11 (70.35)	95.89 (91.75)	49.86 (48.57)
VaDE	94.50 (87.60)	50.39 (59.63)	56.60 (51.20)	29.10 (24.50)	28.10 (20.00)	15.20 (10.80)
GANMM	64.00 (61.00)	34.00 (27.00)	50.12 (49.35)	-	-	-
SDkC	83.30 (77.38)	60.02 (62.30)	-	-	-	-
CC	88.56 (84.21)	**64.52** (61.45)	42.15 (58.89)	79.00 (70.50)	85.00 (76.40)	42.90 (43.10)
DSL	96.22 (90.66)	62.90 (63.58)	45.31 (63.00)	83.40 (71.32)	96.02 (91.90)	50.30 (49.80)
DML-DSL	**96.36±0.15 (91.24±0.22)**	63.20±0.21 (**64.80±0.25**)	**45.83±0.34 (63.40±0.23)**	**84.15±0.36 (71.70±0.29)**	**96.45±0.13 (92.11±0.18)**	**50.68±0.22 (50.19±0.15)**

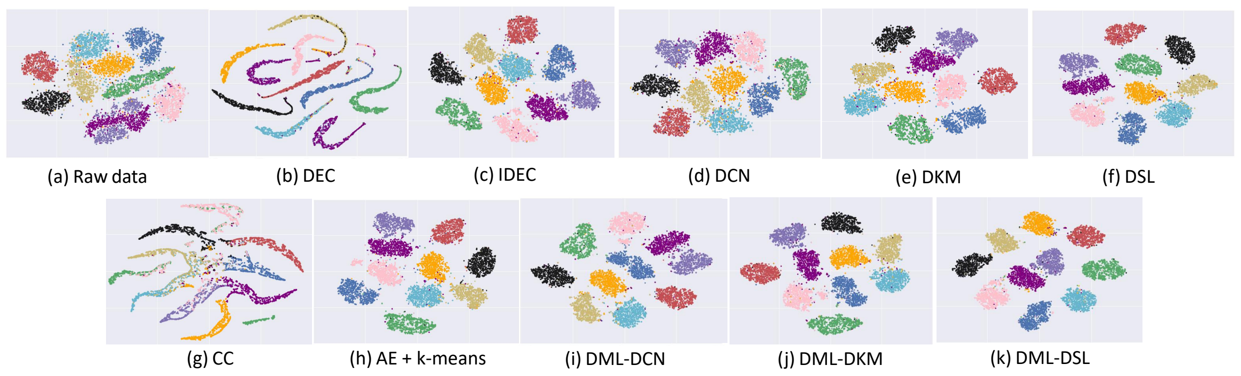


Fig. 2. Clustering visualization of different methods using t-SNE, for MNIST dataset. Axes range from -100 to 100.

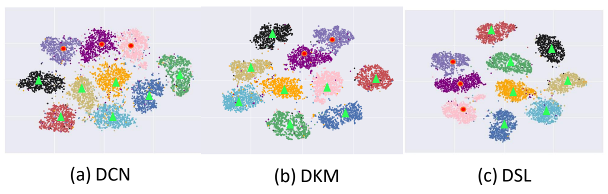


Fig. 3. Visualization of the easy and difficult clusters. Cluster centers of easy and difficult clusters are shown by green triangles and red circles, respectively.

DSL still trains a single common latent space for all of the data clusters. To further show the effectiveness of the proposed DML framework, considering that DSL [67] is the most recent and effective AE-based clustering algorithm, we pick DSL as the base of DML and compare DML-DSL with more SOTA algorithms in Table II. As can be seen, DML-DSL outperforms all comparison methods on 75 out of 76 reported results. Each dataset's best result is shown in bold. The second-top results are denoted by an asterisk (*).

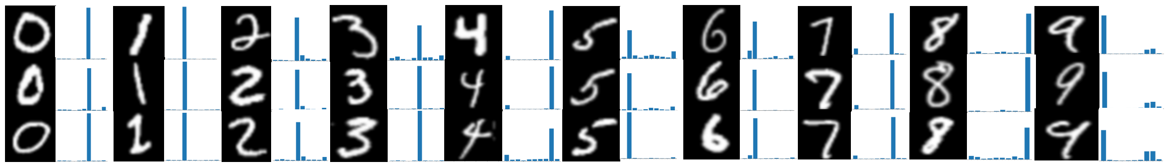
For the DML-$\mathcal{M}$ method, we use the same network structure suggested by the $\mathcal{M}$ method for each dataset, where $\mathcal{M} \in \{$DCN[28], DKM[27], DSL[54]$\}$. We first train a single AE using the $\mathcal{M}$ method to obtain the initial network parameters ($\tilde{\theta}_e$ and $\tilde{\theta}_d$) and cluster centers $\mu^{(k)}$. We then train L AEs and update cluster centers based on our proposed algorithm in Section III. For all datasets $\tau = \frac{d_{min}+d_{max}}{4}$, where d_{min} and d_{max} are the minimum and maximum distance between initial cluster centers, respectively. We choose $\lambda = \{0.001, 0.01, 0.1\}$ based on the accuracy of our model on the validation set for each dataset. Moreover, in DML-$\mathcal{M}$, we use the same data pre-processing technique as what is used in the original $\mathcal{M}$ method.

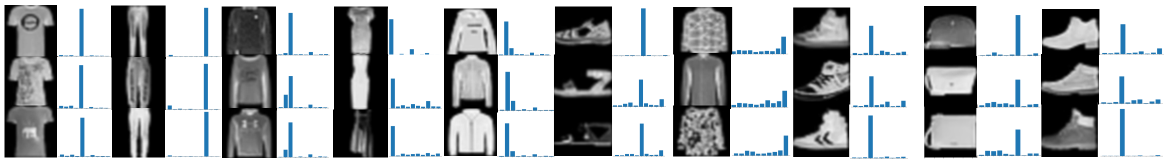
D. *t-SNE Visualization*

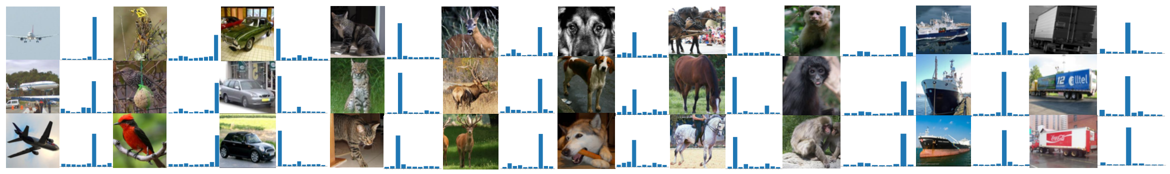
In Fig. 2, we further demonstrate the effectiveness of the proposed DML framework by comparing different data representations of the MNIST dataset using t-SNE visualization [68]. To this end, we use the trained cluster-specific AEs and the general AE to obtain the L latent representations

TABLE III
ACC AND NMI (IN PARENTHESIS) ON IMBALANCED DATASETS FOR DIFFERENT CLUSTERING METHODS. THE MEAN AND THE VARIANCE ARE OBTAINED BY RUNNING TEN INDEPENDENT EXPERIMENTS.

Method \ r	0.1	0.2	0.3	0.4	0.5
AE + k-means	82.61 (78.51)	83.11 (78.99)	84.78 (81.48)	84.83 (81.55)	85.60 (83.70)
DCN	72.74(68.80)	73.02 (72.91)	79.60 (72.80)	74.52 (73.61)	76.32(74.62)
DKM	45.96 (39.21)	46.21 (39.54)	48.56 (38.96)	51.25 (42.10)	50.98 (43.27)
DSL	83.73 (81.38)	89.86 (82.30)	90.40 (83.55)	92.14 (84.71)	94.22 (88.15)
DML-DCN	**75.34±0.35 (69.83±0.38)**	**80.32±0.29 (73.51±0.28)**	**81.23±0.35 (73.05±0.30)**	**79.20±0.20 (75.59±0.26)**	**77.58±0.33 (75.37±0.28)**
DML-DKM	**67.25±0.27 (61.32±0.25)**	**74.35±0.16 (66.24±0.17)**	**68.31±0.28 (63.17±0.25)**	**79.24±0.31 (77.12±0.39)**	**80.16±0.22 (78.12±0.19)**
DML-DSL	**85.35±0.30 (81.92±0.38)**	**90.63±0.27 (83.44±0.27)**	**91.22±0.22 (84.57±0.26)**	**92.44±0.24 (85.72±0.27)**	**94.63±0.19 (88.68±0.21)**


(a) MNIST


(b) Fashion MNIST


(c) STL-10

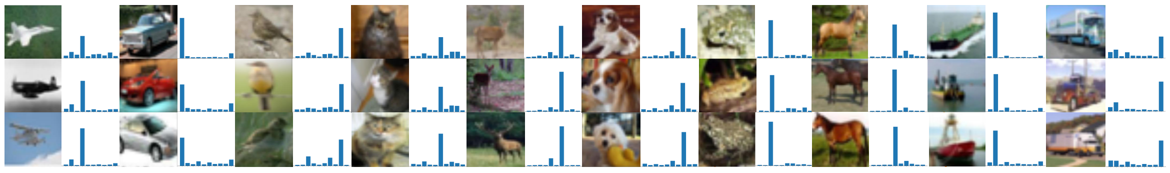

(d) CIFAR-10

Fig. 4. Visualization of soft assignments vector $\mathbf{p_i}$ for samples from (a) MNIST, (b) Fashion MNIST, (c) STL-10, and (d) CIFAR-10 datasets. The vertical axes range from 0 to 1.

of each data point. Then the most probable one is selected as the latent representation of the data, i.e., the latent space that provides the closest representation of the data to its corresponding center is chosen. Afterward, the t-SNE method is utilized to map the latent representation to a 2D space. The benefit of the proposed DML method in providing a clear distribution structure is more apparent if we compare the t-SNE representation.

The effectiveness of our proposed multi-representation learning framework, and the proposed loss function, in minimizing intra-cluster distance(s) is apparent when we compare the clusters obtained by DML-$\mathcal{M}$ shown in Fig.2- (i), (j), (k) with $\mathcal{M}$ shown in Fig. 2- (d), (e), (f). The DML-$\mathcal{M}$ clusters are much more compactly distributed around their corresponding cluster centers. The improved performance of DML-$\mathcal{M}$ compared with $\mathcal{M}$ is more significant when looking at the separation of clusters colored in magenta and purple (digits 4 and 9), as well as the separation of clusters in cyan, olive, and orange (digits 3, 5 and 8) when comparing Fig. 2-(i) with 2-(d), and Fig. 2-(j) with 2-(e), and Fig. 2-(k) with Fig. 2-(f).

In Fig. 3, we further illustrate the properties of easy and difficult clusters by visualizing their cluster centers. The cluster centers of easy and difficult clusters are represented by green triangles and red circles, respectively. As can be seen in the figure, the cluster centers of difficult clusters are located

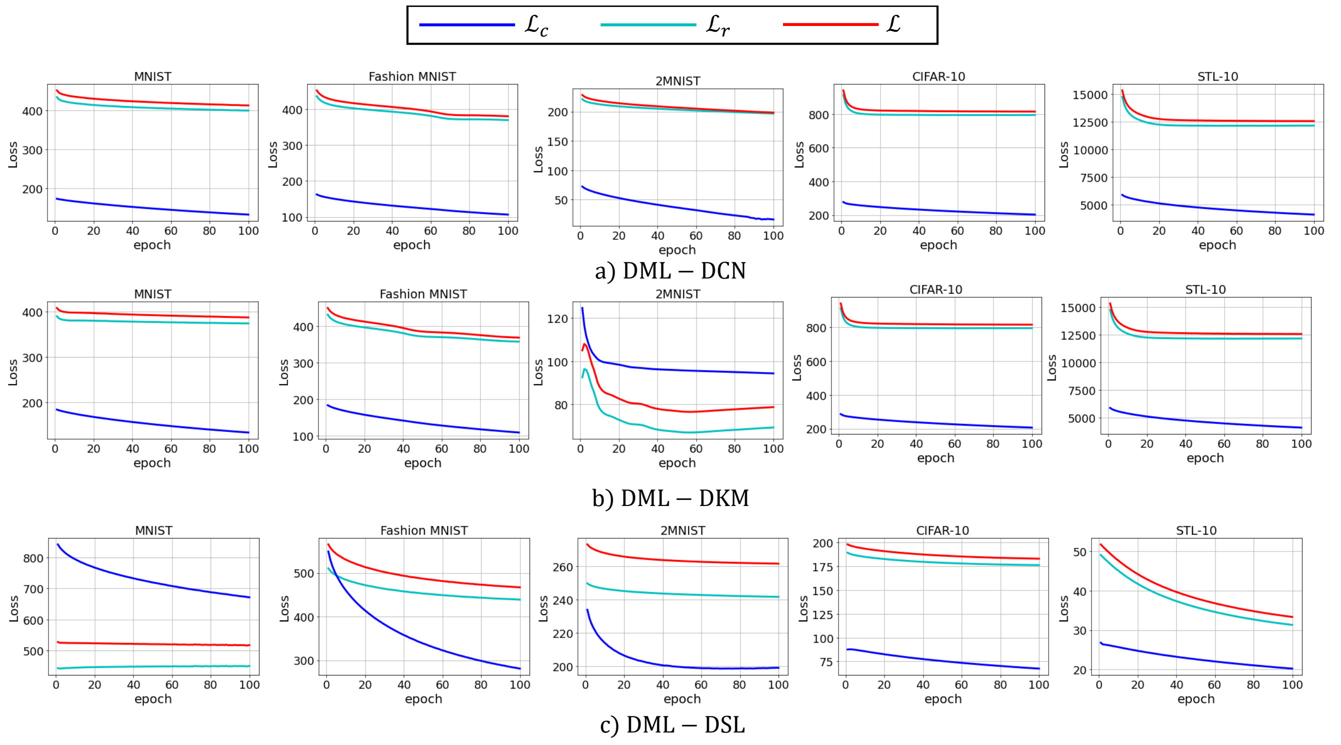


Fig. 5. The average reconstruction loss $\mathcal{L}_r$, clustering loss $\mathcal{L}_c$, and total loss $\mathcal{L}$ of DML-$\mathcal{M}$ methods, $\mathcal{M} \in$ {DCN, DKM, DSL}, for different datasets.

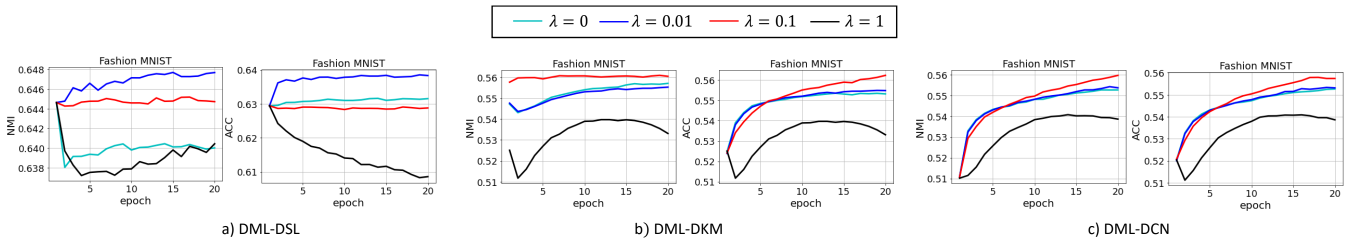


Fig. 6. Effect of hyperparameter λ on clustering performance of DML-$\mathcal{M}$ where $\mathcal{M} \in$ {DCN, DKM, DSL}.

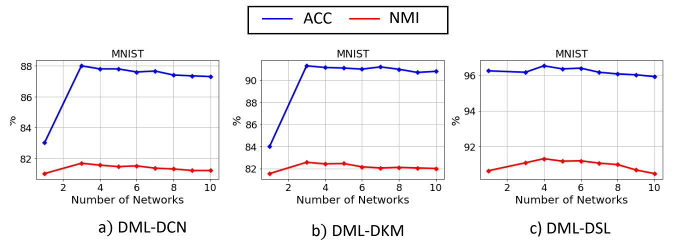


Fig. 7. Effect of number of networks on clustering performance of DML-$\mathcal{M}$ where $\mathcal{M} \in$ {DCN, DKM, DSL}.

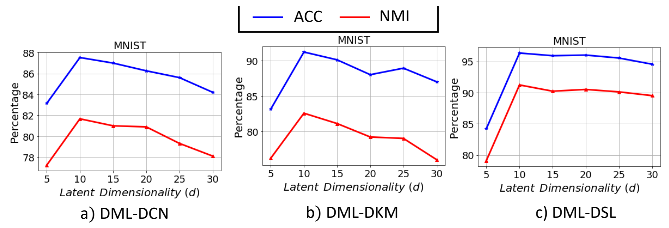


Fig. 8. Effect of the dimension of the latent space on clustering performance of DML-$\mathcal{M}$ where $\mathcal{M} \in$ {DCN, DKM, DSL}. on MNIST dataset

close to each other and far from the easy clusters. For instance, in DCN and DSL, the distances between the magenta, pink, and purple clusters, which are associated with numbers 4, 7, and 9, are much smaller than the distances between other clusters. Therefore, our algorithm considers these clusters as the difficult ones and allocates four autoencoders (three cluster-specific autoencoders for the difficult ones and one general autoencoder for the easy ones) to cluster the data points. In DKM, our algorithm identifies the pink cluster as an easy cluster because its distance from other clusters is sufficient. Thus, three autoencoders (two cluster-specific autoencoders and one general autoencoder) are assigned to perform the clustering task.

E. Performance on Imbalanced Datasets

One of the main advantages of the proposed multi-representation learning method is its outstanding ability to deal with imbalanced datasets since, in contrast with the state-of-the-art methods that provide a common latent space for all clusters, DML dedicates cluster-specific AEs for difficult clusters. To show the effectiveness of the proposed framework on imbalanced data, we sample subsets of MNIST with various retention rates $r \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$, where data points of class 0 are kept with probability r and class 9 with probability 1, with the other classes linearly in between 0 and 1. As such, the smallest cluster is r times smaller than the largest cluster. ACC and NMI for various r on all datasets are shown in Table III. It can be seen that the proposed DML-$\mathcal{M}$ method significantly outperforms $\mathcal{M}$ for all r values, where $\mathcal{M} \in \{\text{DSL, DKM, DCN}\}$. DML-DCN, DML-DKM, and DML-DSL improve the performance of DCN, DKM, and DSL by 0.79% (0.85%), 25.25% (28.57%), and 3.51% (0.94%) in ACC (NMI) in average on the five datasets, which shows the effectiveness of our proposed training procedure of DML in learning useful representation of data points. For reference, the performance of the baseline method AE + k-means is also reported in Table III.

F. Soft assignments visualization

Fig. 4 depicts soft cluster assignment vectors, i.e., $\mathbf{p_i}$, corresponding to samples from various data clusters for DML-DSL. The k^{th} element of $\mathbf{p_i}$ shows the probability of sample x_i belonging to the k^{th} cluster. We observe that, with a high probability, the index of the highest element in vector $\mathbf{p_i}$ of samples from the same ground-truth labels are the same – in other words, with a high probability, samples from the same ground truth class are mapped to the same cluster. Since DML dedicates individual AEs to difficult clusters, it is capable of distinguishing more complex clusters, such as deer and horse. Moreover, DML reasonably recognizes the second most probable cluster for each sample. For example, in the STL-10 dataset, the second most probable cluster for a truck is a car, which is also a vehicle.

G. Loss function convergence

Fig. 5 shows the average of the reconstruction, clustering, and total losses for different DML-$\mathcal{M}$ methods. This average is taken over different networks on different batches of data points. The figure shows the convergence of all losses at the end of training. The remarkable reduction in the clustering loss indicates the ability of our DML framework to gather the data points near their cluster centers to obtain a better latent space for the clustering task.

H. Hyperparameters Sensitivity

In Fig. 6, Fig. 7, and Fig. 8, we investigate the effect of the DML's hyperparameters λ, the number of AE networks and the dimension of the latent space on the clustering performance of DML-$\mathcal{M}$.

In Fig. 6, we scrutinize the effect of hyperparameter λ on the clustering performance of DML-$\mathcal{M}$ for the Fashion MNIST dataset, where $\lambda \in \{0, 0.01, 0.1, 1\}$. λ indicates the importance of the clustering loss in the total loss function of DML (See Section III-C). We observe that for large values of λ, e.g., $\lambda = 1$, DML mainly concentrates on centering the data points near cluster centers and ignores the informative features that provide a low data reconstruction error. For relatively small values of λ, e.g., $\lambda = 0$, DML ignores the clustering loss and only focuses on minimizing the data reconstruction loss, which may mislead the DML in assigning the data points to the correct clusters. The best clustering performance of DML-DCN, DML-DKM, and DML-DSL on the Fashion MNIST dataset are respectively obtained when λ is 0.1, 0.1, and 0.01.

In Fig. 7, we explore the impact of the total number of AEs on the clustering performance of DML-DCN, DML-DKM, and DML-DSL for the MNIST dataset. What we can infer from this figure are as follows. 1) there is a significant improvement in the clustering performance of DML-DCN and DML-DKM when we increase the number of networks from one to three, which shows the effectiveness of having multiple AEs in finding effective representations for data points. The fact that the DSL itself implicitly includes some form of cluster-specific training technique when training its AE could explain the less significant improvement obtained for DSL compared to the DCN and DKM cases. 2) As is expected, increasing the number of networks does not always lead to a model with better performance. For example, the best model performance in terms of ACC and NMI for DML-DSL is achieved when four cluster-specific networks are trained for the MNIST dataset. Note that, in this paper, we proposed an automatic approach, using τ, to determine the number of required AEs per dataset (see Section IV-C).

In Fig. 8, the impact of the latent space dimensionality d on the clustering performance of DML-DCN, DML-DKM, and DML-DSL is investigated using the MNIST dataset. We made the following observations: **(1)** when d has a low value (e.g., $d = 5$), the network lacks the ability to capture well enough information of the input data, resulting in high reconstruction and clustering errors. **(2)** when d has a high value (e.g., $d = 30$), the network may capture unnecessary details that might be data noises; these details may reduce the reconstruction loss but result in overfitting for clustering. The figure shows that a proper choice would be a mid-value, e.g., 10 in this case. Note that one may get better performance by fine-tuning the d value for each dataset; however, for the sake of generality and fair comparison, in all our experiments, we set d to 10 for all datasets.

I. Experiments on small datasets

In this section, we compare the performance of our proposed DML framework with graph convolutional neural networks (GCN) and deep subspace clustering (DSC) methods.

GCN models, such as variational graph autoencoder (GAE) [69], marginalized graph autoencoder (MGAE) [70], unsupervised graph representation learning (UGRL) [71], and structural deep clustering network (SDCN) [72], have often been trained on small-scale datasets due to their very high computational complexity. The common practice in graph-based clustering techniques is to consider every data point as a graph node. This results in large graphs with a sheer number of trainable parameters when dealing with large datasets [73, 69, 70].

TABLE IV
ACC AND NMI (IN PARENTHESIS) ON SMALL SIZE BENCHMARK DATASETS FOR DIFFERENT CLUSTERING METHODS.

Method \ Dataset	COIL-20	UMIST	USPS
GAE	69.10 (86.45)	61.91 (80.24)	76.63 (76.02)
MGAE	60.99 (73.59)	49.19 (68.00)	64.13 (62.18)
UGRL	80.00 (87.71)	41.39 (63.71)	67.64 (71.50)
SDCN	41.04 (60.07)	27.65 (38.65)	37.43 (38.97)
AE + SSC	87.11 (89.90)	70.42 (75.15)	70.23 (72.15)
DSCNet-L1	93.10 (93.70)	72.42 (75.46)	77.33 (76.78)
DSCNet-L2	93.70 (94.10)	73.12 (76.62)	77.54 (78.00)
MRL-DSC	73.25 (84.27)	63.65 (75.18)	74.15 (75.13)
DML-DSL	**93.80 (94.53)**	**74.14 (80.29)**	**78.37 (79.22)**

Similarly, the number of parameters in a network for deep subspace clustering (DSC) algorithms highly depends on the size of the dataset, making almost all DSC algorithms, such as AE + SSC that applies sparse subspace clustering (SSC) [74] on the latent space of a trained autoencoder, the two versions of the deep subspace clustering network denoted by DSCNet-L1 and DSCNet-L2 [75], and multi-level representation learning for deep subspace clustering (MRL-DSC) [76] , only applicable to small datasets (or subsets of larger datasets). For example, [77] uses the first 2,000 train images and the first 2,000 test images of the MNIST dataset to report their results on this dataset, and [78] uses 6,000 train and 1,000 test images of the MNIST dataset in its experiments; while as is mentioned in Section IV.A, the MNIST dataset contains 70,000 data points.

To have a fair comparison of the proposed DML framework with the GCN and DSC methods, we report the clustering performance on the commonly used benchmark datasets in the field, namely COIL-20 [59], UMIST [60], and USPS [61]. The results displayed in Table IV demonstrate that our DML framework outperforms both GCN and DSC methods in terms of ACC and NMI. This also highlights the effectiveness of our framework in finding appropriate representations for small-scale datasets.

V. Conclusion and Future Works

In this paper, we present an effective and practical method for obtaining cluster-specific latent spaces for data clustering. Unlike most deep learning-based clustering methods, which provide a single global latent space, the proposed algorithm provides multiple latent spaces, where an individual cluster-specific latent space is assigned to a difficult cluster, and a common latent space is devised for easy clusters. The idea is realized by devising cluster-specific losses based on the weighted reconstruction and clustering losses. Effectiveness of the proposed multi-representation learning framework and the proposed loss function is demonstrated on multiple benchmark datasets through an extensive set of experiments.

The current DML method assigns an individual AE to the difficult clusters. There might be applications for which there are not enough data points within a difficult cluster to be used for training their associated network. In such scenarios, one may consider artificially increasing the number of data points through data augmentation [79]. Another interesting approach would be to reduce the number of DML parameters leveraging a multi-task learning strategy where a shared encoder is assigned to *all* clusters. L cluster-specific heads then follow the encoder for the difficult clusters and one head for the easy clusters. A head can be made of fully connected layers or CNN layers. The head outputs provide the required multiple representations. For the decoder part, we suggest employing L different decoders: $L - 1$ for data samples of the difficult-to-cluster and one general decoder for samples of the easy-to-cluster data groups.

VI. Acknowledge

The authors would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Department of Electrical and Computer Engineering at McGill University. This work would not have been possible without their financial support. The authors would also like to thank Calcul Quebec and Compute Canada for providing the necessary computational resources to conduct our experiments.

References

[1] Zechao Li, Jinhui Tang, and Tao Mei. “Deep Collaborative Embedding for Social Image Understanding”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.9 (2019), pp. 2070–2083. DOI: 10.1109/TPAMI.2018.2852750.

[2] Man Zhang et al. “A survey of semi-and weakly supervised semantic segmentation of images”. In: *Artificial Intelligence Review* 53 (2020), pp. 4259–4288.

[3] Xiaoli Wang et al. “MMatch: Semi-Supervised Discriminative Representation Learning for Multi-View Classification”. In: *IEEE Transactions on Circuits and Systems for Video Technology* 32.9 (2022), pp. 6425–6436. DOI: 10.1109/TCSVT.2022.3159371.

[4] Jesper E Van Engelen and Holger H Hoos. “A survey on semi-supervised learning”. In: *Machine learning* 109.2 (2020), pp. 373–440.

[5] Ashley J Ross et al. “The clustering of the SDSS DR7 main Galaxy sample–I. A 4 per cent distance measure at z= 0.15”. In: *Monthly Notices of the Royal Astronomical Society* 449.1 (2015), pp. 835–847.

[6] Nguyen Dang Thanh, Mumtaz Ali, et al. "Neutrosophic recommender system for medical diagnosis based on algebraic similarity measure and clustering". In: *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE. 2017, pp. 1–6.

[7] S Selva Kumar and H Hannah Inbarani. "Analysis of mixed C-means clustering approach for brain tumour gene expression data". In: *International Journal of Data Analysis Techniques and Strategies* 5.2 (2013), pp. 214–228.

[8] Florian Schroff, Dmitry Kalenichenko, and James Philbin. "Facenet: A unified embedding for face recognition and clustering". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 815–823.

[9] Kaiye Wang et al. "Joint feature selection and subspace learning for cross-modal retrieval". In: *IEEE transactions on pattern analysis and machine intelligence* 38.10 (2015), pp. 2010–2023.

[10] Linan Feng and Bir Bhanu. "Semantic concept co-occurrence patterns for image annotation and retrieval". In: *IEEE transactions on pattern analysis and machine intelligence* 38.4 (2015), pp. 785–799.

[11] Syed Sameed Husain and Miroslaw Bober. "Improving large-scale image retrieval through robust aggregation of local descriptors". In: *IEEE transactions on pattern analysis and machine intelligence* 39.9 (2016), pp. 1783–1796.

[12] Stuart Lloyd. "Least squares quantization in PCM". In: *IEEE transactions on information theory* 28.2 (1982), pp. 129–137.

[13] James C Bezdek, Robert Ehrlich, and William Full. "FCM: The fuzzy c-means clustering algorithm". In: *Computers & Geosciences* 10.2-3 (1984), pp. 191–203.

[14] Meng Jianliang, Shang Haikun, and Bian Ling. "The application on intrusion detection based on k-means cluster algorithm". In: *2009 International Forum on Information Technology and Applications*. Vol. 1. IEEE. 2009, pp. 150–152.

[15] OJ Oyelade, OO Oladipupo, and IC Obagbuwa. "Application of k Means Clustering algorithm for prediction of Students Academic Performance". In: *arXiv preprint arXiv:1002.2425* (2010).

[16] Mahnaz EtehadTavakol, Saeed Sadri, and EYK Ng. "Application of K-and fuzzy c-means for color segmentation of thermal infrared breast images". In: *Journal of medical systems* 34.1 (2010), pp. 35–42.

[17] Jerome H Friedman. "On bias, variance, 0/1—loss, and the curse-of-dimensionality". In: *Data mining and knowledge discovery* 1.1 (1997), pp. 55–77.

[18] M Pavithra and R Parvathi. "A survey on clustering high dimensional data techniques". In: *International Journal of Applied Engineering Research* 12.11 (2017), pp. 2893–2899.

[19] John R Hershey et al. "Deep clustering: Discriminative embeddings for segmentation and separation". In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2016, pp. 31–35.

[20] Xia Hu, Qiaoyu Tan, and Ninghao Liu. "Deep representation learning for social network analysis". In: *Frontiers in Big Data* 2 (2019), p. 2.

[21] Mei Wang and Weihong Deng. "Deep face recognition with clustering based domain adaptation". In: *Neurocomputing* (2020).

[22] Mathilde Caron et al. "Deep clustering for unsupervised learning of visual features". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 132–149.

[23] Seunghyoung Ryu et al. "Convolutional autoencoder based feature extraction and clustering for customer load analysis". In: *IEEE Transactions on Power Systems* 35.2 (2019), pp. 1048–1060.

[24] Chunfeng Song et al. "Auto-encoder based data clustering". In: *Iberoamerican congress on pattern recognition*. Springer. 2013, pp. 117–124.

[25] Pierre Baldi. "Autoencoders, unsupervised learning, and deep architectures". In: *Proceedings of ICML workshop on unsupervised and transfer learning*. JMLR Workshop and Conference Proceedings. 2012, pp. 37–49.

[26] Xifeng Guo et al. "Improved deep embedded clustering with local structure preservation." In: *International Joint Conference on Artificial Intelligence(IJCAI)*. 2017, pp. 1753–1759.

[27] Maziar Moradi Fard, Thibaut Thonet, and Eric Gaussier. "Deep k-means: Jointly clustering with k-means and learning representations". In: *Pattern Recognition Letters* 138 (2020), pp. 185–192.

[28] Bo Yang et al. "Towards k-means-friendly spaces: Simultaneous deep learning and clustering". In: *international conference on machine learning*. 2017, pp. 3861–3870.

[29] Junyuan Xie, Ross Girshick, and Ali Farhadi. "Unsupervised deep embedding for clustering analysis". In: *International conference on machine learning*. 2016, pp. 478–487.

[30] Mohammadreza Sadeghi and Narges Armanfard. "IDECF: Improved Deep Embedding Clustering With Deep Fuzzy Supervision". In: *IEEE International Conference on Image Processing (ICIP)*. 2021, accepted to be published.

[31] Dongkuan Xu and Yingjie Tian. "A comprehensive survey of clustering algorithms". In: *Annals of Data Science* 2.2 (2015), pp. 165–193.

[32] Qiyue Yin, Shu Wu, and Liang Wang. "Incomplete multi-view clustering via subspace learning". In: *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 2015, pp. 383–392.

[33] Zechao Li et al. "Robust Structured Subspace Learning for Data Representation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.10 (2015), pp. 2085–2098. DOI: 10.1109/TPAMI.2015.2400461.

[34] Weihua Hu et al. "Learning discrete representations via information maximizing self-augmented training". In:

International conference on machine learning. PMLR. 2017, pp. 1558–1567.

[35] Sungwon Park et al. “Improving Unsupervised Image Clustering With Robust Learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 12278–12287.

[36] Jianlong Chang et al. “Deep Self-Evolution Clustering”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42 (2020), pp. 809–823.

[37] Yunfan Li et al. “Contrastive Clustering”. In: *arXiv preprint arXiv:2009.09687* (2020).

[38] Ting Chen et al. “A simple framework for contrastive learning of visual representations”. In: *International conference on machine learning*. PMLR. 2020, pp. 1597–1607.

[39] Erxue Min et al. “A survey of clustering with deep learning: From the perspective of network architecture”. In: *IEEE Access* 6 (2018), pp. 39501–39514.

[40] Peihao Huang et al. “Deep embedding network for clustering”. In: *2014 22nd International conference on pattern recognition*. IEEE. 2014, pp. 1532–1537.

[41] Fei Tian et al. “Learning deep representations for graph clustering”. In: *28th AAAI Conference on Artificial Intelligence*. 2014.

[42] Satu Elisa Schaeffer. “Graph clustering”. In: *Computer science review* 1.1 (2007), pp. 27–64.

[43] Maria CV Nascimento and Andre CPLF De Carvalho. “Spectral methods for graph clustering–a survey”. In: *European Journal of Operational Research* 211.2 (2011), pp. 221–231.

[44] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[45] Ian Goodfellow et al. “Generative adversarial nets”. In: *Advances in neural information processing systems* 27 (2014).

[46] Zhuxi Jiang et al. “Variational deep embedding: An unsupervised and generative approach to clustering”. In: *arXiv preprint arXiv:1611.05148* (2016).

[47] Warith Harchaoui, Pierre-Alexandre Mattei, and Charles Bouveyron. “Deep adversarial Gaussian mixture auto-encoder for clustering”. In: (2017), pp. 1–5.

[48] Alireza Makhzani et al. “Adversarial Autoencoders”. In: *International Conference on Learning Representations*. 2016. URL: http://arxiv.org/abs/1511.05644.

[49] Xi Chen et al. “Infogan: Interpretable representation learning by information maximizing generative adversarial nets”. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. 2016, pp. 2180–2188.

[50] Yang Yu and Wen-Ji Zhou. “Mixture of GANs for Clustering.” In: *IJCAI*. 2018, pp. 3047–3053.

[51] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[52] Yi Yang et al. “Image clustering using local discriminant models and global integration”. In: *IEEE Transactions on Image Processing* 19.10 (2010), pp. 2761–2773.

[53] Wei Xu, Xin Liu, and Yihong Gong. “Document clustering based on non-negative matrix factorization”. In: *26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. 2003, pp. 267–273.

[54] Mohammadreza Sadeghi and Narges Armanfard. “Deep Clustering with Self-supervision using Pairwise Data Similarities”. In: (2021).

[55] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[56] Han Xiao, Kashif Rasul, and Roland Vollgraf. “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms”. In: *arXiv preprint arXiv: 1708.07747* (2017).

[57] Alex Krizhevsky, Geoffrey Hinton, et al. “Learning multiple layers of features from tiny images”. In: (2009).

[58] Adam Coates, Andrew Ng, and Honglak Lee. “An analysis of single-layer networks in unsupervised feature learning”. In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2011, pp. 215–223.

[59] Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. “Columbia object image library (coil-20)”. In: (1996).

[60] Chenping Hou et al. “Joint embedding learning and sparse regression: A framework for unsupervised feature selection”. In: *IEEE transactions on cybernetics* 44.6 (2013), pp. 793–804.

[61] Jonathan J. Hull. “A database for handwritten text recognition research”. In: *IEEE Transactions on pattern analysis and machine intelligence* 16.5 (1994), pp. 550–554.

[62] Xinlei Chen and Deng Cai. “Large scale spectral clustering with landmark-based representation”. In: *25th AAAI conference on artificial intelligence*. 2011.

[63] Deng Cai et al. “Locality preserving nonnegative matrix factorization”. In: *Twenty-first international joint conference on artificial intelligence*. 2009.

[64] Zhuxi Jiang et al. “Variational deep embedding: An unsupervised and generative approach to clustering”. In: *arXiv preprint arXiv:1611.05148* (2016).

[65] Yang Yu and Wen-Ji Zhou. “Mixture of GANs for Clustering.” In: *IJCAI*. 2018, pp. 3047–3053.

[66] Tong Zhang et al. “Scalable deep k-subspace clustering”. In: *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part V 14*. Springer. 2019, pp. 466–481.

[67] Mohammadreza Sadeghi and Narges Armanfard. “Deep Successive Subspace Learning for Data Clustering”. In: *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2021, pp. 1–8.

[68] Laurens van der Maaten and Geoffrey Hinton. "Visualizing data using t-SNE". In: *Journal of machine learning research* 9.Nov (2008), pp. 2579–2605.

[69] Thomas N Kipf and Max Welling. "Variational graph auto-encoders". In: *arXiv preprint arXiv:1611.07308* (2016).

[70] Chun Wang et al. "Mgae: Marginalized graph autoencoder for graph clustering". In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 2017, pp. 889–898.

[71] Jiwoong Park et al. "Symmetric graph convolutional autoencoder for unsupervised graph representation learning". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 6519–6528.

[72] Deyu Bo et al. "Structural deep clustering network". In: *Proceedings of the web conference 2020*. 2020, pp. 1400–1410.

[73] Xu Yang et al. "Deep spectral clustering using dual autoencoder network". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4066–4075.

[74] Ehsan Elhamifar and René Vidal. "Sparse subspace clustering: Algorithm, theory, and applications". In: *IEEE transactions on pattern analysis and machine intelligence* 35.11 (2013), pp. 2765–2781.

[75] Pan Ji et al. "Deep subspace clustering networks". In: *Advances in neural information processing systems* 30 (2017).

[76] Mohsen Kheirandishfard, Fariba Zohrizadeh, and Farhad Kamangar. "Multi-level representation learning for deep subspace clustering". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2020, pp. 2039–2048.

[77] Xi Peng et al. "Deep subspace clustering". In: *IEEE transactions on neural networks and learning systems* 31.12 (2020), pp. 5509–5521.

[78] Juncheng Lv et al. "Pseudo-supervised deep subspace clustering". In: *IEEE Transactions on Image Processing* 30 (2021), pp. 5252–5263.

[79] Connor Shorten and Taghi M Khoshgoftaar. "A survey on image data augmentation for deep learning". In: *Journal of big data* 6.1 (2019), pp. 1–48.

Mohammadreza Sadeghi is currently a Ph.D. Candidate at the Department of Electrical and Computer Engineering, McGill University. He is also affiliated with Mila Quebec AI-Institute, Montreal, QC, Canada. He received his B.Sc. degree in Electrical Engineering from the University of Tehran, Tehran, Iran, in 2019. His research interest includes deep learning, machine learning, and deep subspace learning for data clustering.

Narges Armanfard is currently an Assistant Professor (tenure-track) at the Department of Electrical and Computer Engineering, McGill University and Mila Quebec AI-Institute, Montreal, Quebec, Canada. She received her Ph.D. degree in Electrical and Computer Engineering from McMaster University, Hamilton, ON., Canada. She completed her postdoctoral studies at the University of Toronto and the University Health Network. She performs fundamental and applied research in machine learning. Her current research interests include machine learning and related areas in computer vision, reinforcement learning, subspace learning for data clustering and classification, and anomaly detection.