Deep Clustering with Self-supervision using Pairwise Data Similarities

Mohammadreza Sadeghi [1] and Narges Armanfard [2]

[1]McGill University
[2]Affiliation not available

October 30, 2023

Abstract

Deep clustering incorporates embedding into clustering to find a lower-dimensional space appropriate for clustering. Most of the existing methods try to group similar data points through simultaneously minimizing clustering and reconstruction losses, employing an autoencoder (AE). However, they all ignore the relevant useful information available within pairwise data relationships. In this paper we propose a novel deep clustering framework with self-supervision using pairwise data similarities (DCSS). The proposed method consists of two successive phases. First, we propose a novel AE-based approach that aims to aggregate similar data points near a common group center in the latent space of an AE. The AE's latent space is obtained by minimizing weighted reconstruction and centering losses of data points, where weights are defined based on similarity of data points and group centers. In the second phase, we map the AE's latent space, using a fully connected network MNet, onto a K-dimensional space used to derive the final data cluster assignments, where K is the number of clusters. MNet is trained to strengthen (weaken) similarity of similar (dissimilar) samples. Experimental results on multiple benchmark datasets demonstrate the effectiveness of DCSS for data clustering and as a general framework for boosting up state-of-the-art clustering methods.

Deep Clustering with Self-supervision using Pairwise Data Similarities

Mohammadreza Sadeghi*, Narges Armanfard

Abstract—Deep clustering incorporates embedding into clustering to find a lower-dimensional space appropriate for clustering. Most of the existing methods try to group similar data points through simultaneously minimizing clustering and reconstruction losses, employing an autoencoder (AE). However, they all ignore the relevant useful information available within pairwise data relationships. In this paper we propose a novel deep clustering framework with self-supervision using pairwise data similarities (DCSS). The proposed method consists of two successive phases. First, we propose a novel AE-based approach that aims to aggregate similar data points near a common group center in the latent space of an AE. The AE's latent space is obtained by minimizing weighted reconstruction and centering losses of data points, where weights are defined based on similarity of data points and group centers. In the second phase, we map the AE's latent space, using a fully connected network MNet, onto a K-dimensional space used to derive the final data cluster assignments, where K is the number of clusters. MNet is trained to strengthen (weaken) similarity of similar (dissimilar) samples. Experimental results on multiple benchmark datasets demonstrate the effectiveness of DCSS for data clustering and as a general framework for boosting up state-of-the-art clustering methods.

Index Terms—Deep clustering, Autoencoder

1 Introduction

In many science and practical applications, information about category (aka label) of data samples is non-accessible or expensive to collect. Clustering, as a major data analysis tool in pattern recognition and machine learning, endeavors to gather essential information from unlabeled data samples. Clustering can address many issues in different real-world applications, such as the celestial data analysis [1], the medical analysis [2] the tumor's gene analysis [3], and the data retrieval [4, 5, 6, 7]. The main goal of clustering methods is to partition data points based on a similarity metric. Although recently a wide variety of clustering algorithms have been developed, the more traditional clustering algorithms k-means [8] and fuzzy c-means [9] are still appealing due to their simplicity. Nevertheless, these algorithms do not provide proper clustering performance when dealing with uneven distribution of samples. Moreover, they usually fail to properly cluster high-dimensional data points due to the curse of dimensionality; this makes them inappropriate in many recent applications where data points have many features [10]. Nowadays, novel deep-learning-based clustering methods have been widely developed in many applications such as image segmentation [11], social network analysis [12], face recognition [13], and machine vision [14]. The ultimate goal of these methods is to embed original data samples into a lower-dimensional space (aka latent space), where groups of data points could be distinguished by applying common algorithms such as k-means and fuzzy c-means. In literature, many research studies have been devoted to find an optimal lower-dimensional space in an unsupervised manner using different autoencoder (AE) structures [15, 16, 17]. This provides a highly nonlinear transformation of data points. An AE consists of two networks: 1) an encoder network that maps the original input space to a lower-dimensional space; 2) a decoder network that tries to reconstruct the original space using the output of the encoder network. Accordingly, encoder and decoder networks are trained to minimize reconstruction loss. As is discussed in Section 2, almost all conventional deep-learning-based algorithms aim to grouping data points near their corresponding cluster center using an AE. They attempt to include clustering loss in addition to the reconstruction loss of the AE to make the latent representation more suitable for data clustering [18, 19, 20]. The main common disadvantages of these algorithms are:

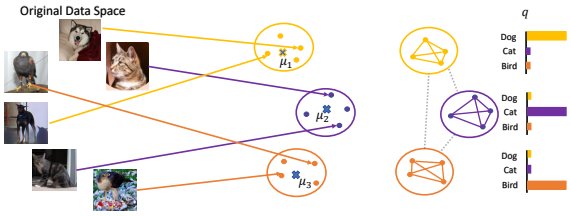


Fig. 1. The motivation of the proposed DCSS method. Arrows show a nonlinear mapping, using an AE, from the original input space to the AE's latent space (i.e the u space). In the first phase of DCSS, we aim to gathering data points near their corresponding group centers in the u space. In the second phase, DCSS employs pairwise samples similarity and dissimilarity to create a new space q in which similar pairwise samples are tightly packed together and dissimilar pairwise samples sit as far away as possible. Similar samples are connected with solid lines and dashed lines represent dissimilar samples/clusters.

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

(1) At each training epoch, they first perform crisp cluster assignment, i.e. assign each data point to only one cluster, then compute the data clustering loss using the obtained crisp assignments assuming they are correct assignments. This is not a correct assumption because of the unsupervised nature of the clustering task where the true cluster labels are unknown during the training phase. This issue would be more serious when the non-crisp (aka soft) K-dimensional cluster assignment vector obtained prior to snapping to the one-hot vector is far from the one-hot vector imposed by the crisp assignment, because this results in misleading of the training process. K is the number of clusters. (2) In all of the former methods, a single common loss function is utilized for all data clusters, without considering the possible existence of differences between characteristics of the different clusters (3) All the conventional deep-learning-based clustering algorithms seek an optimal latent space suitable for data clustering. To this end, they try to locate data points close to their corresponding cluster center without considering relations between the data points, a relation such as pairwise data similarities.

In this paper, we propose a deep clustering with self-supervision using pairwise data similarities (DCSS) framework. DCSS employs similar and dissimilar pairwise samples to supervise the training process, where similarities are measured in reliable subspaces. DCSS performs clustering in two phases. First, the original data points are mapped into a reliable latent space u, i.e. latent space of an AE, in which samples of the same cluster are encouraged to form a group through sitting close to the corresponding group center. The second phase is to create an improved subspace q which is trained under supervision of similar and dissimilar pairwise samples. Since similar and dissimilar samples are not recognizable in the original input feature space, due to the curse of dimensionality [21], we propose to use the (partially) trained subspaces u and q for similarity measurement. More specifically, at the q training outset, similar and dissimilar pairwise samples are determined within the u subspace which is previously trained in the first phase. After a few training epochs, when the subspace q becomes a reliable space, the similarities are measured in the q subspace itself. The subspace q is trained such that similar pairwise samples are tightly packed together in the q space and dissimilar pairwise samples sit in q as far away as possible. Pairwise samples that are neither similar nor dissimilar, i.e. those that are in ambiguous region, do not contribute in the second phase of DCSS. Our experimental results confirm that as the training in the second phase progresses, through performing more and more training epochs, more and more pairwise samples contribute to the training, i.e. very few pairs remain in the ambiguous region at the end of the training phase. Our experiments show that such training process results in sample representations, in the q space, that are very close to one-hot vectors where the active element of the vector indicates the true cluster assignment, assuming the dimension of the q subspace is equal to the true number of data clusters K. An intuitive motivation of the proposed method is illustrated in Fig. 1. Theoretical supports for the proposed DCSS method are provided in Appendixes A. Training at both of the DCSS phases is performed in an end-to-end manner employing gradient descent and back-propagation. In the second phase which is mainly for q training, the u space also receives small updates to be refined and more helpful for the q space training.

In the first phase of DCSS, to obtain the subspace u, we propose to train an autoencoder by simultaneously minimizing two losses: samples reconstruction loss and centering loss; these two losses are linearly combined in a weighted manner and form the final loss function. The latent space of the trained AE is then taken as the DCSS's u subspace. Minimizing the reconstruction loss helps to preserve the local structure of the data points [20] and constraints too much manipulation of the AE latent space by the centering loss. Since pairwise similarities measured in the u space are used for supervising the q space training, data distribution in the u space should be close to the desired data distribution in the q space, i.e. having dense pure groups of similar data in q. This goal is realized by incorporating the data centering loss in the loss function of Phase 1. The subspace u is trained in K successive runs where at each run a specific loss function, corresponding to a specific data group, is minimized. More specifically, at the kth run, $k \in 1, ..., K$, the AE network focuses on centering and reconstruction of those samples that are more likely to belong to the kth data group. This implicitly means that, at the kth run, those AE's parameters are optimized that have more impact on centering of the kth data group, around their corresponding center, in the u space. Despite the previous deep clustering methods that use their estimated crisp assignments for training a latent space, we propose to train the u subspace employing the original soft group assignments (without snapping to 0 and 1). The soft assignments are used as samples weight when computing the reconstruction and centering losses. Block diagram of Phase 1 for training the u space is shown in Fig. 2(a). Throughout this paper, we refer to the proposed method for finding u as Deep Successive Learning (DSL). Details of the DSL formulation are presented in Section 3.1.

In the second phase of the proposed DCSS method, despite the conventional AE-based clustering methods that define the final cluster assignments based on the data representations in the AE's latent space [18, 19, 20, 22], we propose to employ the pairwise relationship between the data points as a supervisor to map the u space representations to an improved space q. Such map is realized through a fully connected network called mutual net (MNet) that its objective is to strengthen pairwise similarities and dissimilarities, where the inner product is used for similarity measurement. Overview of the second phase of the DCSS method is shown in Fig. 2(b). Final cluster assignment is then performed in the discriminative space q (see Fig. 2(c)).

To sum up, the main contributions of this work are:

- We discover the novel framework of employing similarities of pairwise samples as a means of self-supervision for data clustering.
- To realize this, since the original data space suffers from the curse of dimensionality, we propose to measure the similarities in two reliable subspaces u and q that are trained successively.
- To create the reliable space u, we propose a novel AE-

based subspace learning approach DSL that trains the AE through minimizing weighted reconstruction and centering losses of data points, where soft group assignments are used as samples weight. See Fig. 2(a). The latent space of the AE is used as the u space required in the DCSS.
- We propose to create the more reliable subspace q, through appending a fully connected network MNet to the encoder part of the AE trained by DSL, where the process of training q (and refining u) is supervised by the similarity of the pairwise samples. See Fig. 2(b).
- Final crisp cluster assignment of a given data point is realized by mapping it to the q space. The map is performed by feeding the data to the trained encoder followed by the trained MNet. Our experiments show that samples representation in the q space, are very close to one-hot vectors. Assuming the dimension of the q space is set to the true number of clusters K, the index of the maximum element of the q vector indicates the input data cluster label. See Fig. 2(c).
- The proposed method can be considered as a general framework that can be employed to improve the clustering performance of the existing subspace-based clustering methods such as [22, 20, 19, 18]. More specifically, the reliable subspace u can be obtained with other state-of-the-art methods (instead of DSL) followed by the proposed MNet.
- We performed an extensive set of experiments, on seven benchmark datasets, to demonstrate the effectiveness of the proposed DCSS (and DSL). Our results outperform both traditional and state-of-the-art deep network-based clustering methods.

Appendix A presents mathematical proofs that show, under certain assumptions, DCSS can effectively capture pairwise similarities and dissimilarities; hence DCSS performs a final reasonable cluster assignment. We proved that samples' q vector is very close to an one-hot vector (Corollary 1.2), similar samples are assigned to the same cluster (Theorem 2), dissimilar samples are assigned to different clusters (Corollary 1.3), and if two samples are similar to another sample then the two samples are not dissimilar (Theorem 3).

The rest of this paper is organized as follows. In Section 2, we present a brief review of conventional and deep-learning-based clustering methods. Section 3 presents details of our proposed DCSS framework. Extensive experimental results that demonstrates the effectiveness of the DCSS is presented in Section 4. Section 5 conveys the gist of this paper.

2 Related Work

Clustering have been widely studied in machine learning from different aspects such as feature selection [23, 24, 25], distance metric [26, 27], and categorizing methods [28, 29, 30]. K-means [8] and fuzzy c-means [9] are the two popular conventional methods that are applicable to a wide range of tasks [31, 32, 33]. However, because of their distance metric, they only can extract local information of the data dealing with high dimensional feature spaces. Some conventional algorithms, such as [34], aim to handle this difficulty by jointly performing subspace selection and a clustering algorithm in an iterative manner. At each iteration, they group data points using k-means and endeavor to maximize the inter-cluster variance employing the data projected in a lower-dimensional space. They repeated this process until convergence. Another group of conventional methods, called spectral clustering such as [35, 36], address the high input dimension issue by embedding the high dimensional data into a lower-dimensional space. They then apply clustering algorithms in the new space. For their embedding phase, first, they construct a weighted graph in which nodes are data samples and weights are defined based on pairwise relationship between them in the original space. Then, they define a minimization problem using the Laplacian matrix of the weighted graph. Although they could surpass the clustering performance of k-means in different applications, the complexity of solving the optimization problem limited the application of these algorithms to only small datasets. In order to make spectral clustering more applicable on large datasets, [37, 38] propose stochastic optimization methods that try to estimate the original optimization problem. However, these conventional methods only consider linear embedding that is not successful when dealing with complex datasets. In order to take into consideration the nonlinear embedding of the data points, deep clustering has been broadly studied during the past few years. They utilize deep autoencoders to embed original data points in a lower-dimensional space. In some algorithms, learning the lower representation of the data points is separated from the clustering task. For example, deep embedding network (DEN) [39] uses an AE to find a lower-dimensional representation of data points by enforcing group sparsity and locality-preserving constraints. They then obtain cluster assignments by applying k-means algorithm to the obtained lower-dimensional space. As another example, [40] takes advantage of a deep autoencoder to find lower-dimensional representation of a graph; it then utilizes k-means algorithm to define clusters. In order to further improve clustering performance, more recent algorithms simultaneously embed data points in a lower-dimensional feature space and perform clustering algorithms to assign data points to different clusters using the new feature space. E.g. Deep embedded clustering (DEC) [22] initializes parameters of a stacked autoencoder layer by layer using [41, 42]. After removing the decoder part, it then updates encoder part of the AE through minimizing a Kullback–Leibler (KL) divergence loss consisting of soft assignments and estimated target distributions. Soft assignments measure degree of similarity between data points and cluster centers using Students' t-distribution following [43]. In an unsupervised learning problem, since true cluster assignments (aka target distributions) of data points are unknown, they perform a form of self-training [44] to estimate target distributions using the soft cluster assignments. There have been a few research studies, e.g [19, 18, 20], proposed to take advantage of the decoder part of an AE to combine reconstruction loss with clustering loss to maintain the local structure of the original data points. For example, improved deep embedding clustering (IDEC) [20] improved the clustering

performance of DEC by considering the reconstruction loss of an AE besides the KL divergence loss of DEC to preserve local dependencies and structures between the original data points. Deep convolutional embedded clustering (DCEC) [45] could enhance the performance of IDEC by changing the fully connected structure of IDEC to a deep convolutional autoencoder. Moreover, DCEC proposed an end-to-end pre-training scheme by minimizing the reconstruction loss instead of pre-training a stacked autoencoder proposed in DEC and IDEC. Some other works, instead of applying self-training using soft cluster assignments, improve clustering performance of DEC by proposing an independent approach of finding target distributions. For example, improved deep embedding clustering with fuzzy supervision (IDECF) [46] aims to estimating the target distributions by training a deep fuzzy c-means network, which is specifically designed and trained for this purpose. Deep clustering network (DCN) [19] method, which jointly learns lower-dimensional feature representation and performs clustering, aims to find a new representation of data points in which data points are separable by applying k-means algorithm. To this end, DCN updates an autoencoder's parameters by minimizing a combination of the reconstruction loss and the objective function of k-means algorithm; this is to find a k-means friendly space. DCN updates AE's parameters and cluster centers separately. The latter is based on solving a discrete optimization problem. However, in deep k-means (DKM) [18], which has the same objective function as DCN while minimizing the parameters of an AE, weights and cluster centers are updated through minimizing a continuous optimization problem using stochastic gradient descent (SGD). Some recent works such as Deep Spectral Clustering (DSC) [47] proposed a joint learning framework to create a discriminative embedded space using a dual autoencoder. They devise a common encoder part and two decoder parts for their autoencoder. The first decoder tries to reconstruct the original input and the second encoder endeavors to denoise the encoder latent space. They consider reconstruction, mutual information, and spectral clustering losses while updating parameters of their network. They utilized mutual information to exploit more discriminative information from the inputs. Moreover, a deep spectral clustering approach is applied to extract the mutual relationship between data points in the latent space. Contrastive learning methods, such as [48, 49], have been widely attracted researchers' attention in the recent year due to their promising performance. They first construct negative and positive pairs by applying data augmentation on data points. Then, they map them in the feature space and endeavor to maximize similarity (minimize dissimilarity) between positive (negative) pairs. For example, contrastive clustering (CC) [48] defines instance- and cluster-level losses respectively on rows and columns of the feature space in order to maximize similarity while minimizing dissimilarity.

3 Proposed Method

Consider a K-clustering problem which aims to partitioning a given dataset $X = \{x_1, x_2, ...x_N\}$ into K disjoint clusters, where x_i indicates the ith data sample, N is the number of data points, and K is a predefined user-settable parameter. DCSS utilizes an AE, consisting of an encoder and a decoder network respectively represented by $f(.)$ and $g(.)$. Latent representation of X is denoted by $U = \{u_1, u_2, ..., u_N\}$, where $u_i = f(x_i; \theta_e) \in R^d$, d indicates dimension of the latent space, and θ_e denotes parameters of the encoder network. The reconstructed output of the AE is denoted by $\hat{x}_i = g(u_i; \theta_d)$, where θ_d represents the decoder parameters. Center of the kth data group in the u space is denoted by $\mu^{(k)}$. As aforementioned, in the second phase of DCSS, to investigate the mutual relationship between the data points, we propose to employ a fully connected network MNet, which takes the latent representation of the AE for each data point, i.e. u_i, as input and maps it to a K-dimensional vector q_i which its kth element indicates the probability of x_i belonging to the kth data cluster. In this paper, the output of MNet for the ith data point is denoted by $q_i = M(u_i; \theta_M)$, where $M(.)$ and θ_M respectively shows MNet and its corresponding parameters.

3.1 The first phase of DCSS (DSL)

To obtain a reliable low dimensional space u in which detection of the similar and dissimilar samples is possible, we propose to train an AE with a novel and effective loss function consisting of weighted reconstruction and centering losses. The latent space of the AE is in fact the DCSS's u space. At each training data batch, we propose to train the AE in K successive runs where at the kth run, the AE focuses on reconstruction and centering of the data points that are more probable to belong to the kth data group. Note that data clustering is an unsupervised task and the group/cluster membership of the data points is unknown at the problem outset. As such, at the kth run of DSL, we use Euclidean distance between u_i and $\mu^{(k)}$ (obtained in the previous training iteration), i.e. $||u_i - \mu^{(k)}||_2$, as a means of measuring the degree of membership of x_i to the kth data group in the u space. The group memberships are used as the sample weight in the (k+1)th run. More specifically, the closer a sample is to the group center $\mu^{(k)}$, the higher contribution that sample has in minimizing the loss function at the kth run.

Formulation of the proposed DSL method is shown in equation set (1). (1a) presents the DSL's loss function at the kth run, i.e. $\mathcal{L}_u^{(k)}$, where weighted summation of the samples reconstruction and centering losses are respectively denoted by $\mathcal{L}_r^{(k)}$ and $\mathcal{L}_c^{(k)}$. α is a hyperparameter indicating the importance of centering loss vs. the reconstruction loss. m indicates the level of fuzziness and is set to 1.5 in all experiments. In the kth run, to motivate the AE to concentrate on the kth group data points, in $\mathcal{L}_r^{(k)}$ and $\mathcal{L}_c^{(k)}$, higher weights are assigned to the samples closer to the group center $\mu^{(k)}$. Membership of the ith data point to the kth data group, in the u space, is shown as p_{ik} defined in (2). p_{ik} is used as the weight of the ith sample reconstruction and centering losses, at the kth run of DSL.

$$\mathcal{L}_u^{(k)} = \mathcal{L}_r^{(k)} + \alpha \mathcal{L}_c^{(k)} \tag{1a}$$

$$\mathcal{L}_r^{(k)} = \sum_{x_i \in \mathfrak{B}} p_{ik}^m ||x_i - \hat{x}_i||_2^2 \tag{1b}$$

$$\mathcal{L}_c^{(k)} = \sum_{x_i \in \mathfrak{B}} p_{ik}^m ||u_i - \mu^{(k)}||_2^2 \tag{1c}$$

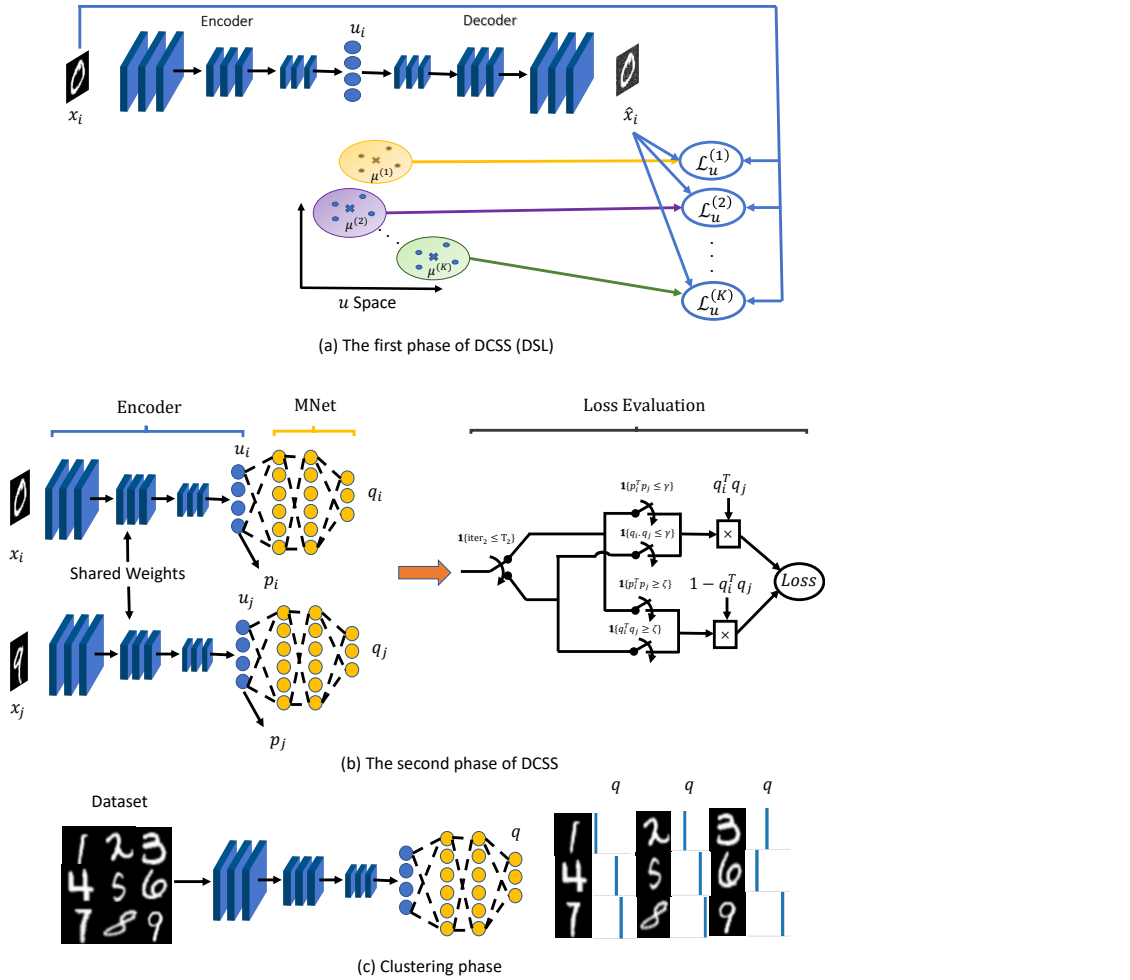


Fig. 2. (a) Training scheme of the first phase of DCSS, called deep successive learning (DSL). (b) Training procedure of the second phase of DCSS. At the outset, when $\text{iter}_2 \le \text{T}_2$, MNet is trained based on the pairwise similarities defined in the u space – i.e. The similarity between two data points x_i and x_j is determined using the dot product of p_i and p_j. At the later stage of MNet training, when $\text{iter}_2 > \text{T}_2$, the pairwise similarities are measured in the q space itself using $q_i^T q_j$. (c) Visualization of the final cluster assignment using DCSS. After completing the training phase shown in (a) and (b), we cluster a data point x_i by locating the largest element in q_i.

Every T_1 training iterations, we update the group centers to average of weighted samples in the u space, as is shown in (3); i.e. samples closer to $\mu^{(k)}$ contribute more in updating.

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

$$\mu^{(k)} = \frac{\sum_{x_i \in X} p_{ik}^m u_i}{\sum_{x_i \in X} p_{ik}^m} \tag{3}$$

Block diagram of the DSL framework is shown in Fig. 2(a). A preliminary version of DSL is presented in [50].

3.2 The second phase of DCSS

After completing the first phase of the DCSS method, we discard the decoder part of the DSL's autoencoder and append a fully connect network MNet to the trained encoder. More specifically, the MNet's input is the latent space of the DSL's encoder, i.e. the u space. We take pairwise samples similarity to supervise the MNet training phase. The MNet output space q is used for data clustering; hence, it is trained to strengthen similarities and dissimilarities. The MNet's output layer, i.e. the q space, consists of K neurons where each neuron corresponds to one data cluster. We utilize the softmax function at the output layer to obtain probability values for clusters assignment. More specifically, for input sample x_i, the output value at the jth neuron, i.e. q_{ij}, is the probability of x_i belonging to the jth cluster.

At the problem outset, MNet parameters, θ_M, are initialized with random values. Hence, at the first few epochs of the q training process, when q is not yet a reliable space, we measure pairwise similarities in the u space, using the inner product of the group assignments shown in (2). More specifically, knowing that $p_i = [p_{i1}, .., p_{iK}]^T$ represents the assignment vector of the ith data point x_i to the different data groups in the u space, the inner product of p_i and p_j shows the pairwise similarity of the two data points x_i and x_j. The loss function proposed for the MNet training at the first T_2 training epochs is shown in (4) where ζ and γ are two user-settable hyperparameters, and $\mathbb{1}\{.\}$ is the indicator function.

$$\mathcal{L}_M = \sum_{x_i, x_j \in \mathfrak{B}} \mathbb{1}\{p_i^T p_j \geq \zeta\}(1 - q_i^T q_j) + \mathbb{1}\{p_i^T p_j \leq \gamma\}(q_i^T q_j) \tag{4}$$

As can be inferred from (4), only similar and dissimilar samples contribute in the MNet training phase – i.e., a pair of samples contributes to the training if their similarity is greater than ζ or less than γ. A Pair of samples with a similarity value between ζ and γ, i.e. in the ambiguity region, does not contribute to the current training epoch. Therefore, minimizing $\mathcal{L}_M$ strengthens (weakens) the similarity of similar (dissimilar) samples. Note that, along with training the MNet parameters, the encoder parameters θ_e are also updated through back-propagation, in an end-to-end manner. Group centers in the u space, i.e. $\mu^{(k)}$ are also updated using (3) after completing each training epoch.

After training MNet (and refining encoder) parameters through minimizing $\mathcal{L}_M$ for T_2 epochs, when q becomes a reliable space for pairwise similarities measurement, we use the loss function $\mathcal{L}'_M$ defined in (5) to complete the MNet training. Similar to (4), a pair contributes in $\mathcal{L}'_M$ if its corresponding similarity value is not in the ambiguity region. As is shown in Section 4, as the MNet training progresses, more and more pairs contribute to the training procedure. Again, the u space receives small updates through the backpropagation process when minimizing $\mathcal{L}'_M$.

$$\mathcal{L}'_M = \sum_{x_i, x_j \in \mathfrak{B}} \mathbb{1}\{q_i^T q_j \geq \zeta\}(1 - q_i^T q_j) + \mathbb{1}\{q_i^T q_j \leq \gamma\}(q_i^T q_j) \tag{5}$$

As is proved in Appendix A, a proper choice for of hyperparameters ζ and γ is $\frac{2}{3} < \zeta$ and $\gamma < \zeta^2$. In our experiments ζ and γ are set to 0.8 and 0.2, respectively.

Fig. 2(b) shows the overall training procedure of the DCSS's second phase. Moreover, Appendix A presents several mathematical proofs that show, under certain assumptions, the final q vector for a query sample is very close to an one-hot vector where the index of the maximum element of the vector indicates the cluster label. This also shows that similar (dissimilar) samples tend to sit in the same (different) data cluster(s).

3.3 Final cluster assignments

To determine the final cluster assignment of a data point x_i, we utilize the trained encoder and MNet networks to compute the data representation in the K-dim q space, i.e. q_i. x_i is assigned to the most probable cluster, i.e. the index corresponding to the highest q_i element. This clustering assignment process is shown in Fig. 2(c).

The pseudo-code of the DCSS algorithm is presented in Algorithm 1.

Algorithm 1 Clustering procedure using DCSS

Input: Data points X, θ_e, θ_d, θ_M, $\mu^{(k)}$ for $k = 1, \ldots, K$
Output: θ_e, θ_M

The first phase:
Initialize θ_e and θ_d with a pre-trained network (see Section 4.4).
for iter$_1 \in \{1, 2, ..., \text{MaxIter}_1\}$ **do**
  for $k \in \{1, 2, ..., \text{K}\}$ **do**
    Compute soft group assignments p_{ik} using (2), for $i \in \mathfrak{B}$
    Update AE's parameters by employing (1a) as loss function
  end for
  Every T_1 iterations, update cluster centers using (3)
end for

The second phase:
for iter$_2 \in \{1, 2, ..., \text{MaxIter}_2\}$ **do**
  if iter$_2 \leq \mathrm{T}_2$ **then**
    Compute soft group assignment vectors p_i for $i \in \mathfrak{B}$
    Compute soft cluster assignments vectors q_i for $i \in \mathfrak{B}$
    Update θ_e and θ_M to minimize (4)
    Update group centers $\mu^{(k)}$, $k = 1, \ldots, K$, using (3)
  else
    Compute q_i for $i \in \mathfrak{B}$
    Update θ_e and θ_M to minimize (5)
  end if
end for

Final Cluster Assignments:
Compute q_i for x_i, $i = 1, \ldots N$
Assign each data sample to the most probable cluster

4 Experiments

In this section, the effectiveness of our proposed DCSS framework is demonstrated on seven benchmark datasets through conducting a rigorous set of experiments. The DCSS clustering performance on these seven benchmark datasets is compared with ten conventional and state-of-the-art deep-learning-based clustering methods. The DCSS code is available: **https://github.com/Armanfard-Lab/DCSS**.

4.1 Datasets

The effectiveness of the proposed method is shown on seven widely used datasets. Considering unsupervised nature of the clustering task, we concatenate training and test sets when applicable. Combining train and test datasets is a common practice in the clustering research field [22, 18, 20, 19, 47]. The datasets are:
(1) MNIST [51] consists of 60,000 training and 10,000 test gray-scale handwritten images with size 28×28. This dataset has 10 classes.
(2) Fashion MNIST [52] has the same image size and number of samples as of MNIST. However, instead of handwritten images, it consists of different types of fashion products. This makes it fairly more complicated for data clustering compared to the MNIST dataset. It has 10 classes of data.
(3) 2MNIST is a more challenging dataset created through concatenation of the two MNIST and Fashion MNIST datasets. Thus, it has 140,000 gray-scale images from 20

TABLE 1
ACC and NMI (in parenthesis) on the benchmark datasets for different clustering methods.

Method \ Datasets	MNIST	Fashion MNIST	2MNIST	USPS	CIFAR-10	STL-10	CIFAR-100
k-means	53.20 (50.00)	47.40 (51.20)	32.31 (44.00)	65.67 (62.00)	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)	63.14 (58.94)	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)	60.82 (54.47)	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)	75.81 (76.91)	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)	75.86 (77.68)	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)	73.00 (71.90)	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)	75.70 (77.60)	35.26 (26.12)	32.61 (29.12)	18.14 (12.30)
DSC	97.80*(94.10*)	66.20*(64.50*)	42.36 (46.53)	86.90*(85.70*)	22.50 (8.65)	25.60 (15.69)	21.12 (13.00)
AE + k-means	86.03 (80.25)	57.94 (57.15)	44.01 (62.80)	75.11 (74.45)	80.11 (70.35)	95.89 (91.75)	49.86 (48.57)
CC	88.56 (84.21)	64.52 (61.45)	42.15 (58.89)	81.21 (79.45)	79.00 (70.50)	85.00 (76.40)	42.90 (43.10)
DSL	95.99 (89.95)	62.90 (63.58)	45.31*(63.00*)	82.93 (81.84)	83.40*(71.32*)	96.02*(91.90*)	50.30*(49.80*)
DCSS	**98.00 (94.71)**	**66.40 (66.94)**	**48.57 (67.80)**	**87.21 (86.10)**	**85.32 (73.35)**	**97.58 (93.74)**	**51.10 (50.59)**

TABLE 2
ACC and NMI (in parenthesis) on the benchmark datasets when employing DCSS as a framework to improve state-of-the-art AE based clustering methods.

Method \ Datasets	MNIST	Fashion MNIST	2MNIST	USPS	CIFAR-10	STL-10	CIFAR-100
DEC+MNet	89.13 (86.97)	61.25 (56.30)	44.25 (57.35)	77.58 (78.15)	83.40 (69.25)	96.10 (92.00)	48.86(50.11)
IDEC+MNet	90.51 (85.42)	60.12 (57.16)	44.83 (58.00)	76.58 (78.14)	83.95 (69.93)	96.18 (92.89)	47.86 (50.00)
DCN+MNet	87.49 (83.25)	54.23 (58.69)	45.62 (48.24)	76.90 (77.59)	71.23 (62.38)	92.59 (86.23)	47.92 (45.81)
DKM+MNet	88.31 (84.52)	57.23 (56.26)	44.34 (49.50)	77.13 (78.02)	82.26 (69.50)	96.00 (91.88)	40.22 (49.81)

classes.
(4) USPS [53] contains of 9,298 16×16 handwritten images from the USPS postal service. It contains 10 classes of data.
(5) CIFAR-10 [54] is comprised of 60,000 RGB images of 10 different items, where the size of each image is 32×32.
(6) STL-10 [55] is a 10-class image recognition dataset comprising of 13,000 96×96 RGB images.
(7) CIFAR-100 [54] is similar to the CIFAR-10, except it has 20 super groups based on similarity between images instead of 10 classes.

4.2 Evaluation Metrics

We utilize two standard metrics to evaluate clustering performance, including clustering accuracy (ACC) [57] and normalized mutual information (NMI) [58]. ACC finds the best mapping between the true and predicted cluster labels. NMI finds normalized measure of similarity between two different labels of the same data point. The ACC and NMI formulations are shown below:

$$ACC = \max_m \frac{\sum_{i=1}^{N} \mathbb{1}\{l_i = map(c_i)\}}{N} \tag{6a}$$

$$NMI = \frac{I(l;c)}{max\{H(l), H(c)\}} \tag{6b}$$

where l_i and c_i denote the true and predicted labels for the data point x_i. $map(.)$ indicates the best mapping between the predicted and true labels of data points. $I(l;c)$ denotes the mutual information between true labels $l = \{l_1, l_2, ..., l_N\}$ and predicted cluster assignments $c = \{c_1, c_2, ..., c_N\}$ for all data points. $H(.)$ presents the entropy function. ACC and NMI range in the interval [0,1] where higher scores indicate higher clustering performance.

4.3 Networks Architecture

The proposed DCSS method includes an autoencoder and a fully connected MNet. This section presents structure of these networks. We use two variations of autoencoders, depending on the dataset nature (i.e. RGB or gray-scale), when training the proposed DCSS framework.

For gray-scale datasets, we propose to use an asymmetric autoencoder; where, following [56], we propose to use the bottleneck layer shown in Fig. 3(c) in the encoder structure. Fig. 3(a) and (b) respectively show the encoder and decoder structures of the proposed asymmetric AE. Employing such asymmetric structure provides a more discriminative latent space. Hyperparameters of the proposed AE for each dataset are indicated in Section 4.4

For the RGB datasets, we first apply a ResNet-152 [56], pre-trained on ImageNet [59], to extract abstract features. Then we feed the extracted features to a symmetric fully connected AE. Inspired by [22], we set the AE architecture to 2048-500-500-2000-d for RGB datasets, where ReLU activation function is utilized in all layers.

MNet is a fully connected network that takes the d dimensional latent space of the AE (u space) as input and generates a K dimensional output q. The architecture of MNet is d-128-128-128-K for all datasets except CIFAR-100. Since CIFAR-100 is a more complicated dataset, it needs a more complex MNet architecture; so we set the MNet architecture for CIFAR-100 to d-1000-1000-1000-K. Batch normalization and ReLU activation function is utilized in for all datasets in all layers of MNet except the last layer in which we used softmax function.

4.4 Implementation Details

In this section, we discuss hyperparameter values and implementation details of DCSS.

Network's hyperparameters n, p_1, s_1, p_2, s_2, p_3, s_3, f_1, and f_2 (shown in Fig. 3) are respectively set to 28, 2, 2, 1,

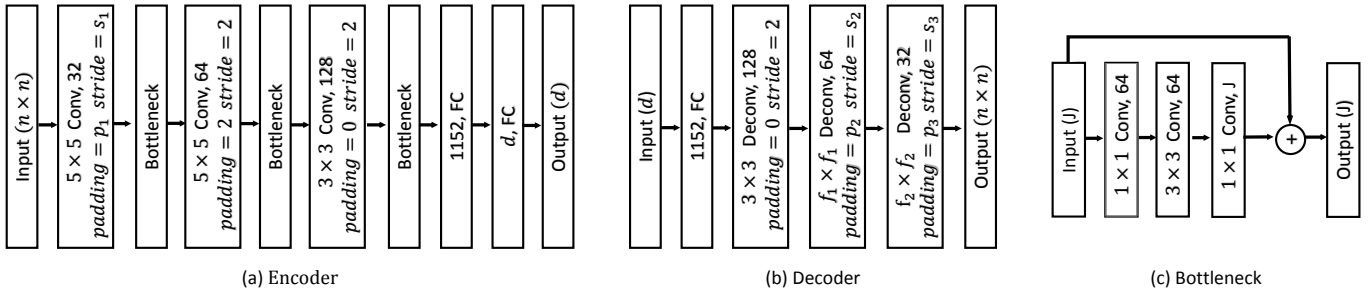


Fig. 3. Structure of the proposed asymmetric autoencoder. In the encoder part, in order to obtain an informative lower-dimensional representation of data points, we proposed to use a Bottleneck layer. Following [56], we use the bottleneck layer after 5×5 and 3×3 convolutional layers. the value of hyperparameters is presented in Section 4.4.

2, 2, 2, 5, and 4 for MNIST, Fashion MNIST, and 2MNIST; these parameters are set to 16, 1, 1, 2, 2, 0, 1, 4, and 5 for the USPS dataset. The latent space dimension d is set to 10 for gray-scale images and 20 for RGB images.

Following [18, 19, 20, 22], in order to initialize the parameters of DSL's, i.e. θ_e, θ_d and $\mu^{(k)}$ for $k = 1, \ldots, K$, we train an autoencoder where the end-to-end training is performed by only minimizing the samples reconstruction losses. Adam optimization method [60], with the same parameters mentioned in the original paper are used for training. θ_e, θ_d are then initialized with the parameters of the trained autoencoder's parameters. We apply k-means algorithm [8] to the latent space of the trained autoencoder and initialize $\mu^{(k)}, k = 1, \ldots, K$ to the centers defined by k-means.

For all datasets, in the first phase of DCSS, α, Maxiter_1, T_1, and m are respectively set to 0.1, 200, 2, and 1.5. The second phase hyperparameters ζ, γ, T_2, and MaxIter_2 are respectively set to 0.8, 0.2, 5, and 20. We utilize Adam optimizer for updating weights of the AE and MNet and their learning rates are set to 10^{-5} and 10^{-3}, respectively.

All algorithms were implemented in Python using PyTorch framework. All codes are run on Google Colaboratory GPU (Tesla K80) with 12GB RAM. The proposed algorithm codes are included in the supplementary materials.

4.5 Clustering Performance

The effectiveness of our proposed DCSS method is compared against ten well-known algorithms, including conventional and state-of-the-art deep-learning-based clustering methods, using the commonly used evaluation metrics ACC and NMI.

The conventional clustering methods are k-means [8], large-scale spectral clustering (LSSC) [61], and locality preserving non-negative matrix factorization (LPMF) [62]. Deep-learning based algorithms are deep embedding clustering (DEC) [22], improved deep embedding clustering (IDEC) [20], deep clustering network (DCN) [19], deep k-means (DKM) [18], deep spectral clustering (DSC) [47], and the very recent contrastive clustering method (CC) [48]. In addition, we report clustering performance of a baseline method AE + k-means in which k-means is simply applied to the latent representation of an AE, with similar architecture as of the AE used in the DCSS method, trained based on minimizing the dataset reconstruction loss. More details about the comparing algorithms can be find in Section 2.

We also demonstrate the success of the DSL method, presented in Section 3.1, in creating a reliable subspace u where the data points are effectively grouped around their corresponding center. To this end, we only implement the first phase of the DCSS algorithm – i.e. we train the DCSS's AE through minimizing the loss function presented in (1), where the AE architecture and its initialization are similar to those presented in Section 4.3 . After training the u space, we perform a crisp cluster assignment by considering each data group, in the u space, as a data cluster and assigning each data point to the cluster (group) with closest center.

the clustering performance of our proposed DSL and DCSS along with our comparison algorithms are shown in Table 1. For the comparison methods, if the ACC and NMI results for a dataset are not reported in the corresponding original paper, we ran the released code with the same hyper-parameters discussed in the original paper. The best result for each dataset is shown in bold. The second top results are shown with *. Several observations can be made from this table. **(1)** The proposed DCSS method outperforms all of our comparison methods on all datasets. **(2)** The proposed DSL approach effectively centers the data around the group centers. This can be inferred from its ACC and NMI results. Indeed DSL outperforms all the clustering methods except DSC which is one of the very most recent state-of-the-art AE-based clustering methods. DSL outperforms DSC in 4 out of the 7 datasets and provides competitive results on the remaining three ones. **(3)** Effectiveness of the second phase of the proposed DCSS framework, where we append MNet to the latent space of the DSL, can be inferred by comparing DCSS with DSL. It can be seen that DCSS significantly outperforms DSL on all the datasets. **(4)** Effectiveness of the AE loss function proposed in equation (1) compare to the case of training AE with only reconstruction loss can be inferred by comparing the DSL performance with the baseline method AE+K-means. As can be seen, the DSL clearly outperforms AE+K-means on all the datasets.

4.6 t-SNE visualization

Fig. 4 illustrates the effectiveness of different phases of our proposed DCSS framework for all datasets, where t-SNE [63] is used to map the output of DCSS's encoder/MNet to a 2D-space. The different colors correspond to the different data groups/clusters.

The first row of Fig. 4 shows the representation of different data points in the u space, i.e. the latent space of the

TABLE 3
ACC and NMI (in parenthesis) for different extracted features.

Dataset	Method	Resnet-34	Resnet-50	Resnet-101	Resnet-152
STL-10	DEC	90.10 (83.40)	91.50 (84.20)	95.30 (90.20*)	95.88 (91.01)
	IDEC	92.70 (84.00)	93.40 (86.15)	95.81* (89.41)	96.00 (92.14)
	DCN	78.12 (77.51)	80.25 (81.12)	90.14 (86.43)	90.25 (84.03)
	DKM	92.40 (88.20*)	91.10 (88.14*)	92.51 (88.57)	95.50 (91.30)
	AE+kmeans	92.60 (86.00)	93.20 (87.10)	95.00 (89.90)	95.89 (91.75)
	DSL	92.85* (86.26)	93.51* (87.63)	95.20 (90.07)	96.02* (91.90*)
	DCSS	**94.51 (88.35)**	**94.60 (89.37)**	**96.40 (92.00)**	**97.58 (93.74)**
CIFAR-10	DEC	72.51 (64.21)	73.25 (61.10)	78.24 (67.41)	81.10 (68.21)
	IDEC	71.41 (64.01)	74.41 (62.30)	79.65 (68.00)	81.24 (68.57)
	DCN	54.71 (53.29)	56.20 (51.23)	71.00 (46.20)	64.20 (59.02)
	DKM	69.52 (60.41)	70.20 (61.25)	80.31 (68.00)	81.90 (69.10)
	AE+kmeans	78.80 (67.81)	75.60 (62.80)	82.90 (70.80)	80.11 (70.35)
	DSL	79.02* (68.01*)	76.00* (62.95*)	83.12* (71.20*)	83.40* (71.32*)
	DCSS	**79.80 (70.61)**	**76.40 (64.21)**	**84.50 (73.13)**	**85.32 (73.35)**
CIFAR-100	DEC	41.56 (47.52*)	45.10 (44.25)	43.25 (46.61)	45.38 (49.41)
	IDEC	42.51 (46.41)	45.61 (45.00)	43.45 (47.29)	44.91 (49.25)
	DCN	40.36 (42.58)	41.25 (41.58)	43.15 (42.64)	44.68 (43.00)
	DKM	36.80 (46.82)	35.61 (40.29)	37.84 (47.25)	37.40 (46.20)
	AE+kmeans	47.30 (46.10)	47.36 (45.09)	47.35 (47.01)	49.86 (48.57)
	DSL	47.66* (46.52)	47.80* (45.44*)	47.43* (47.53*)	50.30* (49.80*)
	DCSS	**50.10 (48.90)**	**48.76 (46.40)**	**48.82 (49.00)**	**51.10 (50.59)**

DCSS's AE, only after completing the first phase discussed in Section 3.1. As it can be seen, after completing the first phase of DCSS, different groups of data points are fairly separated and sit near group centers; however, this phase is not adequate for defining cluster assignments. For example, in the USPS dataset, data groups shown in pink, purple, and magenta are mixed together. This indicates insufficiency of the reconstruction and centering losses.

The second row of Fig. 4 shows the data representations in the u space after completing the second phase of DCSS discussed in Section 3.2, where u is refined by minimizing (4) and (5). As can be seen, refining the u space employing pairwise similarities results in a more clear and separate group distributions. For example, the pink, purple, and magenta groups of USPS are now well distinguishable in the new refined u space. As another example, see samples from three groups shown in red, olive, and brown of the 2MNIST dataset. These groups are more separated in the refined u space, shown in the second row, compare to the representation shown in the first row of Fig 4.

The last row in Fig. 4 depicts the output space of MNet (q space), in which we make decisions about cluster assignments of data points. As is expected, clusters in this space have low within- and high between- cluster distances. As an example, consider the navy blue and the purple groups of the Fashion MNIST dataset, at the end of the second phase. Although these groups are mixed in the u space, they are completely isolated in the q space.

4.7 Effect of Pre-trained Network

In this section, we investigate the effect of the structure of the employed pre-trained network for extracting features from RGB images. To this end, we compare the clustering performance of all the deep-learning-based algorithms presented in Section 4.5 using four different ResNet architectures, namely ResNet-34 [56], ResNet-50 [56], ResNet 101 [56] and ResNet-152 [56]. All ResNets are pre-trained on the ImageNet dataset [56]. The corresponding ACC and NMI are reported in Table 3. The best result for each dataset is shown in boldface and the second top results are shown with *. As can be seen, regardless of the employed structure, the proposed DCSS method outperforms all other algorithms on all datasets. In addition, the DSL method is the second top method in 19 out of the 24 reported values.

4.8 Loss function convergence

Fig. 5. depicts the average, over different groups on different batches of data points, of the reconstruction, centering, and total losses corresponding to the first phase of DCSS (i.e. DSL) shown in (1). As can be seen, all losses are converged at the end of training. The noticeable reduction in the centering loss shows the effectiveness of our proposed approach in creating a reliable u space in which the data points are gathered around the group centers. Moreover, the figures show that at the first training epochs, our method trades the reconstruction loss for an improved centering performance. This proves insufficiency of the reconstruction loss in creating a reliable latent space.

In Fig. 6, we investigate convergence of the second phase's loss of our proposed DCSS method, shown in equations (4) and (5). Since we initialize the MNet randomly, at the first few epochs, MNet knows nothing about the lower-dimension representation of the data points in the q space; thus, we face a high loss value. As the q training process progresses, the loss value drops and converges to zero at the end of the training process. In the first 5 epochs, the algorithm minimizes the loss presented in (4) and minimizes (5) in the remainings. The continuity of the loss reduction over epochs along with the sharp loss drop at the 5th epoch, confirms the effectiveness of our proposed strategy in employing the reliable space u for supervision in the early epochs and then employing the skilled space q for supervision in the later epochs.

4.9 DCSS as a General Framework

In this section, we demonstrate the effectiveness of the DCSS as a general framework where the u space can be trained with other AE-based clustering techniques. To this end, we substitute the DSL technique proposed in Section 3.1 with other deep-learning-based techniques that train an effective subspace using an AE for the purpose of data clustering. Among our comparison methods, algorithms DEC, IDEC, DCN, and DKM are AE-based. For each dataset, we train AEs using these algorithms, then take their encoder part and append our proposed MNet to their latent space. Then we run the second phase of DCSS. Results of such implementation are reported in Table 2 where X+MNet indicates the performance of DCSS employing the X method's latent space as the DCSS's u space. Note that, for the RGB datasets, we construct features using the pre-trained ResNet-152. Comparing the clustering results reported in Tables 1, 3 and 2 confirms the effectiveness of DCSS as a general framework to improve the existing state-of-the-art AE-based clustering methods. On average, MNet improves clustering performance of DEC, IDEC, DCN, and DKM respectively by 3.58% (1.87%), 2.93% (1.54%), 4.04% (2.98%), and 2.56% (1.65%) in terms of ACC (NMI).

4.10 Performance on Imbalanced Dataset

To demonstrate effectiveness of our proposed DCSS method on imbalanced dataset, we randomly collect five subsets

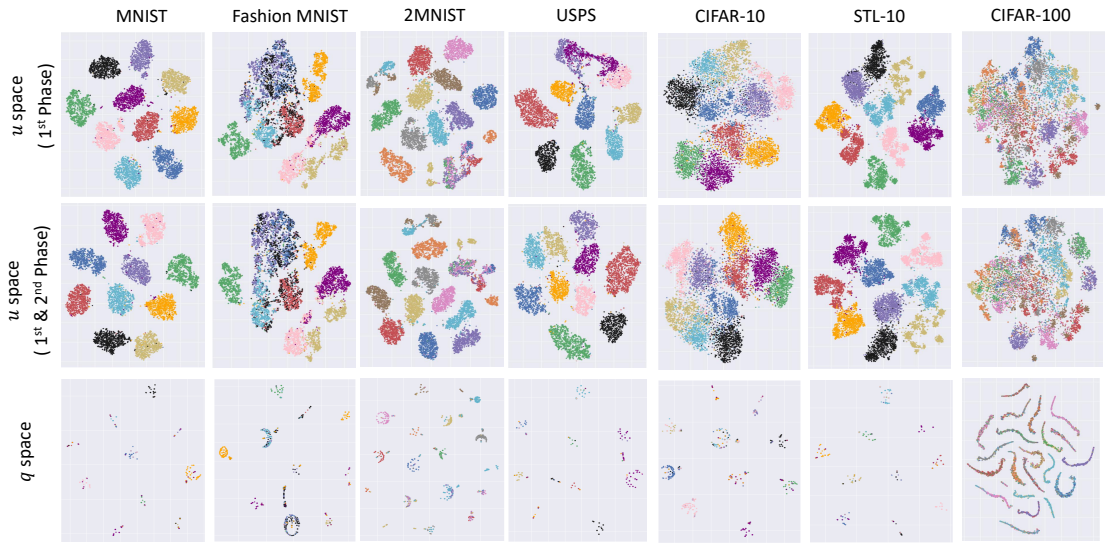


Fig. 4. The grouping and clustering visualization of different phases of DCSS using t-SNE, for different benchmark datasets. The first row shows the grouping result of the first phase of DCSS, where a sample sits close to its corresponding group center by minimizing weighted reconstruction and centering losses. The final u space of DCSS, obtained by completing the first and second phases of DCSS, is shown in the second row. In the second phase, DCSS aims to refining the u space (obtained in the first phase) and training the q space by employing pairwise similarity between data points. The last row depicts the final output of MNet (q space) for all data points. Axes range from -100 to 100.

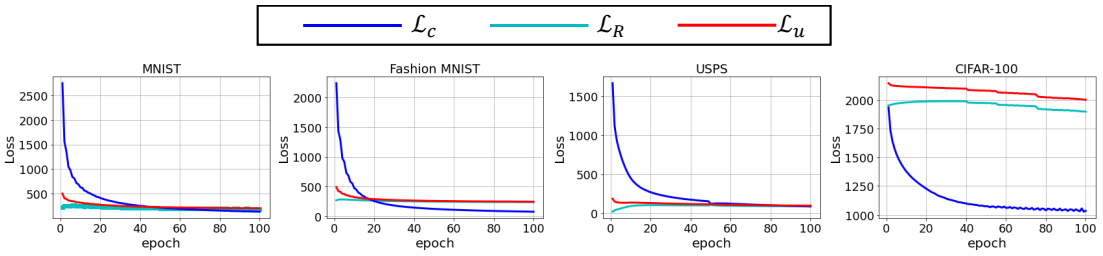


Fig. 5. The reconstruction loss $\mathcal{L}_r$, centering loss $\mathcal{L}_c$, and total loss $\mathcal{L}_u$ of the first phase of DCSS (DSL) vs. training epochs, for different datasets. At the earlier epochs, the latent representation of the data points are irregularly scattered around the group centers, which causes the high centering loss value. During the first step of DCSS, our proposed algorithm aims to simultaneously minimizing centering and reconstruction losses; this leads to an increase in the reconstruction loss (and decrease in the centering loss), in the later training epochs. When the first phase is complete, the latent representation of data points sit close to the group centers, which causes convergence of centering loss to a small value.

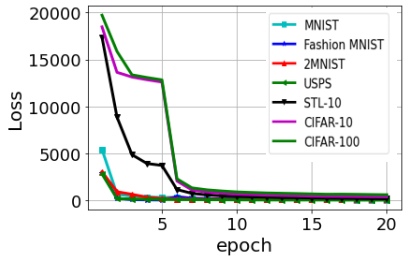


Fig. 6. The second phase's loss function of DCSS for different benchmark datasets. In the first T_2 epochs, when MNet is naïve, we make use of the u space to measure pairwise similarities; hence, the first $T_2 = 5$ epochs show the loss value defined in (4). After the first T_2 epochs all datasets show an eye-catching decrease in the loss values since we switch from using the u space to using the more reliable space q when measuring pairwise similarities by minimizing the loss defined in (5).

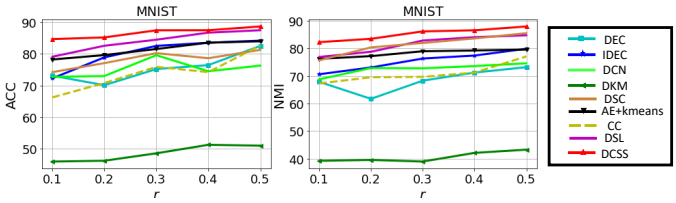


Fig. 7. The clustering performance of different methods on imbalanced samples of MNIST.

of the MNIST dataset with different retention rates $r \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$, where samples of the first class are chosen with probability of r and the last class with probability of 1, with the other classes linearly in between. Hence, on average, the number of samples for the first cluster is r times less than that of the last cluster. As is shown in Fig. 7, our proposed DCSS framework significantly outperforms our comparison methods for all r values. This indicates the robustness of DCSS on imbalanced data. As is expected, in general, for all methods, increasing r results in a higher performance because the dataset gets closer to a balanced one.

4.11 Visualization of representations in the q space

Fig. 9 shows the q vector corresponding to samples from different data clusters. As can be seen, the proposed method

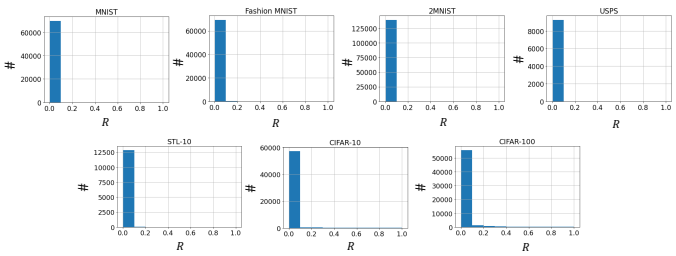


Fig. 8. Histogram plot of $R_i = ||I_i - q_i||_1$, $i = 1, \ldots, N$ where I_i is the one-hot crisp assignment corresponding to q_i.

usually results in q space representations close to an one-hot vector. Note that the kth element of q_i shows the probability of sample x_i being in the kth cluster. The closer q_i is to the one-hot vector, the more confident crisp assignment can be performed. As is proved in Corollary 1.2 of Appendix A, if a data point has at least one similar neighbor, the maximum element of its corresponding q is greater than ζ. In our experiments, ζ is set to 0.8. This can justify aggregation of the data points near one-hot representations in the q space.

To further demonstrate convergence of the q representations to one-hot vectors, histogram of residuals $R_i = ||I_i - q_i||_1$, $i = 1, \ldots, N$ for all datasets are shown in Fig. 8; where $||.||_1$ indicates the ℓ_1-norm and I_i is the one-hot crisp assignment corresponding to q_i – i.e. the index of the non-zero element of I_i is equal to the index of the maximum element of q_i. As it can be seen in Fig. 8, the q representation of almost all data points are very close to their corresponding one-hot vector.

4.12 Hyperparameters Sensitivity

In Fig. 10, we investigate the effect of different hyperparameters on DCSS clustering performance. For hyperparameters of the first phase (i.e. α, m and T_1), we report performance of clustering using DSL (as is presented in Section 4.5).

In Fig. 10(a), we explore importance of the centering loss in the first phase's loss function shown in (1) by changing $\alpha \in \{0, 0.01, 0.1, 1\}$ for the MNIST dataset. As is shown in this figure, by increasing the value of α from 0 to 0.01, our DCSS performance significantly enhances in terms of ACC and NMI, which demonstrates the effectiveness of incorporating the centering loss beside the reconstruction loss in the first phase's loss function. The best clustering performance is achieved for $\alpha = 0.1$ for the MNIST dataset.

Fig. 10(b) shows the impact of the level of fuzziness m on clustering performance of DSL, where $m \in \{1.1, 1.3, 1.5, 1.7\}$. In case m$\rightarrow 1$ (m$\rightarrow \infty$), group membership vectors converge to one-hot (equal probability) vectors. As it is shown in this figure, the best performance is obtained for m $= 1.5$ for the Fashion MNIST dataset.

In Fig. 10(c), we scrutinize the effect of update interval T_1 in clustering performance of the first step for $T_1 \in \{2, 5, 10, 15\}$. As is expected, better clustering performance in terms of ACC and NMI is acquired for smaller value of T_1, i.e. $T_1 = 2$.

In Fig. 10(d), we change the number of training epochs T_2, defined in Section 3.2, where $T_2 \in \{1, 5, 10, 20\}$. As is expected, for very small T_2 value, e.g. $T_2 = 1$, where training the q space is mainly supervised by the q space itself even at the MNet training outset, DCSS cannot provide a proper q space, since q is not a reliable subspace to be used for self-supervision. The figure also shows that for a very large T_2 value, e.g. $T_2 = 20$, when we only trust the u space for supervising the q space, we cannot train an effective q space. As it is shown, the best clustering performance is obtained when T_2 is set to a moderate value, e.g. $T_2 = 5$. This demonstrates the effectiveness of the proposed strategy in supervising the MNet training using both u and q spaces.

In Fig. 11, we change ζ and γ in range [0,1], where $\zeta + \gamma = 1$, to observe model convergence and accuracy for different lengths of the ambiguity interval, defined as $\zeta - \gamma$, ranging from 1 (when $\zeta = 1$) to 0 (when $\zeta = 0.5$). Fig. 11(a) shows the number of pairs participating in minimizing the loss functions defined in (4) and (5). As can be seen, at the beginning of the second phase, our model can make a decisive decision only about a few pairs, and the remaining pairs are in the ambiguous region. As the second phase training process progresses, more and more pairs are included in the loss functions optimization process. Finally, at the end of the second phase, almost all pairs contribute to the training.

Furthermore, in Fig. 11(b), we investigate the influence of ζ and γ in clustering performance. As it can be seen, as is desired, the final clustering performance of our DCSS framework is not highly sensitive to the choice of ζ and γ when are set to reasonable values. In all our experiments $\zeta = 0.8$ and $\gamma = 0.2$.

4.13 Features visualization

In order to investigate the effectiveness of our model in extracting useful features for different datasets, we train a deep neural network with the same structure as we proposed in Section 4.3 in a supervised manner, and then we compare the output of the first convolutional layers for the trained model and our proposed DCSS model. As it can be seen in Fig. 12, our DCSS learns a variety of low- and high-frequency features, which are similar to features learned in a supervised manner. This demonstrates the effectiveness of our framework in finding informative features in an unsupervised manner.

5 Conclusion

In this research study, we developed a novel and effective AE-based deep-clustering framework, i.e. deep clustering with self-supervision (DCSS), which considers pairwise similarity and dissimilarity between data points for data clustering task. DCSS is based on a two-phase training procedure. In the first phase, through K successive runs, DCSS tries to obtain an optimal lower-dimensional representation for data points, where at the kth run, DSL concentrates on reconstruction and grouping of those data points that are more probable to belong to the the kth data group. In the second phase, DCSS aims to finding the pairwise relationship between data points employing the lower-dimensional representations obtained in the first step. To this end, we train a fully connected network, i.e. MNet, in an unsupervised manner through minimizing a novel and effective loss function that only considers similar

Fig. 9. Visualization of q for samples from (a) MNIST, (b) Fashion MNIST, (c) STL-10, and (d) CIFAR-10 datasets. The q vector (i.e. the MNET output) for each image is depicted beside the image. The vertical axes range from 0 to 1.

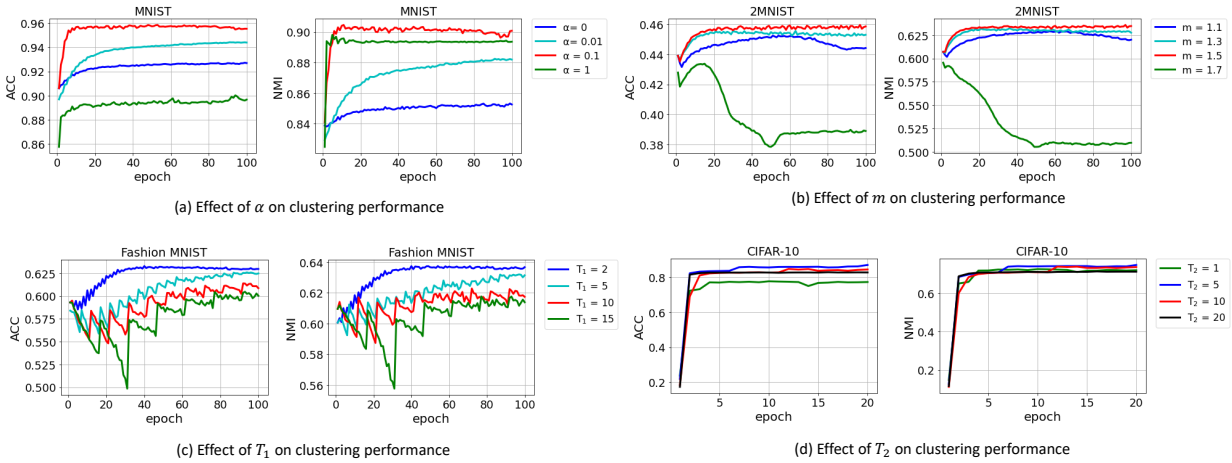


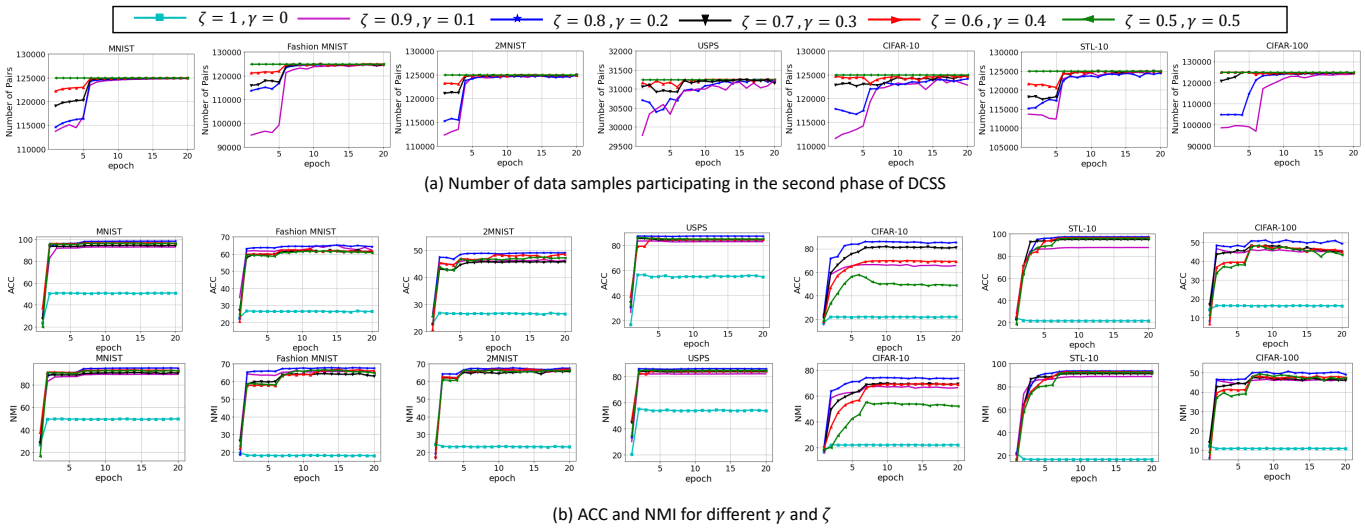
Fig. 10. Sensitivity of DCSS to different hyperparameters.

and dissimilar pairs identified in the AE's latent space or the output space of the MNet itself. We evaluate our DCSS framework on several benchmark datasets. Empirical results show that DCSS outperforms state-of-the-art data clustering methods. The effectiveness of the proposed method is also supported by severe mathematical proofs discussed in Appendix A. Moreover, the results shown in Section 4.9 demonstrate the effectiveness of the DCSS framework in improving the clustering performance of the state-of-the-art AE-based clustering algorithms.

APPENDIX A

Notation clarification: Representation of the ith sample in the q space is shown by q_i. The kth element of q_i is shown by $q_{ik}, k = 1, \ldots, K$, where K is the number of data clusters. Note that q_i is the MNet output when the input sample is x_i. Since we employed soft-max as the final layer of MNet, $0 \leq q_{il} \leq 1$ and the ℓ_1-norm of q_i is equal to 1. Furthermore, as is discussed before, parameters ζ and γ are values between 0 and 1.

Definition A.1. Two data points, i.e. i and j, are adjacent



(a) Number of data samples participating in the second phase of DCSS

(b) ACC and NMI for different γ and ζ

Fig. 11. Changing hyperparameters ζ and γ for different datasets. (a) Number of data pairs participating in the second phase of DCSS. In the first epochs, MNet can barely recognize relation between data points using soft group assignments defined in the u space. After completing the first $T_2 = 5$ epochs, we switch to measure similarities in the more reliable space q hence the number of participating pairs increases dramatically. In the final epochs for different values for ζ and γ, almost all pairs contribute in the loss function defined in (5). For the maximum ambiguity region 1, i.e. $\zeta = 1$ and $\gamma = 0$, all pairs do not participate in the second phase of DCSS and stay on the ambiguity region. Hence, 0 pairs contributes in this phase. (b) Clustering performance in terms of ACC and NMI for different datasets for different value of ζ and γ. The clustering performance of DCSS is less sensitive to the choice of ζ and γ in range of [0.5,0.9] and [0.1,0.5], respectively.

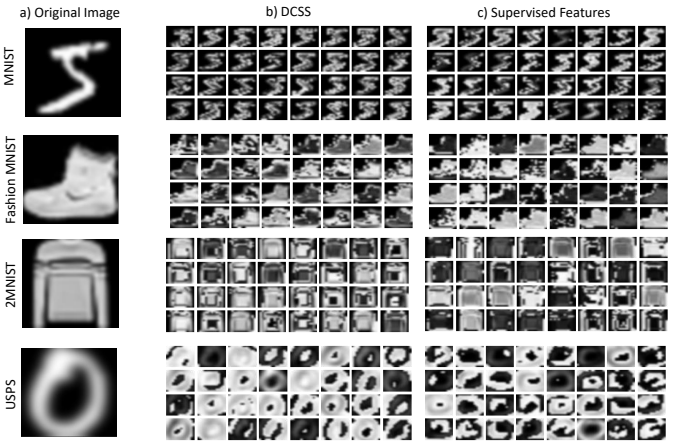


Fig. 12. (a) samples of MNIST, Fashion MNIST, 2MNIST, and USPS. The output of the first convolutional layer using (b) the unsupervised DCSS method, (c) a supervised manner employing the same network structure as of DCSS shown in Section 4.3.

(aka similar) if and only if $q_i^T q_j \geq \zeta$.

Definition A.2. Two data points, i.e. i and j, are in the same cluster if and only if the index of the maximum value in their corresponding q (i.e. q_i and q_j) are equal.

Theorem 1. *Consider the ith and jth data points. Then :*

$$q_i^T q_j \leq \min\{\max_l\{q_{il}\}, \max_l\{q_{jl}\}\} \tag{7}$$

where $q_i^T q_j$ is the inner product of the two vector q_i and q_j. q_{il} and q_{jl} denote the lth element of q_i and q_j , respectively.

Proof. Assume q_j^* is a maximal vector that satisfies the below inequality:

$$q_i^T q_j \leq q_i^T q_j^*, \tag{8}$$

where $||q_j^*||_1 = 1$. In addition, assume the index of the maximum element of q_i is r:

$$r = \arg\max_l\{q_{il}\} \tag{9}$$

In the following, we first prove by contradiction that q_j^* must be a one-hot vector. Then we prove (7).
Assume q_j^* is not a one-hot vector. Therefore, there exists at least one index, i.e. e, that its corresponding element q_{je}^* is non-zero:

$$\exists\ e : e \neq r \ \text{ and } \ q_{je}^* \neq 0. \tag{10}$$

Now, let's define a vector $\hat{q}_j$ as follows:

$$\hat{q}_{jl} = \begin{cases} q_{je}^* + q_{jr}^* & \text{if } l = r \\ 0 & \text{if } l = e \\ q_{jl}^* & O.W \end{cases}, \tag{11}$$

where $\hat{q}_{jl}$ denotes the l^{th} element of $\hat{q}_j$. Since $||q_j^*||_1 = 1$, we can immediately show that $||\hat{q}_j||_1 = 1$. Moreover, we can represent the inner products $q_i^T q_j^*$ and $q_i^T \hat{q}_j$ as shown in (12) and (13), respectively.

$$q_i^T q_j^* = q_{ir} q_{jr}^* + q_{ie} q_{je}^* + \sum_{l \neq r,e} q_{il} q_{jl}^* \tag{12}$$

$$q_i^T \hat{q}_j = q_{ir}(q_{jr}^* + q_{je}^*) + q_{ie} \times 0 + \sum_{l \neq r,e} q_{il} q_{jl}^* \tag{13}$$

Since q_{ir} is the maximum element of q_i, we can readily show that $q_i^T q_j^* \leq q_i^T \hat{q}_j$, which contradicts the assumption shown in equation (8); thus, q_j^* must be a one-hot vector. Therefore:

$$q_i^T q_j^* \leq \max_l\{q_{il}\} \tag{14}$$

Considering (14) and (8), we have:

$$q_i^T q_j \leq \max_l\{q_{il}\}. \tag{15}$$

Similarly, for the ith sample, we can show that $q_i^{*T} q_j \leq \max_l\{q_{jl}\}$, hence:

$$q_i^T q_j \leq \max_l\{q_{jl}\}. \tag{16}$$

(15) and (16) proves (7).

□

Corollary 1.1. *If two samples i and j are adjacent, the maximum value among their elements is greater than ζ.*

Proof. This statement is a corollary of Theorem 1, that can be proved from (7), (15), (16) and the adjacency definition provided in Definition A.1. In other words:

$$\zeta \leq q_i^T q_j \leq min(q_{ir}, q_{jo}) \quad \rightarrow \quad \begin{cases} \zeta \leq q_{ir} \\ \zeta \leq q_{jo} \end{cases} \tag{17}$$

where the index of the maximum element of q_i and q_j are respectively shown by r and o – i.e., $r = \arg\max_l\{q_{il}\}$ and $o = \arg\max_l\{q_{jl}\}$. □

Corollary 1.2. *If a data point has at least one adjacent neighbor, the maximum element of its corresponding q is greater than ζ.*

Proof. We first empirically test the validity of the employed assumption, i.e. existence of at least one adjacent (i.e. similar) sample for a data point, on our datasets. Fig. 13 shows the number of data samples that are not similar to any other data points in the q space. As it can be seen, at the beginning of the second phase of DCSS, since MNet is initialized randomly, many data points do not have any adjacent neighbor (i.e. almost $\forall\ i, j,\ i \neq j:\ q_i^T q_j < \zeta$.). By minimizing (4) and (5) in the second phase of DCSS, similar samples are tightly packed in in the q space; therefore, almost all samples have at least one adjacent neighbor. For example, there are only 330 data points, out of the 60,000 samples of the CIFAR-100 dataset, that are not similar to any other data samples at the end of the second phase. Note that CIFAR-100 presents the worst case among the other datasets shown in Fig 13. All in all, we can roughly assume that each data point has at least one adjacent sample.

Lets consider an arbitrary data point i, and one of its adjacent data points j. From Corollary 1.1, we can conclude that:

$$\begin{cases} \zeta \leq \max_l\{q_{il}\} \\ \zeta \leq \max_l\{q_{jl}\} \end{cases}. \tag{18}$$

Thus, we proved that the maximum element of q_i, where i is an arbitrary data point, is greater than ζ. □

Corollary 1.3. *Assume each data point has at least one adjacent neighbor and $\gamma < \zeta^2$. If two data points i and k are dissimilar, i and k are not from the same cluster.*

Proof. We prove this corollary by contradiction where the contradiction assumption is: i and k are dissimilar, yet from the same cluster where $\gamma < \zeta^2$.
Since i and k are in the same cluster, the index of the maximum element of q_i and q_k are the same. (i.e. $\beta = \arg\max_l\{q_{il}\} = \arg\max_l\{q_{kl}\}$). Since each data point has at least one adjacent neighbor, from Corollary 1.2, we can conclude that:

$$\begin{cases} \zeta \leq q_{i\beta} \\ \zeta \leq q_{k\beta} \end{cases}. \tag{19}$$

and we can represent $q_i^T q_k$ as follow:

$$q_i^T q_k = q_{i\beta} q_{k\beta} + \sum_{l \neq \beta} q_{il} q_{kl} \tag{20}$$

Therefore, $q_{i\beta} q_{k\beta} \leq q_i^T q_k$. Also, we know i and k are dissimilar. From (19), we can conclude that:

$$\zeta^2 \leq q_{i\beta} q_{k\beta} \leq q_i^T q_k \leq \gamma \tag{21}$$

(21) contradicts the assumption of $\gamma < \zeta^2$. Hence, i and k are in different clusters. □

Theorem 2. *For $\frac{2}{3} \leq \zeta$, if i and j are adjacent, they are in the same cluster – i.e. r is equal to o where $r = \arg\max_l\{q_{il}\}$ and $o = \arg\max_l\{q_{jl}\}$.*

Proof. First we find an upper bound for $q_i^T q_j$, when the ith and jth samples are adjacent, but from different clusters.
Since i and j are not from the same cluster, we can represent $q_i^T q_j$ as follows:

$$\begin{aligned} q_i^T q_j &= q_{ir} q_{jr} + q_{io} q_{jo} + \sum_{l \neq o,r} q_{il} q_{jl} \\ &\xrightarrow{\forall l:\ q_{jl} \leq q_{jo}} \leq q_{ir} q_{jr} + q_{io} q_{jo} + \sum_{l \neq o,r} q_{il} q_{jo} \\ &= q_{ir} q_{jr} + q_{jo}\Big(\sum_{l \neq r} q_{il}\Big) \\ &\xrightarrow{\sum_{l \neq r} q_{il} = 1 - q_{ir}} = q_{ir} q_{jr} + q_{jo}(1 - q_{ir}) \\ &\xrightarrow{q_{jr} \leq 1 - q_{jo}} \leq q_{ir}(1 - q_{jo}) + q_{jo}(1 - q_{ir}) \\ &\xrightarrow{\text{from (17)}} \leq q_{ir}(1 - \zeta) + q_{jo}(1 - \zeta) \\ &\xrightarrow{\{q_{ir},\ q_{jo}\} \in [0,1]} \leq 2(1 - \zeta). \end{aligned} \tag{22}$$

Note that $q_{jr} \leq 1 - q_{jo}$ because $\sum_{l \neq o} q_{jl} + q_{jr} + q_{jo} = 1$. Note that all elements of a q vector are probability values between 0 and 1.
Hence, as is shown (22), if two samples are not from the same cluster then the inner product of their corresponding q has an upper bound of $2(1 - \zeta)$. Therefore, if two samples i and j are adjacent (see Definition 1) but from different clusters, then:

$$\begin{aligned} &\zeta \leq q_i^T q_j \leq 2(1 - \zeta) \\ &\rightarrow \zeta \leq 2(1 - \zeta) \rightarrow \zeta \leq \frac{2}{3}. \end{aligned} \tag{23}$$

Thus, for $\frac{2}{3} < \zeta$, i and j cannot be from two different clusters. In other words, if two samples i and j are adjacent AND the user-settable parameter ζ is set to a value greater than $\frac{2}{3}$, then the two samples are from similar clusters, i.e. $r = o$. In this paper we set $\zeta = 0.8 > \frac{2}{3}$. □

Corollary 2.1. *Assume $\zeta > \frac{2}{3}$. Consider three data points i, j, and k. If i and j also i and k are adjacent, then j and k are from the same cluster.*

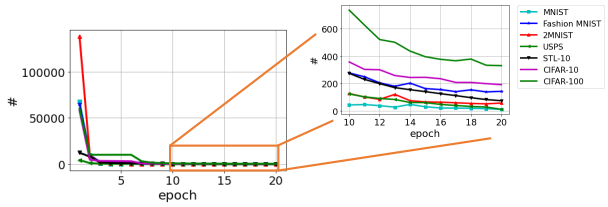


Fig. 13. Number of data points that do not have any adjacent neighbors during DCSS training in the second phase.

Proof. Since i and j (i and k) are adjacent and $\zeta > \frac{2}{3}$, from Theorem 2, we can conclude that i and j (i and k) are in the same cluster; hence, the three samples i, j, and k all are in the same cluster. □

Theorem 3. *Consider three data points i, j, and k where i and j also i and k are adjacent (aka similar). Assume $\zeta > \frac{2}{3}$. If $\gamma < \zeta^2$, then the two samples j and k are not dissimilar (i.e. $q_j^T q_k \not< \gamma$).*

Proof. Considering Theorem 2, i and j and k are in the same cluster. Therefore, we do not want to include the pair of j and k samples as a dissimilar pair when minimizing the loss function defined in equation (5).
Since j and k are in the same cluster and knowing that the index of the maximum element in a q vector shows cluster of the corresponding sample, we have:

$$\eta = \arg\max_l \{q_{jl}\} = \arg\max_l \{q_{kl}\}. \tag{24}$$

Thus,

$$q_j^T q_k = \sum_{l \neq \eta} q_{jl} q_{kl} + q_{j\eta} q_{k\eta}. \tag{25}$$

Therefore,

$$q_j^T q_k \geq q_{j\eta} q_{k\eta}. \tag{26}$$

Since $\zeta \leq q_i^T q_j$ and $\zeta \leq q_i^T q_k$, we can infer (27) from (17).

$$\begin{cases} \zeta \leq q_{j\eta} \\ \zeta \leq q_{k\eta} \end{cases}. \tag{27}$$

From (26) and (27), we can obtain below:

$$\zeta^2 \leq q_{j\eta} q_{k\eta} \leq q_j^T q_k. \tag{28}$$

Thus, if we choose $\gamma < \zeta^2$, we will not include the pair of samples j and k as a dissimilar pair in equation (5). In this paper γ is set to 0.2, i.e. $\gamma = 0.2 < 0.8^2$. □

REFERENCES

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

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

[9] 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.

[10] 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.

[11] 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.

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

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

[14] 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.

[15] 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.

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

[17] 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.

[18] 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.

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

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

[21] 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.

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

[23] Christos Boutsidis, Petros Drineas, and Michael W Mahoney. "Unsupervised feature selection for the k-means clustering problem". In: *Advances in Neural Information Processing Systems*. 2009, pp. 153–161.

[24] Huan Liu and Lei Yu. "Toward integrating feature selection algorithms for classification and clustering". In: *IEEE Transactions on knowledge and data engineering* 17.4 (2005), pp. 491–502.

[25] Salem Alelyani, Jiliang Tang, and Huan Liu. "Feature selection for clustering: A review". In: *Data Clustering* (2018), pp. 29–60.

[26] Eric P Xing et al. "Distance metric learning with application to clustering with side-information". In: *NIPS*. Vol. 15. 505–512. Citeseer. 2002, p. 12.

[27] Shiming Xiang, Feiping Nie, and Changshui Zhang. "Learning a Mahalanobis distance metric for data clustering and classification". In: *Pattern recognition* 41.12 (2008), pp. 3600–3612.

[28] James MacQueen et al. "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. Oakland, CA, USA. 1967, pp. 281–297.

[29] Ulrike Von Luxburg. "A tutorial on spectral clustering". In: *Statistics and computing* 17.4 (2007), pp. 395–416.

[30] Tao Li, Sheng Ma, and Mitsunori Ogihara. "Entropy-based criterion in categorical clustering". In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, p. 68.

[31] 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.

[32] 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).

[33] 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.

[34] Jieping Ye, Zheng Zhao, and Mingrui Wu. "Discriminative k-means for clustering". In: *Advances in neural information processing systems* 20 (2007), pp. 1649–1656.

[35] Andrew Y Ng, Michael I Jordan, and Yair Weiss. "On spectral clustering: Analysis and an algorithm". In: *Advances in neural information processing systems*. 2002, pp. 849–856.

[36] Santo Fortunato. "Community detection in graphs". In: *Physics reports* 486.3-5 (2010), pp. 75–174.

[37] Y. Han and M. Filippone. "Mini-batch spectral clustering". In: *2017 International Joint Conference on Neural Networks (IJCNN)*. 2017, pp. 3888–3895.

[38] M. El Gheche, G. Chierchia, and P. Frossard. "Stochastic Gradient Descent for Spectral Embedding with Implicit Orthogonality Constraint". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 3567–3571.

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

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

[41] Pascal Vincent et al. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." In: *Journal of machine learning research* 11.12 (2010).

[42] Geoffrey E Hinton and Ruslan R Salakhutdinov. "Reducing the dimensionality of data with neural networks". In: *science* 313.5786 (2006), pp. 504–507.

[43] Laurens Van Der Maaten. "Accelerating t-SNE using tree-based algorithms". In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 3221–3245.

[44] Kamal Nigam and Rayid Ghani. "Analyzing the effectiveness and applicability of co-training". In: *Proceedings of the ninth international conference on Information and knowledge management*. 2000, pp. 86–93.

[45] Xifeng Guo et al. "Deep clustering with convolutional autoencoders". In: *International conference on neural information processing*. Springer. 2017, pp. 373–382.

[46] 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.

[47] 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.

[48] Yunfan Li et al. "Contrastive Clustering". In: *arXiv preprint arXiv:2009.09687* (2020).

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

[50] Mohammadreza Sadeghi and Narges Armanfard. "Deep Successive Subspace Learning for Data Clustering". In: *The International Joint Conference on Neural Networks (IJCNN)*. 2021, accepted to be published.

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

[52] 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).

[53] 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.

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

[55] 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.

[56] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[57] 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.

[58] 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.

[59] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. IEEE. 2009, pp. 248–255.

[60] Diederik P Kingma and Jimmy Ba. “Adam: a method for stochastic optimization”. In: *arXiv preprint arXiv: 1412.6980* (2014).

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

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

[63] Laurens Van der Maaten and Geoffrey Hinton. “Visualizing data using t-SNE.” In: *Journal of machine learning research* 9.11 (2008).

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 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, in 2016. She completed her postdoctoral studies at the University of Toronto and University Health Network in 2018. 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.