Deep Clustering with Associative Memories

Bishwajit Saha
Department of CS
RPI
Troy, NY, USA
sahab@rpi.edu

Dmitry Krotov
MIT-IBM Watson AI Lab
IBM Research
Cambridge, MA, USA
krotov@ibm.com

Mohammed J. Zaki
Department of CS
RPI
Troy, NY, USA
zaki@cs.rpi.edu

Parikshit Ram
IBM Research
Yorktown Heights, NY, USA
Parikshit.Ram@ibm.com

Abstract

Deep clustering – joint representation learning and latent space clustering – is a well studied problem especially in computer vision and text processing under the deep learning framework. While the representation learning is generally differentiable, clustering is an inherently discrete optimization, requiring various approximations and regularizations to fit in a standard differentiable pipeline. This leads to a somewhat disjointed representation learning and clustering. Recently, Associative Memories were utilized in the end-to-end differentiable **`ClAM`** clustering scheme (Saha et al., 2023). In this work, we show how Associative Memories enable a novel take on deep clustering, **`DClAM`**, simplifying the whole pipeline and tying together the representation learning and clustering more intricately. Our experiments showcase the advantage of **`DClAM`**, producing improved clustering quality regardless of the architecture choice (convolutional, residual or fully-connected) or data modality (images or text).

1 Introduction

Clustering is a common unsupervised task to find hidden structure in unlabeled data. At a technical level, it critically relies on a notion of (pairwise) distance or similarity to distinguish pairs of data samples as being "similar" or "different" (Xu & Wunsch, 2005; Saxena et al., 2017; Xu & Tian, 2015). Diverse formulations and methods have been explored to find effective data clustering over time, including well-known methods like k-means (MacQueen, 1967), fuzzy c-means (Bezdek et al., 1984), Hierarchical Clustering (Johnson, 1967), Expectation Maximization (Dempster et al., 1977) and Spectral Clustering (Donath & Hoffman, 1973). When dealing with numerical data $S \subset \mathbb{R}^d$ with d dimensions, metrics such as Euclidean distance are commonly used. The insights from clustering can be unintuitive or misleading without a meaningful distance. Nevertheless, even with numerical data and an appropriate (meaningful) notion of distance, increasing data dimensionality (that is, increasing d) makes clustering computationally hard as well as conceptually difficult since the separation between similar pairs and dissimilar ones can start to vanish (Verleysen & François, 2005; Steinbach et al., 2004; Assent, 2012).

In various domains, both these problems manifest – first, the raw representation of samples can be extremely high dimensional (consider the number of pixels in an image, or the number of words in a vocabulary for a bag-of-words representation of documents); second, while we have an *ambient* representation, standard notions of vector distances (such as Euclidean one) do not necessarily make sense – for example, Euclidean distance based on pixels can be large between an image and a slightly shifted version of it, which can be problematic if the content of the images are translation or rotation invariant. **Deep clustering** (Min et al., 2018; Ren et al., 2024; Zhou et al., 2022) tries to address both

Machine Learning and Compression Workshop @ 38th Conference on Neural Information Processing Systems

these issues simultaneously, by both learning a low dimensional *latent* space, and ensuring standard distance metrics are meaningful in that space.

For the latent representations to be faithful to the original samples, deep clustering ensures that there is no significant information loss in the latent space, leading to the common use of autoencoders (Rumelhart et al., 1985; Baldi, 2012; Bank et al., 2023) that learn latent representations (via an encoder) which can be used to reconstruct the original samples (via a decoder). The goal of deep clustering is to discover a cluster structure in the latent space while ensuring low reconstruction loss. This is a widely studied problem, especially in image datasets, which is amenable to end-to-end differentiability (Caron et al., 2018; Chang et al., 2017).

While the autoencoder is usually differentiable, standard clustering schemes (such as k-means or agglomerative ones) are inherently discrete methods since *hard* clustering (where each sample is only assigned to a single cluster) is a discrete optimization problem. To incorporate it in a differentiable deep learning pipeline, clustering is often "softened" by allowing samples to be partially assigned to multiple clusters, although various "regularizations" push the soft assignments to match hard assignments approximately (Xie et al., 2016; Guo et al., 2017a). Recent work (Saha et al., 2023) handles this dichotomy between hard assignments and differentiability with **Associative Memories** (Hopfield, 1982), a neuro-inspired recurrent network, proposing the **`ClAM`** clustering scheme which outperforms both discrete clustering baselines and differentiable soft clustering ones. See detailed related works in section A in Apendix.

In this paper, we explore the use of associative memories for deep clustering and make the following contributions, demonstrating *how associative memories critically enable a more elegant form of deep clustering*:

- We propose **`DClAM`**, an extension of **`ClAM`**, that learns representations and clusters in a latent space.
- We demonstrate how associative memories enable a simplified deep clustering **`DClAM`** that improves the learning while being closely related to standard deep clustering.
- We conduct a thorough evaluation on image and text data and multiple encoder architectures, demonstrating that **`DClAM`** significantly improves clustering quality over existing baselines, with the improvements being agnostic to the encoder/decoder architecture choice.

2 Preliminaries

We denote an input set as $S \subset \mathbb{R}^d$ in the ambient space, with an input $x \in S$, and $[\![n]\!]$ a n-length index set $\{1, \ldots, n\}$.

2.1 Deep Clustering Basics

Deep clustering is an unsupervised task, where we have to learn (usually lower dimensional) representations such that (i) no (critical) information is lost in the latent lower dimensional representations, and (ii) the data in the latent space forms well-separated clusters. To ensure that no information is lost in the latent space, we learn an encoder $\mathbf{e} : \mathbb{R}^d \to \mathbb{R}^m$ ($m < d$) that maps the input $x \in \mathbb{R}^d$ to a latent space (that is, $\mathbf{e}(x) \in \mathbb{R}^m$), along with a decoder $\mathbf{d} : \mathbb{R}^m \to \mathbb{R}^d$ that maps the latent representation back to the original ambient space. Encoder $\mathbf{e}$ and decoder $\mathbf{d}$ together give us an autoencoder, and the loss of information is often measured as the *reconstruction loss*:

$$\mathcal{L}_r(\mathbf{e}, \mathbf{d}) \triangleq \sum_{x \in S} \ell_r(x, \mathbf{e}, \mathbf{d}) = \sum_{x \in S} \|x - \mathbf{d}(\mathbf{e}(x))\|^2. \tag{1}$$

This loss term does not account for the cluster structure in the latent space. For that purpose, we consider k cluster centers $\boldsymbol{\rho} = \{\rho_1, \ldots, \rho_k\} \subset \mathbb{R}^m$ in the latent space, so that the corresponding *clustering loss* is given by:

$$\mathcal{L}_c(\mathbf{e}, \boldsymbol{\rho}) \triangleq \sum_{x \in S} \ell_c(x, \mathbf{e}, \boldsymbol{\rho}) = \sum_{x \in S} \min_{i \in [\![k]\!]} \|\mathbf{e}(x) - \rho_i\|^2, \tag{2}$$

which measures how close a sample is to its closest cluster center in the latent space with a $\min_{i \in [\![k]\!]}$ on a per-sample basis to denote the discrete assignment. A small value of $\mathcal{L}_c(\mathbf{e}, \boldsymbol{\rho})$ implies that all points in the latent space are close to their respective cluster centers.

Unsupervised deep clustering is often considered in the following form (Guo et al., 2017a,b; Cai et al., 2022)

$$\min_{\mathbf{e}, \mathbf{d}, \boldsymbol{\rho}} \mathcal{L}_r(\mathbf{e}, \mathbf{d}) + \gamma \mathcal{L}_c(\mathbf{e}, \boldsymbol{\rho}) \tag{3}$$

where $\gamma \geq 0$ is a hyperparameter that balances the clustering loss $\mathcal{L}_c$ and the reconstruction loss $\mathcal{L}_r$. Nevertheless γ can be difficult to select since the terms $\mathcal{L}_c$ and $\mathcal{L}_r$ are not inherently comparable, with $\mathcal{L}_c$ being computed between entities in the latent space $\mathbb{R}^m$, and $\mathcal{L}_r$ computed between items in the ambient space $\mathbb{R}^d$.

To handle this challenge (though rarely introduced in this manner to the best of our knowledge), usual implementations of deep clustering (Guo et al., 2017a,b; Golzari Oskouei et al., 2023) do the following: (i) First, an autoencoder (that is, $\mathbf{e}$ and $\mathbf{d}$) is "pretrained" with the data to achieve low reconstruction error (that is, low $\mathcal{L}_r$ by setting $\gamma = 0$ in Eq. (3)), and (ii) second, the γ is set to a positive value in Eq. (3), and the clustering loss $\mathcal{L}_c$ is minimized by learning the cluster centers $\boldsymbol{\rho}$, and "fine-tuning" the encoder $\mathbf{e}$, while the reconstruction loss $\mathcal{L}_r$ stays low by changing the decoder $\mathbf{d}$ accordingly *if the balancing hyperparameter γ is set appropriately*.

Evaluation of deep clustering. A common metric to evaluate and benchmark deep clustering algorithms is by computing the overlap between the obtained clusters in the latent space (thus, partitions) and a semantic partitioning of the data with metrics such as the Normalized Mutual Information or NMI. While this is a fair metric to compare methods on, *it is critical to ensure that NMI (or similar label-dependent metrics) is not utilized for hyperparameter selection* since that is leaking supervision into the unsupervised task of deep clustering, making the overall process a supervised learning pipeline. To the best of our knowledge, it is not clear how hyperparameters are typically selected. Even for the purposes of just evaluation, NMI like metrics might only tell us how the learned clusters in the latent space match some semantic partitioning (often manual) of the data, it does not provide any information regarding the reconstruction quality (and thus the information loss in the latent space). Thus, it is easily possible to have high NMI with poor reconstruction loss, which may not align with the primary goals of deep clustering. If we employ autoencoder pretraining, then we could optimize for the clustering quality with some unsupervised metric (such as SC) while ensuring that the reconstruction loss is within some margin (say 10%) of the reconstruction loss of the pretained autoencoder. We believe that the hyperparameters should be selected based on unsupervised metrics – metrics that do not utilize any ground-truth label information to evaluate clustering quality – given the unsupervised nature of the deep clustering problem. Thus, we consider the above strategy of optimizing for SC while keeping the reconstruction loss below some user-defined threshold. Existing literature typically report NMI without explicitly discussing reconstruction loss.

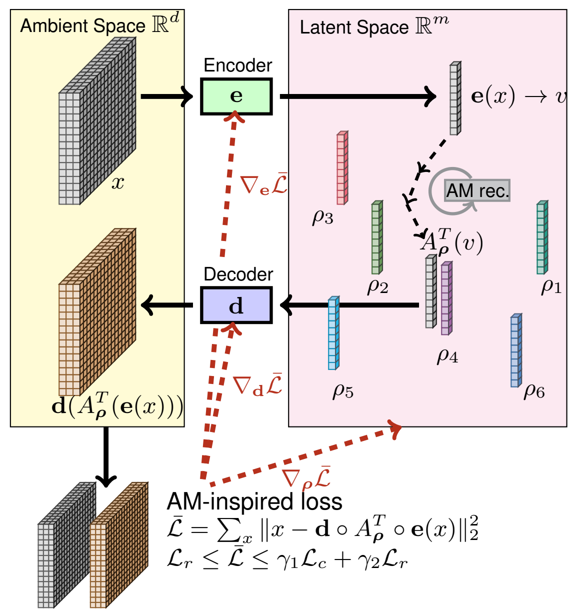


Figure 1: **DClAM**: AM-enabled simplified deep clustering. The solid arrows $\longrightarrow$ denote the forward-pass to compute the single loss term in Eq. (6). The dashed arrows $- \to$ denote the backward pass showing the single loss driving all updates.

2.2 Dense Associative Memories and Clustering

Given k memories $\{\boldsymbol{\rho}_1, \ldots, \boldsymbol{\rho}_k\}$, $\boldsymbol{\rho}_i \in \mathbb{R}^d$, and a point or particle $v \in \mathbb{R}^d$, **ClAM** (Saha et al, 2023) defines the energy function for v as follows:

$$E(v) = -\frac{1}{\beta} \log \left(\sum\nolimits_{i \in [\![k]\!]} \exp(-\beta \|\boldsymbol{\rho}_i - v\|^2) \right) \tag{4}$$

with the scalar $\beta > 0$ playing the role of inverse "temperature". As β increases, the $\exp(\cdot)$ function emphasizes the leading term, suppressing the others. In **ClAM**, the attractor dynamics are driven by gradient descent on the energy landscape. This controls the movement of v over time through dv/dt, ensuring a decrease in energy:

$$\tau \frac{dv}{dt} = -\frac{1}{2} \nabla_v E = \sum_{i \in [\![k]\!]} (\boldsymbol{\rho}_i - v) \, \mathsf{softmax}(-\beta \|\boldsymbol{\rho}_i - v\|^2) \tag{5}$$

Here, $\tau > 0$ is a characteristic time constant that determines how quickly the particle will move on the energy landscape. The function $\mathsf{softmax}(\cdot)$ represents the softmax function applied to the scaled distances to the memories. We use the notation $A_{\boldsymbol{\rho}}^T(v)$ to denote $A_{\boldsymbol{\rho}}(A_{\boldsymbol{\rho}}(\cdots A_{\boldsymbol{\rho}}(v)))$, where the operator $A_{\boldsymbol{\rho}}$ is applied to v recursively for T steps. Thus, $v^{t+1} = A_{\boldsymbol{\rho}}(v^t) = v^t + \tau \frac{dv}{dt}|_{v=v^t}$, via gradient descent on the energy. The attractor dynamics ensure that every memory $\rho_i, i \in [\![k]\!]$, forms a "basin of attraction", and with enough recursions T, any particle will usually converge to exactly one of these memories ρ_i, which thus act as cluster centers. The differentiability of the recursive dynamics is what makes **`ClAM`** an end-to-end differentiable clustering scheme, with the memories learned via standard backpropagation.

3 Deep Clustering with Associative Memories

One key limitation of **`ClAM`** is that is works only in the ambient space, since it lacks representation learning. In this work, we propose novel approach to deep clustering that leverages the attractor dynamics and combines it with latent space learning.

3.1 `DClAM`: AM enabled Deep Clustering

Existing deep clustering needs to solve Eq. (3) explicitly, which involves the critical γ hyperparameter to appropriately balance the clustering and reconstruction losses. Here, we will show how AM enables the removal of the critical γ hyperparameter in the deep clustering objective (Eq. (3)), while still maintaining the intent of Eq. (3) to balance the clustering loss and the reconstruction loss.

Consider the pipeline depicted in Fig. 1: The input x is mapped into the latent space as $\mathbf{e}(x)$ by the encoder $\mathbf{e}$, and then the attractor dynamics operator $A_{\boldsymbol{\rho}} : \mathbb{R}^m \to \mathbb{R}^m$ based on the current centers $\boldsymbol{\rho} = \{\rho_1, \ldots, \rho_k\}$ is applied to $\mathbf{e}(x)$ for T recursions, resulting in $A_{\boldsymbol{\rho}}^T(\mathbf{e}(x)) \approx \rho_4$. Then this representation (effectively of a cluster center) is passed through the decoder $\mathbf{d}$ to get $\mathbf{d}(A_{\boldsymbol{\rho}}^T(\mathbf{e}(x))) \in \mathbb{R}^d$ in the ambient space. We can then optimize for the following loss:

$$\min_{\mathbf{e},\mathbf{d},\boldsymbol{\rho}} \bar{\mathcal{L}}(\mathbf{e},\mathbf{d},\boldsymbol{\rho}) \triangleq \sum_{x \in S} \underbrace{\left\| x - \mathbf{d}\left(A_{\boldsymbol{\rho}}^T\left(\mathbf{e}\left(x\right)\right)\right) \right\|^2}_{\bar{\ell}(x,\mathbf{e},\mathbf{d},\boldsymbol{\rho})}. \tag{6}$$

Here AM becomes the intricate part of the encoder that transforms the embedding space (obtained by the encoder) into a clustering-friendly new space to find clusters (as opposed to the existing deep clustering schemes that use different additional loss functions e.g. clustering loss in Eq. (3) and/or regularizations to get a similar effect). This AM enabled *novel deep clustering loss* $\bar{\mathcal{L}}$ is a single term involving all parameters in the deep learning pipeline – the encoder $\mathbf{e}$, the cluster centers $\boldsymbol{\rho}$ and the decoder $\mathbf{d}$.

Our **`DClAM`** deep clustering provides various advantages – (i) First, it does not involve any balancing hyperparameter γ since the loss involves all parameters in a single term in the per-sample $\bar{\ell}(x, \mathbf{e}, \mathbf{d}, \boldsymbol{\rho})$. (ii) Second, the updates for all the parameters in the pipeline are more explicitly tied together with the $\mathbf{d} \circ A_{\boldsymbol{\rho}}^T \circ \mathbf{e}$ composition in the $\mathbf{d}(A_{\boldsymbol{\rho}}^T(\mathbf{e}(x)))$ term. *This ties the representation learning and clustering objectives more intricately.* (iii) Third, it continues to have all the advantages of traditional deep clustering, being end-to-end differentiable since all operators in the above composition are differentiable, and performing a discrete cluster center assignment with T recursions of the attractor dynamics operator $A_{\boldsymbol{\rho}}$. (iv) Forth, this deep clustering is completely architecture agnostic – we can select a problem dependent encoder and decoder (for example, convolutional or residual networks for images or fully-connected feed-forward networks for text or tabular data). Furthermore, this setup can easily handle already trained encoders (for example, one trained via contrastive learning (Chen et al., 2020; Van Gansbeke et al., 2020). (v) Fifth, it does not involve any additional entropy regularization based hyperparameters as with existing deep clustering algorithms. (vi) Finally, on a less technical level, Fig. 1 clearly highlights how the overall information flow in the deep clustering pipeline is simplified. *The AM plays a critical role in this pipeline with the ability to obtain the actual closest center $A_{\boldsymbol{\rho}}^T(\mathbf{e}(x))$; without it, this new pipeline and loss cannot be utilized.*

Although **`DClAM`** can be viewed as an extension (namely) of **`ClAM`**, there are fundamental difference between how **`ClAM`** uses AMs and how **`DClAM`** utilizes them. In **`ClAM`** AMs are utilized to act as differentiable arg min solver for the k-means objective whereas in **`DClAM`**, which involves representation learning, AM recursion actually has a more elaborate effect. The AM augmented encoder ($A_{\boldsymbol{\rho}}^T \circ \mathbf{e}$) explicitly creates basins of attraction in the latent space, and moves/pushes the latent representations of the points into these basins, thereby explicitly inducing a clustered data distribution in the latent space. While the encoder is moving points into basins of attraction, the **DClAM** loss tries to minimize the information loss in the latent representations by having the decoder reconstruct these relocated latent representations.

Upon solving Eq. (6), we will obtain a trained encoder and decoder, and memories in the latent space, and we can utilize them to obtain the final partition the data (see the **Infer** subroutine in Alg. 1 in Appendix). See Appendix B.5 for an understanding how **DClAM** loss typically relates to existing deep clustering loss.

4 Empirical Evaluation

We evaluate the performance of **DClAM** on a diverse set of 8 datasets (6 images and 2 text sets), ranging in size from 296 to 49152 (raw) features and containing 2007 to 60000 samples. The selection of the number of clusters for each dataset is based on its intrinsic class count, with no reliance on class information during clustering or hyperparameter selection (see dataset details in Appendix B.1). We conduct a comparative analysis of **DClAM** against established clustering methods, including k-means (Lloyd, 1982), agglomerative clustering (or Agglo.) (Müllner, 2011), **ClAM** (Saha et al., 2023), DCEC (Guo et al., 2017b), DEKM (Guo et al., 2021) and EDCWRN (or EDC) Golzari Oskouei et al. (2023). We evaluate k-means, agglomerative clustering, and **ClAM** in the ambient space (denoted as NAE) and in the latent space obtained through a pretrained Convolutional Autoencoder (CAE) as used in DCEC (Guo et al., 2017b). For DCEC amd DEKM, we consider a ResNet-based AE (RAE) (Wickramasinghe et al., 2021) along with their original CAE. For **DClAM**, we extend our exploration to include not only the CAE and RAE architectures but also EDCWRN-based (Golzari Oskouei et al., 2023) Autoencoder (EAE) (originally proposed by Guo et al. (2017a)) to analyze its impact on the algorithm. We also compare **DClAM** with state-of-the-art SimCLR (Chen et al., 2020) based (contrastive learning) SCAN (Van Gansbeke et al., 2020) and NNM (Dang et al., 2021) deep clustering schemes. Detailed parameter setting of the networks are in Appendix B.3, while implementation details are in Appendix B.4.

Table 1: Per-method best SC across all architectures (while RRL is within 10% of the respective pretrained AE loss), comparing **DClAM** to baselines. Best for each dataset is in bold. See text for further details. *Higher SC is better, but lower RRL is better.* The top set of rows are vision datasets, and the bottom set are text datasets. A '-' indicates not applicable (NA); e.g., DCEC, DEKM, SCAN, NNM work only on image datasets. Further, we report SCAN and NNM results only on C-10, C-100 and STL, since these are the only datasets for which pretrained contrastive encoders are available. x▼ indicates negative RRL which means the RL of the method is x% less than the pretrained AE loss.

Dataset	SC									RRL			
	k-means	Agglo.	ClAM	DCEC	DEKM	EDC	SCAN	NNM	DClAM	DCEC	DEKM	EDC	DClAM
FM	0.257	0.201	0.279	0.873	0.296	0.483	-	-	**0.932**	9.8	9.8	10	**1.6▼**
C-10	0.084	0.372	0.208	0.787	0.104	0.511	0.541	0.587	**0.863**	9.6	9.6	10	**0.5**
C-100	0.015	0.149	0.053	0.487	-0.018	0.311	0.321	0.358	**0.553**	10	10	10	**10**
USPS	0.195	0.158	0.194	0.871	0.256	0.461	-	-	**0.898**	10	10	**0.0**	10
STL	0.079	0.270	0.108	0.771	0.112	0.411	0.552	0.540	**0.891**	10	9.5	**4.9▼**	10
CBird	-0.019	0.094	-0.026	0.311	-0.032	0.171	-	-	**0.448**	10	**0.0**	10	9.1
R-10k	-0.010	0.114	-0.002	-	-	0.023	-	-	**0.564**	-	-	10	**10**
20NG	-0.021	0.114	-0.008	-	-	0.101	-	-	**0.197**	-	-	10	**10**

Q1: How does DClAM compare against baselines? We present the best Silhouette Coefficient or SC achieved (while constraining the reconstruction loss or RL to be within 10% of the pretrained AE loss) by **DClAM**, and the baselines for all 8 datasets in Table 1. As it is hard to compare the raw RL numbers if the base AE is different for different methods, we consider relative RL (RRL) defined as $(RL - RL\_PAE)/RL\_PAE)$ where RL_PAE is the pretrained/base RL. Then we present the best SC per method with $RRL <= 10\%$. From Table 1, we see across both image and text datasets, **DClAM** consistently outperforms traditional and deep clustering baselines in terms of SC while keeping RRL relatively low. To provide a comprehensive view alongside SC, we also present the best RRL (while constraining the SC to be within 10% of the best/peak SC of the method) in Table 4 in Appendix and visualize both SC and RL in Fig. 3 for all six image datasets. Both Table 4 and figures demonstrate that **DClAM** excels not only in achieving the best SC but also in minimizing RL compared to the baselines. Note that SCAN and NNM do not have a reconstruction loss term as they work on the pre-trained (pretext) model by SimCLR (Chen et al., 2020) and utilize only

Table 2: SC for image datasets, comparing **DClAM** to baselines with different encoder/decoder architectures. Best for each dataset is in bold. See text for details. *Higher is better.*

Dataset	Convolutional AE			ResNet AE			EAE	
	DCEC	DEKM	DClAM	DCEC	DEKM	DClAM	EDC	DClAM
FM	0.896	0.831	**0.932**	0.800	0.784	0.897	0.521	0.715
C-10	0.787	0.489	**0.863**	0.664	0.443	0.676	0.541	0.731
C-100	0.406	0.025	0.518	0.501	0.027	**0.684**	0.337	0.636
USPS	0.920	**0.946**	0.912	0.896	0.931	0.921	0.491	0.911
STL	0.822	0.675	0.919	0.854	0.824	0.881	0.431	**0.923**
CBird	0.386	0.018	**0.448**	0.282	0.035	0.377	0.188	0.446

the encoder (discarding the decoder) for clustering purpose. For additional insights, we present the best SC (while keeping RL within 10% of the pretrained AE loss) and its corresponding NMI, RL, and cluster sizes and balance obtained by all schemes in Table 5 in Appendix C.1. Simultaneously, Table 6 displays the best RL (while keeping SC within 10% of the best SC of the method) and its associated SC, NMI, and cluster sizes. We also present Table 7 which displays the best NMI and its associated SC, RL, and cluster sizes. **DClAM** consistently outperforms traditional and deep clustering baselines in terms of all SC, RL and NMI metrics.

Q2: Is DClAM's improvement agnostic to selected architecture? Table 2 shows that the performance improvements achieved by **DClAM** is independent of the Autoencoder (AE) architecture choice. **DClAM** with *all three architectures* – CAE, EAE, and RAE – consistently outperform their respective baselines, DCEC, DEKM and EDCWRN with similar architecture. This not only underscores the superiority of the internal algorithm of **DClAM** over the corresponding baselines but also suggests the potential for further improvement with some more advanced AE architecture.

Further results. We qualitatively evaluate the clusters found by **DClAM** in Fig. 2 for Fashion MNIST (10 clusters) and Caltech Birds (10 out of 200 clusters), visualizing the learned memories (centers), and the corresponding closest and farthest cluster members (as measured in the latent space). In most cases, the memories form a blurry image that match the closest images well. The farthest cluster members still appear similar to their memories in most cases, but do start changing significantly in some cases: (i) In the 7th row for FMNIST an image that looks like a pants is classified as a dress though the overall image shape is still similar. (ii) In the 5th row for CBirds, the memory and the closest are very similar but the farthest appears significantly different. In addition to the above, we discuss our thorough empirical evaluation in Appendix C, reporting various clustering metrics in Appendix C.1, and visualizing the evolution of the latent memories (cluster centers) in Appendix C.3.

Figure 2: Visualizing **DClAM++** clusters for Fashion MNIST (left block) and Caltech Birds (right block), with the learned memories (left column in block) and the corresponding closest (center column in block) and farthest (right column in block) images within their clusters.

5 Limitations and Future Work

In this paper, we introduce a fresh integration of associative memories in a deep neural network module to create the innovative deep clustering algorithms: **DClAM**, leveraging the AM attractor dynamics. Our findings demonstrate that **DClAM** significantly surpasses standard prototype-based clustering and existing deep clustering methods. However, it is worth noting that **DClAM** is still sensitive to hyperparameters and requires pretraining to avoid latent space collapse. Inspired by **DClAM**'s outstanding performance, our future work aims to extend it to multimodal deep clustering. We plan to explore new energy functions and update dynamics to enhance spectral and semantic clustering. Additionally, leveraging **DClAM**'s flexibility, we intend to automate the estimation of the number of clusters directly from the data.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pp. 265–283, 2016.

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

Amit, D. J., Gutfreund, H., and Sompolinsky, H. Storing infinite numbers of patterns in a spin-glass model of neural networks. *Physical Review Letters*, 55(14):1530, 1985.

Assent, I. Clustering high dimensional data. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(4):340–350, 2012.

Baldi, P. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pp. 37–49. JMLR Workshop and Conference Proceedings, 2012.

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

Bein, B. Entropy. *Best Practice & Research Clinical Anaesthesiology*, 20(1):101–109, 2006.

Bezdek, J. C., Ehrlich, R., and Full, W. Fcm: The fuzzy c-means clustering algorithm. *Computers & geosciences*, 10(2-3):191–203, 1984.

Cai, J., Wang, S., Xu, C., and Guo, W. Unsupervised deep clustering via contractive feature representation and focal loss. *Pattern Recognition*, 123:108386, 2022.

Caron, M., Bojanowski, P., Joulin, A., and Douze, M. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 132–149, 2018.

Chang, J., Wang, L., Meng, G., Xiang, S., and Pan, C. Deep adaptive image clustering. In *Proceedings of the IEEE international conference on computer vision*, pp. 5879–5887, 2017.

Chazan, S. E., Gannot, S., and Goldberger, J. Deep clustering based on a mixture of autoencoders. In *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6. IEEE, 2019.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223. JMLR Workshop and Conference Proceedings, 2011.

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

Demircigil, M., Heusel, J., Löwe, M., Upgang, S., and Vermet, F. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168(2):288–299, 2017.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

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

Donath, W. E. and Hoffman, A. J. Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17(5):420–425, 1973. doi: 10.1147/rd.175.0420.

Golzari Oskouei, A., Balafar, M. A., and Motamed, C. Edcwrn: efficient deep clustering with the weight of representations and the help of neighbors. *Applied Intelligence*, 53(5):5845–5867, 2023.

Guo, W., Lin, K., and Ye, W. Deep embedded k-means clustering. In *2021 International Conference on Data Mining Workshops (ICDMW)*, pp. 686–694. IEEE, 2021.

Guo, X., Gao, L., Liu, X., and Yin, J. Improved deep embedded clustering with local structure preservation. In *Ijcai*, pp. 1753–1759, 2017a.

Guo, X., Liu, X., Zhu, E., and Yin, J. Deep clustering with convolutional autoencoders. In *International conference on neural information processing*, pp. 373–382. Springer, 2017b.

Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

Huang, X., Hu, Z., and Lin, L. Deep clustering based on embedded auto-encoder. *Soft Computing*, 27(2):1075–1090, 2023.

Hull, J. J. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994.

Johnson, S. C. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

Kaufman, L. and Rousseeuw, P. J. *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Krotov, D. and Hopfield, J. J. Dense associative memory for pattern recognition. *Advances in neural information processing systems*, 29, 2016.

Lloyd, S. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2): 129–137, 1982.

Lucibello, C. and Mézard, M. The exponential capacity of dense associative memories. *arXiv preprint arXiv:2304.14964*, 2023.

MacQueen, J. Classification and analysis of multivariate observations. In *5th Berkeley Symp. Math. Statist. Probability*, pp. 281–297, 1967.

McEliece, R., Posner, E., Rodemich, E., and Venkatesh, S. The capacity of the hopfield associative memory. *IEEE transactions on Information Theory*, 33(4):461–482, 1987.

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

Müllner, D. Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*, 2011.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Ramsauer, H., Schäfl, B., Lehner, J., Seidl, P., Widrich, M., Adler, T., Gruber, L., Holzleitner, M., Pavlović, M., Sandve, G. K., et al. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*, 2020.

Ren, Y., Pu, J., Yang, Z., Xu, J., Li, G., Pu, X., Philip, S. Y., and He, L. Deep clustering: A comprehensive survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2024. URL `https://ieeexplore.ieee.org/abstract/document/10585323`.

Ronen, M., Finder, S. E., and Freifeld, O. Deepdpm: Deep clustering with an unknown number of clusters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9861–9870, 2022.

Rousseeuw, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

Rumelhart, D. E., Hinton, G. E., Williams, R. J., et al. Learning internal representations by error propagation, 1985.

Saha, B., Krotov, D., Zaki, M. J., and Ram, P. End-to-end differentiable clustering with associative memories. *arXiv preprint arXiv:2306.03209*, 2023.

Sammut, C. and Webb, G. I. (eds.). *TF–IDF*, pp. 986–987. Springer US, Boston, MA, 2010. ISBN 978-0-387-30164-8. doi: 10.1007/978-0-387-30164-8_832. URL https://doi.org/10.1007/978-0-387-30164-8_832.

Saxena, A., Prasad, M., Gupta, A., Bharill, N., Patel, O. P., Tiwari, A., Er, M. J., Ding, W., and Lin, C.-T. A review of clustering techniques and developments. *Neurocomputing*, 267:664–681, 2017.

Schaeffer, R., Khona, M., Zahedi, N., Fiete, I. R., Gromov, A., and Koyejo, S. Associative memory under the probabilistic lens: Improved transformers & dynamic memory creation. In *Associative Memory $\{\backslash\&\}$ Hopfield Networks in 2023*, 2023.

Steinbach, M., Ertöz, L., and Kumar, V. The challenges of clustering high dimensional data. In *New directions in statistical physics: econophysics, bioinformatics, and pattern recognition*, pp. 273–309. Springer, 2004.

Van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Van Gansbeke, W., Vandenhende, S., Georgoulis, S., Proesmans, M., and Van Gool, L. Scan: Learning to classify images without labels. In *European conference on computer vision*, pp. 268–285. Springer, 2020.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Verleysen, M. and François, D. The curse of dimensionality in data mining and time series prediction. In *International work-conference on artificial neural networks*, pp. 758–770. Springer, 2005.

Vinh, N. X., Epps, J., and Bailey, J. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th annual international conference on machine learning*, pp. 1073–1080, 2009.

Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., and Perona, P. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

Wickramasinghe, C. S., Marino, D. L., and Manic, M. Resnet autoencoders for unsupervised feature learning from high-dimensional data: Deep models resistant to performance degradation. *IEEE Access*, 9:40511–40520, 2021.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

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

Xu, D. and Tian, Y. A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2: 165–193, 2015.

Xu, R. and Wunsch, D. Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16 (3):645–678, 2005.

Zaki, M. J. and Meira Jr, W. *Data mining and machine learning: Fundamental concepts and algorithms*. Cambridge University Press, 2020.

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

A Related Work

Clustering is a long-studied and well-reviewed problem in computer science, with various formulations and several applications (Kaufman & Rousseeuw, 2009; Zaki & Meira Jr, 2020). Here we will review existing and relevant literature on deep clustering and associative memories.

Deep clustering. This has been extensively studied over the past decade (Ren et al., 2024; Aljalbout et al., 2018; Zhou et al., 2022). Inspired by t-SNE (Van der Maaten & Hinton, 2008), Xie et al. (2016) introduced DEC, enhancing clustering and feature representation by minimizing the Kullback-Leibler Divergence (KLD) to an auxiliary target distribution. However, a drawback is abandoning the decoder layer after pre-training, impacting the embedded space and clustering performance. Guo et al. (2017a) showed that keeping the decoder layer improves clustering (IDEC), and Guo et al. (2017b) proposed DCEC using convolutional autoencoders (CAE). Chazan et al. (2019) proposed DAMIC, a mixture of autoencoders for clustering, determined by minimizing the reconstruction loss without needing a regularization term. However, they leverage multiple AEs to their model, while we focus on schemes using single AE. Huang et al. (2023) introduced an innovative embedded auto-encoder architecture by incorporating it into both the encoding and decoding units of the outer auto-encoder. Guo et al. (2021) proposed DEKM which works on the embedding space (after pretraining) and transforms it to a new cluster-friendly space using an orthonormal transformation matrix. However, discarding the decoder after pretraining for both of these methods may lead to the distortion of the embedded space, consequently hurting clustering performance. In addressing the automatic inference of the number of clusters in a dataset, Ronen et al. (2022) introduced DeepDPM. They proposed a novel loss inspired by EM in the Bayesian Gaussian Mixture Model, facilitating a new amortized inference in mixture models. It is worth noting that DeepDPM diverges from the typical encoder-decoder architecture, opting instead for a multilayer perceptron model.

While many deep clustering methods utilize KLD as a clustering objective, it falls short in preserving the global data structure (which implies that only within-cluster distances are prioritized, leaving uncertainties regarding between-cluster similarities), leading Golzari Oskouei et al. (2023) (EDCWRN) to advocate for cross-entropy over KLD. They incorporate feature weighting to emphasize essential features for clustering and employ a neighborhood technique to encourage similar representations for samples within the same cluster. Addressing another challenge with KLD regarding the presence of hard, misclassified samples, Cai et al. (2022) introduced focal loss to enhance label assignment in deep clustering methods and improved the representation learning module with a contractive penalty term, capturing more discriminative representations. However, it could lead to unintentional bias in the optimization focus between the representation learning and clustering modules. Dang et al. (2021) introduces a novel deep clustering framework (NNM) based on a two-level nearest neighbors matching approach. Distinguishing itself from prior methods (Van Gansbeke et al., 2020), NNM incorporates matching at both local and global levels, resulting in a notable enhancement in clustering performance. Both studies leverage SimCLR (Chen et al., 2020) to pretrain a representation learning model using the state-of-the-art contrastive learning loss. In our work, we rethink the deep clustering problem at a architecture agnostic level by leveraging the capabilities of associative memories. Thus, various architectural and pretraining advancements would also benefit our proposed scheme.

Associative Memory (AM) and Clustering. AMs adeptly store multidimensional vectors as fixed point attractor states in a recurrent dynamical system. AMs form associations between the initial state and a final state (memory), creating disjoint basins of attractions which are crucial for clustering. Initially conceptualized as the classical Hopfield Network (Hopfield, 1982), AM exhibits limited memory capacity, approximately storing only $\approx 0.14d$ arbitrary memories in a d dimensional data domain (McEliece et al., 1987; Amit et al., 1985). Subsequently, Dense AM or Modern Hopfield Network was suggested by Krotov & Hopfield (2016), introducing rapidly expanding non-linearities (activation functions) into the system. This advancement enables a more concentrated memory arrangement and attains super-linear (in d) memory capacity (Demircigil et al., 2017; Ramsauer et al., 2020; Lucibello & Mézard, 2023). With softmax activation, Dense AMs can serve as a unique limiting case of the attention mechanism used in transformers (Vaswani et al., 2017) and BERT (Devlin et al., 2018) model (Ramsauer et al., 2020). Recently, Saha et al. (2023) introduced `ClAM`, an end-to-end differentiable clustering approach, utilizing AMs for clustering. `ClAM` presents a versatile mathematical framework, introducing a novel continuous unconstrained relaxation of the discrete optimization challenge in clustering. Schaeffer et al. (2023) demonstrates that the energy function of `ClAM`'s AM network resembles a scaled negative log-likelihood of a Gaussian mixture

Algorithm 1: Deep clustering a dataset $S \in \mathbb{R}^d$ in a latent space $\mathbb{R}^m$ into k clusters with encoder $\mathbf{e}$ and decoder $\mathbf{d}$. The cluster assignment is done with T recursion of the AM attractor dynamics operator $A_{\boldsymbol{\rho}}$ parameterized with the centers $\boldsymbol{\rho} = \{\rho_i, i \in [\![k]\!]\}$. The per-sample loss of **`DClAM`** (line 10) is highlighted in Sepia. We optimize for N epochs with learning rates $\{\epsilon_{\mathbf{e}}, \epsilon_{\mathbf{d}}, \epsilon_{\boldsymbol{\rho}}\}$ for $\mathbf{e}, \mathbf{d}, \boldsymbol{\rho}$ respectively. The hyperparameters of $A_{\boldsymbol{\rho}}$ are not shown here for the ease of exposition.

1 **Train**$(S, k, N, T, \epsilon_{\mathbf{e}}, \epsilon_{\mathbf{d}}, \epsilon_{\boldsymbol{\rho}}, \gamma)$
2   Pretrain $(\mathbf{e}, \mathbf{d})$ as autoencoder, minimizing $\mathcal{L}_r(\mathbf{e}, \mathbf{d})$
3     $\boldsymbol{\rho} \leftarrow \{\mathbf{e}(x), x \in M\}$, M are random k samples from S
4   **for** *epoch* $n = 1, \ldots, N$ **do**
5   **for** *batch* $B \in S$ **do**
6   Batch loss $\mathcal{L}_B \leftarrow 0$
7   **for** *example* $x \in B$ **do**
8                 $v \leftarrow \mathbf{e}(x)$ `//encode input`
9                 $\bar{v} \leftarrow A^T_{\boldsymbol{\rho}}(v)$ `//AM recursion`
10                 $\ell \leftarrow \|x - \mathbf{d}(\bar{v})\|^2$
11                 $\mathcal{L}_B \leftarrow \mathcal{L}_B + \ell$
12             $\rho_i \leftarrow \rho_i - \epsilon_{\boldsymbol{\rho}} \nabla_{\rho_i} \mathcal{L}_B \ \forall i \in [\![k]\!]$
13             $\mathbf{e} \leftarrow \mathbf{e} - \epsilon_{\mathbf{e}} \nabla_{\mathbf{e}} \mathcal{L}_B$
14             $\mathbf{d} \leftarrow \mathbf{d} - \epsilon_{\mathbf{d}} \nabla_{\mathbf{d}} \mathcal{L}_B$
15   **return** $\mathbf{e}, \mathbf{d}, \boldsymbol{\rho}$
16 **Infer**$(S, \mathbf{e}, \mathbf{d}, \boldsymbol{\rho})$
17   Cluster assignments $C \leftarrow \emptyset$
18   **for** $x \in S$ **do**
19         $\bar{v} \leftarrow A^T_{\boldsymbol{\rho}}(\mathbf{e}(x))$ `//encode` $\rightarrow$ `AM recursion`
20         $C \leftarrow C \cup \left\{ \arg\min_{i \in [\![k]\!]} \|\rho_i - \bar{v}\|^2 \right\}$
21   **return** *Per-point cluster assignments* C

model and that the dynamics of the AM network can be viewed as expectation maximization via gradient ascent. In our work, we study the interaction of clustering with latent AMs and representation learning previously not considered in literature.

B Experimental Details

B.1 Dataset details

To evaluate **`DClAM`**, we conducted our experiments on eight standard benchmark data sets. The datasets are taken from various sources such as USPS from Kaggle[1] (Hull, 1994), Fashion-MNIST from Zalando[2] (Xiao et al., 2017), CIFAR-10 & CIFAR-100 from Krizhevsky[3] (Krizhevsky et al., 2009), STL-10 from Coates et al. (2011)[4], Caltech_birds2010 from Welinder et al. (2010)[5], 20-NG from sklearn[6] and Reuters-10k from TensorFlow datasets[7]. The later two are text datasets, whereas the others are image datasets. For both text datasets, we calculate TFIDF (Sammut & Webb, 2010) features based on the 2000 most frequent words, following a similar approach as Golzari Oskouei et al. (2023) (originally proposed by Xie et al. (2016)). However, unlike their methodology, we diverge by not employing four root categories to represent four clusters in the case of Reuters-10k. Instead, we consider the original number of categories as the true number of clusters, which is 46 for Reuters-10k and 20 for 20-NG. For Caltech_birds2010, as there are images of various shapes, we

[1] https://www.kaggle.com/datasets/bistaumanga/usps-dataset
[2] https://github.com/zalandoresearch/fashion-mnist
[3] https://www.cs.toronto.edu/~kriz/cifar.html
[4] https://cs.stanford.edu/~acoates/stl10/
[5] https://www.tensorflow.org/datasets/catalog/caltech_birds2010
[6] https://scikit-learn.org/0.19/datasets/twenty_newsgroups.html
[7] https://www.tensorflow.org/api_docs/python/tf/keras/datasets/reuters/load_data

resize all images to (128, 128, 3) for uniformity and ease of implementation. Table 3 provides the statistics for the datasets used in our experiments.

Table 3: Descriptions of various benchmark datasets, used in our experiments.

Dataset	Short name	# Points	Shape	# Classes	# Type
Fashion MNIST	FM	60000	(28, 28, 1)	10	Image
CIFAR-10	C-10	50000	(32, 32, 3)	10	Image
CIFAR-100	C-100	50000	(32, 32, 3)	100	Image
USPS	USPS	2007	(16, 16, 1)	10	Image
STL-10	STL	5000	(96, 96, 3)	10	Image
Caltech_birds2010	CBird	3000	(128, 128, 3)	200	Image
Reuters-10k	R-10k	11228	2000	46	Text
20-NG	20NG	18846	2000	20	Text

B.2 Metrics used

To assess the performance of **DClAM**, we utilize the Silhouette Coefficient (SC) (Rousseeuw, 1987) as an unsupervised metric for measuring clustering quality. SC scores range from -1 to 1, where 1 indicates perfect clustering and -1 indicates completely incorrect labels. A score close to 0 suggests the existence of overlapping clusters. We also employ Normalized Mutual Information (NMI) (Vinh et al., 2009) to evaluate the alignment between the partition obtained by **DClAM** and the ground truth clustering labels. NMI scores range from 0 (completely incorrect) to 1 (perfect clustering). Additionally, we compute Reconstruction Loss (RL), representing the mean squared error between original and reconstructed points, where lower is better. Entropy (ETP) (Bein, 2006) and Cluster Size (CS) are computed to assess cluster balance. In clustering, higher entropy (the highest value is $\log_2(k)$ for each dataset, where k is the number of true cluster) indicates more balanced clusters, while lower values suggest potential imbalance, possibly involving singleton or very small clusters. Entropy ($H(X)$) is calculated based on the distribution of data points across clusters like below:

$$H(X) = -\sum_{i=1}^{k} P(C_i) \log_2(P(C_i))$$

where, $P(C_i)$ is the proportion of data points in cluster C_i relative to the total number of data points. Cluster Size (CS) indicates the largest and smallest clusters (in terms of the number of data points) identified in the dataset where the difference should not be so large.

B.3 Parameter setting

For CAE with k-means, Agglomerative, **ClAM**, DCEC, DEKM, and **DClAM**, we adopt an architecture identical to DCEC. The encoder network structure follows $\text{conv}^5_{32} \rightarrow \text{conv}^5_{64} \rightarrow \text{conv}^3_{128} \rightarrow \text{FC}_d$, where conv^k_n represents a convolutional layer with n filters and a kernel size of $k \times k$. Here, d denotes the number of true clusters in the dataset, serving as the latent dimension. The decoder mirrors the encoder.

In RAE with DCEC, DEKM, and **DClAM** a streamlined configuration is employed using two filters with sizes 32 and 64. The size of the embedded representation is maintained at d, corresponding to the number of clusters in the dataset, as in the previous setup. In this experiment, the number of repeating layers in the ResNet block is set to 2. To enhance model performance, batch normalization and leakyReLU are incorporated. For a given number of repeats (f), the total number of hidden layers is calculated as 2 + (f * number of filters), resulting in 6 layers in our case. This approach draws inspiration from the standard ResNet block described by Wickramasinghe et al. (2021).

For EAE with EDCWRN, and **DClAM**, we follow exactly similar architecture as EDCWRN where the encoder network is configured as a fully connected multilayer perceptron (MLP) with dimensions i-500-500-2000-d for all datasets, where i represents the dimension of the input space (features), and d is the number of clusters in the dataset. Similarly, the decoder network mirrors the encoder, constituting an MLP with dimensions d-2000-500-500-i. All internal layers, except for the input, output, and embedding layers, are activated by the ReLU nonlinearity function.

All three architectures described above are pretrained end-to-end for 100 epochs using Adam (Kingma & Ba, 2014) with default parameters.

B.4 Implementation details

We implement and evaluate **`DClAM`** using the Tensorflow (Abadi et al., 2016) library while employing `scikit-learn` (Pedregosa et al., 2011) for clustering baselines and quality metrics. We train our models on a single node with 1 NVIDIA RTX A6000 (48GB RAM) and a 16-core 2.4GHz Intel Xeon(R) Silver 4314 CPU. Hyperparameters are tuned individually for each dataset to maximize the Silhouette Coefficient (Rousseeuw, 1987). Table 8 illustrates the chosen hyperparameters, their roles, and respective values/ranges.

For baseline schemes like k-means and agglomerative, we use the `scikit-learn` library implementation, adjusting hyperparameters for optimal performance on each dataset. For DCEC (Guo et al., 2017b) and DEKM (Guo et al., 2021), we leverage their Tensorflow implementation[8] [9]and for EDCWRN (Golzari Oskouei et al., 2023), we utilze their Python implemtentation[10]. We adopt a similar hyperparameter tuning strategy for the baseline schemes as employed in **`ClAM`** (Saha et al., 2023).

B.5 How `DClAM` loss relates to traditional deep clustering loss

Here, we show how the **`DClAM`** loss $\bar{\mathcal{L}}$ in Eq. (6) is related to the loss $\mathcal{L} = \mathcal{L}_r + \gamma \mathcal{L}_c$ in Eq. (3). If the encoder $\mathbf{e}$ and decoder $\mathbf{d}$ form a decent autoencoder (for example, if they are pretrained, as is common practice), then for a input $x \in S$, the single sample loss can be compared as follows:

$$\ell_r(x, \mathbf{e}, \mathbf{d}) \triangleq \|x - \mathbf{d}(\mathbf{e}(x))\|^2 \le \|x - \mathbf{d}(A_{\boldsymbol{\rho}}^T(\mathbf{e}(x)))\|^2 \triangleq \bar{\ell}(x, \mathbf{e}, \mathbf{d}, \boldsymbol{\rho}), \tag{7}$$

since $A_{\boldsymbol{\rho}}^T(\mathbf{e}(x))$ will be some distortion of $\mathbf{e}(x)$, and thus its decoded version will generally be worse than the decoded version of $\mathbf{e}(x)$. Let us now assume that the decoder $\mathbf{d} : \mathbb{R}^m \to \mathbb{R}^d$ is $C_{\mathbf{d}}$-Lipschitz continuous. Then, considering the per-sample loss $\bar{\ell}$ in Eq. (6), and applying the triangle inequality and the AM–GM inequality, we can show that

$$\begin{aligned}\bar{\ell}(x, \mathbf{e}, \mathbf{d}, \boldsymbol{\rho}) &= \|x - \mathbf{d}(A_{\boldsymbol{\rho}}^T(\mathbf{e}(x)))\|^2 \\ &\le 2\left(\|x - \mathbf{d}(\mathbf{e}(x))\|^2 + \|\mathbf{d}(\mathbf{e}(x)) - \mathbf{d}(A_{\boldsymbol{\rho}}^T(\mathbf{e}(x)))\|^2\right) \\ &\le 2\left(\|x - \mathbf{d}(\mathbf{e}(x))\|^2 + C_{\mathbf{d}}^2\|\mathbf{e}(x) - A_{\boldsymbol{\rho}}^T(\mathbf{e}(x))\|^2\right) \\ &= 2\ell_r(x, \mathbf{e}, \mathbf{d}) + 2C_{\mathbf{d}}^2 \ell_c(x, \mathbf{e}, \boldsymbol{\rho}),\end{aligned} \tag{8}$$

where the last inequality uses the Lipschitz continuity, and the last equality comes from the definition of the clustering loss in the latent space with the AM dynamics operator. Summing the above inequalities in Eqs. (7) and (8) over $x \in S$ gives us $\mathcal{L}_r \le \bar{\mathcal{L}} \le \gamma_1 \mathcal{L}_r + \gamma_2 \mathcal{L}_c$, where the upperbound of $\bar{\mathcal{L}}$ is (a scaled version of) the standard deep clustering objective of the weighted combination of the reconstruction loss $\mathcal{L}_r$ and the clustering loss $\mathcal{L}_c$ in Eq. (3).

We would like to clarify that **`DClAM`** does not impose any specific constraints on the structure of the encoder and decoder (refer to Algorithm 1). In our discussion regarding Lipschitz continuity, our main goal is to highlight the relationship between the novel loss of **`DClAM`** and the loss of traditional deep clustering (Eq. (3) that consists of reconstruction and clustering losses). This comparison serves to underscore how the novel loss is related to the better intertwining of the different components of the deep clustering pipeline – the encoder, decoder, cluster centers. The novel **`DClAM`** loss provides significant improvements over Eq. (3) which uses the standard loss. Also note that if it is a decoder that we can differentiate through with auto-grad, the decoder is Lipschitz continuous. Additionally, there exists a more general notion called the modulus of continuity, which extends beyond Lipschitz continuity. We can substitute Lipschitz continuity with the modulus of continuity in our discussion, maintaining the same inequality but with potentially different constants.

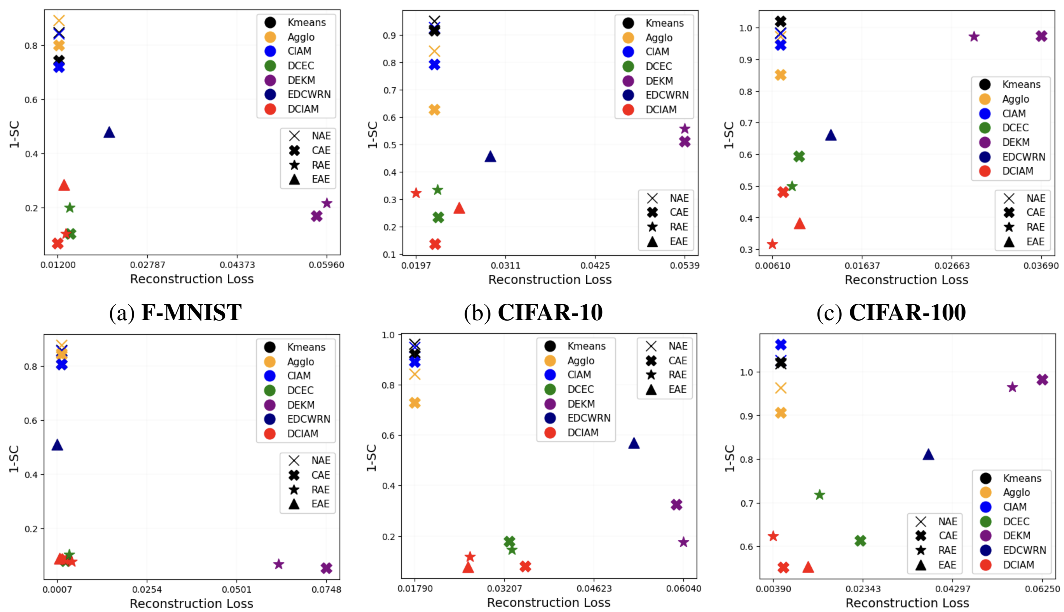
[8] https://github.com/XifengGuo/DCEC

[9] https://github.com/spdj2271/DEKM/blob/main/DEKM.py

[10] https://github.com/Amin-Golzari-Oskouei/EDICWRN

Table 4: Per-method best RRL across all architectures (while SC is within 10% of the best SC of the method) comparing **DC1AM** to baselines. Best for each dataset is in bold. See text for further details. *Higher SC is better, but lower RRL is better.* x$^{\blacktriangledown}$ indicates negative RRL which means the RL of the method is x% less than the pretrained AE loss.

Dataset	SC									RRL			
	k-means	Agglo.	C1AM	DCEC	DEKM	EDC	SCAN	NNM	DC1AM	DCEC	DEKM	EDC	DC1AM
FM	0.257	0.201	0.279	**0.896**	0.831	0.521	-	-	0.865	16.4	374	143	**4.1**$^{\blacktriangledown}$
C-10	0.084	0.372	0.208	0.766	0.489	0.541	0.541	0.587	**0.809**	2.3	145	74.3	**0.5**$^{\blacktriangledown}$
C-100	0.015	0.149	0.053	0.406	0.025	0.337	0.321	0.358	**0.476**	30	427	33.3	**17.1**$^{\blacktriangledown}$
USPS	0.195	0.158	0.194	0.920	**0.931**	0.491	-	-	0.912	52.6	2582	**40**	42.1
STL	0.079	0.270	0.108	0.822	0.675	0.431	0.552	0.540	**0.923**	83.2	231	155	**27.7**
CBird	-0.019	0.094	-0.026	0.282	0.018	0.188	-	-	**0.413**	286	1036	102	**1.8**
R-10k	-0.010	0.114	-0.002	-	-	0.035	-	-	**0.673**	-	-	**60**	120
20NG	-0.021	0.114	-0.008	-	-	0.099	-	-	**0.287**	-	-	**25**$^{\blacktriangledown}$	50



(a) **F-MNIST** (b) **CIFAR-10** (c) **CIFAR-100**

(d) **USPS** (e) **STL-10** (f) **CBird**

Figure 3: Reconstruction loss and clustering quality (1-SC) for CIFAR-100, STL-10 and CBird. Different markers stand for various AE architectures, and different colors signify distinct methods. *Lower is better for both axes*, since we plot 1-SC on the y-axis.

C Additional Experimental Results

C.1 Detailed results with various clustering quality metrics

Table 5 provides a comprehensive overview of the metrics (SC, NMI, RL, ETP, and CS) for **DC1AM**, and corresponding baselines, focusing on the best SC in each method across various AE architectures where RL is constrained to 10% of the pretrained AE loss. RL is not presented for k-means, Agglomerative and **C1AM** for the original space and for CAE as it remains consistent across the three methods after pre-taining. Similarly, Table 6 provides a similar overview of the metrics (SC, NMI, RL, ETP, and CS) for **DC1AM**, and corresponding baselines, focusing on the best Relative RL (RRL) in each method across various AE architectures where SC is constrained to 10% of the best/peak SC of the method. Table 7 represents all corresponding metrics focusing on the best NMI in each method. These tables highlight that **DC1AM** exhibits strong performance not only in terms of SC and RL, but also when compared to the ground truth labels via NMI. In fact, for NMI, **DC1AM** has the best values in 5 out of the 8 datasets (DCEC has the best values on the other 3). Additionally, **DC1AM** clusters maintain reasonable entropy (ETP) and cluster size (CS), ensuring a balanced clustering outcome.

Table 5: **Metrices obtained by `DClAM` and baselines corresponding to the best SC (RL within 10% of the pretrained AE loss).** The best performance for each dataset is in **boldface**. (note abbreviations DCEC→DC, EDCWRN→EDC, Entropy→ETP, Cluster-size→CS, No-AE→NAE, Conv-AE→CAE, EDCWRN-AE→EAE, Resnet-AE→RAE). '-' denotes NA. x▼ indicates negative RRL which means the RL of the method is x% less than the pretrained AE loss.

Data	Met	Kmeans		Agglo		ClAM		DC		DEKM		EDC	DClAM		
		NAE	CAE	NAE	CAE	NAE	CAE	CAE	RAE	CAE	RAE		CAE	EAE	RAE
FM	SC	0.154	0.257	0.109	0.201	0.158	0.279	0.873	0.712	0.296	0.285	0.483	**0.932**	0.663	0.715
	NMI	0.511	0.643	0.534	0.624	0.521	0.622	0.564	0.624	0.648	0.619	0.495	0.472	0.511	0.379
	RL	-	0.0122	-	0.0122	-	0.0122	0.0134	0.0090	0.0134	0.0089	0.0096	0.0120	0.0096	0.0091
	RRL	-	0.0	-	0.0	-	0.0	9.8	8.4	9.8	7.2	10	**1.6**▼	10	9.6
	ETP	3.17	3.17	3.14	3.2	2.81	2.80	3.23	3.23	3.14	3.15	3.11	2.83	3.14	2.99
	CS	9617-2361	11145-2744	11830-1860	10298-2544	19032-1524	15679-2	10421-2779	8975-3218	10771-1118	11196-2789	12118-1478	15458-422	11734-2251	11878-1319
C-10	SC	0.050	0.084	0.158	0.372	0.073	0.208	0.787	0.645	0.104	0.095	0.511	**0.863**	0.632	0.676
	NMI	0.078	0.122	0.0005	0.0004	0.073	0.015	0.074	0.094	0.123	0.121	0.112	0.075	0.061	0.079
	RL	-	0.0220	-	0.0220	-	0.0220	0.0241	0.0198	0.0241	0.0197	0.0184	0.0221	0.0184	0.0197
	RRL	-	0.0	-	0.0	-	0.0	9.6	10	9.6	9.4	10	**0.5**	10	2.2
	ETP	3.27	3.19	0.006	0.003	2.50	0.24	3.22	2.99	2.99	3.15	3.24	2.83	2.65	2.50
	CS	7105-2734	9779-2524	49979-1	49991-1	23544-582	48234-1	8511-2610	11341-1689	12710-1165	11731-2107	8198-2632	17430-380	13771-570	18125-465
C-100	SC	0.015	-0.020	0.028	0.149	0.018	0.053	0.388	0.487	-0.007	-0.018	0.311	0.518	0.636	**0.553**
	NMI	0.161	0.183	0.036	0.004	0.153	0.156	0.111	0.119	0.180	0.184	0.181	0.110	0.202	0.125
	RL	-	0.0070	-	0.0070	-	0.0070	0.0077	0.0044	0.0074	0.0044	0.0106	0.0073	0.0099	0.0044
	RRL	-	0.0	-	0.0	-	0.0	10	10	5.7	10	4.3	10	**3.1**	10
	ETP	6.53	6.48	0.940	0.052	6.51	4.38	6.17	4.08	5.23	6.46	6.49	4.16	5.85	3.21
	CS	1160-129	1395-23	38814-1	49834-1	1317-177	13950-11	1312-152	14731-24	2312-121	1592-73	999-216	12195-10	4116-32	10003-10
USPS	SC	0.143	0.195	0.124	0.158	0.144	0.194	0.871	0.867	0.256	0.255	0.461	**0.898**	0.872	0.869
	NMI	0.573	0.628	0.627	0.680	0.475	0.619	0.706	0.701	0.712	0.684	0.467	0.444	0.347	0.428
	RL	-	0.0019	-	0.0019	-	0.0019	0.0021	0.0026	0.0021	0.0024	0.0005	0.0021	0.0006	0.0025
	RRL	-	0.0	-	0.0	-	0.0	10	10	10	4.3	**0.0**	10	10	8.7
	ETP	3.27	3.23	3.26	3.27	3.10	3.16	3.26	3.27	3.23	3.25	3.29	3.12	2.78	2.99
	CS	284-121	359-89	333-121	328-104	420-53	375-64	297-105	281-110	288-91	321-96	295-134	438-69	841-76	519-47
STL	SC	0.039	0.079	0.158	0.270	0.051	0.108	0.753	0.771	0.112	0.093	0.411	0.814	**0.891**	0.821
	NMI	0.127	0.152	0.007	0.004	0.106	0.139	0.187	0.165	0.162	0.161	0.066	0.147	0.073	0.109
	RL	-	0.0179	-	0.0179	-	0.0179	0.0197	0.0191	0.0198	0.0191	0.0196	0.0192	0.0227	0.0190
	RRL	-	0.0	-	0.0	-	0.0	10	10	10	10	**4.9**▼	7.3	10	9.8
	ETP	3.26	3.25	0.069	0.025	2.43	1.4	3.23	3.27	3.23	3.21	2.92	2.48	2.99	2.87
	CS	764-312	830-287	4969-1	4991-1	2586-82	3888-38	831-242	657-348	841-213	817-256	2611-33	2170-33	912-45	1469-71
CBird	SC	-0.019	-0.021	0.037	0.094	-0.026	-0.062	0.311	0.251	-0.032	-0.037	0.171	**0.448**	0.446	0.312
	NMI	0.412	0.353	0.206	0.132	0.423	0.485	0.347	0.299	0.372	0.370	0.471	0.221	0.467	0.211
	RL	-	0.0055	-	0.0055	-	0.0055	0.0061	0.0040	0.0055	0.0036	0.0206	0.0060	0.0115	0.0039
	RRL	-	0.0	-	0.0	-	0.0	10	10	0.0	0.0	10	9.1	**39**▼	8.3
	ETP	6.34	5.59	2.71	0.958	6.56	7.21	5.41	5.04	5.81	5.80	7.41	5.68	7.02	5.07
	CS	131-1	245-1	1722-1	2773-1	101-2	99-2	241-1	291-1	168-1	197-1	37-2	213-1	99-1	676-1
R-10k	SC	-0.010	-	0.114	-	-0.002	-	-	-	-	-	0.023	-	**0.564**	-
	NMI	0.398	-	0.012	-	0.383	-	-	-	-	-	0.152	-	0.367	-
	RL	-	-	-	-	-	-	-	-	-	-	0.0011	-	0.0011	-
	RRL	-	-	-	-	-	-	-	-	-	-	**10**	-	**10**	-
	ETP	5.13	-	0.072	-	5.10	-	-	-	-	-	5.51	-	4.77	-
	CS	916-20	-	11172-1	-	885-18	-	-	-	-	-	721-51	-	1046-1	-
20NG	SC	-0.021	-	0.114	-	-0.008	-	-	-	-	-	0.101	-	**0.197**	-
	NMI	0.155	-	0.003	-	0.166	-	-	-	-	-	0.019	-	0.181	-
	RL	-	-	-	-	-	-	-	-	-	-	0.0009	-	0.0009	-
	RRL	-	-	-	-	-	-	-	-	-	-	**10**	-	**10**	-
	ETP	4.03	-	0.022	-	3.86	-	-	-	-	-	4.32	-	4.21	-
	CS	2217-107	-	18818-1	-	3428-26	-	-	-	-	-	1131-599	-	1812-199	-

For an understanding of the importance of ETP and CS in clustering, consider the case of Agglomerative clustering in the latent space (CAE) on the CIFAR-10 dataset (see Table 5). In this instance, almost all points (49991 out of 50000) belong to one cluster, while the other 9 clusters contain only one data point each, indicating very poor clustering. The low entropy (0.003) further highlights the deficiency of the clustering.

In certain situations, when comparing two clustering methods, it can happen that a method performs better in terms of SC and RL but still exhibits a lower NMI compared to another method (see Table 5 for USPS where **`DClAM`** outperforms DCEC in both CAE and RAE architecture in both SC and RL, however, the NMI is worse than DCEC in both cases). This indicates that the alignment of semantic class (ground truth or true underlying structure) with the geometric characteristics of the data might not be consistent or straightforward.

C.2 Hyperparameter dependency for `DClAM`

We extensively tune all hyperparameters (Table 8) for the optimal results in **`DClAM`**. We found that the inverse temperature β serves as the most critical hyperparameter, which we explore in the range of $[10^{-5}, ..., 5]$ for tuning. We employ the Adam optimizer while keeping separate initial learning rates for the AM and AE networks. If the training loss does not improve for a certain number of epochs, we decrease the learning rate by a factor of 0.8 until it reaches the minimum threshold (10^{-6}). Each hyperparameter configuration is run mostly for 300 epochs (in certain cases longer training is needed for better results) with 5 restarts using different random seeds. Throughout each epoch, we track the training loss. The set of hyperparameters and the associated model yielding the lowest training loss are chosen during the inference step. The best hyperparameter values for various datasets for **`DClAM`** are detailed in Table 9.

Table 6: **Metrices obtained by `DClAM` and baselines corresponding to the best RL (SC within 10% of the best SC of the method).** The best performance for each dataset is in **boldface**. (note abbreviations DCEC→DC, EDCWRN→EDC, Entropy→ETP, Cluster-size→CS, No-AE→NAE, Conv-AE→CAE, EDCWRN-AE→EAE, Resnet-AE→RAE). '-' denotes NA. x$^{\blacktriangledown}$ indicates negative RRL which means the RL of the method is x% less than the pretrained AE loss.

Data	Met	Kmeans		Agglo		`ClAM`		DC		DEKM		EDC	`DClAM`		
		NAE	CAE	NAE	CAE	NAE	CAE	CAE	RAE	CAE	RAE		CAE	EAE	RAE
	SC	0.154	0.257	0.109	0.201	0.158	0.279	0.896	0.800	0.831	0.784	0.521	0.865	0.715	**0.897**
	NMI	0.511	0.643	0.534	0.624	0.521	0.622	0.561	0.623	0.585	0.639	0.493	0.472	0.522	0.377
FM	RL	-	0.0122	-	0.0122	-	0.0122	0.0142	0.0141	0.0578	0.0596	0.0211	0.0117	0.0131	0.0134
	RRL	-	0.0	-	0.0	-	0.0	16.4	69.9	374	618	143	**4.1**$^{\blacktriangledown}$	54.0	61.4
	ETP	3.17	3.17	3.14	3.2	2.81	2.80	3.22	3.24	3.07	3.16	3.09	2.83	3.16	2.98
	CS	9617-2361	11145-2744	11830-1860	10298-2544	19032-1524	15679-2	10523-2775	8877-3061	12986-119	11023-2652	13199-1391	15458-422	11886-2148	12836-1378
	SC	0.050	0.084	0.158	0.372	0.073	0.208	0.766	0.664	0.489	0.443	0.541	**0.809**	0.731	0.592
	NMI	0.078	0.122	0.0005	0.0004	0.073	0.015	0.073	0.094	0.098	0.115	0.111	0.075	0.060	0.082
C-10	RL	-	0.0220	-	0.0220	-	0.0220	0.0225	0.0224	0.0539	0.0539	0.0291	0.0219	0.0252	0.0182
	RRL	-	0.0	-	0.0	-	0.0	2.3	24.4	145	199	74.3	**0.5**$^{\blacktriangledown}$	50.9	1.1
	ETP	3.27	3.19	0.006	0.003	2.50	0.24	3.22	2.99	2.90	2.93	3.25	2.83	2.64	2.50
	CS	7105-2734	9779-2524	49979-1	49991-1	23544-582	48234-1	8514-2701	11322-1646	14800-975	16091-1905	8172-2562	17430-380	14890-120	18125-465
	SC	0.015	-0.020	0.028	0.149	0.018	0.053	0.406	0.501	0.025	0.027	0.337	0.476	0.617	**0.684**
	NMI	0.161	0.183	0.036	0.004	0.153	0.156	0.110	0.119	0.162	0.164	0.186	0.112	0.201	0.121
C-100	RL	-	0.0070	-	0.0070	-	0.0070	0.0091	0.0083	0.0369	0.0292	0.0128	0.0058	0.0092	0.0061
	RRL	-	0.0	-	0.0	-	0.0	30	108	427	630	33.3	**17.1**$^{\blacktriangledown}$	4.2$^{\blacktriangledown}$	52.5
	ETP	6.53	6.48	0.940	0.052	6.51	4.38	6.19	4.06	5.12	5.02	6.51	4.02	5.83	3.22
	CS	1160-129	1395-23	38814-1	49834-1	1317-177	13950-11	1299-160	14936-3	2514-101	2613-132	996-156	11191-10	4350-10	11132-10
	SC	0.143	0.195	0.124	0.158	0.144	0.194	0.920	0.896	**0.946**	0.931	0.491	0.912	0.911	0.921
	NMI	0.573	0.628	0.627	0.680	0.475	0.619	0.737	0.736	0.728	0.699	0.451	0.444	0.339	0.437
USPS	RL	-	0.0019	-	0.0019	-	0.0019	0.0029	0.0039	0.0748	0.0617	0.0007	0.0027	0.0013	0.0047
	RRL	-	0.0	-	0.0	-	0.0	52.6	69.6	3837	2582	**40**	42.1	160	104.3
	ETP	3.27	3.23	3.26	3.27	3.10	3.16	3.27	3.27	3.24	3.24	3.29	3.12	2.55	2.99
	CS	284-121	359-89	333-121	328-104	420-53	375-64	284-108	282-107	298-80	314-88	294-156	438-69	947-27	514-46
	SC	0.039	0.079	0.158	0.270	0.051	0.108	0.822	0.854	0.675	0.824	0.431	0.919	**0.923**	0.881
	NMI	0.127	0.152	0.007	0.004	0.106	0.139	0.188	0.164	0.161	0.158	0.065	0.144	0.072	0.107
STL	RL	-	0.0179	-	0.0179	-	0.0179	0.0328	0.0332	0.0593	0.0604	0.0525	0.0354	0.0263	0.0266
	RRL	-	0.0	-	0.0	-	0.0	83.2	91.9	231	249	155	97.8	**27.7**	53.8
	ETP	3.26	3.25	0.069	0.025	2.43	1.4	3.24	3.28	3.22	3.12	2.90	2.48	2.98	2.86
	CS	764-312	830-287	4969-1	4991-1	2586-82	3888-38	849-232	669-328	822-224	870-244	2641-23	2280-27	929-34	1466-69
	SC	-0.019	-0.021	0.037	0.094	-0.026	-0.062	0.386	0.282	0.018	0.035	0.188	0.413	**0.441**	0.377
	NMI	0.412	0.353	0.206	0.132	0.423	0.485	0.333	0.297	0.316	0.273	0.484	0.222	0.466	0.209
CBird	RL	-	0.0055	-	0.0055	-	0.0055	0.0229	0.0139	0.0625	0.0560	0.0377	0.0056	0.0104	0.0039
	RRL	-	0.0	-	0.0	-	0.0	316	286	1036	1455	102	1.8	**44.4**$^{\blacktriangledown}$	8.3
	ETP	6.34	5.59	2.71	0.958	6.56	7.21	5.51	5.03	5.16	4.47	7.43	5.68	7.01	5.06
	CS	131-1	245-1	1722-1	2773-1	101-2	99-2	248-1	297-1	312-1	519-1	35-2	211-1	100-1	701-1
	SC	-0.010	-	0.114	-	-0.002	-	-	-	-	-	0.035	-	**0.673**	-
	NMI	0.398	-	0.012	-	0.383	-	-	-	-	-	0.147	-	0.378	-
R-10k	RL	-	-	-	-	-	-	-	-	-	-	0.0016	-	0.0022	-
	RRL	-	-	-	-	-	-	-	-	-	-	**60**	-	120	-
	ETP	5.13	-	0.072	-	5.10	-	-	-	-	-	5.55	-	4.79	-
	CS	916-20	-	11172-1	-	885-18	-	-	-	-	-	727-56	-	1026-1	-
	SC	-0.021	-	0.114	-	-0.008	-	-	-	-	-	0.099	-	**0.287**	-
	NMI	0.155	-	0.003	-	0.166	-	-	-	-	-	0.018	-	0.180	-
20NG	RL	-	-	-	-	-	-	-	-	-	-	0.0006	-	0.0012	-
	RRL	-	-	-	-	-	-	-	-	-	-	**25**$^{\blacktriangledown}$	-	50	-
	ETP	4.03	-	0.022	-	3.86	-	-	-	-	-	4.31	-	4.19	-
	CS	2217-107	-	18818-1	-	3428-26	-	-	-	-	-	1142-582	-	1809-197	-

C.3 How interpretable are the memories of `DClAM`?

We explore the prototype-based representation of the learned memories in latent space for **`DClAM`** for Fashion-MNIST and USPS in figure 4. For Fashion-MNIST, the 60k images are partitioned into 10 clusters, and the evolution of memories is visualized in figure 4b during the training process outlined in algorithm 1 for **`DClAM`**. In each sub-figure of figure 4b, we observe the evolution over epochs. At epoch 0, there are no distinct memories for clustering; instead, there are pairs of pullover (rows 3 & 5), shirts (rows 7 & 8), and t-shirts/tops (rows 6 & 9). However, discernible patterns emerge at epoch 10, refining further by epoch 20. By epoch 100, all ten memories represent distinct shapes, representing different cluster centroids (explore the additional sub-figures of Fig. 4 to observe the evolution of memories across epochs for **`ClAM`** in the latent space, an **`DClAM`**). Fig. 5 displays 20-closest points for each of the memory of Fashion-MNIST.

Table 7: **Metrices obtained by `DClAM` and baselines corresponding to the best NMI.** The best performance for each dataset is in **boldface**. (note abbreviations DCEC→DC, EDCWRN→EDC, Entropy→ETP, Cluster-size→CS, No-AE→NAE, Conv-AE→CAE, EDCWRN-AE→EAE, Resnet-AE→RAE). '-' denotes NA. $x^{\blacktriangledown}$ indicates negative RRL which means the RL of the method is x% less than the pretrained AE loss.

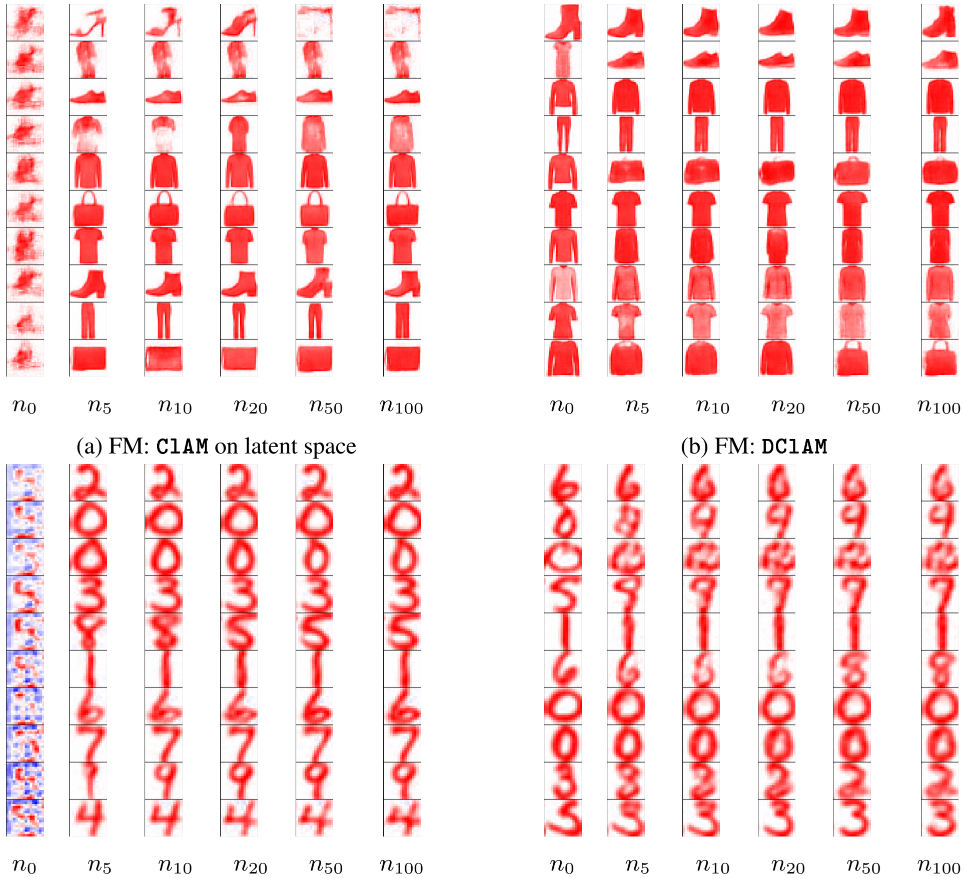
Data	Met	Kmeans		Agglo		`ClAM`		DC		DEKM		EDC	`DClAM`		
		NAE	CAE	NAE	CAE	NAE	CAE	CAE	RAE	CAE	RAE		CAE	EAE	RAE
FM	SC	0.154	0.251	0.109	0.201	0.140	0.262	0.861	0.716	0.819	0.784	0.430	0.817	0.619	0.825
	NMI	0.511	0.643	0.534	0.625	0.525	0.631	0.629	**0.668**	0.586	0.639	0.457	0.610	0.534	0.597
	RL	-	0.0122	-	0.0122	-	0.0122	0.0138	0.0139	0.0574	0.0596	0.0263	0.0406	0.0327	0.0387
	RRL	-	0.0	-	0.0	-	0.0	13.1	67.5	370	618	202	233	276	366
	ETP	3.17	3.17	3.14	3.20	3.13	2.98	3.22	3.20	3.07	3.16	3.00	3.16	3.22	3.18
	CS	9617-2361	11145-2744	11830-1860	10298-2544	14068-2435	15262-2100	10886-3030	9734-2847	12974-1191	11023-2652	17140-1578	11028-2658	10332-3054	10404-2610
C-10	SC	0.050	0.072	0.014	0.020	0.064	0.101	0.118	0.653	0.276	0.262	0.541	0.713	0.632	0.420
	NMI	0.078	0.122	0.071	0.101	0.086	0.105	0.121	0.120	0.116	0.122	0.111	**0.123**	0.114	0.119
	RL	-	0.0220	-	0.0220	-	0.0220	0.0221	0.0245	0.0426	0.0362	0.0291	0.0403	0.0379	0.0326
	RRL	-	0.0	-	0.0	-	0.0	0.5	36.1	93.6	101	74.3	83.2	127	81.1
	ETP	3.27	3.19	3.17	3.02	3.23	2.21	3.07	3.21	3.19	3.11	3.25	3.18	2.98	3.28
	CS	7105-2734	9779-2524	10505-1650	11278-1764	9587-2925	26395-361	11022-3374	10235-1968	10275-2454	13746-2168	8172-2562	8595-2365	10721-289	6843-3144
C-100	SC	0.015	-0.014	-0.018	-0.043	0.018	0.001	0.048	0.002	-0.011	-0.028	0.308	0.354	0.200	0.130
	NMI	0.161	0.183	0.150	0.167	0.153	0.170	0.162	0.179	0.186	0.189	0.186	0.219	0.225	**0.239**
	RL	-	0.0070	-	0.0070	-	0.0070	0.0072	0.0049	0.0112	0.0074	0.0398	0.0257	0.0250	0.0226
	RRL	-	0.0	-	0.0	-	0.0	2.9	22.5	60	85	315	267	160	465
	ETP	6.53	6.48	6.45	6.30	6.51	6.27	6.41	6.41	5.23	6.45	5.51	6.33	6.33	6.36
	CS	1160-129	1395-23	1299-77	2308-17	1317-177	2535-39	1623-14	1380-21	2213-32	1440-60	996-156	1210-5	2105-15	1315-5
USPS	SC	0.143	0.195	0.124	0.159	0.142	0.180	0.920	0.896	0.946	0.465	0.43	0.865	0.660	0.857
	NMI	0.573	0.628	0.627	0.680	0.564	0.640	**0.737**	0.736	0.728	0.701	0.451	0.689	0.583	0.660
	RL	-	0.0019	-	0.0019	-	0.0019	0.0074	0.0039	0.0748	0.0374	0.0006	0.0451	0.0322	0.0409
	RRL	-	0.0	-	0.0	-	0.0	289	69.6	3836	1526	20	2274	6340	1678
	ETP	3.27	3.23	3.26	3.27	3.27	3.21	3.27	3.27	3.24	3.24	3.29	3.11	3.24	3.23
	CS	284-121	359-89	333-121	328-104	290-132	343-73	284-108	282-107	298-80	318-91	294-156	396-35	385-107	308-72
STL	SC	0.039	0.074	0.024	0.021	0.042	0.069	0.822	0.837	0.109	0.079	0.332	0.388	0.597	0.280
	NMI	0.127	0.152	0.121	0.138	0.130	0.169	**0.188**	0.165	0.170	0.166	0.103	0.149	0.151	0.159
	RL	-	0.0179	-	0.0179	-	0.0179	0.0328	0.0362	0.0315	0.0174	0.0433	0.0409	0.0454	0.0364
	RRL	-	0.0	-	0.0	-	0.0	83.2	109	76.0	0.6	110	128	120	110
	ETP	3.26	3.25	3.02	3.02	3.24	2.82	3.24	3.28	3.21	3.20	2.62	3.13	3.18	3.15
	CS	764-312	830-287	1379-205	1373-130	945-317	1212-2	849-232	671-326	807-250	876-264	2173-121	982-46	929-232	938-181
CBird	SC	-0.019	-0.021	-0.018	-0.064	-0.026	-0.062	0.248	0.152	-0.041	-0.038	0.188	0.135	0.068	0.167
	NMI	0.412	0.353	0.469	0.439	0.423	0.485	0.356	0.320	0.364	0.370	0.484	0.421	**0.493**	0.385
	RL	-	0.0055	-	0.0055	-	0.0055	0.0229	0.0152	0.0066	0.0036	0.0377	0.0255	0.0237	0.0249
	RRL	-	0.0	-	0.0	-	0.0	316	322	20	0.0	102	364	26.7	592
	ETP	6.34	5.59	6.97	6.58	6.56	7.21	5.84	5.12	5.71	5.80	7.43	6.48	7.39	6.05
	CS	131-1	245-1	93-1	232-1	101-2	99-2	167-1	570-1	177-1	197-1	35-2	143-1	58-2	180-1
R-10k	SC	-0.010	-	-0.012	-	-0.007	-	-	-	-	-	0.013	-	0.647	-
	NMI	0.398	-	0.404	-	0.394	-	-	-	-	-	0.169	-	**0.414**	-
	RL	-	-	-	-	-	-	-	-	-	-	0.0014	-	0.0020	-
	RRL	-	-	-	-	-	-	-	-	-	-	40	-	100	-
	ETP	5.13	-	5.15	-	5.22	-	-	-	-	-	5.47	-	5.2	-
	CS	916-20	-	845-18	-	650-41	-	-	-	-	-	478-76	-	540-1	-
20NG	SC	-0.021	-	-0.186	-	-0.103	-	-	-	-	-	0.066	-	0.199	-
	NMI	0.155	-	0.167	-	0.176	-	-	-	-	-	0.018	-	**0.229**	-
	RL	-	-	-	-	-	-	-	-	-	-	0.0006	-	0.0012	-
	RRL	-	-	-	-	-	-	-	-	-	-	$25^{\blacktriangledown}$	-	50	-
	ETP	4.03	-	3.64	-	3.77	-	-	-	-	-	4.31	-	3.87	-
	CS	2217-107	-	4024-52	-	4227-103	-	-	-	-	-	1142-582	-	3203-105	-

Table 8: Hyperparameters, their roles and range of values for `DClAM`.

Hyperparameter	Used Values
Inverse temperature, β	$[10^{-5}, ..., 5]$
Number of layers, $T = \tau/dt$	$[5, ..., 25]$
Batch size	$[16, 32, 64, 128, 256]$
Initial learning rate (AM), ϵ_{am}	$[10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}]$
Initial learning rate (AE), ϵ_{ae}	$[10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}]$
Reduce LR by factor	0.8
Reduce LR patience (epochs)	$[5, 10, 15]$
Minimum LR	10^{-6}
Reduce LR loss threshold	10^{-4}
Maximum Number of epochs	300
Latent dimension	Number of true clusters as per dataset
No of cluster, k	Number of true clusters as per dataset
Number of restarts	5

Table 9: Best hyperparameters for different datasets for `DClAM`. '-' denotes NA.

Dataset	Inverse temperature, β			Layers, T			Batch size			Learning rate(AM)			Learning rate (e)			Learning rate (d)		
	CAE	RAE	EAE	CAE	RAE	EAE	CAE	RAE	EAE	CAE	RAE	EAE	CAE	RAE	EAE	CAE	RAE	EAE
FM	0.5	0.09	0.7	15	15	10	64	64	64	0.001	0.001	0.1	0.0000001	0.0000001	0.0000001	0.001	0.001	0.001
C-10	2	0.02	0.5	15	15	12	64	64	64	0.001	0.001	0.01	0.0000001	0.0000001	0.0000001	0.001	0.001	0.001
C-100	1	0.005	5	10	10	10	64	64	64	0.001	0.001	0.001	0.0000001	0.0000001	0.0000001	0.001	0.001	0.001
USPS	0.5	1	1	15	10	15	64	64	32	0.01	0.01	0.1	0.0000001	0.0000001	0.0000001	0.001	0.001	0.01
STL	0.5	0.003	0.1	15	10	12	64	64	128	0.001	0.01	0.1	0.0000001	0.0000001	0.0000001	0.001	0.001	0.0001
CBird	0.05	0.00015	0.005	15	10	15	64	64	64	0.01	0.001	0.1	0.0000001	0.0000001	0.0000001	0.001	0.001	0.001
R-10K	-	-	10	-	-	10	-	-	64	-	-	0.01	-	-	0.0000001	-	-	0.1
20-NG	-	-	1.5	-	-	15	-	-	64	-	-	0.1	-	-	0.0000001	-	-	0.1



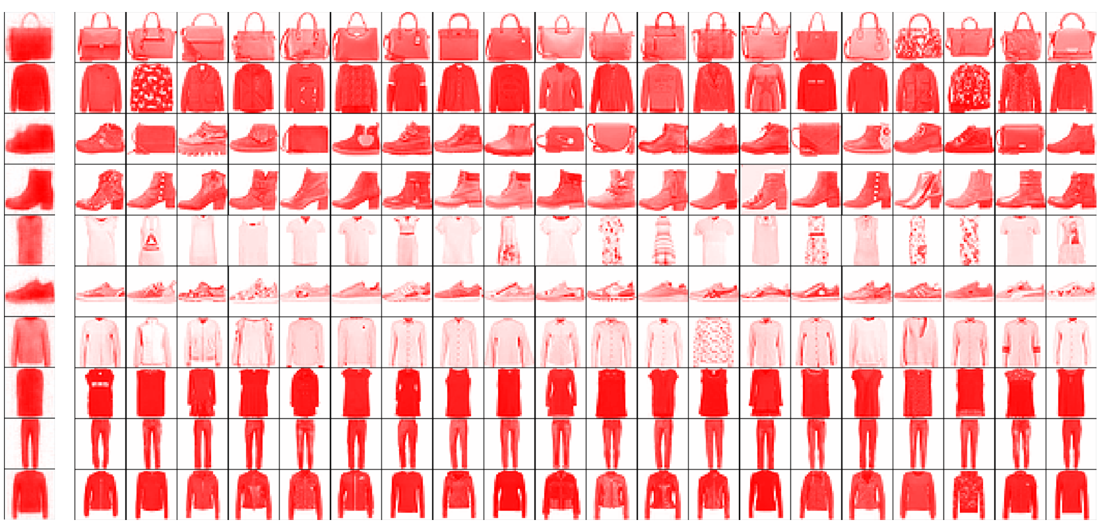
(a) FM: `ClAM` on latent space

(b) FM: `DClAM`

(c) USPS: `ClAM` on latent space

(d) USPS: `DClAM`

Figure 4: **Evolution of prototypes for Fashion-MNIST & USPS in `ClAM` on latent space and `DClAM`.** We visualize the prototypes at the n^{th} training epoch for $n = 0, 5, 10, 20, 50, 100$ (with $T = 10$).



(a) M

(b) 20-closest points

Figure 5: `DClAM`: **Final memories (left column) and the 20-closest points for each memory in F-MNIST.**

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We lay out clearly our contribution in proposing novel formulation of deep clustering using associative memories.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5 discusses limitations.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: This is mostly an empirical paper, but we do discuss the upper bound on the unified loss proposed in `DClAM` in Eq. 8.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: In Appendix B.1 we describe in detail all the methods, (hyper)parameters and training details.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

Justification: All datasets used are open and publicly available. We will also share our code after publication.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Yes, the paper lists the whole methodology for hyperparameter seletion and optimizer used, as well as the dataset details.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [NA]

Justification: We report the best results for all the methods based on the best set of hyperparameters. For this reason there are no error bars or significance testing required.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We detail in section B.1 the experimental platform used.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our work does not require human subjects. All datasets are public and the nature of the evaluation is not dependent on any inherent biases. The work does not have anticipated societal harms or other harmful consequences requiring mitigation.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: We do not anticipate any additional societal harms or other harmful consequences requiring mitigation beyond the well-known risks.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point

out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: No proprietary datasets were used, and the models are not anticipated to have a high risk potential for misuse.

    Guidelines:
    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All the original asset owners are properly cited in this work. No licensed work has been used.

    Guidelines:
    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The paper mainly contributes new clustering algorithm.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Not applicable

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: IRB is no applicable, since there is no human (or animal) study involved.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.