Expert Systems With Applications 238 (2024) 121923

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Review

Unified embedding and clustering

Mebarka Allaoui [a,b,*], Mohammed Lamine Kherfi [a,c], Abdelhakim Cheriet [a,b], Abdelhamid Bouchachia [d]

[a] *Kasdi Merbah University Ouargla, Ouargla, 30000, Algeria*
[b] *LINATI Laboratory, Kasdi Merbah University Ouargla, Ouargla, 30000, Algeria*
[c] *LAMIA Laboratory, University of Quebec in Trois-Rivières, 3351 Boulevard des Forges, C.P. 500, Trois-Rivières, Canada*
[d] *Bournemouth University, Poole BH12 5BB, Bournemouth, UK*

A R T I C L E   I N F O

Keywords:
Manifold embedding
Clustering
Joint optimisation
Deep representation learning

A B S T R A C T

This paper investigates the problem of treating embedding and clustering simultaneously to uncover data structure reliably by constraining manifold embedding through clustering. Conversely, most existing methods perform embedding sequentially, followed by clustering, which leads to a clustering that sometimes pushes data towards a direction induced by embedding. Instead, we perform them simultaneously through an original formulation, which allows for preserving the data's original structure in the embedding space and producing a better clustering assignment. To achieve this goal, we introduce a novel algorithm that unifies manifold embedding and clustering (UEC). The proposed UEC algorithm is based on a bi-objective loss function that combines data embedding and clustering, which is optimised using three different ways: (1) Comma Variant, (2) Plus Variant, and (3) Light Plus Variant. The experimental results with several real-world datasets show that UEC is competitive with the state-of-the-art embedding and clustering methods showing in particular that UEC allows for better preservation of the structure of the dataset resulting in better clustering performance.

1. Introduction

Clustering is a fundamental operation in unsupervised machine learning that has received considerable attention from various research communities. Clustering algorithms can either operate on data in the original space or after projecting it into a new "embedding" space. This projection is usually accompanied by a dimensionality reduction, which can be linear, like PCA (Pearson, 1901), or non-linear such as Isomap (Tenenbaum, De Silva, & Langford, 2000), and UMAP (McInnes, Healy, & Melville, 2018). Isomap (Tenenbaum et al., 2000) is a well-known technique that aims to preserve the global geometric structure through its shortest path distance between all pairs of points on the neighbourhood graph. UMAP (McInnes et al., 2018) is a recent dimension reduction technique that represents the original data by a high-dimensional neighbourhood graph and then generates an equivalent low-dimensional graph such that the cross entropy between the two graphs is minimal. Once embedding is done, different operations can be applied to the resulting transformed data, including clustering. For example, in Allaoui, Kherfi, and Cheriet (2020), they proved that the performance of different clustering algorithms could considerably increase when applied to the embedded data rather than the original one.

The clustering field has taken a new direction with deep learning development. Several algorithms are proposed: Deep Embedding Network for Clustering (DENC) (Huang, Huang, Wang, & Wang, 2014), Deep Embedding for Clustering (DEC) (Xie, Girshick, & Farhadi, 2016), Deep subspace clustering networks (DSCN) (Ji, Zhang, Li, Salzmann, & Reid, 2017), Deep Semantic Embedding and Clustering (DSEC) (Guo, Yuan, Xu, Bai, & Liu, 2020), Joint Unsupervised LEarning (JULE) (Yang, Parikh, & Batra, 2016), DEeP Embedded RegularIsed ClusTering (DEPICT) (Ghasedi Dizaji, Herandi, Deng, Cai, & Huang, 2017), Information Maximising Self-Augmented Training (IMSAT) (Hu, Miyato, Tokui, Matsumoto, & Sugiyama, 2017), Deep Adaptive Clustering (DAC) (Chang, Wang, Meng, Xiang, & Pan, 2017), and Deep Embedded Dimensionality Reduction Clustering (Yan, Hao, Xu, Zhao, & Shen, 2020). These algorithms can be categorised into two families, which are (a) the sequential optimisation family and (b) the interlaced optimisation family.

The first family proceeds: solve the embedding optimisation problem until convergence, then pass the estimated outputs to the clustering optimisation problem. Then, solve the second optimisation problem until convergence, which means two different convergence criteria are

* Corresponding author at: Kasdi Merbah University Ouargla, Ouargla, 30000, Algeria.
E-mail addresses: allaoui.mebarka@univ-ouargla.dz (M. Allaoui), Mohammedlamine.Kherfi@uqtr.ca (M.L. Kherfi), Abdelhakim.cheriet@univ-ouargla.dz (A. Cheriet), abouchachia@bournemouth.ac.uk (A. Bouchachia).

https://doi.org/10.1016/j.eswa.2023.121923
Received 21 December 2022; Received in revised form 26 September 2023; Accepted 28 September 2023
Available online 11 October 2023

met sequentially. To this family belong DENC (Huang et al., 2014), DEC (Xie et al., 2016), DSCN (Ji et al., 2017), DSEC (Guo et al., 2020), which rely on an encoding network (after the pre-training step of an autoencoder). However, removing the decoding part could lead to incorrect training results, which means that the encoder could produce random discriminative features. The reconstruction capability of autoencoders captures prominent features without forcing any randomness, which could mitigate the effect of generating random features (Mrabah, Khan, Ksantini, & Lachiri, 2020).

The second family of algorithms proceeds in what is known as the alternating minimisation approach, which has one convergence criterion only. The autoencoder reconstruction and clustering loss are combined and optimised in such algorithms. The interlaced family takes one gradient step for the first optimisation problem, then passes the new parameters to the second optimisation problem to take one gradient step for the second optimisation problem. This process is repeated alternatively between the two optimisation problems until convergence. To this family belong DEPICT (Ghasedi Dizaji et al., 2017), DAC (Chang et al., 2017), JULE (Yang et al., 2016), IMSAT (Hu et al., 2017), DERC (Yan et al., 2020).

However, the inevitable trade-off between clustering and reconstruction makes integrating the two difficult. In contrast to the clustering loss function, which aims to reduce non-discriminative features, the reconstruction loss function allows the recovery of all features.

In contrast to the families mentioned above, The embedding and clustering loss functions are optimised simultaneously in one step by UEC. It is motivated by the fact that jointly learning the embedding and clustering manifold improves clustering quality (Allaoui et al., 2020; McConville, Santos-Rodriguez, Piechocki, & Craddock, 2019; Yan et al., 2020). To start the optimisation process that boosts the quality of embedding and clustering, UEC initialises the low dimensional space and the cluster assignments.

However, this optimisation process can be done differently, yielding different versions of the UEC algorithm. Here, we propose three different variants. In the first variant, called **Comma** variant, the algorithm alternates between optimising the embedding loss and the clustering loss functions. In the second and the third variants, called **Plus** and **Light Plus** variants, respectively, the optimisation is done in one combined step. The difference between the second and third variants lies in calculating the derivative (please see Section 5). The Light Plus variant is computationally faster than the Plus variant, but the latter is more accurate than the Light Plus variant.

The contributions of this paper are:

- The proposal of a novel manifold-based learning algorithm that simultaneously learns the embedding space and clustering of the data. The joint optimisation of both the embedding and clustering objective functions leads to better preservation of the similarities in the original data and clusterability in the embedding space.
- The theoretical derivations of the optimisation process are associated with the proposed UEC variants.
- Extensive evaluation against state-of-the-art clustering methods to show the superiority of UEC.

The rest of this paper is organised as follows: Section 2 reviews related work. Section 3 describes the proposed UEC algorithm, and Section 4 gives more details about the ingredients of the algorithm. Section 5 presents the details of the UEC's optimisation variants. Section 6 discusses the experimental results before concluding the paper in Section 7.

2. Related work

Dimensionality reduction (DR) techniques have been used to map data into a lower dimensional space. According to Huang, Wu, and Ye (2019), DR methods can be categorised into several families. Among them, we have projection, manifold embedding, and deep learning methods.

Principal Component Analysis (PCA) (Pearson, 1901) is a projection method that transforms the data into a new feature space. However, PCA does not perform well in cases where relationships are non-linear.

Manifold Embedding (ME) methods can focus on local or global structures and overcome the shortcomings of linearity. ME examples include Locally Linear Embedding (LLE) (Roweis & Saul, 2000), Multidimensional Scaling (MDS), Isomap (Tenenbaum et al., 2000), and the most recent algorithm UMAP (McInnes et al., 2018). These techniques aim to preserve the similarity between datapoints in the low dimensional space by learning from the relationships between the original data points to discover the underlying structure. These techniques aim to capture the most pertinent information. However, discriminative information loss compromises the clustering performance.

In deep DR methods, various autoencoder architectures (Hinton & Salakhutdinov, 2006; Ranzato, Huang, Boureau, & LeCun, 2007) and CNN architectures (He, Zhang, Ren, & Sun, 2016; Radford, Metz, & Chintala, 2015) have been proposed showing significant improvements in many computer vision tasks (Girshick, 2015; Girshick, Donahue, Darrell, & Malik, 2014; Krizhevsky, Sutskever, & Hinton, 2012; Russakovsky et al., 2015; Simonyan & Zisserman, 2014). Deep DR methods can deal with large-scale datasets. However, they need high computational power and ample memory space.

Clustering has been extensively studied in the literature, resulting in many algorithms. Those clustering algorithms can be categorised into seven families depending on their conceptual idea (Xu & Tian, 2015): (1) Partitioning clustering such as the well-known K-means (MacQueen et al., 1967; Wang, Wang, Ke, Zeng and Li, 2015), (2) Distribution clustering as GMM (Bishop, 2006), (3) Hierarchical clustering like the agglomerative clustering (Gowda & Krishna, 1978), and (4) Density-based clustering such as DBSCAN (Campello, Moulavi, & Sander, 2013; Wang, Wu, Tang and Hor, 2015). Despite their widespread use, these techniques are not always effective and have a significant computational complexity when working with huge datasets.

Three other families have not been mentioned, and they are (5) Manifold clustering, (6) Deep manifold clustering, and (7) Deep clustering. (5) Manifold clustering (Ng, Jordan, Weiss, et al., 2002; Shi & Malik, 2000) combines discriminative DR with clustering. Manifold clustering techniques calculate the similarity matrix to map the non-linear data to low-dimensional manifolds. On the other hand, spectral clustering is a popular manifold clustering approach that, under mild conditions, can be seen as the equivalent of kernelised K-Means (MacQueen et al., 1967). Another manifold-based method, called Sparse Manifold Clustering and Embedding (SMCE) (Elhamifar & Vidal, 2011), connects each datapoint to its small neighbourhood with appropriate weights. However, while this approach leads to better embedding, it does not improve the clustering performance (Patel & Vidal, 2014). (6) Deep manifold clustering such as Spectral-Net (Shaham et al., 2018) is a CNN that maps the data onto the eigenspace of the graph Laplacian matrix. Then, the Siamese network learns the weights of the connections between the graph's nodes, which is passed to k-means as the final clustering step. However, manifold clustering techniques suffer from a vast computational time when dealing with large-scale datasets. The representational capability of such techniques is also constrained, and the similarity matrix's discriminative power typically underfits the semanticity of natural data (Mrabah et al., 2020).

The last family is Deep clustering. Whereas Recent clustering research investigated the use of deep learning for both DR and clustering, known as deep clustering. Aljalbout, Golkov, Siddiqui, Strobel, and Cremers (2018) and Min et al. (2018). We divide the deep clustering methods into two classes according to how they optimise both loss functions: sequential and interlaced deep algorithms.

The deep sequential methods first perform the optimisation of the embedding loss and then that of the clustering one. To this class belongs Deep Embedding and Clustering (DEC) (Xie et al., 2016), which trains an autoencoder to learn features and imposes a soft assignment constraint on these features. The self-learning strategy of DEC could mislead the learning process by producing non-representative features of data clustering (Mrabah et al., 2020). Deep Semantic Embedding and Clustering (DSEC) (Guo et al., 2020) is quite similar to DEC. The significant contribution of DSEC is concatenating the original feature and semantic space of training data after the pre-training step.

The main drawback of the sequential deep clustering methods is that discarding the decoder part makes less discrimination in the latent space. The Autoencoder-based clustering techniques retain the reconstruction loss to mitigate the impact of the random discriminative features during the clustering phase.

In the interlaced deep algorithms family, optimisation of the embedding and clustering objective functions is performed alternatively until a convergence criterion is met.

DEeP Embedded RegularIsed ClusTering (DEPICT) (Ghasedi Dizaji et al., 2017) consists of a convolutional autoencoder and a single clustering layer, which learns the latent features and the distribution of the cluster assignments. DEPICT is optimised by minimising the reconstruction error and the relative entropy between the distributions of the cluster assignments and their priors. Deep Embedded Dimensionality Reduction Clustering (DERC) (Yan et al., 2020) is a combination of a deep autoencoder and the t-SNE embedding method (Van der Maaten & Hinton, 2008) to learn data representation. Centres of the clusters are initialised using GMM, and a probability-based triplet loss measure is used to retrain the model and improve the clustering performance of the model.

Joint Unsupervised LEarning (JULE) (Yang et al., 2016) uses a CNN with an agglomerative clustering loss. In every iteration, hierarchical clustering is performed on the forward pass using an affinity measure, while data representation is optimised on the backward pass of the CNN. However, the limitations of JULE are the computational and memory complexity issues associated with the computation of the affinity matrix constructed by the agglomerative clustering loss function. Information Maximising Self-Augmented Training (IMSAT) (Hu et al., 2017) is based on data augmentation, where the net is trained to maximise the mutual information between data and predicted clusters. The network is regularised so that the cluster assignment of the original data will be consistent with the assignment of augmented data.

Deep Adaptive Image Clustering (DAC) (Chang et al., 2017) generates the label of the points using a CNN. Then, it calculates the cosine similarities between points based on the label of points. Those points are sampled as pairs of points according to the cosine similarities. The CNN is trained using the chosen samples based on the developed binary pairwise classification model.

The combination of clustering and reconstruction in the interlaced model loss can cause a trade-off between eliminating the non-discriminative details and preserving all information. In contrast to these methods, the proposed UEC optimises a multi-objective function that controls embedding and clustering in one combined step.

3. Unification of manifold embedding and clustering

Before discussing the details of the proposed method, the list of symbols to be used in the rest of this paper is introduced (see Table 1).

UEC is an unsupervised non-linear technique that aims to preserve the closeness in the input space when mapping the data into the output space. Thus, datapoints with similar characteristics are projected nearby, and dissimilar ones are mapped apart. As general manifold embedding techniques, UEC considers the original data as a high-dimensional graph to be transformed into a lower-dimensional one.

UEC is about mapping high D-dimensional input data onto a d-dimensional embedded space while considering clustering constraints,

Table 1
Description of all used symbols.

Symbols	Description
Y	Input datapoints: $Y = \{y\}_{i=1}^{n}, y_i \in R^f$
Z	The representation of the data in the embedding space: $Z = \{z\}_{i=1}^{n}, z_i \in R^d$
M	The set of cluster centres: $M = \{\mu\}_{k=1}^{C}$, where C is the number of clusters
Γ_i	Local neighbourhood of y_i datapoint
$p_{i\|j}$	Probability distribution between the input datapoints y_i and their jth nearest neighbour. These form the matrix: $P = \{p_{ij}\}$ to be defined later
q_{ij}	The probability distribution between the embedded datapoint z_i and its jth nearest neighbour: $Q = \{q_{ij}\}$
S_{ik}	The Soft assignment distribution that indicates the probability between the points z_i and the cluster centre μ_k: $S = \{s_{ik}\}$
T_{ik}	Auxiliary target distribution: $T = \{T_{ik}\}$
CE	Binary cross entropy representing the adopted embedding loss function
KL	Kullback–Leibler divergence representing the adopted clustering objective function
F	Total loss as a Weighted sum function of both CE and KL

Algorithm 1 UEC

```
Input: Dataset $Y = \{y_i\}_{i=1}^{n}$, number of cluster $C$, dimension of the embedding space $d$, min-dist, number of nearest neighbours $nn$
Output: Embedded dataset $Z = \{z_i\}_{i=1}^{n}$, set of centres $M = \{\mu_k\}_{k=1}^{C}$.

1: Construct the graph of the original data
2: Compute the affinity matrix for the input data $Y$ Eq. (3) and (5).
3: Initialise the embedding space of dimension $d$.
4: Initialise the cluster centres.
5: while The convergence criterion is not met do
6:    Compute the affinity matrix for the embedded data $Z$, Eq. (6).
7:    Compute the soft assignment of the embedded data to the clusters $\mathcal{S}(Z, M)$, Eq. (7).
8:    Compute the probabilistic point-to-cluster assignments using the obtained soft assignment $\mathcal{T}(Z, M)$, Eq. (8).
9:    Update the coordinates of the embedded data $Z$, as in Eq. (14).
10:   Update the centres, as in Eq. (12).
11: end while
```

where $d \ll D$. To achieve that, UEC optimises the following objective function:

$$F = \alpha \; CE \oplus \beta \; KL \tag{1}$$

The first term (CE) is the cross-entropy that assesses the quality of the embedding in the low-dimensional space. It is referred to as *the embedding loss function*. The second term (KL) is the Kullback–Leibler divergence and is used to assess the clustering quality. It is referred to as *the clustering loss function*. The quantities α and β define the relative weights of the overall function's two-loss component and control the trade-off between embedding and clustering.

We can optimise this function in two different ways. Each of them follows an interpretation of the $\oplus$ symbol:

1. $\oplus$ = ',' (comma) UEC takes one gradient step for the embedding optimisation problem, passes the new parameters to the clustering optimisation problem, takes one gradient step for the second optimisation problem, then passes the estimated parameters to the first optimisation problem again, and repeats this alternating sequence. This scenario is referred to as a *sequential update*.
2. $\oplus$ = '+' (plus) that indicates that the evaluation of Eq. (1) is done in one combined step (multi-objective function) to update

the sought quantities simultaneously. This scenario is referred to as *joint update*.

Eq. (1) depends on several quantities (matrices), namely P, Q, T and S described in Table 1. The matrix P forms the affinity scores between the individual datapoints and their nearest neighbours in the input space. The Q matrix represents the similarity between the embedding space's datapoints. The matrix S is a soft assignment of embedded data to clusters. At the same time, T represents the probabilistic point-to-cluster assignments using the obtained soft assignment S (The formal definition of these matrices will follow below). Eq. (1) can be rewritten as follows:

$$F(P, Q, T, S) = \alpha\ CE(P, Q) \oplus \beta\ KL(T \parallel S) \tag{2}$$

CE computes the total entropy between P and Q, while KL measures the relative entropy (divergence) between S and T. This objective function will be used to analytically compute the coordinates of the set Z (coordinates of the data in the embedded space) and the centres M. Before delving into the details, the UEC algorithm is presented in Alg. 1.

4. Formulation of the algorithm ingredients

4.1. Computation of the affinity matrix P

The affinity matrix P represents the similarity scores between pairs of datapoints using the exponential probability:

$$p_{i|j} = \exp(-\frac{d(y_i, y_j) - \rho_i}{\sigma_i}) \tag{3}$$

Given an input hyper-parameter *nn*, which represents the number of neighbours, $d(y_i, y_j)$ is the distance between the ith datapoint and its jth nearest neighbour. ρ_i is defined by:

$$\rho_i = min\{d(y_i, y_j) \mid 1 \leq j \leq nn, d(y_i, y_j) > 0\} \tag{4}$$

Which represents the distance between ith datapoint and its first nearest neighbour. It ensures the local connectivity of the manifold. For each y_i, σ_i is defined by:

$$\sum_{j=1}^{nn} exp\left(\frac{-max(0, d(y_i, y_{i_j}) - \rho_i)}{\sigma_i}\right) = log_2(nn)$$

We use a symmetrisation of the high-dimensional probability since the weight of the edge between the ith and jth nodes is not equal to the weight between jth and ith nodes. The final formulation of P is given as follows:

$$p_{ij} = p_{i|j} + p_{j|i} - p_{i|j}p_{j|i} \tag{5}$$

We define the number of Nearest Neighbour as nn, a hyper-parameter, and its default value is 15. We utilise the Nearest-Neighbour-Descent (k-NN) algorithm (Dong, Moses, & Li, 2011) to represent the underlying structure of the data and to construct local neighbourhood Γ of point y_i. Therefore local neighbourhood is $\Gamma_i = \{y_j | y_j \in knn(y_i), y_i \in knn(y_j)\}$. However, the resulting k-NN graph often contains disconnected components and potentially isolated vertices. Isolated vertices violate the assumption that the underlying manifold is locally connected, and disconnected components negatively impact the initialisation of the low dimensional space. Inspired from Dalmia and Sia (2021), we refine the graph construction and avoid the problem of the isolated vertices by increasing the connectivity of the k-NN graph using The maximum spanning tree (MST) (Ozaki, Shimbo, Komachi, & Matsumoto, 2011).

4.2. Initialisation

We use a spectral layout (Luo, Wilson, & Hancock, 2003) to initialise the embedding. We use the standard spectral method to compute the eigenvectors of the graph's Laplacian matrix. This provides both faster convergence and more excellent stability of the algorithm. To initialise the centres of the clusters, we use a centroid-based algorithm (e.g., k-means, GMM, etc.).

4.3. Computation of Q

The affinity matrix Q represents the similarity scores between each embedded datapoint and its neighbours. It is computed using a smooth approximation of the membership strength:

$$q_{ij} = (1 + a\|z_i - z_j\|_2^{2b})^{-1} \tag{6}$$

The parameters a and b are defined using the piece-wise non-linear least-square fitting $\psi : \mathbb{R}_d \times \mathbb{R}_d \to [0, 1]$ where

$$\psi(z_i, z_j) = \begin{cases} 1 & if \|z_i - z_j\|_2 \leq min - dist \\ \exp(-(\|z_i - z_j\|_2 - min - dist)) & \text{otherwise} \end{cases}$$

where $min - dist$ is a hyper-parameter representing the desired separation between close points in the embedding space.

4.4. Computation of the soft assignments S

The matrix S is computed using a smooth approximation of the membership strength between the embedded points z_i and the cluster centres μ_k as follows:

$$S_{ik} = (1 + a\|z_i - \mu_k\|_2^{2b})^{-1} \tag{7}$$

where a and b are the same ones used in Eq. (6). We could also consider the following form like in DEC (Xie et al., 2016) (which is inspired from t-SNE Van der Maaten & Hinton, 2008):

$$S_{ik} = \frac{(1 + a\|z_i - \mu_k\|_2^{2b})^{-1}}{\sum_k (1 + a\|z_i - \mu_k\|_2^{2b})^{-1}}$$

However, it has been shown in UMAP (McInnes et al., 2018) that the normalisation increases processing time without improving accuracy.

4.5. Computation of the auxiliary target distribution T

The matrix T should satisfy three constraints: (1) improving the cluster purity, (2) ensuring high-confidence assignments to get more emphasis, and (3) preventing large clusters from distorting the embedding space by normalising the loss contribution of each centre. A formulation that addresses these constraints is given as follows:

$$T_{ik} = \frac{S_{ik} / \sum_l S_{lk}}{\sum_m (S_{lm} / \sum_l S_{lm})} \tag{8}$$

The numerator of T_{ik} ensures that the first condition is met. Normalising the membership of a datapoint by the sum of the memberships of other datapoints in the same cluster makes this point have a hard assignment (closer to 0 or 1). The denominator guarantees the third constraint, which enforces our preference for having balanced assignments and preventing large clusters from distorting the embedding space. Points that have hard assignments are involved in the process of finding the assignment of stray points.

4.6. Formulation of the optimisation problem

The coordinates of the datapoints and the centres of the clusters are updated in light of the minimisation of the two objective functions: the embedding and the clustering loss functions. These functions are coupled in Eq. (2). Before discussing the two optimisation options (see Section 3) presented earlier, we formulate the first term, which is the embedding objective loss represented as binary cross-entropy (CE). CE is given as follows:

$$CE(P, Q) = \sum_i \sum_j [p_{ij} \log(\frac{p_{ij}}{q_{ij}}) + (1 - p_{ij}) \log(\frac{1 - p_{ij}}{1 - q_{ij}})] \tag{9}$$

where P and Q are the probabilistic similarity scores of the input data Y and that of the embedded data Z respectively. The second term is the clustering objective function, which is defined as the Kullback–Leibler (KL) divergence function between the soft assignments S and the auxiliary target distribution T:

$$KL(T \parallel S) = \sum_i \sum_k T_{ik} \log \frac{T_{ik}}{S_{ik}} \tag{10}$$

Due to the relative entropy of the KL divergence function and the ability to learn from high-confidence assignments and auxiliary target distribution, it is used to refine clusters. The combination of cross entropy and KL divergence loss functions makes our algorithm capable of capturing the local and global data structure. The terms and all quantities used in the objective function have been defined. Now, We discuss the optimisation problem to update the coordinates of Z and the clusters' centres.

5. Optimisation of the objective function

The coordinates of a datapoint z_i and cluster centres μ_j are updated at each time step t until the criterion convergence parameter is met. Due to the rapid convergence and low memory consumption of Stochastic Gradient Descent (SGD), we use it as an optimisation algorithm. As illustrated in Alg. 1, the optimisation steps are repeated until the change in the cluster assignment of the points is less than a threshold $1e-5$.

5.1. Comma (sequential) variant

Here, the terms of the objective function are sequentially evaluated.

5.1.1. Update of the embedding

The data coordinates in the embedded space will be updated twice in a sequential manner. The first update stems from the embedding loss and is given as follows:

$$z_i(t+1) = z_i(t) - \eta \frac{\delta CE}{\delta z_i}$$

where η is a learning rate. The quantity $\frac{\delta CE}{\delta z_i}$ is expressed as follows:

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[\frac{2bp_{ij}}{1/(a\|z_i - z_j\|_2^{2(b-1)}) + \|z_i - z_j\|_2^2} - \frac{2b(1-p_{ij})}{\|z_i - z_j\|_2^2(1 + a\|z_i - z_j\|_2^{2b})} \right] \|z_i - z_j\| \tag{11}$$

For more details, please see Appendix A. While the second update of the coordinates of z_i comes from the clustering loss.

$$z_i(t+1) = z_i(t) - \eta \frac{\delta KL}{\delta z_i}$$

The partial derivative of the clustering loss function (KL) given by Eq. (10) w.r.t. z_i as follows:

$$\frac{\partial KL}{\partial z_i} = \sum_k \frac{2ab\|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}}$$

5.1.2. Update of the cluster centres

The centres μ_k does not depend on the embedding loss function. Hence, the centres of the clusters are updated using the derivative of the clustering loss function, the KL-divergence (Eq. (10)), as follows:

$$\mu_k(t+1) = \mu_k(t) - \eta \frac{\delta KL}{\delta \mu_k} \tag{12}$$

where η is the learning rate, and $\frac{\delta KL}{\delta \mu_k}$ is as follow:

$$\frac{\delta KL}{\delta \mu_k} = \sum_i - \frac{2ab\|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}} \tag{13}$$

5.2. Plus (combined) variant

The main difference between the combined and the sequential variants is that in the former, the clustering influences the computation of z_i and μ_j.

5.2.1. Update of the embedding

According to this second variant, the coordinates of the datapoints z_i are updated using the embedding and the clustering loss functions simultaneously.

From Eq. (1), the update is executed as follows:

$$z_i(t+1) = z_i(t) - \eta \frac{\partial F}{\partial z_i} \tag{14}$$

$$= z_i(t) - \eta \left(\alpha \frac{\delta CE}{\delta z_i} + \beta \frac{\delta KL}{\delta z_i} \right) \tag{15}$$

where $\frac{\delta CE}{\delta z_i}$ is given by Eq. (11) and $\frac{\delta KL}{\delta z_i}$ is given as follows:

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i}(1 + \log \frac{T_{ik}}{S_{ik}}) + \frac{2bT_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2} \right] + \sum_{i' \neq i} \sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1 + \log \frac{T_{i'k}}{S_{i'k}}) \right] \tag{16}$$

The derivative of KL is the sum of 2 terms since the derivation of T_{ik} with respect to z_i is computed according to two cases: (1) When $i' = i$, (2) When $i' \neq i$ (please see Appendix B for more details).

The final formulation of the update is obtained by combining Eqs. (11) and (16):

$$\frac{\partial F}{\partial z_i} = \alpha \frac{\partial CE}{\partial z_i} + \beta \frac{\partial KL}{\partial z_i} \tag{17}$$

5.2.2. Update of the cluster centres

Since CE does not depend on μ_k, only the clustering loss function, KL is relevant:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i \left[\frac{\partial T_{ik}}{\partial \mu_k}(1 + \log \frac{T_{ik}}{S_{ik}}) - \frac{2ab\|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}} \right] + \sum_i \sum_{k' \neq k} \left[\frac{\partial T_{ik'}}{\partial \mu_k}(1 + \log \frac{T_{ik'}}{S_{ik'}}) \right]$$

The derivative of the KL divergence is the sum of two parts requiring the consideration of two cases: (1) When $k' = k$, and (2) When $k' \neq k$. For more details, please see Appendix B.

5.3. Light plus variant

In the *combined variant*, we consider that the update of z_i and μ_k depends on the auxiliary target distribution, T. While this variant significantly improves the proposed UEC's performance, it is not highly efficient in terms of computational time due to the heavy computation involved in the gradient descent update. Hence, we proposed the light version of the *combined variant*. The underlying assumption of this third variant is to consider T_{ik} not dependent on z_i and μ_k. More details are provided in Appendix C.

5.3.1. Update of the embedding

KL divergence is derived with respect to z_i:

$$\frac{\partial KL}{\partial z_i} = \sum_k \frac{2ab\|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}} \tag{18}$$

By substituting the derivative of CE (Eq. (11)) and the derivative of KL divergence (Eq. (18)) in the total loss function (Eq. (17)), we then obtain the final update of $z_i(t+1)$ as shown in Eq. (15).

5.3.2. Update of the cluster centres

The derivative of total loss F (Eq. (1)) with respect to μ_k is computed only for the KL-divergence function, which depends on the cluster centroids. The formulation obtained in the sequential variant derivations will be applied here for updating the centres (See Eqs. (12) and (13)).

5.4. Implementation

Following the work of Dong et al. (2011) and McInnes et al. (2018), the practical implementation of this algorithm requires k-nearest neighbour calculation and efficient optimisation via stochastic gradient descent. As mentioned above, we used the Nearest-Neighbour-Descent algorithm of Dong et al. (2011) to obtain the optimal number of the nearest neighbours. The authors of the Nearest-Neighbour-Descent algorithm report an empirical complexity of $O(N^{1.14})$. In optimising the embedding under the provided objective function, we follow the work of McInnes et al. (2018). This gives a very efficient approximate stochastic gradient descent algorithm since there is no normalisation requirement. From what was mentioned above, the overall complexity is approximately $O(N^{1.14})$.

6. Experiments and discussion

This section will show the performance of the proposed 3 variants of UEC on a set of benchmark datasets. Specifically, we will discuss the following experiments:

- The first experiment evaluates the three variants and compare their performance in term of external validation, internal validation and computational time, see Section 6.2.
- In the second experiment, we study the effect of the number of clusters and the initialisation of the clusters' centres on the performance of UEC variants, see Section 6.3.
- In the third experiment, we study the sensitivity of UEC towards the initialisation of embedding space, see Section 6.4.
- In the fourth experiment, we assess the ability of UEC to Preserve the Local and Global structure of the data, see Section 6.5.
- In the fifth experiment, we discuss the parameters α and β in Eq. (1) and their effect, see Section 6.6.
- In the sixth experiment, we compare the performance of UEC variants against the state-of-the-art manifold and deep clustering methods, see Section 6.7.
- In the seventh experiment, we evaluate the performance of UEC variants challenging datasets, see Section 6.8.
- Qualitative results are presented in the last section, see Section 6.9

6.1. Experiment requirement

6.1.1. Datasets

We conduct experiments on six benchmark datasets given as follows:

1. **IRIS**: is a dataset of 150 instances with 4 dimensions categorised into 3 different classes, where each class refers to a type of iris plant.
2. **Spiral**: consists of 300 points with 2 coordination which forms three spirals.
3. **Atom** consists of two clusters in 3-dimensional space. The first cluster consists of a dense core of 400 points surrounded by a well-separated 400 points from the second cluster (Ultsch, 2005).
4. **EngyTime** is a dataset that contains 4096 points with two variables, "Engy" and "Time", as a two-dimensional mixture of Gaussians. These points belong to two clusters (Baggenstoss, 2002).
5. **USPS**: consists of 9298 images belonging to 10 different classes. Each image is a 16×16 grey image (Hull, 1994).
6. **MNIST**: consists of 10 handwritten digits with 70,000 images. Each image is a 28×28 grey image (LeCun, Bottou, Bengio, & Haffner, 1998).
7. **CIFAR10**: is a dataset of 60 000 samples with 10 classes, where each sample is a 32×32 RGB image (Krizhevsky, Hinton, et al., 2009).
8. **Reuters 10K**: is English news dataset (Lewis, Yang, Russell-Rose, & Li, 2004) consisting of 10 700 documents. Similar to Xie et al. (2016), we use four (4) categories/classes and discard all documents labelled by multiple root categories. We removed stop words and used the tf-idf representation of the 2000 most frequent words.

6.1.2. Evaluation metrics

Accuracy. We evaluate all clustering methods with clustering ACCuracy (ACC), which is defined as the best match between the ground truth and predicted labels:

$$ACC = max_m \frac{\sum_{i=1}^{n} 1\{GT_i = m(PL_i)\}}{n}$$

where GT_i and PL_i are the ground truth and predicted label of example z_i respectively. All conceivable one-to-one mappings between clusters and labels are covered by the range m. The Hungarian algorithm can effectively calculate the optimal mapping (Kuhn, 1955).

Normalised Mutual Information (NMI). is a normalisation of the mutual information score to scale the results between 0, which means no mutual information, and 1 represents the perfect correlation. NMI formula is given by:

$$NMI = \frac{2I(GT_i, PL_i)}{[H(GT_i) + H(PL_i)]}$$

where GT_i is the ground truth label, PL_i is the predicted label, H(.) is the Entropy and $I(GT_i, PL_i)$ is the Mutual Information between GT_i and PL_i.

Internal validation measures. We use three well-known internal validation indices, which are Davis–Bouldin (DB) (Davies & Bouldin, 1979), Silhouette coefficient (S) (Rousseeuw, 1987) and Calinski–Harabaz (CH) (Caliński & Harabasz, 1974). The values of the DB index are in $[0, +\infty]$, and lower values imply better clustering. The values for the S index are in $[-1, 1]$ where values close to 1 and -1 indicate better and worse results, respectively. The values of the CH index are in $[0, +\infty]$, where higher values indicate better clustering.

6.1.3. Implementation details

In the optimisation step of UEC, our weighted sum function is minimised using Stochastic Gradient Descent (SDG). We set the learning rate to 1.0 and then decreased it by 10 every 10 epochs. The number of epochs is set to 200 for big datasets and 500 for small ones.

Remark. All UEC source codes will be made publicly available as soon as the paper has been accepted.

6.2. Comparison of the variants

In this experiment, We evaluate the performance of each variant of the UEC (*comma variant*, *plus variant*, *light plus variant*) in terms of three aspects: (1) External validation, (2) internal validation, and (3) computational time.

We used the accuracy and NMI measures as external validation measures to evaluate the performance of UEC's three variants. Through Table 2, We can see that the clustering performance of *Plus variant* is better than the other two versions. However, Table 3 shows that *Comma variant* and *Light plus variant* are better than *Plus variant* in the

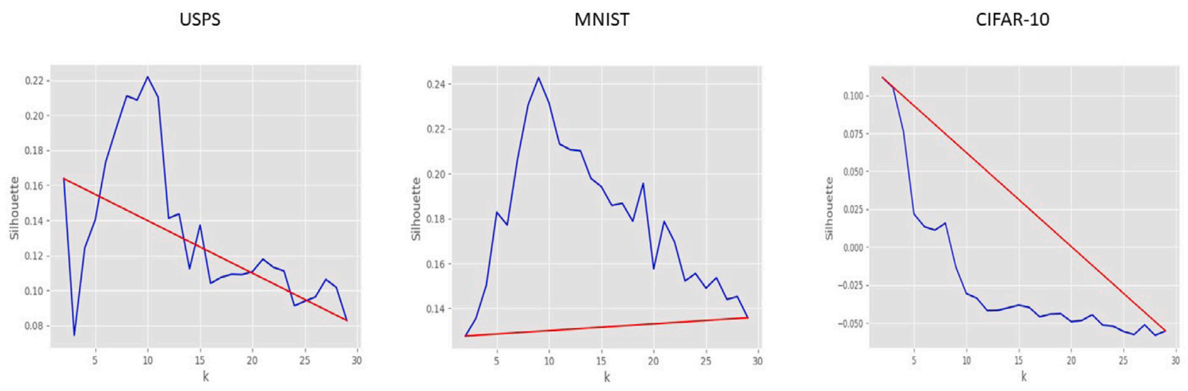

Fig. 1. The effect of changing the number of clusters measured by Silhouette score.

Table 2
Evaluating the performance of the UEC's three variants in Accuracy (ACC) and NMI.

Models	Dataset			
	USPS		MNIST	
	ACC	NMI	ACC	NMI
Comma variant	0.967	0.928	0.959	0.932
Plus variant	0.982	0.948	0.988	0.956
Light plus variant	0.979	0.937	0.986	0.950

Table 3
Evaluating the performance of the UEC's three variants in terms of Execution time (in seconds).

Models	Dataset	
	USPS	MNIST
Comma variant	70	150
Plus variant	180	1226
Light plus variant	47	115

Table 4
Clustering quality using the Davies–Bouldin index.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
Comma variant	1.676	1.593	5.148	0.701
Plus variant	1.581	1.494	4.264	0.559
Light plus variant	1.634	1.555	4.721	0.623

Table 5
Clustering quality using the Silhouette index.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
Comma variant	0.201	0.219	−0.001	0.315
Plus variant	0.278	0.282	0.009	0.495
Light plus variant	0.222	0.231	0.006	0.417

Table 6
Clustering quality using the Calinski–Harabaz index.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
Comma variant	1123.754	9682.256	997.84	16 481.130
Plus variant	1398.425	10 189.580	1289.475	20 142.254
Light plus variant	1211.068	9819.213	1179.962	18 248.961

execution time as expected given its high computational requirements involving many equations as shown in Appendix B. *Comma variant* and *Light plus variant* are approximately near to each other in the execution time. However, *Light plus variant* is better than *Comma variant* in the clustering performance. Therefore, these results support our claim that the joint optimisation of embedding and the clustering loss functions improves the clustering performance of our algorithm.

Internal validation intends to quantify clustering quality, usually using two criteria: *Compactness* and *Separation*. The first criterion measures the data objects' similarity in the individual clusters. The second criterion measures how distinct or well-separated clusters are from each other. Often, these two criteria are embedded in various clustering quality indices. In this study, we use three internal validation indices, which are Davis–Bouldin (DB) (Davies & Bouldin, 1979), Silhouette coefficient (S) (Rousseeuw, 1987), and Calinski–Harabaz (CH) (Caliński & Harabasz, 1974).

Tables 4, 5, and 6 show the clustering quality scores of the UEC variants. They indicate that the *Plus variant* is the best among the three variants across the three indices. The UEC variants obtain approximately the same DB and S results on MNIST and USPS, presumably because of the similar nature of these two datasets. The results on CIFAR-10 are less competitive, maybe because of the poor quality of the images leading to cluster overlapping.

On the other hand, the three variants perform well on the Reuters dataset using each of the three indices. In general, the UEC variants can preserve the between and within-cluster distances because they are designed to preserve the global and local structures of the data.

Remark. Based on the results achieved by the three variants of UEC, we use only the *Light plus variant* in the upcoming experiments (in Sections 6.3, 6.4, 6.5, 6.6, and 6.9), since *Light plus variant* is the fastest one among the three versions and its clustering performance is near to *Plus variant.*

6.3. Effect of the number of clusters

In this experiment, we study two aspects. First, We analyse the sensitivity of the UEC algorithm towards changing the number of clusters and then study the effect of different initialisation of the clusters' centres.

To study The effect of changing the number of clusters on the performance of UEC on three datasets: USPS, MNIST, and CIFAR10. We use the Average Silhouette method to evaluate the performance of UEC for different values of the number of clusters k in a range of $[2, 30]$. We observe that the best number of clusters for the three datasets is 10 according to the Silhouette method, see Fig. 1. We made one more step to be sure that 10 is the optimal number of clusters for the three datasets. We evaluate UEC on the same range using Accuracy and NMI metrics. Fig. 2 shows that the best performance of UEC on the three datasets is 10 clusters. We can observe that the performance of UEC is coherent and consistent in both experiments.

The second part of this experiment is to study the sensitivity of the UEC algorithm towards the initialisation of the clusters' centres. We perform three types of initialisation using k-means, Gaussian mixture

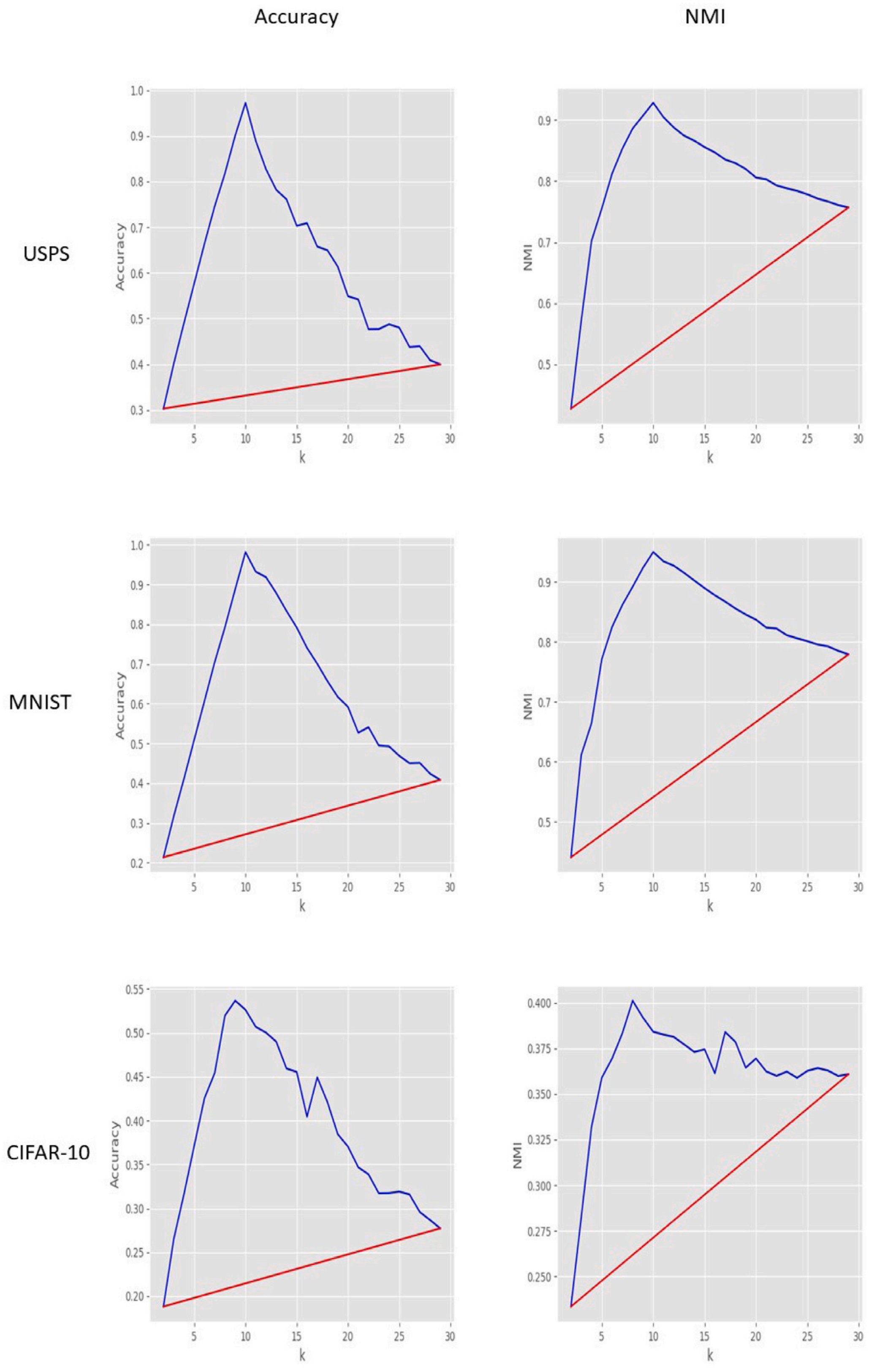

Fig. 2. The effect of changing the number of clusters measured by Accuracy and NMI.

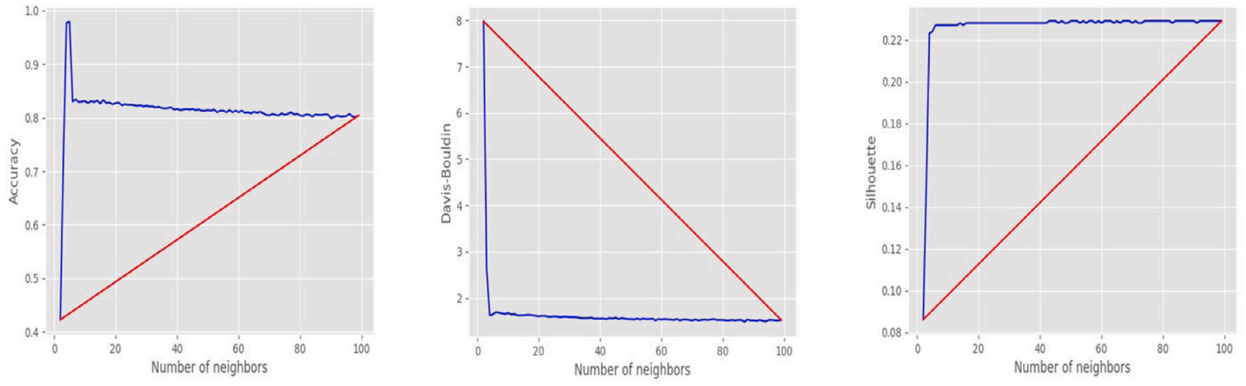

Fig. 3. Preservation of the local structure using k-NN graph.

Table 7
Effect of centre initialisation on the performance.

Algorithm	Dataset			
initialisation	USPS	MNIST	CIFAR-10	Reuters
Random	0.975 ± 0.004	0.981 ± 0.005	0.515 ± 0.01	0.955 ± 0.002
K-means	0.979 ± 0.002	0.986 ± 0.003	0.526 ± 0.002	0.962 ± 0.001
GMM	0.980 ± 0.003	0.985 ± 0.002	0.524 ± 0.002	0.961 ± 0.002

models (GMM) and randomly chosen medoids. The experiments are conducted using *Light plus variant* on the considered datasets (see Section 6.2). Table 7 shows our algorithm's performance using the three initialisations evaluated by the accuracy measure. We can see that the UEC algorithm is not sensitive to the type of initialisation adopted (even with the random centroid initialisation, it achieves good results).

Notice: *Based on the achieved results, in the upcoming experiments, we use k-means to initialise the centres.*

6.4. Embedding space initialisation

In this experiment, we analyse the sensitivity of the UEC algorithm towards the initialisation of the embedding space. We perform two types of initialisation using spectral embedding algorithm and random initialisation. The experiments are conducted using *Light plus variant* on the considered datasets. Table 8 shows the performance of our algorithm using both initialisations evaluated by the accuracy measure. We can see that the UEC algorithm is not sensitive to the type of initialisation adopted (even with random initialisation, it achieves good results).

6.5. Preserving the local and the global structures of the data

In the first part of this experiment, we assess the ability of k-NN graphs to capture the local structure and study the effect of the number of neighbours nn. In addition, this experiment is important to decide the best value of nn for the upcoming experiments. To do so, we conduct an extensive sensitivity analysis experiment, varying the values of nn in a range $[2, 100]$ and evaluating the performance of UEC using three measures: two are internal validation metrics, namely Davis–Bouldin and Silhouette coefficient and the third one is an external metric which is the accuracy. The Internal metrics assess how well the data points within the same cluster are similar to each other and how distinct different clusters are from each other. Internal measures indicate how well the local structure of data is preserved. The external metrics provide insights into the extent to which obtained clusters fit known or expected patterns so that external measures are a good indicator of how well the global structure of the data is maintained.

In this experiment, we consider the USPS dataset to obtain the results shown in Fig. 3. The best performance in terms of accuracy is obtained when the number of neighbours is set to 4 and 5. The internal measures, on the other hand, in the last two sub-figures show that the local structure is well captured since nn after a certain threshold value, 4, does not affect the internal measures becoming stable.

In the upcoming experiments, we fixed nn for the small datasets to be 5 and for the large datasets to be 30 as the default value.

In the second part of this experiment, we will study how the cross entropy (CE) function can maintain the global structure of the data. First, let us observe the following cases to understand the behaviour of CE:

- If the distances between points in both the high-dimensional space and the low-dimensional space are small, then CE must be low or non-existent.
- If the distances between points in both the high-dimensional space and the low-dimensional space are large, then CE must be low or non-existent.
- If the distances between points in the high-dimensional space are small (resp. large) and the distances in the low-dimensional space are large (resp. small), then CE must be large.

On the other hand, from the definition of CE, Eq. (9):

$$CE(P,Q) = \sum_i \sum_j [P(X)\log(\frac{P(X)}{Q(Y)}) + (1-P(X))\log(\frac{1-P(X)}{1-Q(Y)})]$$

where X is the distance between the data points in the high-dimensional space and Y is the distance between the data points in the low-dimensional space. Considering the USPS dataset, X and Y are plotted in Fig. 4 along with the CE loss between them. Moreover, when the X distances are large, and the Y distances are small, CE is large (meaning that far apart points in the high dimensional projected far to each other in the low dimensional space). In conclusion, the $CE(X,Y)$ function is able to preserve both local and global distances.

6.6. Effect of α and β

Recall that α is related to the embedding loss, while β is related to clustering. To study the effect of the parameters α and β on the loss function, we vary them in the unit interval $[0,1]$ and observe their effect on the UEC's performance and execution time. The experiments are conducted using *Light plus variant* following the remark in Section 6.2. To perform such analysis, two experiments are designed as follows:

1. **Experiment 1:** we study the effect of the parameters on the gradient magnitude order of both embedding and clustering loss functions with respect to z_i (more details about the magnitude of both gradients are presented in Appendix D) by varying α and β in $[0,1]$. Using the USPS dataset, the accuracy and NMI results are presented in Tables 9 and 10, respectively. For computational reasons, we have considered only USPS due to its size. The algorithm achieves better

Table 8
Effect of the embedding space initialisation on the performance.

Algorithm initialisation	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
Random initialisation	0.976 ± 0.03	0.984 ± 0.02	0.520 ± 0.04	0.949 ± 0.01
Spectral embedding	0.979 ± 0.02	0.986 ± 0.01	0.526 ± 0.02	0.962 ± 0.005

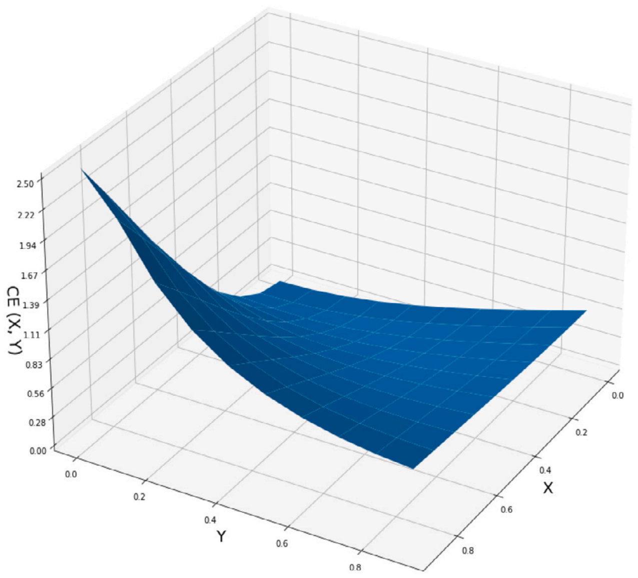

Fig. 4. Preservation of the global structure using CE function.

Table 9
Effect of α and β on the performance of the algorithm measured by accuracy.

β	α					
	0	0.2	0.4	0.6	0.8	1
0	0.615	0.951	0.953	0.956	0.958	0.961
0.2	0.958	0.961	0.962	0.964	0.965	0.967
0.4	0.961	0.963	0.965	0.967	0.969	0.970
0.6	0.963	0.966	0.968	0.970	0.971	0.973
0.8	0.965	0.968	0.971	0.973	0.975	0.976
1	0.972	0.973	0.974	0.976	0.977	0.979

Table 10
Effect of α and β on the performance of the algorithm measured by NMI.

β	α					
	0	0.2	0.4	0.6	0.8	1
0	0.575	0.904	0.908	0.912	0.916	0.919
0.2	0.910	0.913	0.916	0.919	0.921	0.923
0.4	0.914	0.917	0.920	0.923	0.925	0.927
0.6	0.918	0.921	0.923	0.926	0.928	0.930
0.8	0.921	0.924	0.927	0.929	0.932	0.934
1	0.923	0.926	0.929	0.932	0.935	0.937

results when α and β both go to 1 and worst results when they go to 0 (since only the centroids move). If we allow datapoints to move, the model makes a quantum leap in its performance as we see in Table 9 comparing the outcome when the parameters are set to 0 and 0.2, the accuracy increases from 0.615 to 0.951. As β increases, the performance improves, but as α increases, the performance remains relatively constant. We can also observe that the execution time increases when α decreases (because the algorithm needs more epochs to converge).

2. **Experiment 2:** We study the effect of varying α and β on a modified gradient magnitude order of both embedding and clustering loss

Table 11
Effect of different values of α and β on the algorithm's performance measured by the accuracy.

β	α					
	0	0.2	0.4	0.6	0.8	1
0	0.594	0.944	0.948	0.951	0.954	0.957
0.2	0.956	0.957	0.958	0.960	0.961	0.962
0.4	0.960	0.962	0.963	0.964	0.965	0.966
0.6	0.963	0.966	0.967	0.968	0.969	0.970
0.8	0.967	0.968	0.970	0.971	0.972	0.973
1	0.969	0.971	0.972	0.973	0.975	0.976

Table 12
Effect of different values of α and β on the performance of the algorithm measured by the NMI.

β	α					
	0	0.2	0.4	0.6	0.8	1
0	0.571	0.900	0.904	0.907	0.912	0.915
0.2	0.912	0.914	0.916	0.918	0.920	0.922
0.4	0.915	0.917	0.919	0.921	0.924	0.926
0.6	0.918	0.920	0.922	0.924	0.927	0.929
0.8	0.921	0.923	0.925	0.927	0.929	0.931
1	0.924	0.926	0.928	0.930	0.932	0.933

functions. This experiment aims to show if we clip the values of the gradients of the loss function, this modification could improve the algorithm's performance.
The gradient values of the embedding loss are found to be in $]-20, 20[$, while those of the clustering loss are in $]0, 20[$. However, most of the values for the gradient of embedding and clustering loss functions are in $[-4, 4]$ and $[0, 1]$, respectively. We, therefore, clip the values of the embedding loss gradient to $[-4, 4]$, and we divide the values by 4 to bring it to the interval $[-1, 1]$. The values of the clustering loss gradient are clipped to $[0, 1]$. Tables 11 and 12 show the accuracy and NMI of the algorithm after the clipping transformation. We observe that the algorithm performance in this experiment is lower than in the **Experiment 1**.

6.7. Comparative study

6.7.1. Baseline methods

The proposed UEC is compared with a set of deep and manifold-based clustering methods including state-of-the-art deep clustering methods: k-means (MacQueen et al., 1967), deep embedded clustering (DEC) (Xie et al., 2016), Joint Unsupervised Learning (JULE) (Yang et al., 2016), Deep Embedded Regularised Clustering (DEPICT) (Ghasedi Dizaji et al., 2017), Deep Adaptive Clustering (DAC) (Chang et al., 2017), Information Maximising Self-Augmented Training (IMSAT) (Hu et al., 2017). Spectral Net (Shaham et al., 2018), Deep Embedded Dimensionality Reduction Clustering (DERC) (Yan et al., 2020), Dynamic Autoencoder (DynAE) (Mrabah et al., 2020), and Generalised Clustering and Multi-manifold Learning (GCML) (Wu et al., 2022). For these methods, the performance results are taken from the original publications.

6.7.2. Experiment results

Tables 13 and 14 outline the performance in terms of accuracy and NMI, respectively, where the top three accuracy scores are highlighted. Table 13 shows that both variants of UEC (Plus and Light Plus

Table 13
UEC vs. other baselines: accuracy scores.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
k-means	0.668	0.572	0.228	0.524
DEC	0.619	0.843	0.301	0.722
JULE	0.950	0.964	0.271	–
IMSAT	0.976	0.984	0.456	0.719
DEPICT	0.964	0.965	0.342	–
DAC	0.972	0.978	0.522	–
Spectral Net	0.965	0.971	0.322	0.803
DERC	0.977	0.975	0.374	–
DynAE	**0.981**	**0.987**	**0.530**	–
GCML	0.958	0.980	0.431	**0.836**
UEC (Plus variant)	**0.982**	**0.988**	**0.534**	**0.965**
UEC (Light plus variant)	**0.979**	**0.986**	**0.526**	**0.962**

Table 14
UEC vs. other baselines: NMI scores.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
k-means	0.450	0.499	0.087	0.497
DEC	0.586	0.816	0.256	0.703
JULE	0.913	0.913	0.192	–
IMSAT	0.945	0.953	0.431	0.594
DEPICT	0.927	0.917	0.306	–
DAC	0.928	0.935	0.395	–
Spectral Net	0.914	0.924	0.288	**0.674**
DERC	**0.942**	0.927	0.316	–
DynAE	**0.948**	**0.964**	**0.403**	–
GCML	0.902	0.946	0.374	0.590
UEC (Plus variant)	**0.948**	**0.956**	**0.412**	**0.894**
UEC (Light plus variant)	0.937	**0.950**	**0.398**	**0.879**

Table 15
Evaluating the performance of our three variants against the other techniques in terms of Execution time (in seconds).

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
k-means	12	112	152	15
DEC	53	693	1035	80
JULE	2540	12 500	21 250	–
IMSAT	1160	4675	9687	1856
DEPICT	1778	9561	13 570	–
DAC	1690	9670	12 280	–
Spectral Net	6480	11 430	19 430	9720
DERC	3247	10 195	17 400	–
DynAE	7910	10 808	26 270	–
GCML	8040	15 000	32 500	7880
UEC (Plus variant)	180	1226	3065	160
UEC (Light plus variant)	47	115	965	40

variant) are competitive with other algorithms across all benchmarks. It outperforms most of the deep clustering methods by a significant margin. UEC does not need any fine-tuning, in contrast to the deep clustering models, which require tweaking several hyper-parameters and fine-tuning to achieve better results. This advantage makes UEC significantly a better embedding-and-clustering choice than the other alternative clustering models. In addition, JULE, DEPICT, and DAC techniques are designed for image datasets, so these techniques cannot be performed on other datasets, such as Reuters. In contrast, UEC can be applied to any dataset. In Table 15, we compare UEC to several deep clustering methods regarding run-time. Based on our comparison, we can observe that the execution time of our model is lower than the execution time of the deep clustering techniques except for DEC and k-means.

6.8. *Evaluation on challenging datasets*

The objective of this experiment is to evaluate the performance of UEC on challenging datasets, such as spiral, IRIS, EngyTime, and Atom, compared to a set of clustering algorithms. Those datasets address specific challenges to clustering algorithms, such as lack of linear separability, different or small internal class spacing, classes defined by data density rather than data spacing, no cluster structure, outliers, or touching classes (Thrun, 2018; Thrun & Ultsch, 2020). Through Table 16, it can be seen we compared UEC with conventional algorithms in this table. There is no Deep Clustering (DC) algorithm because it is known that these techniques require large datasets to be able to train the DC techniques on these datasets. Data augmentation can be done, but DC techniques still fall into the problem of over-fitting because DC techniques have large numbers of hidden features whose values are radically under-determined by small data. The ability to cluster small datasets is a plus point in favour of UEC against deep clustering algorithms. In addition, through Table 16, we can observe that UEC outperform most of the compared algorithms with a significant margin in both datasets. In addition, we observe that the performance of UEC is competitive to HDBSCAN on the Atom dataset, which means both techniques performs well on dataset defined by data density. UEC has successfully overcome the mentioned challenges posed by these datasets.

Fig. 5 represents 2D visualisation of IRIS, Spiral, Engytime and Atom datasets using UEC versus original visualisation. We note that UEC always seeks to maintain the similarity between the points so that the original space's close points are mapped to each other in the embedding space, and the diverging points fall far from each other. We can also notice that UEC can separate overlapped clusters from each other well, as the case in IRIS and Engytime datasets.

6.9. *Qualitative results*

The discriminative ability of UEC can be illustrated in Fig. 6. The figure represents a 2D visualisation of USPS, MNIST, and CIFAR-10 datasets using UEC, ISOMAP, and t-SNE. The visualisation is coloured according to the ground truth label, where each colour represents one particular class. Compared to ISOMAP and t-SNE, we note that UEC is able to preserve the local and global structure of the data very well. We can observe in USPS and MNIST datasets how UEC achieves both inter/intra-clusters where the clusters are projected based on their similarity, e.g. the clusters of numbers '5' and '6', '6' and '8', '4' and '9' mapped close to each other. The clusters with low similarity projected far apart from each other, such as '1' and '0', '6' and '4', etc. The visualisation of the CIFAR-10 dataset is pretty satisfactory in spite of it being a challenging dataset because, within each class in the CIFAR-10 dataset, there can be a large amount of variation in the appearance of images.

7. Conclusion

In this paper, We have proposed a new algorithm called UEC, which jointly optimises the representation and clustering of data. UEC is a manifold-based clustering algorithm that can preserve data's local and global structure and seeks to learn the manifold within the embedding space. It comes with three variants that resulted from the optimisation process: *Comma variant, Plus variant,* and *Light plus variant*. The empirical results obtained through performance and sensitivity analysis have shown the high effectiveness of UEC across large-scale benchmarks and against several baseline algorithms.

In Future work, we will work on optimising the multi-objective function to improve the algorithm's performance in terms of results and computational efficiency. In addition, we could investigate the effect of different embedding methods on the clustering task performance.

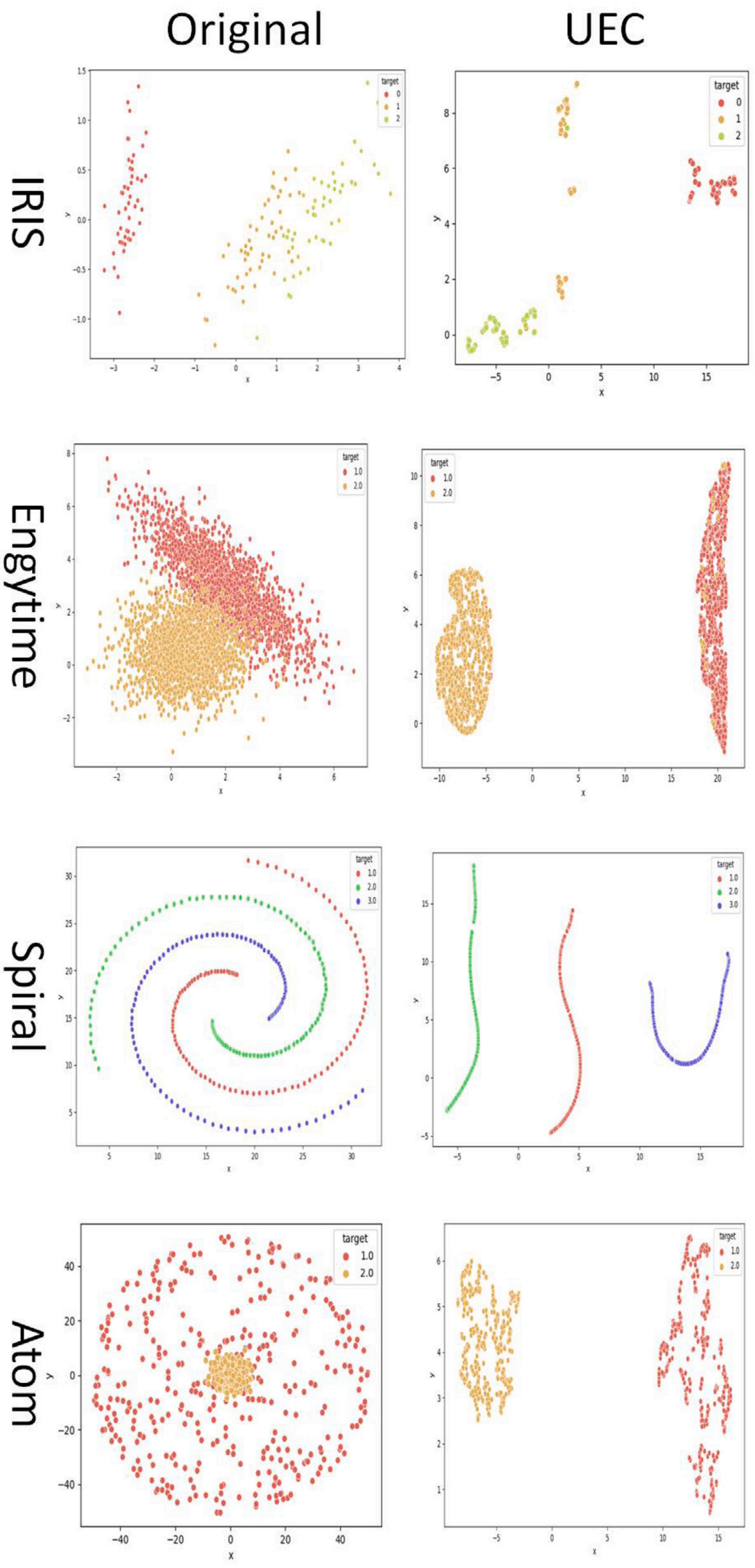

Fig. 5. Visualisation of challenging datasets: Original vs. The UEC visualisation.

Table 16
Evaluating the performance of UEC's variants against the other techniques in terms of Accuracy (ACC) and NMI.

Models	Dataset							
	Spiral		IRIS		EngyTime		Atom	
	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
K-means	0.523	0.218	0.893	0.758	0.951	0.729	0.717	0.298
GMM	0.473	0.202	0.900	0.777	0.951	0.794	0.642	0.200
Spectral clustering	0.486	0.298	0.906	0.805	0.962	0.776	0.501	0.047
Agglomerative	0.570	0.219	0.893	0.770	0.923	0.646	0.657	0.219
HDBSCAN	0.406	0.129	0.906	0.713	0.575	0.385	1.0	1.0
UEC (Plus variant)	0.829	0.753	0.946	0.845	0.987	0.904	1.0	1.0
UEC (Light plus variant)	0.794	0.733	0.934	0.827	0.982	0.895	1.0	1.0

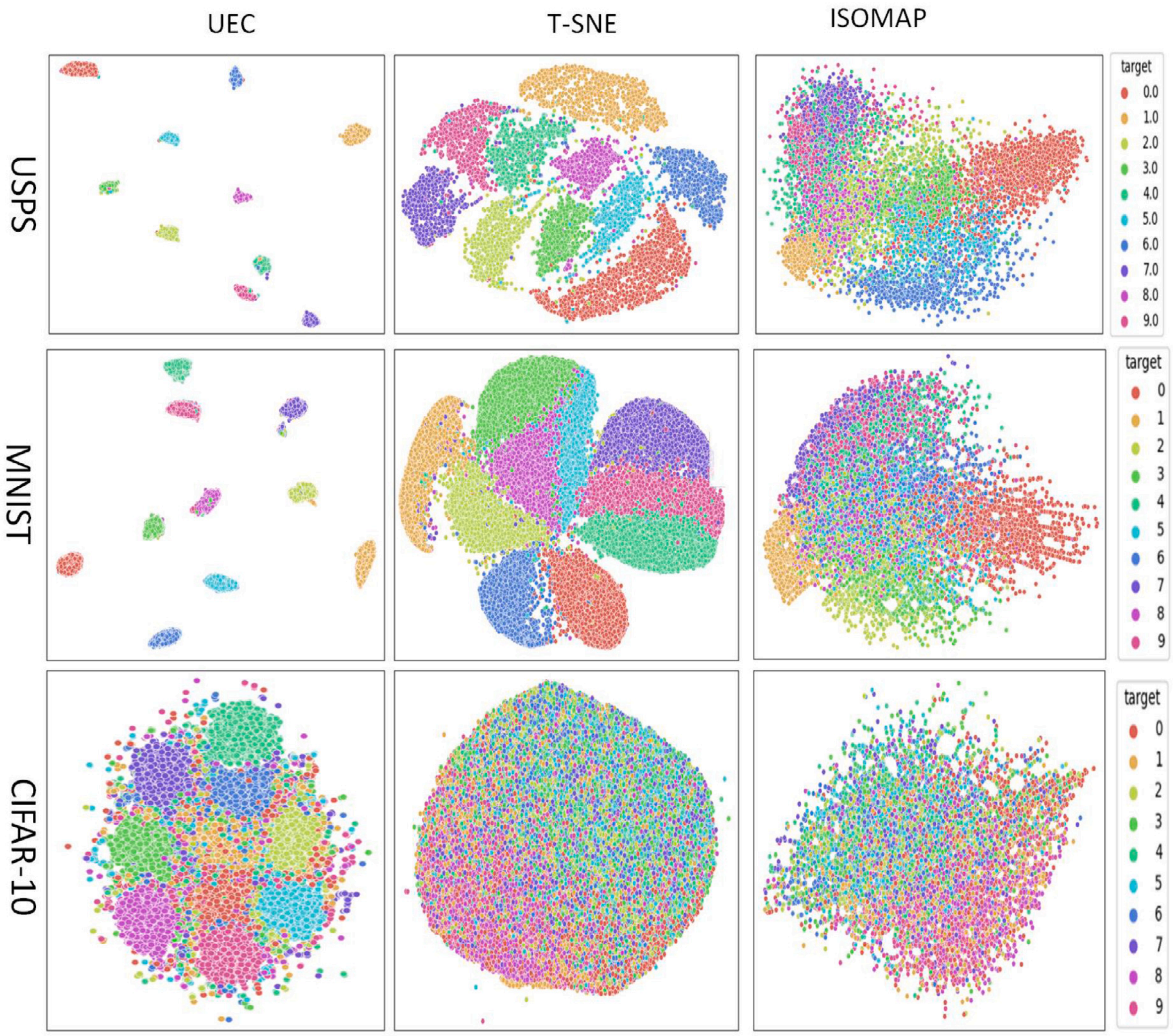

Fig. 6. The visualisation of USPS MNIST, and CIFAR-10 datasets using the UEC against t-SNE and ISOMAP.

CRediT authorship contribution statement

Mebarka Allaoui: Conceptualization, Software, Validation, Data curation, Formal analysis, Writing – original draft. **Mohammed Lamine Kherfi:** Conceptualization, Methodology, Formal analysis, Writing – review & editing. **Abdelhakim Cheriet:** Software, Validation, Formal analysis, Writing – review & editing. **Abdelhamid Bouchachia:** Conceptualization, Methodology, Formal analysis, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The used datasets are publicly published.

Acknowledgements

A. Bouchachia is funded by the EC through the project “ExtremeXP” under Grant Agreement No. 101093164.

Appendix A. The derivation of the embedding objective function (Cross-Entropy)

The Cross-Entropy function depends only on z_i, so in this part, we derive the CE with respect to z_i. For The partial derivative of the CE with respect to z_i, we used the same derivative of UMAP.

Let us first compute the derivative of q_{ij} and $1 - q_{ij}$ we have:

$$q_{ij} = (1 + a\|z_i - z_j\|_2^{2b})^{-1} \tag{A.1}$$

$$\frac{\partial q_{ij}}{\partial z_i} = \frac{-2ab\|z_i - z_j\|_2^{2b-1}}{(1 + a\|z_i - z_j\|_2^{2b})^2} \tag{A.2}$$

Now, we can derive the CE objective function as follows:

$$CE(P,Q) = \sum_i \sum_j \left[p_{ij} \log(\frac{p_{ij}}{q_{ij}}) + (1-p_{ij}) \log(\frac{1-p_{ij}}{1-q_{ij}}) \right]$$

$$= \sum_i \sum_j \left[p_{ij} \log(p_{ij}) - p_{ij}\log(q_{ij}) + (1-p_{ij})\log(1-p_{ij}) \right.$$
$$\left. -(1-p_{ij})\log(1-q_{ij}) \right]$$

Since p_{ij} does not depend on z_i, the derivatives of $p_{ij}\log(p_{ij})$ and $(1-p_{ij})\log(1-p_{ij})$ are zero, so we can derive the CE as follow:

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[-\frac{\partial q_{ij}}{\partial z_i}\frac{p_{ij}}{q_{ij}} - \frac{\partial(1-q_{ij})}{\partial z_i}\frac{1-p_{ij}}{1-q_{ij}} \right] \tag{A.3}$$

We have $1 - q_{ij} = \frac{a\|z_i - z_j\|_2^{2b}}{1+a\|z_i - z_j\|_2^{2b}}$, now we will substitute the last one, Eqs. (A.1) and (A.2) in the Eq. (A.3) so we can rewrite it as follow:

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[-\frac{-2ab\|z_i - z_j\|_2^{2b-1}}{(1 + a\|z_i - z_j\|_2^{2b})^2}\frac{p_{ij}}{(1 + a\|z_i - z_j\|_2^{2b})^{-1}} \right.$$
$$\left. -\frac{2ab\|z_i - z_j\|_2^{2b-1}}{(1 + a\|z_i - z_j\|_2^{2b})^2}\frac{1-p_{ij}}{\frac{a\|z_i - z_j\|_2^{2b}}{1+a\|z_i - z_j\|_2^{2b}}} \right]$$

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[\frac{2ab\|z_i - z_j\|_2^{2(b-1)}p_{ij}}{1 + a\|z_i - z_j\|_2^{2b}} - \frac{2b(1-p_{ij})}{\|z_i - z_j\|_2^2(1 + a\|z_i - z_j\|_2^{2b})} \right]$$
$$\times \|z_i - z_j\|$$

Since the exponential power is taking a long calculation time, we can reduce this term $\frac{2ab\|z_i - z_j\|_2^{2(b-1)}}{1+a\|z_i - z_j\|_2^{2(b)}}$ as shown below:

$$\frac{2ab\|z_i - z_j\|_2^{2(b-1)}}{1 + a\|z_i - z_j\|_2^{2(b)}} = \frac{2b}{1/a\|z_i - z_j\|_2^{2(b-1)} + \|z_i - z_j\|_2^2}$$

Now, we can write the derivative of the embedding loss function as follows:

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[\frac{2bp_{ij}}{1/a\|z_i - z_j\|_2^{2(b-1)} + \|z_i - z_j\|_2^2} \right.$$
$$\left. -\frac{2b(1-p_{ij})}{\|z_i - z_j\|_2^2(1 + a\|z_i - z_j\|_2^{2b})} \right] \|z_i - z_j\| \tag{A.4}$$

Appendix B. Plus variant derivations

We describe how the clustering loss function is derived considering the auxiliary target distribution T_{ik}, which depends on z_i and μ_k. First let us recall that S_{ik} and T_{ik} are expressed as follows:

$$S_{ik} = (1 + a\|z_i - \mu_k\|_2^{2b})^{-1} \tag{B.1}$$

$$T_{ik} = \frac{S_{ik}/\sum_l S_{lk}}{\sum_m (S_{im}/\sum_l S_{lm})} \tag{B.2}$$

The derivative of the objective function F with respect to z_i:

$$\frac{\partial F}{\partial z_i} = \alpha\frac{\partial CE}{\partial z_i} + \beta\frac{\partial KL}{\partial z_i} \tag{B.3}$$

The derivation of the cross entropy is given in Appendix A, while the derivation of the KL divergence with respect to z_i is given in the following.

$$KL(T \parallel S) = \sum_i \sum_k \left[T_{ik}\log T_{ik} - T_{ik}\log S_{ik} \right]$$

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i}\log T_{ik} + T_{ik}\frac{\frac{\partial T_{ik}}{\partial z_i}}{T_{ik}} - \frac{\partial T_{ik}}{\partial z_i}\log S_{ik} - \frac{\partial S_{ik}}{\partial z_i}\frac{T_{ik}}{S_{ik}} \right]$$

$$+ \sum_{i' \neq i}\sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}\log T_{i'k} + T_{i'k}\frac{\frac{\partial T_{i'k}}{\partial z_i}}{T_{i'k}} - \frac{\partial T_{i'k}}{\partial z_i}\log S_{i'k} - \frac{\partial S_{i'k}}{\partial z_i}\frac{T_{i'k}}{S_{i'k}} \right]$$

The derivative is formulated as a sum of two parts since T_{ik} needs to compute its derivative with respect to z_i according to two cases. The first one is when $i' = i$, and the second case is when $i' \neq i$. Since i' is different from i, $S_{i'k}$ does not depend on z_i (see Eq. (B.1)), $\frac{\partial S_{i'k}}{\partial z_i} = 0$. We can simplify the derivative as follows:

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i}(1 + \log\frac{T_{ik}}{S_{ik}}) - \frac{\partial S_{ik}}{\partial z_i}\frac{T_{ik}}{S_{ik}} \right] + \sum_{i' \neq i}\sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1 + \log\frac{T_{i'k}}{S_{i'k}}) \right] \tag{B.4}$$

The derivative of S_{ik} with respect to z_i is:

$$\frac{\partial S_{ik}}{\partial z_i} = -\frac{2ab\|z_i - \mu_k\|_2^{2b-1}}{(1 + a\|z_i - \mu_k\|_2^{2b})^2}$$

We can write $\frac{\partial S_{ik}}{\partial z_i}$ as follows:

$$\frac{\partial S_{ik}}{\partial z_i} = -\frac{2b}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2}S_{ik} \tag{B.5}$$

To derive T_{ik}, we need to distinguish two cases. The first one corresponds to $i' = i$ (written as T_{ik}), and the second case corresponds to $i' \neq i$ (written as $T_{i'k}$). The expression of T_{ik} and $T_{i'k}$ is the same as in Eq. (B.2).

$$T_{ik} = \frac{S_{ik}/\sum_l S_{lk}}{\sum_m (S_{im}/\sum_l S_{lm})}$$

$$T_{i'k} = \frac{S_{i'k}/\sum_l S_{lk}}{\sum_m (S_{i'm}/\sum_l S_{lm})}$$

Let us show the derivation of T_{ik} with respect to z_i by considering the numerator and the denominator separately. The numerator is given as:

$$N_{ik} = \frac{S_{ik}}{\sum_l S_{lk}} \tag{B.6}$$

and its derivative is:

$$\frac{\partial N_{ik}}{\partial z_i} = \frac{\frac{\partial S_{ik}}{\partial z_i}\sum_l S_{lk} - S_{ik}\frac{\partial \sum_l S_{lk}}{\partial z_i}}{(\sum_l S_{lk})^2}$$

Since S_{lk} depends on z_i in the only case where $l = i$, $\frac{\partial \sum_l S_{lk}}{\partial z_i} = \sum_l \frac{\partial S_{lk}}{\partial z_i}$, where $\frac{\partial S_{lk}}{\partial z_i} = \frac{\partial S_{ik}}{\partial z_i}$ only when $l = i$ and $\frac{\partial S_{lk}}{\partial z_i} = 0$, where $l \neq i$. Now we can write $\frac{\partial N_{ik}}{\partial z_i}$ as follows:

$$\frac{\partial N_{ik}}{\partial z_i} = \frac{\frac{\partial S_{ik}}{\partial z_i} \sum_{l \neq i} S_{lk}}{(\sum_l S_{lk})^2} \tag{B.7}$$

For the denominator, which is expressed as:

$$D_i = \sum_m \frac{S_{im}}{\sum_l S_{lm}} \tag{B.8}$$

the derivative with respect to z_i is given as follows:

$$\frac{\partial D_i}{\partial z_i} = \sum_m \frac{\frac{\partial S_{im}}{\partial z_i} \sum_l S_{lm} - S_{im} \frac{\partial \sum_l S_{lm}}{\partial z_i}}{(\sum_l S_{lm})^2}$$

In the same way and for all m, $\frac{\partial \sum_l S_{lm}}{\partial z_i} = \frac{\partial S_{im}}{\partial z_i} = -\frac{2ab(z_i-\mu_m)^{2b-1}}{(1+a(z_i-\mu_m)^{2b})^2}$. Consequently:

$$\frac{\partial D_i}{\partial z_i} = \sum_m \frac{\frac{\partial S_{im}}{\partial z_i} \sum_{l \neq i} S_{lm}}{(\sum_l S_{lm})^2} \tag{B.9}$$

The derivative of T_{ik} with respect to z_i is:

$$\frac{\partial T_{ik}}{\partial z_i} = \frac{\frac{\partial N_{ik}}{\partial z_i} D_i - N_{ik} \frac{\partial D_i}{\partial z_i}}{D_i^2} \tag{B.10}$$

where N_{ik}, $\frac{\partial N_{ik}}{z_i}$, D_i, $\frac{\partial D_i}{z_i}$ are substituted by Eqs. (B.6), (B.7), (B.8), and (B.9) respectively in Eq. (B.10).

Now we compute the derivative of $T_{i'k}$ with respect to z_i. In Eq. (B.4), we use $T_{i'k}$ to denote the case where $i' \neq i$.. The numerator and the denominator are derived separately. Like in Eq. (B.6), the numerator is given:

$$N_{i'k} = \frac{S_{i'k}}{\sum_l S_{lk}} \tag{B.11}$$

Since $S_{i'k}$ does not depend on z_i, its derivative is 0. Also $\frac{\partial \sum_l S_{lk}}{\partial z_i} = \sum_l \frac{\partial S_{lk}}{\partial z_i}$ such that $\frac{\partial S_{lk}}{\partial z_i} = \frac{\partial S_{ik}}{\partial z_i}$ only when $l = i$ and $\frac{\partial S_{lk}}{\partial z_i} = 0$ otherwise. The derivative of N_{ik} with respect to z_i is:

$$\frac{\partial N_{i'k}}{\partial z_i} = \frac{-S_{i'k} \frac{\partial S_{ik}}{\partial z_i}}{(\sum_l S_{lk})^2} \tag{B.12}$$

where $\frac{\partial S_{ik}}{\partial z_i}$ is given by Eq. (B.5).

Like in Eq. (B.7), the denominator is given as follows:

$$D_{i'} = \sum_m \frac{S_{i'm}}{\sum_l S_{lm}} \tag{B.13}$$

Since $S_{i'm}$ does not depend on z_i, its derivative is 0. $\frac{\partial \sum_l S_{lm}}{\partial z_i} = \sum_l \frac{\partial S_{lm}}{\partial z_i}$ where $\frac{\partial S_{lm}}{\partial z_i} = \frac{\partial S_{im}}{\partial z_i}$ only when $l = i$ such that $\frac{\partial S_{im}}{\partial z_i} = -\frac{2ab\|z_i-\mu_k\|_2^{2b-1}}{(1+a\|z_i-\mu_k\|_2^{2b})^2}$ and $\frac{\partial S_{lm}}{\partial z_i} = 0$ otherwise. We can write the derivative of D as follows:

$$\frac{\partial D_{i'}}{\partial z_i} = \sum_m \frac{-S_{i'm} \frac{\partial S_{im}}{\partial z_i}}{(\sum_l S_{lm})^2} \tag{B.14}$$

Now the derivative of $T_{i'k}$ with respect to z_i is:

$$\frac{\partial T_{i'k}}{\partial z_i} = \frac{\frac{\partial N_{i'k}}{\partial z_i} D_{i'} - N_{i'k} \frac{\partial D_{i'}}{\partial z_i}}{D_{i'}^2} \tag{B.15}$$

where $N_{i'k}$, $\frac{\partial N_{i'k}}{z_i}$, $D_{i'}$, $\frac{\partial D_{i'}}{z_i}$ are substituted by Eqs. (B.11), (B.12), (B.13), (B.14) respectively in Eq. (B.15).

Now let us recall the derivative of the KL-divergence, Eq. (B.4):

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i}(1 + \log \frac{T_{ik}}{S_{ik}}) - \frac{\partial S_{ik}}{\partial z_i} \frac{T_{ik}}{S_{ik}} \right] + \sum_{i' \neq i} \sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1 + \log \frac{T_{i'k}}{S_{i'k}}) \right]$$

We substitute S_{ik} and $\frac{\partial S_{ik}}{\partial z_i}$ by Eqs. (B.1) and (B.5) in Eq. (B.4). to obtain:

$$\begin{aligned} \frac{\partial KL}{\partial z_i} = \sum_k \Bigg[& \frac{\partial T_{ik}}{\partial z_i}(1 + \log \frac{T_{ik}}{S_{ik}}) + \frac{2bS_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2} \\ & \times \frac{T_{ik}}{(1 + a\|z_i - \mu_k\|_2^{2b})^{-1}} \Bigg] \\ & + \sum_{i' \neq i} \sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1 + \log \frac{T_{i'k}}{S_{i'k}}) \right] \end{aligned}$$

In Eq. (B.4), The term $\frac{\partial S_{ik}}{\partial z_i} \frac{T_{ik}}{S_{ik}}$ can be more simpler as follow:

$$\frac{\partial S_{ik}}{\partial z_i} \frac{T_{ik}}{S_{ik}} = -\frac{2bT_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2}$$

Substituting $\frac{\partial S_{ik}}{\partial z_i} \frac{T_{ik}}{S_{ik}}$ in $\frac{\partial KL}{\partial z_i}$ by its expression, we obtain the final formulation of the derivative of KL:

$$\begin{aligned} \frac{\partial KL}{\partial z_i} = & \sum_k \left[\frac{\partial T_{ik}}{\partial z_i}(1 + \log \frac{T_{ik}}{S_{ik}}) + \frac{2bT_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2} \right] \\ & + \sum_{i' \neq i} \sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1 + \log \frac{T_{i'k}}{S_{i'k}}) \right] \end{aligned} \tag{B.16}$$

B.1. Derivative of the total loss function with respect to z_i

Now, we assemble the two parts by substituting the derivative of Cross Entropy (Eq. (A.4)) and the derivative of KL divergence (Eq. (B.16)) in the derivative of the total loss function to obtain:

$$\frac{\partial F}{\partial \mu_k} = \alpha \frac{\partial CE}{\partial \mu_k} + \beta \frac{\partial KL}{\partial \mu_k}$$

$$\begin{aligned} \frac{\partial F}{\partial z_i} = \alpha \sum_j \Bigg[& \frac{2bp_{ij}}{1/a\|z_i - z_j\|_2^{2(b-1)} + \|z_i - z_j\|_2^2} \\ & - \frac{2b(1 - p_{ij})}{\|z_i - z_j\|_2^2 (1 + a\|z_i - z_j\|_2^{2b})} \Bigg] \|z_i - z_j\|_2 \\ & + \beta \Bigg(\sum_k \left[\frac{\partial T_{ik}}{\partial z_i}(1 + \log \frac{T_{ik}}{S_{ik}}) + \frac{2bT_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2} \right] \\ & + \sum_{i' \neq i} \sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1 + \log \frac{T_{i'k}}{S_{i'k}}) \right] \Bigg) \end{aligned} \tag{B.17}$$

The update of z_i is performed using $\frac{\partial F}{\partial z_i}$ where α and β are parameters to be set.

However, Eq. (B.17) refers to the two fractions: $\log \frac{T_{ik}}{S_{ik}}$ and $\log \frac{T_{i'k}}{S_{i'k}}$ which could involve division by 0.

To solve this problem, we do some studies and analysis on the definition domain of S_{ik} and T_{ik}, the sign of hyper-parameter a, and the definition domain of $\log \frac{T_{ik}}{S_{ik}}$.

The sign of hyper-parameter 'a'

Define $\phi : \mathbb{R}^d \times \mathbb{R}^d \rightarrow [0, 1]$, a smooth approximation of the membership strength between two points in $\mathbb{R}^d$:

$$\phi(z_i, z_j) = q_{ij} = (1 + \|z_i - z_j\|_2^{2b})^{-1}$$

where 'a' and 'b' are chosen by non-linear least square fitting against the curve $\psi : \mathbb{R} \times \mathbb{R} \rightarrow [0, 1]$ where:

$$\psi(z_i, z_j) = \begin{cases} 1 & if \|z_i - z_j\|_2 \leq min - dist \\ \exp(-(\|z_i - z_j\|_2 - min - dist)) & \text{otherwise} \end{cases}$$

So the sign of 'a' is always positive since $q_{ij} \in [0, 1]$ and that is mean:

$$q_{ij} \rightarrow \begin{cases} 0 & if \quad 1 + a\|z_i - z_j\|_2^{2b} \rightarrow +\infty \\ 1 & if \quad a\|z_i - z_j\|_2^{2b} = 0 \end{cases}$$

So since $\|z_i - z_j\|_2$ is always positive, we conclude that 'a' is also always positive.

Definition domain of S_{ik} and T_{ik}

Define $S_{ik} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow [0,1]$, a soft assignment between the datapoints and the cluster centroid in $\mathbb{R}^d$:

$$S_{ik} = (1 + a\|z_i - \mu_k\|_2^{2b})^{-1} = \frac{1}{1 + a\|z_i - \mu_k\|_2^{2b}}$$

$$S_{ik} \text{ is defined since} \Leftrightarrow \begin{cases} \|z_i - \mu_k\|_2^{2b} & \text{is defined} \\ a\|z_i - \mu_k\|_2^{2b} \neq -1 \end{cases}$$

$$\Leftrightarrow \begin{cases} \|z_i - \mu_k\|_2^{2b} & \text{is always defined} \\ a \neq \frac{-1}{\|z_i-\mu_k\|_2^{2b}} & \text{is defined since } a \text{ is positive} \end{cases}$$

Now, since T_{ik} is based on S_{ik} so it is also defined on the interval $[0,1]$.

Definition domain of $\log \frac{T_{ik}}{S_{ik}}$

Let us set $g(z_i, \mu_k) = \log \frac{T_{ik}}{S_{ik}}$

$$\text{g is defined if and only if} \Leftrightarrow \begin{cases} S_{ik} \neq 0 & (1) \\ \frac{T_{ik}}{S_{ik}} > 0 \end{cases} \tag{B.18}$$

where:

$$\frac{T_{ik}}{S_{ik}} > 0 \Leftrightarrow \begin{cases} T_{ik} \neq 0 & (2) \\ S_{ik} \text{ and } T_{ik} \text{ has the same sign} & (3) \end{cases} \tag{B.19}$$

As we see in the previews Eq. (B.18), we have three conditions that are treated separately. The first condition is satisfied if and only if:

$$(1) \Leftrightarrow \frac{1}{1 + a\|z_i - z_j\|_2^{2b}} \nrightarrow 0 \Leftrightarrow \begin{cases} 1 + a\|z_i - z_j\|_2^{2b} \nrightarrow \infty \\ 1 + a\|z_i - z_j\|_2^{2b} \neq 0 \end{cases}$$

$$\Leftrightarrow \begin{cases} a\|z_i - z_j\|_2^{2b} \nrightarrow \infty & (*) \\ a\|z_i - z_j\|_2^{2b} \neq -1 & \text{is always satisfied} \end{cases}$$

$$(*) \Leftrightarrow a\|z_i - z_j\|_2^{2b} \nrightarrow +\infty \quad \text{since } -\infty \text{ is excluded}$$
$$\Leftrightarrow \|z_i - z_j\|_2^{2b} \nrightarrow +\infty$$

The cases where $\|z_i - z_j\|_2^{2b} \rightarrow +\infty$ are:

$$\|z_i - z_j\|_2^{2b} \rightarrow +\infty \Leftrightarrow \begin{cases} \|z_i - z_j\|_2^{2b} \rightarrow +\infty & \text{and } b \geq 0 \text{ (A)} \\ \|z_i - z_j\|_2^{2b} \rightarrow 0 & \text{and } b < 0 \text{ (B)} \end{cases}$$

Now, to make **condition (1)** $S_{ik} \nrightarrow 0$ satisfied, it should make conditions (A) and (B) not satisfied.

Now we pass to **condition (2)**: $T_{ik} \neq 0$

$$T_{ik} = \frac{S_{ik}/\sum_l S_{lk}}{\sum_m (S_{im}/\sum_l S_{lm})}$$

So T_{ik} is defined and $T_{ik} = 0$ if $S_{ik} = 0$.

Condition (3): T_{ik} and S_{ik} have the same sign, and this condition is always satisfied since $S_{ik} \in [0,1]$ and T_{ik} is computed based on S_{ik} so they have always the same sign.

From what we mentioned above, we can conclude our solutions to solve the problem of division by zero as follows. The first one is shown in algorithm 2:

Algorithm 2 Solving the problem of division by zero

```
if $b \geq 0$ then
    if $\|z_i - \mu_k\|_2 > Dist - Max$ then
        we have two solutions:
        Consider $z_i$ as an outlier
        Replace $\|z_i - \mu_k\|_2$ by Dist-Max
    end if
else
    Solve the cases where $z_i \approx \mu_k$
end if
```

The second solution is as follows:

$$\frac{T_{ik}}{S_{ik}} = \frac{\frac{S_{ik}/\sum_l S_{lk}}{\sum_m (S_{im}/\sum_l S_{lm})}}{S_{ik}}$$
$$= \frac{1}{(\sum_l S_{lk})(\sum_m (S_{im}/\sum_l S_{lm}))}$$

B.2. The derivative of objective function with respect to the μ_k

Now, we will drive our objective function with respect to μ_k. First, let us recall our weighted sum function:

$$F(P,Q,T,S) = \alpha CE(P,Q) + \beta KL(T \parallel S) \tag{B.20}$$

Such that:

$$CE(P,Q) = \sum_i \sum_j \left[p_{ij} \log(\frac{p_{ij}}{q_{ij}}) + (1 - p_{ij}) \log(\frac{1 - p_{ij}}{1 - q_{ij}}) \right]$$

$$KL(T \parallel S) = \sum_i \sum_k T_{ik} \log \frac{T_{ik}}{S_{ik}}$$
$$= \sum_i \sum_k T_{ik} \log T_{ik} - T_{ik} \log S_{ik}$$

Also, let us recall the derivative of our weighted sum function:

$$\frac{\partial F}{\partial \mu_k} = \alpha \frac{\partial CE}{\partial \mu_k} + \beta \frac{\partial KL}{\partial \mu_k}$$

CE does not depend on cluster centroids, so its derivative is zero. Therefore, we only compute the derivative of the KL divergence with respect to μ_k:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i \left[\frac{\partial T_{ik}}{\partial \mu_k} \log T_{ik} + T_{ik} \frac{\frac{\partial T_{ik}}{\partial \mu_k}}{T_{ik}} - \frac{\partial T_{ik}}{\partial \mu_k} \log S_{ik} - \frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} \right]$$
$$+ \sum_i \sum_{k' \neq k} \left[\frac{\partial T_{ik'}}{\partial \mu_k} \log T_{ik'} + T_{ik'} \frac{\frac{\partial T_{ik'}}{\partial \mu_k}}{T_{ik'}} - \frac{\partial T_{ik'}}{\partial \mu_k} \log S_{ik'} - \frac{\partial S_{ik'}}{\partial \mu_k} \frac{T_{ik'}}{S_{ik'}} \right]$$

We divided the derivative into a sum of 2 parts since T_{ik} needs to compute its derivative with respect μ_k in two cases. The first one is when $k' = k$, and the second case is when $k' \neq k$. Also, since $S_{ik'}$ does not depend on μ_k so $\frac{\partial S_{ik'}}{\partial \mu_k} = 0$.

$$\frac{\partial KL}{\partial \mu_k} = \sum_i \left[\frac{\partial T_{ik}}{\partial \mu_k} \log T_{ik} + \frac{\partial T_{ik}}{\partial \mu_k} - \frac{\partial T_{ik}}{\partial \mu_k} \log S_{ik} - \frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} \right]$$
$$+ \sum_i \sum_{k' \neq k} \left[\frac{\partial T_{ik'}}{\partial \mu_k} \log T_{ik'} + \frac{\partial T_{ik'}}{\partial \mu_k} - \frac{\partial T_{ik'}}{\partial \mu_k} \log S_{ik'} \right]$$

$$\frac{\partial KL}{\partial \mu_k} = \sum_i \left[\frac{\partial T_{ik}}{\partial \mu_k} (1 + \log \frac{T_{ik}}{S_{ik}}) - \frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} \right] + \sum_i \sum_{k' \neq k} \left[\frac{\partial T_{ik'}}{\partial \mu_k} (1 + \log \frac{T_{ik'}}{S_{ik'}}) \right] \tag{B.21}$$

To simplify the derivative of the KL divergence with respect to μ_k we calculate the derivatives of $S_{ik} = (1 + a\|z_i - \mu_k\|_2^{2b})^{-1}$ and T_{ik} w.r.t. μ_k in case of $k' = k$. In addition, we compute the derivatives of $S_{ik'}$ and $T_{ik'}$ w.r.t. μ_k in the case $k' \neq k$. Now we start by $\frac{\partial S_{ik}}{\partial \mu_k}$

$$\frac{\partial S_{ik}}{\partial \mu_k} = \frac{2bS_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2} \tag{B.22}$$

And since $S_{ik'}$ does not depend on μ_k, so we can conclude that

$$\frac{\partial S_{ik'}}{\partial \mu_k} = 0 \tag{B.23}$$

Also here we should derive T_{ik} when $k' = k$ and $T_{ik'}$ when $k' \neq k$ with respect to μ_k. Now we are going to start with T_{ik}:

$$T_{ik} = \frac{S_{ik}/\sum_l S_{lk}}{\sum_m (S_{im}/\sum_l S_{lm})}$$

let us derive the numerator and the denominator separately. We have the numerator:

$$N_{ik} = \frac{S_{ik}}{\sum_l S_{lk}}$$

So the derivative of N_{ik} with respect to μ_k is:

$$\frac{\partial N_{ik}}{\partial \mu_k} = \frac{\frac{\partial S_{ik}}{\partial \mu_k}\sum_l S_{lk} - S_{ik}\sum_l \frac{\partial S_{lk}}{\partial \mu_k}}{(\sum_l S_{lk})^2}$$

such that $\frac{\partial S_{ik}}{\partial z_i}$ is given by Eq. (B.22) and $\frac{\partial S_{lk}}{\partial z_i}$ is deduced from Eq. (B.22) as follows:

$$\frac{\partial S_{lk}}{\partial \mu_k} = \frac{2bS_{lk}}{1/\|z_l - \mu_k\|_2^{2b-1} + \|z_l - \mu_k\|_2^2}$$

We have the denominator:

$$D_i = \sum_m \frac{S_{im}}{\sum_l S_{lm}}$$

We have $\frac{\partial \sum_m \frac{S_{im}}{\sum_l S_{lm}}}{\partial \mu_k} = \sum_m \frac{\partial \frac{S_{im}}{\sum_l S_{lm}}}{\partial \mu_k}$ where $\frac{\partial \frac{S_{im}}{\sum_l S_{lm}}}{\partial \mu_k} \neq 0$ when $m = k$. And $\frac{\partial \frac{S_{im}}{\sum_l S_{lm}}}{\partial \mu_k} = 0$ when $m \neq k$, since $\frac{\partial S_{im}}{\partial \mu_k} = \frac{\partial S_{ik}}{\partial \mu_k} = \frac{2ab\|z_i-\mu_k\|_2^{2b-1}}{(1+a\|z_i-\mu_k\|_2^{2b})^2}$ when $m = k$, and $\frac{\partial S_{im}}{\partial \mu_k} = 0$ otherwise. Also $\sum_l \frac{\partial S_{lm}}{\partial \mu_k} = \sum_l \frac{\partial S_{lk}}{\partial \mu_k} = \sum_l \frac{2ab\|z_l-\mu_k\|_2^{2b-1}}{(1+a\|z_l-\mu_k\|_2^{2b})^2}$ when $m = k$, and $\sum_l \frac{\partial S_{lm}}{\partial \mu_k} = 0$ otherwise. So we can write $\frac{\partial D_i}{\partial \mu_k}$ as follow:

$$\frac{\partial D_i}{\partial \mu_k} = \frac{\frac{\partial S_{ik}}{\partial \mu_k}\sum_l S_{lk} - S_{ik}\sum_l \frac{\partial S_{lk}}{\partial \mu_k}}{(\sum_l S_{lk})^2}$$

We can see that $\frac{\partial N_{ik}}{\partial \mu_k} = \frac{\partial D_i}{\partial \mu_k}$. So the derivative of T_{ik} with respect to μ_k is:

$$\frac{\partial T_{ik}}{\partial \mu_k} = \frac{\frac{\partial N_{ik}}{\partial \mu_k}(D_i - N_{ik})}{D_i^2}$$

We do the same thing for the derivative of $T_{ik'}$ with respect to μ_k in the case here is $k' \neq k$. Let us first write $T_{ik'}$:

$$T_{ik'} = \frac{S_{ik'}/\sum_l S_{lk'}}{\sum_m (S_{im}/\sum_l S_{lm})}$$

we are going to derive the numerator and denominator separately:

$$N_{ik'} = \frac{S_{ik'}}{\sum_l S_{lk'}}$$

Since $S_{ik'}$ does not depend on μ_k, we can deduce that $\frac{\partial S_{ik'}}{\partial \mu_k} = 0$ from Eq. (B.1) and (B.22). Also for $\sum_l S_{lk'}$ does not depend on μ_k so $\sum_l \frac{\partial S_{lk'}}{\partial \mu_k} = 0$. We can conclude that the derivative of $N_{ik'}$ with respect to μ_k is zero. Let us move now to the denominator:

$$D_i = \sum_m \frac{S_{im}}{\sum_l S_{lm}}$$

We already computed the derivative of D_i, so let us just recall it:

$$\frac{\partial D_i}{\partial \mu_k} = \frac{\frac{\partial S_{ik}}{\partial \mu_k}\sum_l S_{lk} - S_{ik}\sum_l \frac{\partial S_{lk}}{\partial \mu_k}}{(\sum_l S_{lk})^2}$$

From previous equations we can conclude that the derivative of $T_{ik'}$ is as follows:

$$\frac{\partial T_{ik'}}{\partial \mu_k} = \frac{-N_{ik'}\frac{\partial D_i}{\partial \mu_k}}{D_i^2}$$

Now we are going to substitute the previews equations in the derivative of the KL divergence Eq. (B.21) with respect to μ_k:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i \left[\frac{\partial T_{ik}}{\partial \mu_k}(1 + \log\frac{T_{ik}}{S_{ik}}) - \frac{\partial S_{ik}}{\partial \mu_k}\frac{T_{ik}}{S_{ik}}\right] + \sum_i \sum_{k' \neq k}\left[\frac{\partial T_{ik'}}{\partial \mu_k}(1 + \log\frac{T_{ik'}}{S_{ik'}})\right]$$

We substitute S_{ik} and $\frac{\partial S_{ik}}{\partial \mu_k}$ by Eqs. (B.1) and (B.22), so let us recall $\frac{\partial S_{ik}}{\partial \mu_k}$:

$$\frac{\partial S_{ik}}{\partial \mu_k} = \frac{2bS_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2}$$

Then we are going to multiply it by $\frac{T_{ik}}{S_{ik}}$ in order to get a simpler form:

$$\frac{\partial S_{ik}}{\partial \mu_k}\frac{T_{ik}}{S_{ik}} = \frac{2bT_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2}$$

Substituting $\frac{\partial S_{ik}}{\partial \mu_k}\frac{T_{ik}}{S_{ik}}$ in $\frac{\partial KL}{\partial \mu_k}$ by the last equation, we obtain:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i \left[\frac{\partial T_{ik}}{\partial \mu_k}(1 + \log\frac{T_{ik}}{S_{ik}}) - \frac{2bT_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2}\right] + \sum_i \sum_{k' \neq k}\left[\frac{\partial T_{ik'}}{\partial \mu_k}(1 + \log\frac{T_{ik'}}{S_{ik'}})\right]$$

Appendix C. Light plus variant derivations

Our optimisation purpose is to update the datapoints z_i and the centroids μ_k, so first, we derived the function F with respect to z_i. Then, we compute the derivative of the weighted sum function with respect to μ_k. However, the CE function does not depend on the cluster centres, so we compute only the derivative of the clustering loss function. Now, for the derivative of the weighted sum function, we have:

$$\frac{\partial F}{\partial z_i} = \alpha\frac{\partial CE}{\partial z_i} + \beta\frac{\partial KL}{\partial z_i} \tag{C.1}$$

C.1. Derivation of the KL divergence with respect to z_i

Notice that for the datapoints z_i we have two objective functions that need to compute their derivations with respect to z_i. The derivation of the CE function is already explained in Appendix A. In this part, we provide the details of the derivative of the KL divergence with respect to z_i. Let us recall the KL divergence:

$$KL(T,S) = \sum_i \sum_k T_{ik}\log T_{ik} - T_{ik}\log S_{ik} \tag{C.2}$$

When updating z_i, T_{ik} is already computed and considered a constant number, so T_{ik} does not depend on z_i. Thus, the derivative of $T_{ik}\log T_{ik}$ with respect to z_i is zero. We obtain:

$$\frac{\partial KL}{\partial z_i} = \sum_k -\frac{\partial S_{ik}}{\partial z_i}\frac{T_{ik}}{S_{ik}} \tag{C.3}$$

We can derived S_{ik} with respect to z_i as follow:

$$\frac{\partial S_{ik}}{\partial z_i} = -\frac{2ab\|z_i - \mu_k\|_2^{2b-1}}{(1 + a\|z_i - \mu_k\|_2^{2b})^2} \tag{C.4}$$

Substituting S_{ik} and its derivative $\frac{\partial S_{ik}}{\partial z_i}$ (Eq. (C.4)) in Eq. (C.3). we can conclude $\frac{\partial KL}{\partial z_i}$ like this:

$$\frac{\partial KL}{\partial z_i} = \sum_k -\frac{\partial S_{ik}}{\partial z_i}\frac{T_{ik}}{S_{ik}}$$

$$\frac{\partial KL}{\partial z_i} = \sum_k \frac{2ab\|z_i - \mu_k\|_2^{2b-1}T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}} \tag{C.5}$$

C.2. Derivative of the total loss function with respect to z_i

Now, we assemble the two parts by substituting the derivative of Cross Entropy (Eq. (A.4)) and the derivative of KL divergence (Eq. (C.5)) in the total loss function (Eq. (C.1)) to obtain:

$$\frac{\partial F}{\partial z_i} = \alpha\frac{\partial CE}{\partial z_i} + \beta\frac{\partial KL}{\partial z_i}$$

$$\frac{\partial F}{\partial z_i} = \alpha \sum_j \left[\frac{2ab\|z_i - z_j\|_2^{2(b-1)} p_{ij}}{1 + a\|z_i - z_j\|_2^{2b}} - \frac{2b(1-p_{ij})}{\|z_i - z_j\|_2^2(1 + a\|z_i - z_j\|_2^{2b})} \right] \|z_i - z_j\| + \beta \sum_k \frac{2ab\|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}} \tag{C.6}$$

Then the update of z_i is performed using $\frac{\partial F}{\partial z_i}$, we just have to choose α and β.

C.3. The derivative of objective function with respect to the μ_i

Now, we derivate our objective function with respect to μ_k. First, let us recall our weighted sum function:

$$F(P,Q,T,S) = \alpha CE(P,Q) + \beta KL(T \parallel S) \tag{C.7}$$

Such that:

$$CE(P,Q) = \sum_i \sum_j \left[p_{ij} \log(\frac{p_{ij}}{q_{ij}}) + (1-p_{ij}) \log(\frac{1-p_{ij}}{1-q_{ij}}) \right]$$

$$KL(T \parallel S) = \sum_i \sum_k T_{ik} \log T_{ik} - T_{ik} \log S_{ik}$$

Also, let us recall the derivative of our weighted sum function:

$$\frac{\partial F}{\partial \mu_k} = \alpha \frac{\partial CE}{\partial \mu_k} + \beta \frac{\partial KL}{\partial \mu_k}$$

CE does not depend on cluster centroids, so its derivative is zero. Therefore, we compute only the derivative of the KL divergence with respect to μ_k:

$$KL(T,S) = \sum_i \sum_k T_{ik} \log T_{ik} - T_{ik} \log S_{ik}$$

T_{ik} is considered as a constant number, since T_{ik} does not depend on μ_k. Thus, the derivative of $T_{ik} \log T_{ik}$ with respect to μ_k is zero. The partial derivative of the KL function with respect to μ_k is:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i -\frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} \tag{C.8}$$

To simplify the derivative of the KL divergence with respect to μ_k, we calculate the derivative of S_{ik} w.r.t μ_k:

$$\frac{\partial S_{ik}}{\partial \mu_k} = \frac{2ab\|z_i - \mu_k\|_2^{2b-1}}{(1 + a\|z_i - \mu_k\|_2^{2b})^2} \tag{C.9}$$

Substituting S_{ik} and its derivative $\frac{\partial S_{ik}}{\partial \mu_k}$ (Eq. (C.9)) in Eq. (C.8). we obtain:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i -\frac{2ab\|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}} \tag{C.10}$$

Appendix D. Definition of the values domain of the CE and KL divergence gradients with respect to z_i

D.1. Defining the interval values of the CE gradient with respect to z_i

We have the gradient of the CE as follows:

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[\frac{2bp_{ij}}{1/(a\|z_i - z_j\|_2^{2(b-1)}) + \|z_i - z_j\|_2^2} - \frac{2b(1-p_{ij})}{\|z_i - z_j\|_2^2(1 + a\|z_i - z_j\|_2^{2b})} \right] \|z_i - z_j\| \tag{D.1}$$

We observe that the gradient of the CE has two terms $\frac{2bp_{ij}}{1/(a\|z_i-z_j\|_2^{2(b-1)})+\|z_i-z_j\|_2^2}$ and $\frac{2b(1-p_{ij})}{\|z_i-z_j\|_2^2(1+a\|z_i-z_j\|_2^{2b})}$. So we need to study their values Domain. Let us start with the first term:

$$\frac{2bp_{ij}}{1/(a\|z_i - z_j\|_2^{2(b-1)}) + \|z_i - z_j\|_2^2} \tag{D.2}$$

We have that the values domain of P and Q are $[0,1]$. In addition, the hyper-parameters a and b are always positives. Also, we have $\|z_i - z_j\|_2^{2(b-1)}$ and $\|z_i - z_j\|_2^2$ are always positives so the values domain of the first term (Eq. (D.2)) is $[0,+\infty[$. Now let us move to the second term:

$$-\frac{2b(1-p_{ij})}{\|z_i - z_j\|_2^2(1 + a\|z_i - z_j\|_2^{2b})} \tag{D.3}$$

In the above term (Eq. (D.3)), we observe that it is preceded by the negative sign, so the values domain of this term is $]-\infty,0]$. So the values domain of the CE gradient is $]-\infty,+\infty[$.

D.2. Defining the values domain of the KL divergence gradient with respect to z_i

In this part, we studied the definition domain of the KL divergence gradient w.r.t z_i in the two variants.

D.2.1. The derivative of the KL divergence plus variant

Let us recall the derivative of the KL divergence with respect to z_i:

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i}(1 + \log\frac{T_{ik}}{S_{ik}}) + \frac{2bT_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2} \right] + \sum_{i' \neq i} \sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1 + \log\frac{T_{i'k}}{S_{i'k}}) \right]$$

The same thing for the KL gradient, we proved that the values domain of S_{ik} and T_{ik} are in $[0,1]$ in the division by zero subsection (see Appendix B). so the values domain of KL derivative in $[0,+\infty[$.

D.2.2. The derivative of the KL divergence light plus variant

Let us recall the derivative of the KL divergence with respect to z_i:

$$\frac{\partial KL}{\partial z_i} = \sum_k \frac{2ab\|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}}$$

We have the values domain of T_{ik} is in $[0,1]$, $\|z_i - z_j\|_2^2$ is always positive, and a is also positive, so the values domain of KL derivative in $[0,+\infty[$.

References

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

Allaoui, M., Kherfi, M. L., & Cheriet, A. (2020). Considerably improving clustering algorithms using UMAP dimensionality reduction technique: A comparative study. In *International conference on image and signal processing* (pp. 317–325). Springer.

Baggenstoss, P. M. (2002). *Statistical modeling using gaussian mixtures and hmms with matlab*. Newport RI: Naval Undersea Warfare Center.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Caliński, T., & Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics-Theory and Methods*, *3*(1), 1–27.

Campello, R. J., Moulavi, D., & Sander, J. (2013). Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining* (pp. 160–172). Springer.

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

Dalmia, A., & Sia, S. (2021). Clustering with UMAP: Why and how connectivity matters. arXiv preprint arXiv:2108.05525.

Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (2), 224–227.

Dong, W., Moses, C., & Li, K. (2011). Efficient k-nearest neighbor graph construction for generic similarity measures. In *Proceedings of the 20th international conference on world wide web* (pp. 577–586).

Elhamifar, E., & Vidal, R. (2011). Sparse manifold clustering and embedding. In *Advances in neural information processing systems, Vol. 24*.

Ghasedi Dizaji, K., Herandi, A., Deng, C., Cai, W., & Huang, H. (2017). Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. In *Proceedings of the IEEE international conference on computer vision* (pp. 5736–5745).

Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440–1448).

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580–587).

Gowda, K. C., & Krishna, G. (1978). Agglomerative clustering using the concept of mutual nearest neighbourhood. *Pattern Recognition*, *10*(2), 105–112.

Guo, J., Yuan, X., Xu, P., Bai, H., & Liu, B. (2020). Improved image clustering with deep semantic embedding. *Pattern Recognition Letters*, *130*, 225–233.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504–507.

Hu, W., Miyato, T., Tokui, S., Matsumoto, E., & Sugiyama, M. (2017). Learning discrete representations via information maximizing self-augmented training. In *International conference on machine learning* (pp. 1558–1567). PMLR.

Huang, P., Huang, Y., Wang, W., & Wang, L. (2014). Deep embedding network for clustering. In *2014 22nd international conference on pattern recognition* (pp. 1532–1537). IEEE.

Huang, X., Wu, L., & Ye, Y. (2019). A review on dimensionality reduction techniques. *International Journal of Pattern Recognition and Artificial Intelligence*, *33*(10), Article 1950017.

Hull, J. J. (1994). A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *16*(5), 550–554.

Ji, P., Zhang, T., Li, H., Salzmann, M., & Reid, I. (2017). Deep subspace clustering networks. In *Advances in neural information processing systems, Vol. 30*.

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

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems, Vol. 25* (pp. 1097–1105).

Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, *2*(1–2), 83–97.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Lewis, D. D., Yang, Y., Russell-Rose, T., & Li, F. (2004). Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, *5*(Apr), 361–397.

Luo, B., Wilson, R. C., & Hancock, E. R. (2003). Spectral embedding of graphs. *Pattern Recognition*, *36*(10), 2213–2230.

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, *9*(11).

MacQueen, J., et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, Vol. 1* (pp. 281–297). Oakland, CA, USA.

McConville, R., Santos-Rodriguez, R., Piechocki, R. J., & Craddock, I. (2019). N2d:(not too) deep clustering via clustering the local manifold of an autoencoded embedding. arXiv preprint arXiv:1908.05968.

McInnes, L., Healy, J., & Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426.

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

Mrabah, N., Khan, N. M., Ksantini, R., & Lachiri, Z. (2020). Deep clustering with a dynamic autoencoder: From reconstruction towards centroids construction. *Neural Networks*, *130*, 206–228.

Ng, A. Y., Jordan, M. I., Weiss, Y., et al. (2002). On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems, Vol. 2* (pp. 849–856).

Ozaki, K., Shimbo, M., Komachi, M., & Matsumoto, Y. (2011). Using the mutual k-nearest neighbor graphs for semi-supervised classification on natural language data. In *Proceedings of the fifteenth conference on computational natural language learning* (pp. 154–162).

Patel, V. M., & Vidal, R. (2014). Kernel sparse subspace clustering. In *2014 IEEE international conference on image processing (Icip)* (pp. 2849–2853). IEEE.

Pearson, K. (1901). LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *2*(11), 559–572.

Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.

Ranzato, M., Huang, F. J., Boureau, Y.-L., & LeCun, Y. (2007). Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *2007 IEEE conference on computer vision and pattern recognition* (pp. 1–8). IEEE.

Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, *20*, 53–65.

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*(5500), 2323–2326.

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

Shaham, U., Stanton, K., Li, H., Nadler, B., Basri, R., & Kluger, Y. (2018). Spectralnet: Spectral clustering using deep neural networks. arXiv preprint arXiv:1801.01587.

Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 888–905.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Tenenbaum, J. B., De Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, *290*(5500), 2319–2323.

Thrun, M. C. (2018). *Projection-based clustering through self-organization and swarm intelligence: combining cluster analysis with the visualization of high-dimensional data*. Springer.

Thrun, M. C., & Ultsch, A. (2020). Clustering benchmark datasets exploiting the fundamental clustering problems. *Data in Brief*, *30*, Article 105501.

Ultsch, A. (2005). U* C: Self-organized clustering with emergent feature maps. In *LWA* (pp. 240–244). Citeseer.

Wang, J., Wang, J., Ke, Q., Zeng, G., & Li, S. (2015). Fast approximate k-means via cluster closures. In *Multimedia data mining and analytics* (pp. 373–395). Springer.

Wang, W.-T., Wu, Y.-L., Tang, C.-Y., & Hor, M.-K. (2015). Adaptive density-based spatial clustering of applications with noise (DBSCAN) according to data. In *2015 international conference on machine learning and cybernetics (ICMLC), Vol. 1* (pp. 445–451). IEEE.

Wu, L., Liu, Z., Xia, J., Zang, Z., Li, S., & Li, S. Z. (2022). Generalized clustering and multi-manifold learning with geometric structure preservation. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision* (pp. 139–147).

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

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

Yan, Y., Hao, H., Xu, B., Zhao, J., & Shen, F. (2020). Image clustering via deep embedded dimensionality reduction and probability-based triplet loss. *IEEE Transactions on Image Processing*, *29*, 5652–5661.

Yang, J., Parikh, D., & Batra, D. (2016). Joint unsupervised learning of deep representations and image clusters. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5147–5156).