Unified Embedding and Clustering

Mebarka Allaoui [1], Mohammed Lamine Kherfi [1], Abdelhakim Cheriet [1], and Abdelhamid Bouchachia [1]

[1]Affiliation not available

October 30, 2023

Abstract

In this paper, we introduce a novel algorithm that unifies manifold embedding and clustering (UEC) which efficiently predicts clustering assignments of the high dimensional data points in a new embedding space. The algorithm is based on a bi-objective optimisation problem combining embedding and clustering loss functions. Such original formulation will allow to simultaneously preserve the original structure of the data in the embedding space and produce better clustering assignments. The experimental results using a number of real-world datasets show that UEC is competitive with the state-of-art clustering methods.

Unified Embedding and Clustering

Mebarka Allaoui, Mohammed Lamine Kherfi, Abdelhakim Cheriet,
and Abdelhamid Bouchachia, *Senior Member, IEEE*

***Abstract*—In this paper, we introduce a novel algorithm that unifies manifold embedding and clustering (UEC) which efficiently predicts clustering assignments of the high dimensional data points in a new embedding space. The algorithm is based on a bi-objective optimisation problem combining embedding and clustering loss functions. Such original formulation will allow to simultaneously preserve the original structure of the data in the embedding space and produce better clustering assignments. The experimental results using a number of real-world datasets show that UEC is competitive with the state-of-art clustering methods.**

***Index Terms*—Clustering, manifold embedding, joint optimisation, deep representation learning.**

I. Introduction

CLUSTERING is a fundamental pillar of unsupervised machine learning and it has received increasing popularity among various research communities. There are two ways for applying clustering algorithms, either by operating on data in the original space or projecting data into a new space. Such a projection which can be either linear or non-linear transformation, is often done through dimensionality reduction techniques such as Isomap [1], local linear embedding (LLE) [2], and Laplacian eigen-maps [3]. These techniques embed data in a new space based on the manifold local geometric structure. Once embedding is done, clustering can be applied in a straightforward way on the transformed data [4], [5], [6].

With the advent of deep learning, work on clustering has seen a new direction which is based on deep representation learning leading to a number of algorithms such as Deep Embedding Clustering (DEC) [7], Joint Unsupervised LEarning (JULE) [8], DEeP Embedded RegularIzed ClusTering (DEPICT) [9], Deep Adaptive Image Clustering (DAC) [10], Information Maximising Self-Augmented Training (IMSAT) [11], Spectral-Net [12], and Deep Clustering by Gaussian Mixture Variational Autoencoders with Graph Embedding (DGG) [13]. Based on nonlinear embedding, these algorithms showed excellent clustering results.

M. Allaoui, M. L. Kherfi and A. Cheriet are with Kasdi Merbah University Ouargla, Ouargla, Algeria e-mail: (allaoui.mebarka@univ-ouargla.dz; Mohammedlamine.Kherfi@uqtr.ca; ahcheriet@gmail.com).

M. L. Kherfi was with LAMIA Laboratory, University of Quebec on Trois-Rivières, Trois-Rivières, Canada

Abdelhamid Bouchachia is with the Department of Computing and Informatics, Bournemouth University, Poole BH12 5BB, U.K. e-mail:(abouchachia@bournemouth.ac.uk)

Despite the ability of these techniques to learn the intrinsic structure of data, they suffer the problem of inherent defects that exist in the embedding space. In addition, they do not focus on learning from the similarities between the data points, therefore, the global and local structure of the data set are not preserved in the low dimensional spaces of these techniques. In contrast, the manifold learning techniques, including algorithms such as Isomap [1], t-SNE [14] and UMAP [15], tend to preserve similarities and project similar data points to a nearby representation, while dissimilar ones are projected far apart preserving the local and global structure of data.

In this paper, we propose an algorithm that can learn to simultaneously produce a low-dimensional embedding space and assign data to clusters. It is motivated by the fact that manifold learning-based techniques [16], [17] improve the quality of clustering when jointly learned [18]. The proposed algorithm, called Unified Embedding and Clustering (UEC). It seeks to learn the embedding space in a way similar to UMAP [15]. Firstly, UEC takes the extracted features using deep autoencoder [19], [20]. Then it initialises both the low dimensional space and the cluster assignments. In order to improve the embedding and the quality of the assignment of the data, UEC optimises a two-term objective function that controls both embedding and clustering.

Our contributions in this paper are:

- The proposal of a novel manifold-based learning algorithm that simultaneously learns the embedding space of the data and the clustering assignments. The joint optimisation of both the embedding and clustering objective functions leads to better preservation of the similarities in the original space and clusterability of the embedding space.
- The theoretical derivations of the optimisation process associated with the proposed UEC.
- 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 II reviews related work. Section III describes the proposed algorithm. Section VI discusses the experimental results before concluding the paper in Section Sec VII.

II. Related work

Clustering has been extensively studied in the literature resulting in a plethora of algorithms such as the well-known algorithms K-means [21], [22], GMM

[23], the agglomerative clustering [24], DBSCAN [25], [26], and Spectral Clustering [27], [28]. They can be categorised into distance-based, density-based or connectivity-based algorithms, depending on their conceptual idea [29]. Despite the popularity of these techniques, they are not efficient in all cases and suffer from high computational complexity when dealing with large-scale data sets.

To solve the aforementioned issues, dimensionality reduction techniques have been used to map data in low dimensional space. The dimensionality reduction methods can be categorised to deep or manifold learning methods. Manifold embedding methods can focus on local or global structure. The well-known algorithm Principal Component Analysis (PCA) [30] seeks to produce a linear transformation of data into a new feature space. However, due to its linearity, PCA does not perform well in cases where relationships are non-linear. Thankfully, alternative manifold learning methods exist to overcome the shortcoming of linearity. The most popular ones are Isomap [1] and its precursor Multidimensional Scaling (MDS), Locally Linear Embedding (LLE) [2]. t-distributed Stochastic Neighbour Embedding (t-SNE) [14] and the most recent algorithm UMAP [15]. These methods seek to utilise the distances between the original data points to learn the underlying structure better and preserve the similarity among data points in the low dimensional space. A number of techniques [17], [18], [16], proved powerful to support clustering.

In dimensionality reduction methods, various autoencoders [31], [20] and CNNs [32], [33] have been proposed showing significant improvements on many computer vision tasks [34], [35], [36], [37], [38]. Recent research on clustering has explored the application of deep learning for both dimensionality reduction and clustering, named deep clustering [39], [40]. We divide those techniques into two classes according to how they do their optimisation: The first class includes the techniques, that use one loss function. The second class contains the techniques that use two loss functions (embedding and clustering loss functions). We find in the sequential techniques DEC [7], N2D [17], JULE [8], IMSAT [11], AND DAC [10]. DEC trains an autoencoder to learn features and imposes a soft assignment constraint on them. However, how to effectively pre-train deep networks is an open problem. N2D [17] passes the extracted features using deep autoencoder to UMAP followed by GMM [23] applied on the embedding space to cluster the data. However, its simplicity makes it a trivial method and not suitable for complected data sets.

JULE [8] uses a convolutional neural network with agglomerative clustering loss without the need for any reconstruction loss. In every iteration, hierarchical clustering is performed on the forward pass using an affinity measure and representations are optimised on the backward pass. However, the limitation of JULE is the computational and memory complexity issues, due to the undirected affinity matrix constructed by the agglomerative clustering loss function. IMSAT [11] is based on data augmentation, where the net is trained to maximise the mutual information between data and predicted clusters, while regularising the net so that the cluster assignment of the original data will be consistent with the assignment of augmented data. DAC [10] uses a convolutional neural network with a binary pairwise classification as clustering loss. It is based on the assumption that the relationship between pairwise images is binary. It also adds a regularisation constraint that helps learn label features as one hot encoded features, and the similarity is computed as the dot product of these label features. Spectral-Net [12] is a method based on spectral clustering. It attempts to learn a network that maps the training data into the eigen-space of the graph Laplacian matrix. Then a Siamese network is applied to learn the weights of the connections between the graph's nodes before k-means is used to perform the final clustering.

On the other hand, we find a set of techniques belonging to the interlaced family such as DEPICT [9] and DERC [18]. DEPICT [9] consists of a convolutional autoencoder and a single layer classifier, 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 prior. Deep Embedded Dimensionality Reduction Clustering (DERC) [18] is a combination of deep autoencoder and t-SNE to represent the data. Then, centres of the clusters are initialised using GMM and incorporate a probability-based triplet loss measure to retrain their model and improve the clustering performance of the model.

All the models mentioned above are based on deep learning, which requires tuning of many hyper-parameters. In contrast, our model does not need any supervisory signals for hyper-parameter tuning. The models which use autoencoder to produce the low-dimensional space have the issue of the inherent defects that exist in the embedding space. In addition, their way of optimising the embedding space does not focus on preserving the data's global and local structure. In contrast, our model can maintain the data's global and local structure by learning from the similarities, which leads to a better clustering assignment.

III. 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 as follows (Tab I):

TABLE I
DESCRIPTION OF ALL USED SYMBOLS

Symbols	Description
Y	input data points: $Y = \{y\}_{i=1}^{n}, y_i \in R^f$
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
$p_{i\|j}$	Probability distribution between the input data points 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 data point 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

The proposed algorithm, UEC, aims to preserve the closeness in the input space when mapping the data into the output space. Thus, data points with similar characteristics are projected nearby, and dissimilar ones are mapped apart. Like with general manifold embedding, UEC considers the original data as a high-dimensional graph to be transformed into a lower-dimensional one.

UEC is about mapping high dimensional input data into a lower-dimensional embedded space while considering clustering constraints. To achieve that, UEC tries to optimise 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 two-loss component of the overall function and serve to 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 $\oplus$:

1) $\oplus$ = ',' (comma) that indicates sequential operations: evaluate the embedding using CE, then assess the clustering using KL to update first the embedded data points and then the cluster centres. This scenario is referred to as *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*.

Equation 1 actually depends on a number of quantities (matrices), namely P, Q, T and S described in Table I. The matrix P forms the affinity scores between the individual data points and their nearest neighbours in the input space. The matrix, Q, represents the similarities between the data points in the embedding space. The matrix S is a soft assignment of embedded data to clusters, while T represents the probabilistic point-to-cluster assignments using the obtained soft assignment S (The formal definition of these matrices will follow below). Equation 1 can be rewritten as follows:

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

Clearly, 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 and the centres M. Before delving more into the details, it is worthwhile to portray the structure of the UEC algorithm.

Algorithm 1 UEC

Input: Dataset $Y = \{y_i\}_{i=1}^{n}$, number of cluster C
Output: Embedded dataset $Z = \{z_i\}_{i=1}^{n}$, set of centres $M = \{\mu_k\}_{k=1}^{C}$.

1: Compute the affinity matrix for the input data Y Eq. 3 and 4.
2: Initialise the embedding space of dimension d.
3: Initialise the cluster centres.
4: **while** The convergence criterion is not met **do**
5: Compute the affinity matrix for the embedded data Z, Eq. 5.
6: Compute the soft assignment of the embedded data to the clusters $\mathcal{S}(Z, M)$, Eq. 6.
7: Compute the probabilistic point-to-cluster assignments using the obtained soft assignment $\mathcal{T}(Z, M)$, Eq. 7.
8: Update the coordinates of the embedded data Z, as in Eq. 13.
9: Update the centres, as in Eq. 11.
10: **end while**

IV. FORMULATION OF THE ALGORITHM INGREDIENTS

A. Computation of the affinity matrix P

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

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

where $d(y_i, y_j)$ is the distance between the i^{th} data point and its j^{th} nearest neighbour, ρ_i is the distance between i^{th} data point and its first nearest neighbour. The quantities ρ_i and σ_i ensure the local connectivity of the manifold. We use a symmetrization of the high-dimensional probability, since the weight of the edge between the i^{th} and j^{th} nodes is not equal to the weight between j^{th} and i^{th} nodes. The final formulation of P is given as follows:

$$p_{ij} = p_{i|j} + p_{j|i} - p_{i|j}p_{j|i} \tag{4}$$

B. Initialisation

To initialise the embedding space and the centres of the clusters, we use respectively spectral embedding [41] and a centroid-based algorithm (e.g., k-means, GMM, etc.).

C. Computation of Q

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

$$q_{ij} = (1 + a\left\|z_i - z_j\right\|_2^{2b})^{-1} \tag{5}$$

The parameters a and b are defined using the piece-wise non-linear least-square fitting method.

D. 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:

$$S_{ik} = (1 + a\left\|z_i - \mu_k\right\|_2^{2b})^{-1} \tag{6}$$

We could also consider the following form like in DEC [7] (which is inspired from t-SNE [14]):

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

However, it has been shown in UMAP [15] that the normalisation increases processing time without bringing any improvements in the accuracy. It can be experimentally shown that avoiding normalisation does not affect the quality of clustering.

E. Computation of the auxiliary target distribution T

The matrix T should satisfy three constraints: (1) improving the cluster purity; (2) ensuring high-confidence assignments 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:

$$T_{ik} = \frac{S_{ik}/\sum_l S_{lk}}{\sum_m (S_{lm}/\sum_l S_{lm})} \tag{7}$$

F. Formulation of the optimisation problem

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

$$CE(P, Q) = \sum_i \sum_j [p_{ij} \log(\frac{p_{ij}}{q_{ij}}) + (1 - p_{ij}) \log(\frac{1 - p_{ij}}{1 - q_{ij}})] \tag{8}$$

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 distribution T. KL divergence function refines the clusters by learning from the high-confidence assignments with the help of auxiliary target distribution:

$$KL(T||S) = \sum_i \sum_k T_{ik} \log \frac{T_{ik}}{S_{ik}} \tag{9}$$

Now that the objective function, its individual terms and all quantities used have been defined, we discuss the optimisation problem to update the coordinates of Z and the clusters' centres.

V. Optimisation of the objective function

The coordinates of the data point z_i and cluster centres μ_j are updated at each time step t until the criterion convergence parameter is met. We use Stochastic Gradient Descent (SGD) and negative sampling algorithms as optimisation algorithms due to their rapid convergence and their low memory consumption. The optimisation steps, as illustrated in Alg. 1, are repeated until the change in the cluster assignment of the points is less than a threshold $1e-5$.

As indicated earlier, the update of z_i and μ_j can take place according to two variants.

A. Sequential (Comma) variant

Here the terms of the objective function are sequentially evaluated.

1) Update of the embedding: The coordinates of the data in the embedded space will be updated twice in a sequential manner. The first update stems from the embedding loss 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:

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[\frac{2bp_{ij}}{1/(a\left\|z_i - z_j\right\|_2^{2(b-1)}) + \left\|z_i - z_j\right\|_2^2} - \frac{2b(1 - p_{ij})}{\left\|z_i - z_j\right\|_2^2 (1 + a\left\|z_i - z_j\right\|_2^{2b})} \right] \left\|z_i - z_j\right\| \tag{10}$$

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. 9 w.r.t. z_i reads as follows:

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

2) Update of the cluster centres: The centres μ_k do 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. 9), as follows:

$$\mu_k(t+1) = \mu_k(t) - \eta \frac{\delta KL}{\delta \mu_k} \tag{11}$$

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

$$\frac{\delta KL}{\delta \mu_k} = \sum_i -\frac{2ab\left\|z_i - \mu_k\right\|_2^{2b-1} T_{ik}}{1 + a\left\|z_i - \mu_k\right\|_2^{2b}} \tag{12}$$

B. Combined (plus) variant

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

1) Update of the embedding: According to this second variant, the coordinates of the data points z_i are updated using the embedding and the clustering loss functions simultaneously.

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

$$z_i(t+1) = z_i(t) - \eta \frac{\partial F}{\partial z_i} \tag{13}$$

$$= z_i(t) - \eta \left(\alpha \frac{\delta CE}{\delta z_i} + \beta \frac{\delta KL}{\delta z_i} \right) \tag{14}$$

Where $\frac{\delta CE}{\delta z_i}$ is given by Eq. 10 and $\frac{\delta KL}{\delta z_i}$ is given as follows:

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i} (1 + \log \frac{T_{ik}}{S_{ik}}) + \frac{2bT_{ik}}{1/\left\|z_i - \mu_k\right\|_2^{2b-1} + \left\|z_i - \mu_k\right\|_2^2} \right] + \sum_{i' \neq i} \sum_k \left[\frac{\partial T_{i'k}}{\partial z_i} (1 + \log \frac{T_{i'k}}{S_{i'k}}) \right] \tag{15}$$

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 $i' = i$ and 2). when $i' \neq i$ (please see Appendix B for more details).

The final formulation of the update is obtained by combining Eq. 10 and Eq. 15:

$$\frac{\partial F}{\partial z_i} = \alpha \frac{\partial CE}{\partial z_i} + \beta \frac{\partial KL}{\partial z_i} \tag{16}$$

2) Update of the cluster centres: Since CE doesn't depend on μ_k, only the clustering loss function, KL is relevant:

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

The derivative of the KL divergence is the sum of two parts requiring the consideration of two cases: when $k' = k$ and when $k' \neq k$. For more detail, please, see Appendix B.

C. Light combined 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 improves the performance of the proposed UEC significantly, it is not highly efficient in terms of computational time due to the heavy computation involved in the gradient descent update, hence this 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.

1) Update of the embedding: KL divergence is derived with respect to z_i:

$$\frac{\partial KL}{\partial z_i} = \sum_k \frac{2ab\left\|z_i - \mu_k\right\|_2^{2b-1} T_{ik}}{1 + a\left\|z_i - \mu_k\right\|_2^{2b}} \tag{17}$$

By substituting the derivative of CE (Eq. 10) and the derivative of KL divergence (Eq. 17) in the total loss function (Eq. 16), we then obtain the final update of $z_i(t+1)$ as shown in Eq. 14.

2) Update of the cluster centres: Now for the derivative of total loss F (Eq. 1) with respect to μ_k is computed only for KL-divergence function which depend on the cluster centroids. The formulation obtained in the sequential variant derivations will be applied here for updating the centres (See Eqs. 11 and Eq. 12).

VI. Experiments and discussion

In this section we will show the performance of the proposed 3 variants of UEC on a set of benchmarks. Specifically, we will discuss the following experiments:

- In the first experiment we evaluate the three variants and compare their performance, see Sec. VI-A.
- In the second experiment we study the sensitivity of UEC variants with respect to the initialisation of the clusters' centres, see Sec. VI-B.
- In the third experiment, we discuss the weight α and β in Eq. 1 and their effect, see Sec. VI-C.
- In the fourth experiment, we compare the performance of UEC variants against the state-of-the-art clustering methods, see Sec. VI-D.
- The final experiment discusses the performance of the UEC variants in term of the internal validation clustering, see Sec. VI-E.

Datasets

We conduct experiments on four benchmark datasets given as follows:

1) **USPS**: consists of 9298 images belonging to 10 different classes. Each image is 16x16 gray image [42].
2) **MNIST**: consists of 10 handwritten digits with a total of 70,000 images. Each image is a 28x28 gray image [43].
3) **CIFAR10**: is a dataset of 60000 samples with 10 classes, where each sample is a 32x32 RGB image [44].
4) **Reuters 10K**: is English news data set [45] consisting of 10700 documents. Similar to [7], we use four (4) categories/classes and discard all documents that are labelled by multiple root categories. We removed stop words and used tf-idf representation of 2000 most frequent words.

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, and m is a one to one mapping from predicted label to ground truth label.

1) ***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 represent the perfect correlation. NMI formula is given by:

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

Implementation Details

In the optimisation step of UEC, the minimisation of our weighted sum function is conducted using Stochastic Gradient Descent and Negative Sampling algorithm. We set the learning rate to 1.0 and then decreased it by a factor of 10 every 10 epochs. The number of epochs is set to 200 for big data sets and 500 for small ones. For the image data sets, we use a deep autoencoder to extract the features. Our UEC source code is publicly available on:

A. *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 term of accuracy and run-time execution. Table II represents the results of the three UEC variants evaluated by the accuracy measure, where The experiments are conducted on the original data sets. We can see that the clustering performance of *Plus variant* is better than the other two versions. However, Table III shows that *Comma variant* and *Light plus variant* are better than *Plus variant* in the execution time, due to that Plus variant has a huge amount of calculation operations, as in Appendix B which shows the big number of equations. *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. In contrast, we reduced the execution time.

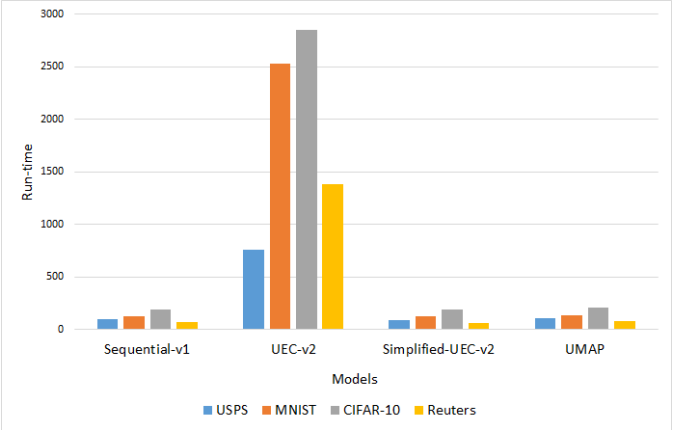

Fig. 1. Execution time (in seconds) using the original data sets.

TABLE II
ACCURACY (ACC) PERFORMANCE USING THE ORIGINAL DATA SETS.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
Comma variant	0.969	0.962	0.478	0.950
Plus variant	0.980	0.988	0.525	0.976
Light plus variant	0.977	0.985	0.513	0.974

TABLE III
EXECUTION TIME (IN SECONDS) USING THE ORIGINAL DATA SETS.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
Comma variant	110	150	215	84
Plus variant	760	2526	2853	1380
Light plus variant	87	115	179	60
UMAP	105	140	210	80

Notice: *Based on the results achieved by the three versions of UEC in the upcoming experiments, in Sec. VI-B and Sec. VI-C, we use only the Light plus variant since it is the fastest one among the three versions and its clustering performance is near to Plus variant.*

B. *Centre Initialisation*

In this experiment, we analyse the sensitivity of our algorithm toward the initialisation of the clusters' cen-

TABLE IV
EFFECT OF CENTRE INITIALISATION ON THE PERFORMANCE.

Algorithm initialisation	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
random	0.972	0.980	0.482	0.970
K-means	0.977	0.985	0.513	0.974
GMM	0.975	0.983	0.510	0.974

TABLE V
EFFECT OF α AND β ON THE PERFORMANCE.

β \ α	0	0.2	0.8	1
0	0.615	0.951	0.955	0.958
0.2	0.957	0.959	0.962	0.965
0.8	0.965	0.968	0.971	0.974
1	0.971	0.973	0.975	0.977

tres. We perform three types of initialisation using k-means, Gaussian mixture models (GMM) and randomly chosen medoids. The experiments are conducted using *Light plus variant* on the original data sets (see Section VI-A). Table IV shows the performance of our algorithm using the three initialisation evaluated by the accuracy measure. We can see that our algorithm is not sensitive to the type of initialisation adopted (even with the random initialisation, it achieves good results).

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

C. *Effect of α and β*

In order to study the effect of the parameter α and β on the loss function. Recall that α is related to the embedding loss, while β is related to clustering. The goal here is to vary them in the unit interval $[0, 1]$ and observe their effect on the algorithm performance and the execution time. The experiments are conducted using *Light plus variant* following the note in Sec. VI-A.

To analyse the effect of the parameters, 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 by varying α and β between the interval $[0, 1]$. Using the USPS dataset, the accuracy results obtained are presented in Tab. V (for more visibility see Fig. 2). Clearly, the algorithm achieves better 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 data points to move, the model makes a quantum leap in its performance. As we see in the Tab. V comparing the outcome when the parameters are set to 0 and 0.2 respectively, the accuracy increases from 0.61 to 0.95. 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).

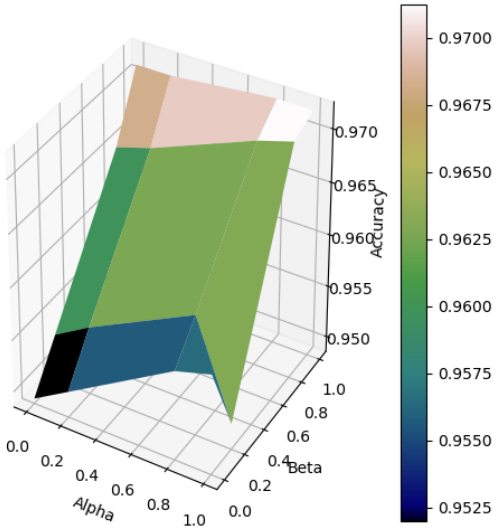

Fig. 2. Effect of α and β on the performance.

TABLE VI
EFFECT OF DIFFERENT VALUES OF α AND β ON THE ACCURACY.

β \ α	0	0.2	0.8	1
0	0.594	0.942	0.948	0.954
0.2	0.949	0.952	0.956	0.960
0.8	0.964	0.966	0.969	0.971
1	0.970	0.971	0.973	0.974

2) **Experiment 2:** we Study the effect of varying α and β on a modified gradient magnitude order of both embedding and clustering loss functions. 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, the majority 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 embedding loss gradient to $[-4, 4]$ and we divide the values by 4 to bring it to the interval $[-1, 1]$. The values of clustering loss gradient are clipped to $[0, 1]$. Table VI (see also Fig. 3 for more visibility) shows the accuracy of the algorithm after the clipping transformation. We can observe that there is a slight positive change in the performance results.

D. *Comparative Study*

1) Baseline Methods: The proposed UEC is compared with a set of clustering methods including state-of-the-art deep clustering methods: k-means [21], deep embedded clustering (DEC) [7], Joint Unsupervised Learning (JULE) [8], Deep Embedded Regularised Clustering (DEPICT) [9], Deep Adaptive Clustering (DAC) [10],

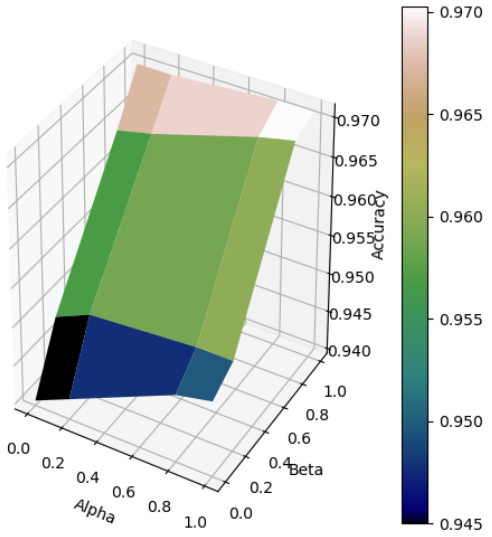

Fig. 3. Effect of α and β on the accuracy.

TABLE VII
UEC VS. OTHER BASELINES: ACCURACY SCORES.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
k-means [21]	0.668	0.572	0.228	0.524
DEC (2016) [7]	0.619⋆	0.843	0.301⋆⋆	0.722
JULE (2016) [8]	0.950⋆	0.964⋆	0.271⋆⋆	-
IMSAT (2017) [11]	-	**0.984**	0.456	0.719
DEPICT (2017) [9]	0.964	0.965	-	-
DAC (2017) [10]	-	0.978	**0.522**	-
Spectral Net (2018) [12]	-	0.971	-	0.803
DGG (2019) [13]	-	0.976	-	**0.823**
DERC (2020) [18]	**0.977**	0.975	-	-
UEC (Plus variant)	**0.980**	**0.988**	**0.525**	**0.976**
UEC (Light plus variant)	**0.977**	**0.985**	**0.513**	**0.974**

TABLE VIII
UEC VS. OTHER BASELINES: NMI SCORES.

Models	Dataset		
	USPS	MNIST	CIFAR-10
k-means [21]	0.450	0.499	0.087
DEC (2016) [7]	0.586⋆	0.816⋆	0.256⋆⋆
JULE (2016) [8]	0.913	0.913	0.192⋆⋆
DEPICT (2017) [9]	0.927	0.917	-
DAC (2017) [10]	-	**0.935**	**0.395**
Spectral Net (2018) [12]	-	0.924	-
DERC (2020) [18]	**0.942**	0.927	-
UEC (Plus variant)	**0.948**	**0952**	**0.407**
UEC (Light plus variant)	**0.933**	**0.948**	**0.394**

Information Maximising Self-Augmented Training IMSAT [11]. Spectral Net [12], and Deep clustering via a Gaussian mixture variations autoencoder (VAE) with Graph embedding (DGG) [13]. For these methods, the performance results are taken from the original publications.

2) Experiment results: Tables VII and VIII outline the performance in terms of accuracy and NMI, respectively. The top three accuracy scores are highlighted, while those marked by ($\star$) and ($\star\star$) are reported in [9] and [10] respectively. The dash mark (-) means that the result was not reported in the referenced paper. Table VII shows that UEC outperforms the other algorithms across all benchmarks. It outperforms most of the deep clustering methods by a significant margin. It is worthwhile to mention that deep autoencoders improved the performance of the UEC algorithm. This later did not need any fine-tuning, in contrast to other deep clustering models, which require tweaking several hyper-parameters and fine-tuning to achieve their better results. This advantage makes UEC significantly a better embedding-and-clustering choice compared to the other alternative clustering models.

Tables VII and VIII outline the performance in terms of accuracy and NMI, respectively. The results achieved by UEC or even by other methods prove that representation learning plays an important role in the clustering process. The fact that UEC is a manifold-based embedding method helps in the clustering process and improves its performance.

E. Internal validation

The goal of clustering is to assign similar data objects to the same cluster and dissimilar ones to different clusters. Internal validation intends to quantify the quality of clustering usually using two criteria: *Compactness* and *Separation*. The first criterion measures how similar the data objects are 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 these experiments we use three well-known internal validation indices which are Davis-Bouldin (DB) [46], Silhouette coefficient (S) [47] and Calinski-Harabaz (CH) [48]. 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.

Tables IX, X and XI show the clustering quality scores of the UEC variants. They indicate that the *Plus variant* is the best among the three variants across comparing the three indices. In fact, the UEC variants obtain approxi-

TABLE IX
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 X
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 XI
CLUSTERING QUALITY USING THE CALINSKI-HARABAZ INDEX.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
Comma variant	1123.754	9682.256	997.84	16481.130
Plus variant	1398.425	10189.580	1289.475	20142.254
Light plus variant	1211.068	9819.213	1179.962	18248.961

mately 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 quite weak, 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 are capable of preserving the between and within-cluster distances because they are designed with the goal of preserving the global and local structures of the data.

VII. Conclusion

We have proposed a new algorithm, UEC, that jointly optimise the representation and clustering of data. UEC is a manifold-based clustering algorithm that has the capability to preserve the local and the global structure of data and that 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 a number of large-scale benchmarks and against a number of baseline algorithms.

Future work investigates the different optimisation techniques and the objective functions to improve the performance of the algorithm in terms of the evaluation measures and run-time.

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:

$$q_{ij} = (1 + a\left\|z_i - z_j\right\|_2^{2b})^{-1} \tag{18}$$

$$\frac{\partial q_{ij}}{\partial z_i} = \frac{-2ab\left\|z_i - z_j\right\|_2^{2b-1}}{(1 + a\left\|z_i - z_j\right\|_2^{2b})^2} \tag{19}$$

Now we can derive the CE objective function as follows:

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

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

Since p_{ij} doesn't 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 derived the CE as follow:

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[-\frac{\partial q_{ij}}{\partial z_i} \frac{p_{ij}}{q_{ij}} - \frac{\partial (1 - q_{ij})}{\partial z_i} \frac{1 - p_{ij}}{1 - q_{ij}} \right] \tag{20}$$

We have $1 - q_{ij} = \frac{a\left\|z_i - z_j\right\|_2^{2b}}{1 + a\left\|z_i - z_j\right\|_2^{2b}}$, now we will substitute the last one, Eq. 18 and Eq. 19 in the Eq. 20 so we can rewrite it as follow:

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

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

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

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

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

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[\frac{2bp_{ij}}{1/a\left\|z_i - z_j\right\|_2^{2(b-1)} + \left\|z_i - z_j\right\|_2^2} - \frac{2b(1-p_{ij})}{\left\|z_i - z_j\right\|_2^2 (1 + a\left\|z_i - z_j\right\|_2^{2b})} \right] \left\|z_i - z_j\right\| \quad (21)$$

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:

$$S_{ik} = (1 + a\left\|z_i - \mu_k\right\|_2^{2b})^{-1} \quad (22)$$

$$T_{ik} = \frac{S_{ik}/\sum_l S_{lk}}{\sum_m (S_{im}/\sum_l S_{lm})} \quad (23)$$

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

$$\frac{\partial F}{\partial z_i} = \alpha \frac{\partial CE}{\partial z_i} + \beta \frac{\partial KL}{\partial z_i} \quad (24)$$

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.

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

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

The derivative is formulated as a sum of two parts since T_{ik} needs to compute its derivative with respect z_i according to two cases. The first one when $i' = i$ and the second case when $i' \neq i$. Since i' is different from i, $S_{i'k}$ does not depend on z_i (see Eq. 22), $\frac{\partial S_{i'k}}{\partial z_i} = 0$. We can simplify the derivative as follows:

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i} (1 + \log \frac{T_{ik}}{S_{ik}}) - \frac{\partial S_{ik}}{\partial z_i} \frac{T_{ik}}{S_{ik}} \right] + \sum_{i' \neq i} \sum_k \left[\frac{\partial T_{i'k}}{\partial z_i} (1 + \log \frac{T_{i'k}}{S_{i'k}}) \right] \quad (25)$$

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

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

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

$$\frac{\partial S_{ik}}{\partial z_i} = -\frac{2b}{1/\left\|z_i - \mu_k\right\|_2^{2b-1} + \left\|z_i - \mu_k\right\|_2^2} S_{ik} \quad (26)$$

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. 23.

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

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

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:

$$N_{ik} = \frac{S_{ik}}{\sum_l S_{lk}} \quad (27)$$

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:

$$\frac{\partial N_{ik}}{\partial z_i} = \frac{\frac{\partial S_{ik}}{\partial z_i} \sum_{l \neq i} S_{lk}}{(\sum_l S_{lk})^2} \quad (28)$$

For the denominator which is expressed as:

$$D_i = \sum_m \frac{S_{im}}{\sum_l S_{lm}} \quad (29)$$

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:

$$\frac{\partial D_i}{\partial z_i} = \sum_m \frac{\frac{\partial S_{im}}{\partial z_i} \sum_{l \neq i} S_{lm}}{(\sum_l S_{lm})^2} \quad (30)$$

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

$$\frac{\partial T_{ik}}{\partial z_i} = \frac{\frac{\partial N_{ik}}{\partial z_i} D_i - N_{ik} \frac{\partial D_i}{\partial z_i}}{D_i^2} \quad (31)$$

where N_{ik}, $\frac{\partial N_{ik}}{z_i}$, D_i, $\frac{\partial D_i}{z_i}$ are substituted by Eqs. 27, 28, 29, and 30 respectively in Eq.31.

Now we compute the derivative of $T_{i'k}$ with respect to z_i. In Eq. 25, we use $T_{i'k}$ to denote the case where

$i' \neq i$.. The numerator and the denominator are derived separately. Like in Eq. 27, the numerator is given:

$$N_{i'k} = \frac{S_{i'k}}{\sum_l S_{lk}} \tag{32}$$

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:

$$\frac{\partial N_{i'k}}{\partial z_i} = \frac{-S_{i'k}\frac{\partial S_{ik}}{\partial z_i}}{(\sum_l S_{lk})^2} \tag{33}$$

where $\frac{\partial S_{ik}}{\partial z_i}$ is given by Eq. 26.

Like in Eq. 28, the denominator is given as follows:

$$D_{i'} = \sum_m \frac{S_{i'm}}{\sum_l S_{lm}} \tag{34}$$

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 $\frac{\partial S_{im}}{\partial z_i} = -\frac{2ab\left\|z_i-\mu_k\right\|_2^{2b-1}}{(1+a\left\|z_i-\mu_k\right\|_2^{2b})^2}$ and $\frac{\partial S_{lm}}{\partial z_i} = 0$ otherwise. We can write the derivative of D as follows:

$$\frac{\partial D_{i'}}{\partial z_i} = \sum_m \frac{-S_{i'm}\frac{\partial S_{im}}{\partial z_i}}{(\sum_l S_{lm})^2} \tag{35}$$

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

$$\frac{\partial T_{i'k}}{\partial z_i} = \frac{\frac{\partial N_{i'k}}{\partial z_i} D_{i'} - N_{i'k}\frac{\partial D_{i'}}{\partial z_i}}{D_{i'}^2} \tag{36}$$

where $N_{i'k}$, $\frac{\partial N_{i'k}}{z_i}$, $D_{i'}$, $\frac{\partial D_{i'}}{z_i}$ are substituted by Eqs. 32, 33, 34, 35 respectively in Eq.36.

Now let's recall the derivative of the KL-divergence, Eq. 25:

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

We substitute S_{ik} and $\frac{\partial S_{ik}}{\partial z_i}$ by Eqs.22 and 26 in Eq. 25. to obtain:

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

In Eq. 25, The term $\frac{\partial S_{ik}}{\partial z_i}\frac{T_{ik}}{S_{ik}}$ can be more simpler as follow:

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

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:

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i}(1+\log\frac{T_{ik}}{S_{ik}}) + \frac{2bT_{ik}}{1/\left\|z_i-\mu_k\right\|_2^{2b-1}+\left\|z_i-\mu_k\right\|_2^2}\right] + \sum_{i'\neq i}\sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1+\log\frac{T_{i'k}}{S_{i'k}})\right] \tag{37}$$

A. 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. 21) and the derivative of KL divergence (Eq. 37) in the derivative of the total loss function to obtain:

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

$$\begin{aligned}\frac{\partial F}{\partial z_i} = \alpha\sum_j \Bigg[&\frac{2bp_{ij}}{1/a\left\|z_i-z_j\right\|_2^{2(b-1)}+\left\|z_i-z_j\right\|_2^2} \\ &-\frac{2b(1-p_{ij})}{\left\|z_i-z_j\right\|_2^2(1+a\left\|z_i-z_j\right\|_2^{2b})}\Bigg]\left\|z_i-z_j\right\|_2 \\ &+\beta\Bigg(\sum_k\Bigg[\frac{\partial T_{ik}}{\partial z_i}(1+\log\frac{T_{ik}}{S_{ik}}) \\ &+\frac{2bT_{ik}}{1/\left\|z_i-\mu_k\right\|_2^{2b-1}+\left\|z_i-\mu_k\right\|_2^2}\Bigg] \\ &+\sum_{i'\neq i}\sum_k\left[\frac{\partial T_{i'k}}{\partial z_i}(1+\log\frac{T_{i'k}}{S_{i'k}})\right]\Bigg)\end{aligned} \tag{38}$$

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

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

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

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

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

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

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

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

Define $S_{ik} : \mathbb{R}^d \times \mathbb{R}^d \to [0,1]$, a soft assignment between the data points 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 $g(z_i, \mu_k) = \log \frac{T_{ik}}{S_{ik}}$

$$\text{g is defined if and only if } \Leftrightarrow \begin{cases} S_{ik} \neq 0 & (1) \\ \frac{T_{ik}}{S_{ik}} > 0 \end{cases} \tag{39}$$

Where:

$$\frac{T_{ik}}{S_{ik}} > 0 \Leftrightarrow \begin{cases} T_{ik} \neq 0 & (2) \\ S_{ik} \text{ and } T_{ik} \text{ has the same sign} & (3) \end{cases} \tag{40}$$

As we see in the previews Eq. 39 we have three condition let us treat them separately. The first condition is satisfied if and only if:

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

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

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

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

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

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 follow, the first one is shown in algorithm B-A:

Algorithm 2 Solving the problem of division by zero

```
if b ≥ 0 then
  if ‖z_i − μ_k‖_2 > Dist − Max then
    we have two solution:
    Consider z_i as an outlier
    Replace ‖z_i − μ_k‖_2 by Dist-Max
  end if
else
  Solve the cases where z_i ≈ μ_k
end if
```

And the second solution is as follow:

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

B. 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's recall our weighted sum function:

$$F(P, Q, T, S) = \alpha CE(P, Q) + \beta KL(T||S) \tag{41}$$

Such that:

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

$$\begin{aligned} KL(T||S) &= \sum_i \sum_k T_{ik} \log \frac{T_{ik}}{S_{ik}} \\ &= \sum_i \sum_k T_{ik} \log T_{ik} - T_{ik} \log S_{ik} \end{aligned}$$

Also let's 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 doesn't depend on cluster centroids, so its derivative is zero. Therefore, we only compute the derivative of the KL divergence with respect to μ_k:

$$\begin{aligned} \frac{\partial KL}{\partial \mu_k} = \sum_i \Bigg[& \frac{\partial T_{ik}}{\partial \mu_k} \log T_{ik} + T_{ik} \frac{\frac{\partial T_{ik}}{\partial \mu_k}}{T_{ik}} \\ & - \frac{\partial T_{ik}}{\partial \mu_k} \log S_{ik} - \frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} \Bigg] \\ + \sum_i \sum_{k' \neq k} \Bigg[& \frac{\partial T_{ik'}}{\partial \mu_k} \log T_{ik'} + T_{ik'} \frac{\frac{\partial T_{ik'}}{\partial \mu_k}}{T_{ik'}} \\ & - \frac{\partial T_{ik'}}{\partial \mu_k} \log S_{ik'} - \frac{\partial S_{ik'}}{\partial \mu_k} \frac{T_{ik'}}{S_{ik'}} \Bigg] \end{aligned}$$

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 when $k' = k$ and the second case when $k' \neq k$. Also, since $S_{ik'}$ doesn't depend on μ_k so $\frac{\partial S_{ik'}}{\partial \mu_k} = 0$.

$$\begin{aligned} \frac{\partial KL}{\partial \mu_k} = \sum_i \Bigg[& \frac{\partial T_{ik}}{\partial \mu_k} \log T_{ik} + \frac{\partial T_{ik}}{\partial \mu_k} - \frac{\partial T_{ik}}{\partial \mu_k} \log S_{ik} - \frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} \Bigg] \\ + \sum_i \sum_{k' \neq k} \Bigg[& \frac{\partial T_{ik'}}{\partial \mu_k} \log T_{ik'} + \frac{\partial T_{ik'}}{\partial \mu_k} - \frac{\partial T_{ik'}}{\partial \mu_k} \log S_{ik'} \Bigg] \end{aligned}$$

$$\begin{aligned} \frac{\partial KL}{\partial \mu_k} = \sum_i \Bigg[& \frac{\partial T_{ik}}{\partial \mu_k} (1 + \log \frac{T_{ik}}{S_{ik}}) - \frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} \Bigg] \\ & + \sum_i \sum_{k' \neq k} \Bigg[\frac{\partial T_{ik'}}{\partial \mu_k} (1 + \log \frac{T_{ik'}}{S_{ik'}}) \Bigg] \end{aligned} \tag{42}$$

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 $\frac{\partial S_{ik}}{\partial \mu_k}$

$$\frac{\partial S_{ik}}{\partial \mu_k} = \frac{2bS_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2} \tag{43}$$

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

$$\frac{\partial S_{ik'}}{\partial \mu_k} = 0 \tag{44}$$

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's derive the numerator and the denominator each separately. We have the numerator is:

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

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. 43 and $\frac{\partial S_{lk}}{\partial z_i}$ is deduced from Eq. 43 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 is:

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

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 derived the numerator and denominator separately:

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

Since $S_{ik'}$ doesn't depend on μ_k, we can deduce that $\frac{\partial S_{ik'}}{\partial \mu_k} = 0$ from Eq. 22 and 43. Also for $\sum_l S_{lk'}$ doesn't 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:

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

And from previews equations we can conclude that the derivative of $T_{ik'}$ as follow:

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

Now we are going to substitute the previews equations in the derivative of the KL divergence Eq. 42 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 Eq. 22 and Eq. 43, so let's recall $\frac{\partial S_{ik}}{\partial \mu_k}$:

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

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

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

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:

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

Appendix C
Light plus variant derivations

Our optimisation purpose is to update the data points 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:

$$\frac{\partial F}{\partial z_i} = \alpha \frac{\partial CE}{\partial z_i} + \beta \frac{\partial KL}{\partial z_i} \tag{45}$$

A. Derivation of the KL divergence with respect to z_i

Notice that for the data points 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's recall the KL divergence:

$$KL(T, S) = \sum_i \sum_k T_{ik} \log T_{ik} - T_{ik} \log S_{ik} \tag{46}$$

When updating z_i, T_{ik} is already computed and is considered as a constant number, T_{ik} doesn't depend on z_i, thus the derivative of $T_{ik} \log T_{ik}$ with respect to z_i is zero. We obtain:

$$\frac{\partial KL}{\partial z_i} = \sum_k - \frac{\partial S_{ik}}{\partial z_i} \frac{T_{ik}}{S_{ik}} \tag{47}$$

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

$$\frac{\partial S_{ik}}{\partial z_i} = - \frac{2ab\left\|z_i - \mu_k\right\|_2^{2b-1}}{(1 + a\left\|z_i - \mu_k\right\|_2^{2b})^2} \tag{48}$$

Substituting S_{ik} and its derivative $\frac{\partial S_{ik}}{\partial z_i}$ (Eq. 48) in Eq. 47. we can conclude $\frac{\partial KL}{\partial z_i}$ like this:

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

$$\frac{\partial KL}{\partial z_i} = \sum_k \frac{2ab\left\|z_i-\mu_k\right\|_2^{2b-1} T_{ik}}{1+a\left\|z_i-\mu_k\right\|_2^{2b}} \tag{49}$$

B. 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. 21), and the derivative of KL divergence (Eq. 49) in the total loss function (Eq. 45) to obtain:

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

$$\frac{\partial F}{\partial z_i} = \alpha\sum_j \left[\frac{2ab\left\|z_i-z_j\right\|_2^{2(b-1)} p_{ij}}{1+a\left\|z_i-z_j\right\|_2^{2b}} - \frac{2b(1-p_{ij})}{\left\|z_i-z_j\right\|_2^2(1+a\left\|z_i-z_j\right\|_2^{2b})}\right]\left\|z_i-z_j\right\| + \beta\sum_k \frac{2ab\left\|z_i-\mu_k\right\|_2^{2b-1} T_{ik}}{1+a\left\|z_i-\mu_k\right\|_2^{2b}} \tag{50}$$

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

C. The derivative of Objective Function with respect to the μ_i

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

$$F(P,Q,T,S) = \alpha CE(P,Q) + \beta KL(T||S) \tag{51}$$

Such that:

$$CE(P,Q) = \sum_i\sum_j\left[p_{ij}\log(\frac{p_{ij}}{q_{ij}}) + (1-p_{ij})\log(\frac{1-p_{ij}}{1-q_{ij}})\right]$$
$$KL(T||S) = \sum_i\sum_k T_{ik}\log T_{ik} - T_{ik}\log S_{ik}$$

Also let's 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 doesn't 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} doesn't 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:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i -\frac{\partial S_{ik}}{\partial \mu_k}\frac{T_{ik}}{S_{ik}} \tag{52}$$

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:

$$\frac{\partial S_{ik}}{\partial \mu_k} = \frac{2ab\left\|z_i-\mu_k\right\|_2^{2b-1}}{(1+a\left\|z_i-\mu_k\right\|_2^{2b})^2} \tag{53}$$

Substituting S_{ik} and its derivative $\frac{\partial S_{ik}}{\partial \mu_k}$ (Eq. 53) in Eq. 52. we obtain:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i -\frac{2ab\left\|z_i-\mu_k\right\|_2^{2b-1} T_{ik}}{1+a\left\|z_i-\mu_k\right\|_2^{2b}} \tag{54}$$

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

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

We have the gradient of the CE as follow:

$$\frac{\partial CE}{\partial z_i} = \sum_j\left[\frac{2bp_{ij}}{1/(a\left\|z_i-z_j\right\|_2^{2(b-1)}) + \left\|z_i-z_j\right\|_2^2} - \frac{2b(1-p_{ij})}{\left\|z_i-z_j\right\|_2^2(1+a\left\|z_i-z_j\right\|_2^{2b})}\right]\left\|z_i-z_j\right\| \tag{55}$$

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

$$\frac{2bp_{ij}}{1/(a\left\|z_i-z_j\right\|_2^{2(b-1)}) + \left\|z_i-z_j\right\|_2^2} \tag{56}$$

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 $\left\|z_i-z_j\right\|_2^{2(b-1)}$ and $\left\|z_i-z_j\right\|_2^2$ are always positives so the values domain of the first term (Eq. 56) is $[0,+\infty[$. Now let us move to the second term:

$$-\frac{2b(1-p_{ij})}{\left\|z_i-z_j\right\|_2^2(1+a\left\|z_i-z_j\right\|_2^{2b})} \tag{57}$$

In above term (Eq. 57), 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 $]-\infty,+\infty[$.

B. 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.

1) The derivative of the KL divergence Plus variant: Let us recall the derivative of the KL divergence with respect to z_i:

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

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

2) The derivative of the KL divergence Light Plus variant: Let us recall the derivative of the KL divergence with respect to z_i:

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

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

References

[1] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[2] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[3] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[4] V. Roth and T. Lange, "Feature selection in clustering problems," *Advances in neural information processing systems*, vol. 16, pp. 473–480, 2003.

[5] F. Tian, B. Gao, Q. Cui, E. Chen, and T.-Y. Liu, "Learning deep representations for graph clustering," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, no. 1, 2014.

[6] Z. Wang, S. Chang, J. Zhou, M. Wang, and T. S. Huang, "Learning a task-specific deep architecture for clustering," in *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM, 2016, pp. 369–377.

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

[8] J. Yang, D. Parikh, and D. Batra, "Joint unsupervised learning of deep representations and image clusters," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5147–5156.

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

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

[11] W. Hu, T. Miyato, S. Tokui, E. Matsumoto, and M. Sugiyama, "Learning discrete representations via information maximizing self-augmented training," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1558–1567.

[12] U. Shaham, K. Stanton, H. Li, B. Nadler, R. Basri, and Y. Kluger, "Spectralnet: Spectral clustering using deep neural networks," *arXiv preprint arXiv:1801.01587*, 2018.

[13] L. Yang, N.-M. Cheung, J. Li, and J. Fang, "Deep clustering by gaussian mixture variational autoencoders with graph embedding," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6440–6449.

[14] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[15] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[16] M. Allaoui, M. L. Kherfi, and A. Cheriet, "Considerably improving clustering algorithms using umap dimensionality reduction technique: A comparative study," in *International Conference on Image and Signal Processing*. Springer, 2020, pp. 317–325.

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

[18] Y. Yan, H. Hao, B. Xu, J. Zhao, and F. Shen, "Image clustering via deep embedded dimensionality reduction and probability-based triplet loss," *IEEE Transactions on Image Processing*, vol. 29, pp. 5652–5661, 2020.

[19] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." *Journal of machine learning research*, vol. 11, no. 12, 2010.

[20] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[21] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14. Oakland, CA, USA, 1967, pp. 281–297.

[22] J. Wang, J. Wang, Q. Ke, G. Zeng, and S. Li, "Fast approximate k-means via cluster closures," in *Multimedia data mining and analytics*. Springer, 2015, pp. 373–395.

[23] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[24] K. C. Gowda and G. Krishna, "Agglomerative clustering using the concept of mutual nearest neighbourhood," *Pattern recognition*, vol. 10, no. 2, pp. 105–112, 1978.

[25] W.-T. Wang, Y.-L. Wu, C.-Y. Tang, and M.-K. Hor, "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. IEEE, 2015, pp. 445–451.

[26] R. J. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Pacific-Asia conference on knowledge discovery and data mining*. Springer, 2013, pp. 160–172.

[27] A. Y. Ng, M. I. Jordan, Y. Weiss *et al.*, "On spectral clustering: Analysis and an algorithm," *Advances in neural information processing systems*, vol. 2, pp. 849–856, 2002.

[28] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[29] D. Xu and Y. Tian, "A comprehensive survey of clustering algorithms," *Annals of Data Science*, vol. 2, no. 2, pp. 165–193, 2015.

[30] K. Pearson, "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*, vol. 2, no. 11, pp. 559–572, 1901.

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

[32] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.
[35] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.
[36] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.
[37] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.
[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[39] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long, "A survey of clustering with deep learning: From the perspective of network architecture," *IEEE Access*, vol. 6, pp. 39 501–39 514, 2018.
[40] E. Aljalbout, V. Golkov, Y. Siddiqui, M. Strobel, and D. Cremers, "Clustering with deep learning: Taxonomy and new methods," *arXiv preprint arXiv:1801.07648*, 2018.
[41] B. Luo, R. C. Wilson, and E. R. Hancock, "Spectral embedding of graphs," *Pattern recognition*, vol. 36, no. 10, pp. 2213–2230, 2003.
[42] J. J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 16, no. 5, pp. 550–554, 1994.
[43] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[44] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.
[45] D. D. Lewis, Y. Yang, T. Russell-Rose, and F. Li, "Rcv1: A new benchmark collection for text categorization research," *Journal of machine learning research*, vol. 5, no. Apr, pp. 361–397, 2004.
[46] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE transactions on pattern analysis and machine intelligence*, no. 2, pp. 224–227, 1979.
[47] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.
[48] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics-theory and Methods*, vol. 3, no. 1, pp. 1–27, 1974.