Deep Clustering for Improved Inter-Cluster Separability and Intra-Cluster Homogeneity with Cohesive Loss

Byeonghak KIM†, Murray LOEW††††, David K. HAN†††, Nonmembers, and Hanseok KO††††, Member

SUMMARY To date, many studies have employed clustering for the classification of unlabeled data. Deep separate clustering applies several deep learning models to conventional clustering algorithms to more clearly separate the distribution of the clusters. In this paper, we employ a convolutional autoencoder to learn the features of input images. Following this, k-means clustering is conducted using the encoded layer features learned by the convolutional autoencoder. A center loss function is then added to aggregate the data points into clusters to increase the intra-cluster homogeneity. Finally, we calculate and increase the inter-cluster separability. We combine all loss functions into a single global objective function. Our new deep clustering method surpasses the performance of existing clustering approaches when compared in experiments under the same conditions.

1. Introduction

In supervised learning, data classification is a particularly important task. However, individually labeling data points requires significant time and effort, and it is often impossible to fully label datasets for research applications. To overcome this problem, clustering via unsupervised learning has been proposed and is widely utilized. Clustering effectively groups unlabeled data based on specific criteria of similarity and can automatically extract semantic information that humans cannot abstract.

Many data-mining researchers have investigated various types of clustering. Both hard and soft clustering is possible depending on whether an observation point belongs to one or to multiple clusters. Hard clustering includes $k$-means clustering [1], $k$-medoid clustering [2], density-based spatial clustering of applications with noise (DBSCAN) [3], hierarchical clustering [4], random binary pattern of patch clustering (RBPPC) [5]. Soft clustering includes Gaussian mixture model-based clustering [6] and fuzzy clustering [7]. Recently, a deep clustering method that extracts features to be used for data representation based on deep learning has been proposed.

An autoencoder is an unsupervised feature extraction tool used in deep clustering. Representative autoencoder-based deep clustering algorithms include autoencoder-based deep clustering (ABDC) [8], deep embedded clustering (DEC) [9], improved deep embedded clustering (IDEC) [10], discriminatively boosted clustering (DBC) [11], and deep embedded regularized clustering (DEPICT) [12]. They are all based on the idea that the neural network learns the features that are suitable for clustering. In addition, deep embedded clustering with data augmentation (DEC-DA) [13] employs random rotation, cropping, shearing, and shifting on the data to generalize the model. Yet, these techniques do not exhibit strong inter-cluster separability nor robust intra-cluster homogeneity, leaving a room for improvements.

In this paper, we propose a new clustering algorithm with the ability to separate clusters more effectively than existing deep clustering algorithms by increasing the inter-cluster separability and the intra-cluster homogeneity. Our deep separate clustering algorithm focuses on making the scattered samples characterized by Gaussian distribution more cohesive between very similar data but, when the similarity is low, it separates the data points belonging to more corresponding clusters. This process can also be effective when processing cluster-overlapping data. Our proposed method greatly improves on the performance of existing deep clustering algorithms when tested on public datasets.

In Sect. 2, the proposed method is described. The process and results of the experiment are presented in Sect. 3. Finally, Sect. 4 provides the conclusions.

2. Proposed Deep Clustering with Cohesive Loss

In this section, we propose an innovative deep clustering algorithm based on the separation between clusters using four loss functions for global optimization. First, deep features are learned while reconstructing unlabeled data with a convolutional autoencoder (CAE). Upon the completion of pretraining of the autoencoder, the second stage is focused on clustering of deep features using the distances between the features assigned to the same cluster and the distances between data points assigned to different clusters in an end-to-end manner.
2.1 Feature Extraction with a Convolutional Autoencoder

An autoencoder is a deep learning approach that learns the features of unlabeled data and that is widely employed as a feature extractor. The structure is shown in Fig. 1.

In training set \( X = \{ x_i \in \mathbb{R}^D \}_{i=1}^m \), \( x_i \) denotes the \( i \)-th training data point from \( m \) data points, and \( D \) is the dimension of \( x_i \). The autoencoder loss function is

\[
L_{ae} = ||X - X'||^2 \quad (1)
\]

In addition, because the autoencoder learns features in order to reconstruct input \( X \) as accurately as possible using output \( X' \) through the encoder and decoder, autoencoder loss is also referred to as reconstruction loss. Equation (1) can thus be expressed as follows:

\[
L_r = \frac{1}{m} \sum_{i=1}^m ||x_i - x'_i||^2 \quad (2)
\]

The embedded features learned by Eq. (2) become the input for the subsequent clustering algorithm described in Sects. 2.2 and 2.3.

2.2 Loss Function for Deep Embedded Clustering

Autoencoder-based clustering is also known as deep embedded clustering. The main idea of deep embedded clustering algorithms is to perform clustering the features obtained from autoencoders. As mentioned earlier, [13] and also improve model generalization using data augmentation techniques. For this reason, we employ the data augmentation.

The features in deep embedded clustering generally have smaller dimensions than the input and output data. In this paper, we perform clustering on embedded features \( Z = \{ z_i \in \mathbb{R}^q \}_{i=1}^m \), where \( q \) is the embedded feature size and \( m \) is the number of data points. The clustering loss function is calculated by applying the Student’s \( t \)-distribution [14] results obtained by performing soft deep clustering using the network in Sect. 2.1 to Kullback-Leibler divergence (KLD) [15].

In more detail, \( k \)-means clustering is performed on \( Z \) to \( M = \{ \mu_j \in \mathbb{R}^q \}_{j=1}^n \), where \( \mu_j \) is the \( j \)-th element of \( n \) centroids. The output distribution \( q_{ij} \) resulting from clustering with \( Z \) and \( M \) can be defined as follows based on the Student’s \( t \)-distribution:

\[
q_{ij} = \frac{(1 + ||z_i - \mu_j||^2)^{-1}}{\sum_j(1 + ||z_i - \mu_j||^2)^{-1}} \quad (3)
\]

The target distribution \( p_{ij} \) can be expressed based on \( q_{ij} \):

\[
p_{ij} = \frac{Q^T \sum_i q_{ij}}{\sum_j(Q^T \sum_i q_{ij})} \quad (4)
\]

where \( Q = (e^{q_{ij}} - 1)/(e - 1) \) and \( T > 0 \). We set this \( T \) value to 3. The range of \( Q^T \) is 0 to 1 because \( q_{ij} \) has a value from 0 to 1. In Eq. (4), the exponential function constructs \( p \) by adding nonlinearity to \( q \). Therefore, the target distribution \( p \) enhances the prediction by giving more emphasis to the cluster assignments with high probability in \( q \). In addition, the loss is regularized to prevent distortion of the entire feature space by different contributions to the loss depending on the density of the cluster, \( \sum_i q_{ij} \). Finally, the clustering loss function is

\[
L_c = D_{KL}(P \| Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad (5)
\]

2.3 The Proposed Discriminative Cluster Loss Function

The final goal of clustering is to assign data points with a strong similarity to the same cluster with a certain similarity measurement. Therefore, to achieve robust clustering, both the intra-cluster homogeneity and the inter-cluster separability should be increased. In this study, \( L_w \) is used to minimize the intra-cluster distance and \( L_B \) reduces the inter-cluster proximity. \( L_w \) is a function that represents the variations between the cluster centroids and the inner cluster points, as in [16]. \( L_w \) increases the homogeneity of the inner cluster points by making the data points in the same cluster gather around the centroid. \( L_w \) is represented by (6):

\[
L_w = \frac{1}{2} \sum_{i=1}^m ||z_i - \mu_j||^2 \quad (6)
\]

where \( y_i \) is the predicted cluster label for the \( i \)-th sample and \( \mu_j \in \mathbb{R}^d \) is the centroid of the \( y_i \)-th cluster. \( \mu_j \) can be updated for an iterative training process. \( L_B \) is a function calculated from the inter-cluster cosine distance. Thus, smaller \( L_B \) means larger inter-cluster distances. \( L_B \) can be derived as follows:

\[
L_B = \frac{1}{2C} \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} \log \left( \text{ReLU} \left( \frac{\mu_j \cdot \mu_k}{||\mu_j||_2 ||\mu_k||_2} \right) + 1 + \epsilon \right) \quad (7)
\]

where \( n \) denotes the number of clusters and \( \mu_j \) and \( \mu_k \) are the \( j \)-th and \( k \)-th centroids of the embedded features from the CAE, respectively. \( nC \) represents the number of all combinations when pairing two in \( n \) clusters.

An ReLU activation function is used to prevent a negative value for the centroids of all clusters in combination; as such, 1 and \( \epsilon \) prevent the output of the log function from becoming negative or zero, respectively. Because \( L_B \) is
significantly larger than our other loss functions, it is log-
transformed for normalization.

Finally, the global loss function in Eq. (8) is created by
combining the loss functions in Eqs. (2), (5), (6), and (7):

\[ L = \alpha \cdot L_r + \beta \cdot L_c + \gamma \cdot L_W + \omega \cdot L_B \]  

(8)

where \( \alpha, \beta, \gamma, \) and \( \omega \) are the heuristically determined loss
weight parameters. Competition among these losses may
cause oscillations in the learning process, however com-
bining them into a global loss and careful selection of the
weight parameters ensure stable and effective learning.

The process of the proposed method is summarized in
Algorithm 1. First, the initial \( z_i \) is obtained through pre-
training the CAE on the public database. Second, \( k \)-means
clustering is performed to obtain initial \( \mu_j \). Then, \( q_{ij} \) and \( p_{ij} \)
are initialized by Eqs. (3) and (4). After that, the network is
updated repeatedly until the appropriate stopping criterion
is satisfied. At this time, the model’s global loss function is
optimized by a stochastic gradient descent algorithm.

3. Experiments

To verify our proposed method, we compare the perfor-
mance of many types of prominent clustering algorithms
for four image datasets. As a result of the experiment,
our proposed method exhibits superior performance in terms
of clustering accuracy (ACC) and normalized mutual infor-
mation (NMI) when compared to the state-of-art clustering
algorithms.

3.1 Databases

- **MNIST**: A total 70,000 handwritten digit images (60,000
  training images and 10,000 test images), 10 classes, 28 \( \times \) 28
  grayscale.

- **MNIST-test**: A total of 10,000 handwritten digit images,
  10 classes, 28 \( \times \) 28 grayscale.

- **USPS**: A total of 9,298 handwritten digit images (7,291
  training images and 2,007 test images), 16 \( \times \) 16 grayscale.

- **Fashion-MNIST**: A total of 70,000 fashion item images
  (60,000 training images and 10,000 test images), 10 classes,
  28 \( \times \) 28 grayscale.

  We normalized all of these to values from 0 to 1.

Fig. 2  If the inter-cluster distance increases, the distribution of the data
points changes and the degree of separation of the clusters increases. The
left side shows the case with small inter-cluster while the right side shows
a large separability between the clusters (i.e., the distance is suitably large).
(a), (c), (e), (g) represents distributions prior to clustering while (b), (d), (f),
(h) shows distributions after clustering of MNIST-full, MNIST-test, USPS,
and Fashion-MNIST datasets, respectively.

3.2 Implementation and Results of the Experiments

As shown in the Fig. 2, after performing the proposed clus-
tering for sample points (b), (d), (f) and (h), inter-cluster
separability and intra-cluster homogeneity were further im-
proved in (a), (c), (e), and (g). In particular, the proposed
method has an effect of separating adjacent clusters well
such as (a), (c), (e), and (g) where the clusters overlap.
In our experiments, the reconstruction, center, separation, and clustering loss functions are all used to improve clustering performance. Each loss function has its own weighting parameters and they can be found heuristically. **MNIST**: \( \alpha = 1.0, \beta = 0.8, \gamma = 1.0, \omega = 1.0 \). **MNIST-test**: \( \alpha = 20.0, \beta = 1.0, \gamma = 0.5, \omega = 0.5 \). **USPS**: \( \alpha = 1.0, \beta = 0.5, \gamma = 0.1, \omega = 0.1 \). **Fashion-MNIST**: \( \alpha = 1.0, \beta = 0.01, \gamma = 0.1, \omega = 0.1 \).

The CAE used to extract the learned features is divided into an encoder and a decoder section, each consisting of three layers. The first and second layers of the encoder have 32 and 64 filters, respectively, with a kernel size of 5 and 5, respectively, and a stride of 2. The third layer has 128 filters and a kernel size of 3 × 3. The decoder section has the same structure as the encoder layers in reverse order.

We compare the results from our experiments with 15 existing state-of-the-art clustering algorithms to verify the performance of our proposed method (Table 1). The numbers in red and blue represent the best and second-best performance, respectively. As shown in the table, our proposed algorithm outperforms the other algorithms for all datasets and metrics (e.g. ACC and NMI). Compared to DEC as a prominent deep clustering algorithm, the proposed method achieved 15.8% improvement in ACC and 19.9% gain in NMI for the Fashion MNIST data set.

### 4. Conclusions

In this paper, we proposed a novel approach for deep clustering. Experimental results have shown that our proposed algorithm can effectively cluster unlabeled datasets by applying cohesive loss functions. The proposed clustering method includes reconstruction, intra-cluster homogeneity, inter-cluster separability, and KL-divergence loss functions and is shown to exceed the performance of the existing clustering algorithms. In addition, our proposed method can be customized according to the task by adjusting the loss weights.

### Acknowledgments

This research was supported by Government-wide R&D Fund project for infectious disease research (GFID), Republic of Korea (grant number: HG19C0682). The work of Murray Loew was supported by GWU (KU-GWU Joint Research Fund).

### Table 1

| Table 1 | MNIST | MNIST-test | USPS | Fashion MNIST |
|---------|-------|------------|------|--------------|
|         | ACC   | NMI        | ACC  | NMI          | ACC  | NMI          |
| k-means [1] | 0.532 | 0.500      | 0.546 | 0.501        | 0.668 | 0.627        | 0.474 | 0.512 |
| SG-LS [17]  | 0.714 | 0.706      | 0.740 | 0.756        | 0.746 | 0.755        | 0.496 | 0.497 |
| NME-LP [18] | 0.471 | 0.452      | 0.479 | 0.467        | 0.652 | 0.693        | 0.434 | 0.425 |
| AC-Zell [19] | 0.113 | 0.017      | 0.810 | 0.693        | 0.657 | 0.798        | 0.100 | 0.010 |
| AC-GLD [20]  | 0.113 | 0.017      | 0.933 | 0.864        | 0.725 | 0.825        | 0.112 | 0.010 |
| DCN [21]     | 0.830 | 0.810      | 0.802 | 0.786        | 0.688 | 0.683        | 0.501 | 0.558 |
| DEC [9]      | 0.863 | 0.834      | 0.856 | 0.850        | 0.762 | 0.767        | 0.518 | 0.546 |
| IDRC [10]    | 0.881 | 0.867      | 0.846 | 0.802        | 0.761 | 0.785        | 0.529 | 0.557 |
| VaDe [22]    | 0.945 | 0.876      | 0.287 | 0.287        | 0.566 | 0.512        | 0.578 | 0.630 |
| DEPICT [12]  | 0.965 | 0.917      | 0.963 | 0.915        | 0.899 | 0.906        | 0.392 | 0.392 |
| FeDEC-DA [13]| 0.985 | 0.959      | 0.979 | 0.945        | 0.980 | 0.945        | 0.577 | 0.652 |
| FeIDEC-DA [13]| 0.985 | 0.960      | 0.983 | 0.958        | 0.987 | 0.962        | 0.586 | 0.636 |
| FeIDEC-DA [13]| 0.986 | 0.962      | 0.981 | 0.950        | 0.984 | 0.954        | 0.580 | 0.652 |
| FeIDEC-DA [13]| 0.983 | 0.955      | 0.979 | 0.950        | 0.984 | 0.955        | 0.584 | 0.638 |
| FeIDC-DA [13]| 0.986 | 0.962      | 0.961 | 0.920        | 0.969 | 0.929        | 0.580 | 0.650 |
| Proposed method | 0.986 | 0.963      | 0.984 | 0.960        | 0.988 | 0.966        | 0.600 | 0.655 |

### References

[1] J. Macqueen, et al., “Some methods for classification and analysis of multivariate observations,” Proc. fifth Berkeley symposium on mathematical statistics and probability, pp. 281–297, 1967.

[2] L. Rousseeuw and P.J. Kaufman, “Clustering by means of medoids,” Faculty of Mathematics an Informatics, 1987.

[3] R.F. Ling, “On the theory and construction of k-clusters,” The Computer Journal, vol.15, no.4, pp.326–332, 1972.

[4] R. Kok and O. Maimon, “Clustering methods,” Data mining and knowledge discovery handbook, Springer, Boston, MA, pp.321–352, 2005.

[5] H. Wang, Z. Xu, and H. Ko, “Random Binary Local Patch Clustering Transforms Based Image Matching for Nonlinear Intensity Changes,” Mathematical Problems in Engineering, 2018.

[6] C. Amédola, J.-C. Faugere, and B. Sturmfels, “Moment varieties of Gaussian mixtures,” arXiv preprint arXiv:1510.04654, 2015.

[7] J.C. Dunn, “A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters,” Journal of Cybernetics, pp.32–57, 1973.

[8] C. Song, F. Liu, Y. Huang, L. Wang, and T. Tan, “Auto-encoder based data clustering,” Iberoamerican congress on pattern recognition, Springer, Berlin, Heidelberg, pp.117–124, 2013.

[9] J. Xie, R. Girshick, and A. Farhadi, “Unsupervised deep embedding for clustering analysis,” International Conference on Machine Learning, pp.478–487, 2016.

[10] X. Guo, L. Gao, X. Liu, and J. Yin, “Improved deep embedded clustering with local structure preservation,” ICAI, pp.1753–1759, 2017.

[11] F. Li, H. Qiao, and B. Zhang, “Discriminatively boosted image clustering with fully convolutional auto-encoders,” Pattern Recognition, vol.83, pp.161–173, 2018.

[12] K.G. Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang “Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization,” Proc. IEEE international conference on computer vision, pp.5736–5745, 2017.

[13] X. Guo, et al., “Deep embedded clustering with data augmentation,” Asian Conference on Machine Learning, pp.478–487, 2016.
[17] X. Chen and D. Cai, “Large scale spectral clustering with landmark-based representation,” Twenty-fifth AAAI Conference on Artificial Intelligence, pp.313–318, 2011.

[18] D. Cai, et al., “Locality preserving nonnegative matrix factorization,” IJCAI, pp.1010–1015, 2009.

[19] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” Proc. 25th international conference on Machine learning, pp.1096–1103, 2008.

[20] W. Zhang, X. Wang, D. Zhao, and X. Tang, “Graph degree linkage: Agglomerative clustering on a directed graph,” European Conference on Computer Vision, Springer, Berlin, Heidelberg, pp.428–441, 2012.

[21] B. Yang, et al., “Towards k-means-friendly spaces: Simultaneous deep learning and clustering,” International Conference on Machine Learning, pp.3861–3870, 2017.

[22] Z. Jiang, et al., “Variational deep embedding: An unsupervised and generative approach to clustering,” arXiv preprint arXiv:1611.05148, 2016.