

## RESEARCH ARTICLE

# Deep Embedded Clustering Framework for Mixed Data

YONGGU LEE<sup>1</sup>, (Member, IEEE), CHULWUNG PARK<sup>1</sup>, (Member, IEEE), AND SHINJIN KANG<sup>2</sup><sup>1</sup>NCSOFT, Bundag-gu, Seongnam-si 13494, Republic of Korea<sup>2</sup>School of Games, Hongik University, Jochiwon, Sejong 30016, South Korea

Corresponding author: Shinjin Kang (directx@hongik.ac.kr)


This work was supported by the Culture, Sports and Tourism Research and Development Program through the Korea Creative Content Agency funded by the Ministry of Culture, Sports and Tourism, in 2022, through the Project Name: Development of Artificial Intelligence-Based Game Simulation Technology to Support Online Game Content Production under Project R2022020070.

**ABSTRACT** Deep embedded clustering (DEC) is a representative clustering algorithm that leverages deep-learning frameworks. DEC jointly learns low-dimensional feature representations and optimizes the clustering goals but only works with numerical data. However, in practice, the real-world data to be clustered includes not only numerical features but also categorical features that DEC cannot handle. In addition, if the difference between the soft assignment and target values is large, DEC applications may suffer from convergence problems. In this study, to overcome these limitations, we propose a deep embedded clustering framework that can utilize mixed data to increase the convergence stability using soft-target updates; a concept that is borrowed from an improved deep Q learning algorithm used in reinforcement learning. To evaluate the performance of the framework, we utilized various benchmark datasets composed of mixed data and empirically demonstrated that our approach outperformed existing clustering algorithms in most standard metrics. To the best of our knowledge, we state that our work achieved state-of-the-art performance among its contemporaries in this field.

**INDEX TERMS** Clustering algorithm, mixed data, deep learning.

## I. INTRODUCTION

Currently, because big data has been utilized extensively in many research fields, data mining experts must analyze databases from multiple perspectives. However, extracting potentially useful information from big data remains challenging. Accordingly, researchers in scientific fields such as pattern recognition, signal processing, bioinformatics, and social network analysis, who inherently work with high-dimensional and large-scale data, utilize various tools such as dimensionality reduction techniques and clustering analysis before conducting exploratory data analysis. In particular, the potential usefulness of clustering analysis cannot be overlooked, as it may provide a new perspective that data analysts may have missed or not yet discovered. However, despite the growing importance of clustering analysis, two problems remain to be addressed.

The associate editor coordinating the review of this manuscript and approving it for publication was Chao Tong .

The first problem is that most existing clustering algorithms assume all data features to be numerical. However, in the real world, data collected from the medical, social, and industrial fields also include nominal attributes, such as gender, occupation, and blood type (i.e., categorical). Hereafter, data with both numerical and categorical features are referred to as *mixed data*. Before carrying out clustering analysis on mixed data, data types are attempted to be unified either by discretizing numerical features into categorical ones or by encoding categorical features into numerical ones. However, preserving the relationship between data attributes before and after transformation and identifying a common evaluation criterion are other issues.

The second problem is the scalability of the clustering algorithms. High-dimensional data can be treated as embedding vectors in a relatively low-dimensional latent space using dimensionality reduction techniques. However, a problem arises in terms of data scalability. Some hierarchical clustering algorithms solve this problem by first performing

coarse clustering and then using the result as an input to another clustering algorithm. Other algorithms, such as mini-batch K-means, solve this problem by iteratively optimizing the clustering objectives by sampling the input data in mini-batch units. This mini-batch training style is natural for deep learning-based methods that are highly scalable and general.

Deep embedded clustering (DEC) is a representative algorithm that uses the deep learning framework [1]. The clustering process in DEC can be divided into the following two steps. In the first step, namely, pre-training, the weights of the stacked autoencoder (SAE) model are learned by embedding the input data as latent vectors in a low-dimensional space and then restoring them. In the second clustering step, with each embedded data and initial cluster centroids as input, the clustering objective function is optimized by solving a classification problem based on a soft assignment that determines which input data belongs to which cluster. Although the clustering accuracy reported by DEC is impressive, it can only handle numerical data, which is a challenge. In addition, because the learning parameters of the classifier model and cluster centroids are jointly optimized, the convergence may be unstable owing to the moving target issue.

In this study, we propose an extended version of deep embedded clustering that also works with categorical input features and mitigates the moving target issue with a soft-target update when optimizing the cluster objective function. The soft-target update is a technique that improves deep reinforcement learning methods, such as DDPG [2] and DQN [3], which helps to train learning parameters in a more stable manner by frequently updating the behavior policy network with respect to the target policy network. When the clustering performance is evaluated on UCI public datasets, it is empirically verified that the proposed method performs better than other existing methods operating on mixed data.

The main contributions of this study are summarized as follows:

- 1) We extend the DEC algorithm so that it can handle categorical data types as well as numerical ones.
- 2) We improve the convergence stability of the DEC algorithm when optimizing clustering objectives with a soft-target update.
- 3) The proposed method outperforms other clustering methods for mixed data in most standard metrics.

The remainder of this paper is organized as follows. The related research section introduces clustering methodologies using categorical data and describes the clustering algorithms used for comparatively evaluating the performance of the proposed method. The methodology section describes the proposed method in detail regarding how it uses mixed data as input and improves the convergence stability of DEC. In the experiments section, the performance of the proposed method is evaluated on UCI public datasets and compared with existing methodologies. Finally, we provide concluding remarks on the proposed algorithm in the discussion & conclusion section.

## II. RELATED RESEARCH

Various investigations have been conducted to apply clustering analysis to mixed data, which can be categorized into the following three categories [4].

In the first category, various approaches have been used to transform the categorical features into numerical ones. The simplest transformation method would be 0-1 encoding, which encodes each categorical feature value with a 0-1 indicator vector [5]. Although it is intuitive and can be easily converted back to the original data, it assumes that every categorical feature value is independent of each other; all pairwise distances are 1 and inner products are 0, which is often violated in real-world data. In addition, as the number of items in the categorical features increases, this approach suffers from high dimensionality [6]. To alleviate this issue, one can apply dimensionality reduction techniques, such as principal component analysis (PCA) [7], or use another embedding method, such as binary encoding, zero-appearance encoding [8], etc. Additionally, by leveraging deep learning frameworks, the distributed representation learning model of Word2Vec-style that yielded record-breaking success in the past decade in the natural language processing field can be used [9].

In contrast, in the second category, algorithms inversely transform numerical features into categorical ones via various discretization methods. This method utilizes similarity-based metrics, graph partitioning, or information entropy when discretizing categorical features [10], [11], [12], [13], [14]. However, as a combination of numerical features is transformed into a single nominal value, a loss of information is inevitable, e.g., the difference between the numerical features in the original space.

The third category attempts to design a generalized clustering criterion that applies to both numerical and categorical features. For example, [15] utilized Gower's distance as a dissimilarity metric for mixed data and applied k-prototypes algorithms for clustering. In another approach, the Goodall similarity metric was used to quantify the similarity between objects [16]. Alternatively, the distance between categorical values can be measured using Hamming distance, Ahmad's distance [17], context-based distance [18], etc. [19]. A deep learning model based on an autoencoder architecture can be utilized to find a suitable feature map that transforms both categorical and numerical features into a common space using locality-preserving projection [20]. A general clustering framework based on unified object-cluster similarity has been proposed recently in the literature [21].

In this study, we mainly conducted experiments using our suggested algorithm on two methods that belong to the first category: one-hot encoding and embedding layer. One-hot encoding was used to verify whether the superiority of the DEC model still holds when categorical features were added with minimal modification. For the embedding layer counterpart, a simple embedding-lookup operation is added just before concatenating all input features at the pre-training step.

### A. CLUSTERING METHODS FOR COMPARISON

We measured the quality of the clusters created using the proposed method in terms of the rand index (RI) and normalized mutual information (NMI) to quantitatively compare the proposed method with some of the existing clustering methods for mixed data reported in [20].

This type of method, called subspace clustering, aims to identify the features of interest that the true clusters might reside in. [22], a.k.a. WKM, proposed a modified version of the K-means by iteratively updating the weights of features based on the current data partitions. By incorporating the weighted entropy along with the clustering objective function, [23] extends the WKM framework, which we denote as EWKM. The OCIL algorithm in [21] is based on object-cluster similarity, which is computed separately for numerical and categorical features. [4], denoted by WOCIL, improves OCIL algorithm by considering the varying contribution of different features when forming the clusters. In this approach, the weights of the features for each cluster are dynamically updated when optimizing the object-cluster similarities.

### III. METHODOLOGY

In this section, we introduce the proposed network architecture and training process for clustering in detail. First, for a better understanding, we explained the overall network architecture. To implement clustering in the latent data space, our approach utilized an autoencoder structure based on a deep neural network for non-linear mapping. The reason for clustering in a latent data space was that it is difficult to directly cluster mixed data using general distance or density-based clustering methods, and a latent data space can efficiently prevent the high dimensionality problem [6]. Therefore, clustering in our method was conducted in the latent vector space instead of the original data space. The original data in space  $X$  were converted to a latent vector in  $Z$  using the non-linear mapping method  $f_{\theta}:X \rightarrow Z$ , where  $\theta$  is the trainable parameter and  $Z$  is the latent space. Second, all training steps were covered. The training steps mainly comprised two parts, namely, the pre-training and clustering steps. In each step, the mixed data were utilized for pre-training the network, and clustering was implemented using the soft assignment [1] and soft-target update methods [2] for better performance.

#### A. NETWORK ARCHITECTURE

The entire network architecture of the proposed algorithm is illustrated in Figure 1. The embedding layer automatically transformed categorical data into machine-interpretable data. In the encoding part, the autoencoder architecture compressed the data information so that it represented in latent space. The compressed latent vector was restored in the decoding part. The numerical features were reconstructed using the linear activation function, and categorical features were restored using the Softmax activation function when the one-hot encoding was used for an embedding method. Additionally, trainable parameters indicating cluster centers

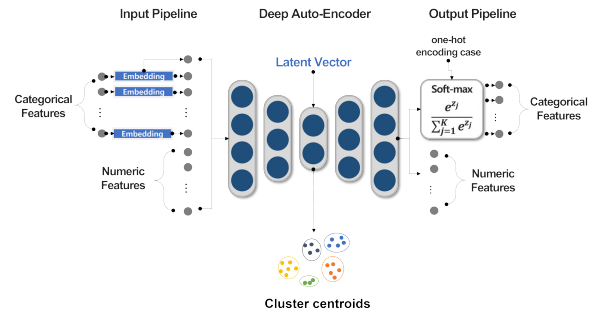


FIGURE 1. The complete network architecture.

existed in the latent space. These parameters were utilized for clustering using the Student's t-distribution [24]. The parameters in the autoencoder and latent space were trained to minimize the sum of the reconstruction and clustering losses.

#### B. TRAINING STEPS

The training processes are mainly composed of two steps: the first step is the pre-training step and the second step is the clustering step. This is depicted in Figure 2 in detail.

##### 1) PRE-TRAIN STEPS

In the pre-training step, a deep autoencoder was pre-trained to represent the original data in the latent vector space. The datasets with mixed data were composed of numerical and categorical data transformed in a machine-interpretable format using an embedding layer or one-hot encoding method. The losses for optimizing the deep autoencoder were differentiated according to the input data type. In our approach, the reconstruction losses were composed of the mean cross-entropy loss and mean squared error based on the input data type. To calculate the reconstruction loss for categorical features, we substituted this with a multi-class classification problem for better performance. Thus, the mean cross-entropy was used for this part because cross-entropy or KL divergence loss was more effective in optimizing the neural network weights in terms of training efficiency. To update the weights responsible for the categorical and numerical inputs equally, we utilized the mean squared error and mean cross-entropy loss.

##### 2) CLUSTERING STEPS

In the clustering step, the k-means method [25] was applied to determine the initial centroids in the latent space. It also calculated the clustering loss defined as the KL divergence between a target distribution and soft assignment using Student's t-distribution [1]. The soft assignment, target distribution, and KL divergence between them were computed as follows:

$$q_{ij} = \frac{(1 + \|z_i - \mu_j\|^2 / a)^{-\frac{\alpha+1}{2}}}{\sum_j (1 + \|z_i - \mu_j\|^2 / a)^{-\frac{\alpha+1}{2}}} \quad (1)$$

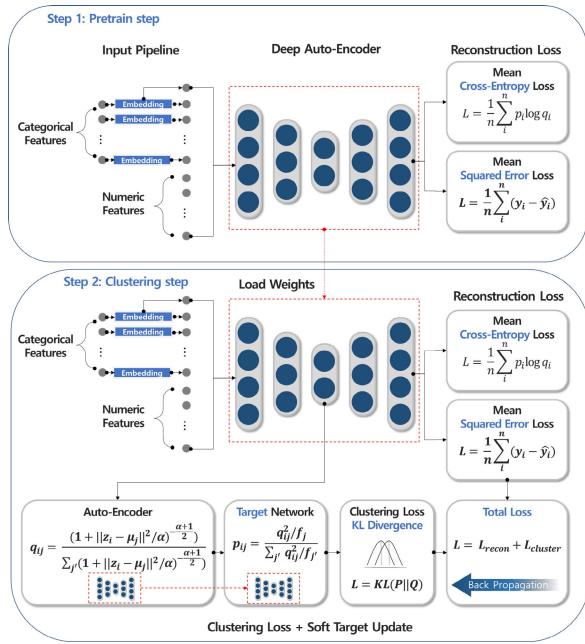


FIGURE 2. Training steps: First step is for pre-training, while second step is for clustering.

$$p_{ij} = \frac{q_{ij}/f_j}{\sum_j q_{ij}^2/f_j} \text{ where } f_j = \sum_i q_{ij} \quad (2)$$

$$L = KL(P \parallel Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad (3)$$

However, we consider that this step probably increased the reconstruction loss because the latent vector distribution trained in the pre-training step would be distorted while minimizing the clustering loss. This eventually decreased the clustering performance. To resolve this problem, we introduced two novel methods. First, we trained our network to minimize reconstruction and clustering losses simultaneously, while implementing the clustering step. This would maintain or change the latent vector distribution in a stable manner even if the clustering step proceeded. Second, we adopted the soft-target network method [2] to achieve better convergence. This is because, to minimize the KL divergence between the soft assignment and target distribution, we needed to calculate the target distribution for each epoch or time-step that we defined. However, if the difference between the target distribution and soft assignment values was quite large, the convergence stability of the network parameters decreased. Therefore, we utilized a soft-target update method using a target network. The entire training algorithm is described in Algorithms 1 and 2.

### 3) SOFT-TARGET UPDATE

During the clustering step, the target network was duplicated using the autoencoder network. After that, at every training time-step, the autoencoder provided the target distribution  $p$ . Accordingly, the weights of the target network were updated gradually [2] following the equation. This update method

### Algorithm 1 The Pre-Training Step for Deep Embedding Clustering With Mixed Data

**Input:** Mixed data for clustering

**Parameter:** Autoencoder model weights

**Output:** Pre-trained autoencoder

- 1: Initialize autoencoder weights  $\theta$
- 2: Let  $epoch = 1$
- 3: **for**  $epoch = 1$  to  $T$  **do**
- 4:   Let  $step = 1, M = ceiling(data\ size / batch\ size)$
- 5:   **for**  $step = 1$  to  $M$  **do**
- 6:     Concatenate input features  $X_{concat}$  according to the input data type from current batch of data
- 7:     Perform encoding  $l = f_{encoder}(X_{concat})$  and decoding  $Y_{concat} = f_{decoder}(l)$
- 8:     Calculate the reconstruction loss according to input data type
- 9:     Update autoencoder parameters  $\theta$  by minimizing reconstruction loss:  $L_{recon} = MSE + MCE$
- 10:   **end for**
- 11: **end for**
- 12: **return** pre-trained autoencoder

improved not only the network convergence stability but also the clustering performance. The difference between the existing method and soft-target update method is depicted in Figure 3.

$$W_{target} \leftarrow \tau * W_{model} + (1 - \tau) * W_{target} \quad (4)$$

## IV. EXPERIMENTS

In this section, the performance of the proposed clustering method is numerically presented. To verify the clustering capability of our algorithm, we assess the performance of existing clustering algorithms, namely, WKM [22], EWKM [23], OCIL [21], and WOCIL [4], on public datasets, and compare the results with that of our clustering method.

### A. DATASET

Four public datasets from the UCI machine learning data repository (URL: <http://archive.ics.uci.edu/ml/>) were chosen to validate the performance of our method. The first dataset was the ‘‘Heart Disease’’ dataset composed of 303 mixed data entries. The features of the ‘‘Heart Disease’’ dataset consisted of seven categorical variables and six numerical variables related to the features of heart disease patients. Thus, the objective of this clustering was to determine whether a person is healthy or sick. The ground-truth label was also defined as healthy and sick. The second dataset is the ‘‘Credit Approval’’ dataset. The number of data entries was 690, which had nine categorical features and six numerical features. The goal of this clustering was to determine whether a credit card application is approved based on related features. Thus, the ground



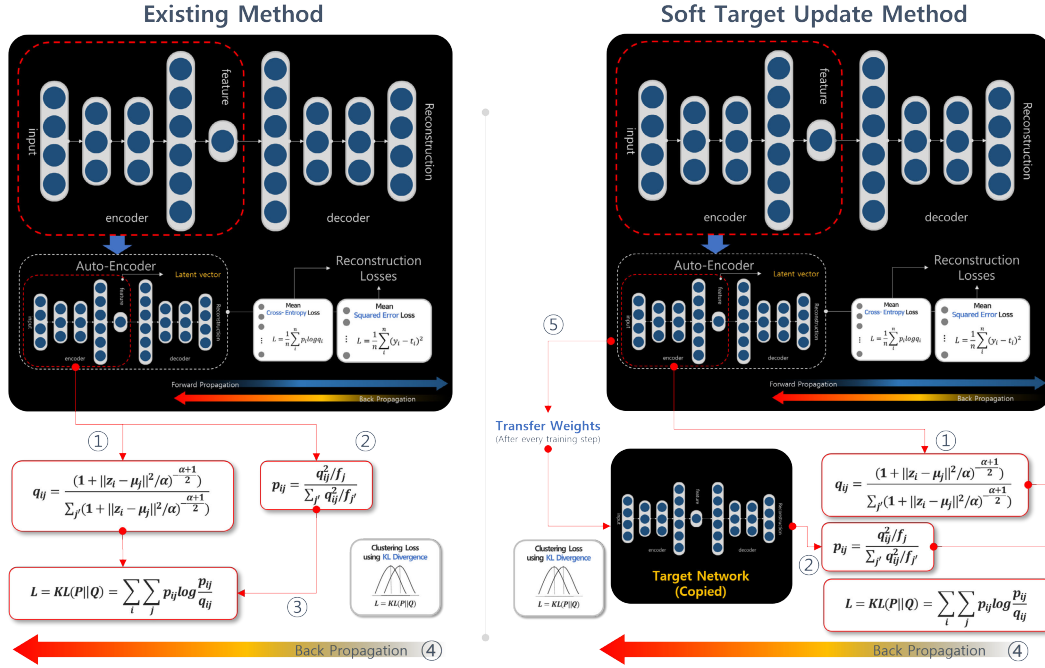


FIGURE 3. Target update methods: A) existing method and b) soft-target update method.

truth labels were credit card approval and non-approval. Third, the ‘‘German Credit’’ dataset was selected to test our method. It contains 1000 individual information data entries consisting of thirteen categorical features and seven numerical features. Similar to the credit approval dataset, the ground truth labels consisted of whether approved or not. Finally, the ‘‘Adult’’ dataset was chosen for the experiment. This dataset consisted of 45222 data entries, each having eight categorical and six numerical features. The ground truth label was whether a person earns more than \$50K/year based on census data.

### B. PERFORMANCE METRICS

Because the ground truth labels for each dataset exist, we adopted two metrics, namely, rand index (RI) and normalized mutual information (NMI), to evaluate the performance of our algorithm against that of existing algorithms. RI values are defined as follows:

$$RI = \frac{TP + TN}{TP + FP + FN + TN} \quad (5)$$

where TP, TN, FP, and FN denote the true positive, true negative, false positive, and false negative, respectively, regarding the class label. NMI values are represented as follows:

$$MI(U, V) = \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} \frac{|U_i \cap V_j|}{N} \log \frac{N|U_i \cap V_j|}{|U_i||V_j|} \quad (6)$$

$$H(U) = - \sum_{i=1}^{|U|} P(i) \log P(i) \quad (7)$$

$$NMI(U, V) = \frac{MI(U, V)}{\text{mean}(H(U), H(V))} \quad (8)$$

where  $|U_i|$  is the number of samples in cluster  $U_i$  and  $|V_i|$  is the number of samples in cluster  $V_i$ .  $N$  is the total number of data points.

### C. HYPER-PARAMETERS

<https://www.overleaf.com/project/62cf6df01713fed84b516-281> As shown in Fig. 1, our approach is fundamentally based on the deep learning framework. Thus, hyper-parameter tuning is a challenging issue when utilizing our approach in general. This is because even if the clustering performance is acceptable, the method is difficult to use if the hyper-parameter tuning process is too complicated. Thus, in our experiments, to prove the generality of our method, we conducted all experiments with the same hyper-parameters except the drop-out rate and batch size, which were tuned according to the data size. This is reported in Table 3. In all experiments, the autoencoder architecture consisted of 50-50-200-5-200-50-50 layers, and the latent vector size was 5. Additionally, the learning rate was 0.01, which generally decreased by a factor of 0.1 every 100 epochs. The soft-target update parameter was defined as 0.01 as well. Finally, we prove that our algorithm can be applied to any dataset in general and shows nearly state-of-the-art performance when performing clustering on datasets with mixed data.

### D. COMPARED ALGORITHMS

We assess the performance of our clustering approach compared with that of existing algorithms, namely, WKM,

**TABLE 1.** Clustering performance with respect to RI of different algorithms on various datasets with mixed data.

Dataset	WKM	EWKM	OCIL	WOCIL	Mixed DEC	Mixed DEC+SU
Heart Disease	0.618±0.061	0.643±0.064	0.647±0.011	0.689±0.055	0.717±0.017	0.742±0.013
Credit Approval	0.667±0.116	0.658±0.072	0.607±0.095	0.655±0.103	0.742±0.0167	0.755±0.007
German Credit	0.505±0.010	0.512±0.012	0.560±0.026	0.574±0.003	0.559±0.0102	0.583±0.003
Adult	0.604±0.000	0.626±0.000	0.625±0.000	0.626±0.000	0.642±0.0009	0.642±0.001

**TABLE 2.** Clustering performance with respect to NMI of different algorithms on various datasets with mixed data.

Dataset	WKM	EWKM	OCIL	WOCIL	Mixed DEC	Mixed DEC+SU
Heart Disease	0.184±0.094	0.225±0.099	0.233±0.019	0.307±0.087	0.350±0.027	0.400±0.025
Credit Approval	0.287±0.196	0.245±0.113	0.182±0.145	0.236±0.174	0.404±0.041	0.434±0.012
German Credit	0.004±0.001	0.013±0.006	0.003±0.002	0.006±0.005	0.020±0.007	0.026±0.005
Adult	0.092±0.001	0.001±0.000	0.004±0.001	0.005±0.000	0.033±0.001	0.033±0.001

### Algorithm 2 The Clustering Step for Deep Embedding Clustering With Mixed Data Using Soft-Target Network

**Input:** Mixed data for clustering, pre-trained autoencoder

**Parameter:** Autoencoder model weights and cluster centroids parameters

**Output:** Trained autoencoder, clustered data according to the number of clustering centroids

- 1: Let  $epoch = 1$ ,  $k =$  number of clusters
- 2: Implement standard  $k$ -means to initialize cluster centroids  $\{\mu_j\}_{j=1}^k$
- 3: Duplicate pre-trained autoencoder  $f_{AE}$  to generate target network  $f_T$
- 4: **for**  $epoch = 1$  to  $T$  **do**
- 5:   Let  $step = 1$ ,  $M = \text{ceiling}(\text{data size} / \text{batch size})$
- 6:   **for**  $step = 1$  to  $M$  **do**
- 7:     Follow the pre-train step to get reconstruction loss
- 8:     Compute  $q_{ij}$  from  $f_{AE}$  and  $p_{ij}$  from  $f_T$ 

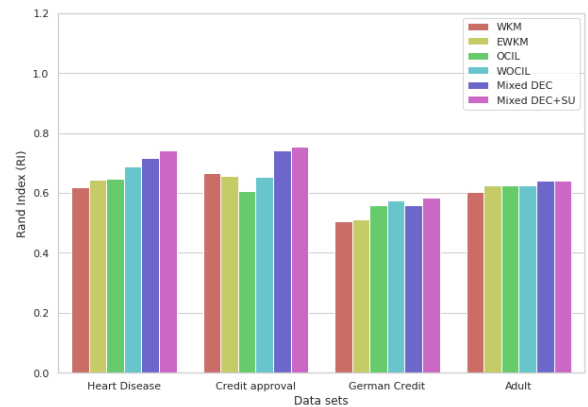
$$q_{ij} = \frac{(1 + \|z_i - \mu_j\|^2 / a)^{-\frac{\alpha+1}{2}}}{\sum_{j'} (1 + \|z_i - \mu_{j'}\|^2 / a)^{-\frac{\alpha+1}{2}}}$$

$$p_{ij} = \frac{q_{ij} / f_j}{\sum_{j'} q_{ij'}^2 / f_j'}$$
- 9:     Calculate clustering loss
$$L_{cluster} = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$
- 10:     Update autoencoder parameters  $\theta$  by minimizing both reconstruction loss and clustering loss
$$L_{total} = L_{recon} + L_{cluster}$$
- 11:     Update target network parameters
$$\theta_T \leftarrow \tau * \theta_{AE} + (1 - \tau) * \theta_T$$
- 12:   **end for**
- 13: **end for**
- 14: Label the data based on  $\text{argmax}(p_{ij})$
- 15: **return** trained autoencoder, clustered data

EWKM, OCIL, and WOCIL. Each stands for the weighted K-means algorithm, entropy weighted K-means algorithm, object-cluster similarity algorithm, attribute-weighted, and attribute-weighted object-cluster similarity. The performances of these algorithms are described in [4]. Clustering performances are reported in Tables 1 and 2 in detail.

**TABLE 3.** Used clustering hyper-parameters in each dataset.

Dataset	Epoch	Batch Size	Dropout
Heart Disease	200	64	0.01
Credit Approval	200	128	0.05
German Credit	200	256	0.3
Adult	200	4096	0.3

**FIGURE 4.** RI of different algorithms on various datasets with mixed data.

To compare the performance of the method with soft-target network update and that of the existing one, we implement an ablation test for each dataset.

### E. CLUSTERING PERFORMANCE

Clustering for datasets with mixed data is performed based on the steps outlined above. In our experiment, we performed clustering using both a trainable look-up embedding layer and raw one-hot encoding. However, we found that the weights of the embedding layer hardly changed from their initial states even when the reconstruction loss was minimized. Therefore, in our experiments, we mainly used the one-hot encoding method for embedding categorical features and report only the results of it. To prove the generality of our algorithm, we proceeded with the pre-training step and then performed the clustering step using the pre-trained model 50 times to estimate the mean and standard deviation of each dataset.

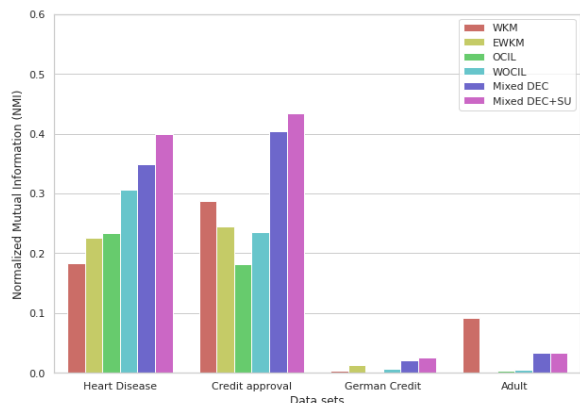


FIGURE 5. NMI of different algorithms on various datasets with mixed data.

From Tables 1 and 2, and in Figures 4 and 5, we can observe that our proposed algorithm outperforms the existing clustering methods in all datasets. In particular, the NMI metric performance was much better than that of the compared methods. Additionally, we found that using the soft-target network strategy is generally better than training without a soft-target network in terms of both NMI and RI. This empirically proves that the soft-target network strategy is effectively applicable to the proposed approach. Only in credit approval dataset, the performance of WKM method is better than that of our approach. The reason might be that the fundamental relevance of the data is more suitable for the WKM method than it is for the mixed DEC method.

## V. DISCUSSION AND CONCLUSION

In this study, we proposed a novel deep embedded clustering framework that is applicable to both numerical and categorical data. This approach successfully extends the established DEC model, yielding state-of-the-art clustering performance on datasets with mixed data. In addition, based on the experimental results, we proved that our method can be effectively used without complicated hyper-parameter tuning and successfully overcome the scalability issues.

In future research, similar to [26], our method can be further extended to constrained clustering, which operates in a semi-supervised setting to guide clustering and increase accuracy. Furthermore, there is room for improvement in the pre-training step, where other efficient embedding methods for categorical features can be used instead of one-hot encoding and embedding layer methods, such as Cat2Vec [9].

## ACKNOWLEDGMENT

The authors would like to thank the NC Soft Corporation for providing such a huge opportunity in pursuing research project and to express special gratitude to Hwanhee Kim for helping their project on Deep Embedded Clustering Framework For Mixed Dataset, also would like to thank the opportunity to work on this deep learning project with his advice, and also would like to thank seniors and colleagues, Jinun Park,

Hyunseok Paeng, and Myeongseop Cha. This work would not have possibly proceeded well without their continuous advice and cooperation during their project.

## REFERENCES

- [1] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proc. ICML*, 2016, pp. 478–487.
- [2] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.
- [3] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.
- [4] H. Jia and Y.-M. Cheung, "Subspace clustering of categorical and numerical data with an unknown number of clusters," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 8, pp. 3308–3325, Apr. 2018.
- [5] F. Kemp, "Applied multiple regression/correlation analysis for the behavioral sciences," *J. Roy. Stat. Soc., D, Statistician*, vol. 52, no. 4, p. 691, Dec. 2003.
- [6] R. Bellman and R. Kalaba, "A mathematical theory of adaptive control processes," *Proc. Nat. Acad. Sci. USA*, vol. 45, no. 8, pp. 1288–1290, 1959.
- [7] I. Jolliffe, "Principal component analysis," in *Encyclopedia of Statistics in Behavioral Science*, 2005.
- [8] G. Pang, K. M. Ting, D. Albrecht, and H. Jin, "ZERO++: Harnessing the power of zero appearances to detect anomalies in large-scale data sets," *J. Artif. Intell. Res.*, vol. 57, pp. 593–620, Dec. 2016.
- [9] Y. Wen, J. Wang, T. Chen, and W. Zhang, "Cat2Vec: Learning distributed representation of multi-field categorical data," Tech. Rep., 2016.
- [10] S. Guha, R. Rastogi, and K. Shim, "ROCK: A robust clustering algorithm for categorical attributes," *Inf. Syst.*, vol. 25, no. 5, pp. 345–366, 2000.
- [11] M. J. Zaki and M. Peters, "CLICKS: Mining subspace clusters in categorical data via  $K$ -partite maximal cliques," in *Proc. 21st Int. Conf. Data Eng.*, 2005, pp. 355–356.
- [12] D. Barbará, Y. Li, and J. Couto, "COOLCAT: An entropy-based algorithm for categorical clustering," in *Proc. 11th Int. Conf. Inf. Knowl. Manage.*, 2002, pp. 582–589.
- [13] P. Andritsos, P. Tsaparas, and R. J. Miller, "LIMBO: Scalable clustering of categorical data," in *Proc. Int. Conf. Extending Database Technol.* Berlin, Germany: Springer, 2004, pp. 123–146.
- [14] N. Tishby, W. Bialek, and F. C. Pereira, "The information bottleneck method: Extracting relevant information from concurrent data," NEC Research Institute, Tech. Rep., 1998.
- [15] Ö. Akay and G. Yüksel, "Clustering the mixed panel dataset using Gower's distance and  $k$ -prototypes algorithms," *Commun. Statist.-imul. Comput.*, vol. 47, no. 10, pp. 3031–3041, Nov. 2018.
- [16] D. W. Goodall, "A new similarity index based on probability," *Biometrics*, vol. 22, no. 4, pp. 882–907, Dec. 1966.
- [17] A. Ahmad and L. Dey, "A  $k$ -mean clustering algorithm for mixed numeric and categorical data," *Data Knowl. Eng.*, vol. 63, no. 2, pp. 503–527, 2007.
- [18] D. Ienco, R. G. Pensa, and R. Meo, "From context to distance: Learning dissimilarity for categorical data clustering," *ACM Trans. Knowl. Discovery Data*, vol. 6, no. 1, pp. 1–25, Mar. 2012.
- [19] H. Jia, Y.-M. Cheung, and J. Liu, "A new distance metric for unsupervised learning of categorical data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 5, pp. 1065–1079, May 2016.
- [20] S. Sahoo and S. Chakraborty, "Learning representation for mixed data types with a nonlinear deep encoder-decoder framework," 2020, *arXiv:2009.09634*.
- [21] Y.-M. Cheung and H. Jia, "Categorical-and-numerical-attribute data clustering based on a unified similarity metric without knowing cluster number," *Pattern Recognit.*, vol. 46, no. 8, pp. 2228–2238, 2013.
- [22] J. Z. Huang, M. K. Ng, H. Rong, and Z. Li, "Automated variable weighting in  $k$ -means type clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 5, pp. 657–668, May 2005.
- [23] L. Jing, M. K. Ng, and J. Z. Huang, "An entropy weighting  $k$ -means algorithm for subspace clustering of high-dimensional sparse data," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 8, pp. 1026–1041, Aug. 2007.
- [24] L. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 1–27, Nov. 2008.

- [25] J. MacQueen, "Classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, 1967, pp. 281–297.
- [26] H. Zhang, S. Basu, and I. Davidson, "A framework for deep constrained clustering-algorithms and advances," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Cham, Switzerland: Springer, 2019, pp. 57–72.



**YONGGU LEE** (Member, IEEE) received the B.S. degree in mechanical engineering from Sungkyunkwan University, Republic of Korea, in 2016, and the M.S. degree in mechanical engineering from the National University of Singapore, Singapore, in 2019. From 2020 to 2021, he was a Research Engineer at NHN Corporation, Pangyo, South Korea. Since 2022, he has been a Research Engineer at NCSOFT, South Korea. He wrote some papers and articles related to robotics using reinforcement learning and deep learning. His current research interests include the development of automated game balancing program utilizing reinforcement learning and clustering algorithms.



**CHULWUNG PARK** (Member, IEEE) received the B.S. degree in computer science (double major: electrical engineering) from the Korea Advanced Institute of Science and Technology, Daejeon, Republic of Korea, in 2015, and the M.S. degree from the School of Computing, Korea Advanced Institute of Science and Technology, in 2018. From 2018 to 2021, he was a Research Engineer at NHN Corporation, Pangyo, South Korea, and since 2021, he has been working at NCSOFT Corporation, South Korea. His research interest includes tackle the challenging problems that exist in games of multiagent environments with reinforcement learning.



**SHINJIN KANG** received the M.S. degree from the Department of Computer Science and Engineering, Korea University, in 2003, and the Ph.D. degree in computer science and engineering, Korea University, in 2011. After graduation, he joined Sony Computer Entertainment Korea (SCEK) as a Game Developer. Since 2006, he has been working at NCsoft Korea as a Lead Game Designer. He is currently a Professor at the School of Games, Hongik University.

...