Academic Journal of Computing & Information Science
ISSN 2616-5775 Vol. 7, Issue 6: 44-52, DOI: 10.25236/AJCIS.2024.070607

# TabClusterNet: Enhanced Deep Clustering for Tabular Data Analysis

**Shuwei Xu**[1,a,*], **Zhi Hu**[1,b], **Xiaowei Wang**[1,c]

[1]*Software College, Shenyang Normal University, Shenyang, Liaoning, 110034, China*
[a]*17355360809@163.com*, [b]*hu-zhi521@163.com*, [c]*wangxwvv@gmail.com*
[*]*Corresponding author*

***Abstract:*** *Clustering high-dimensional tabular data is a complex and challenging problem. Traditional clustering techniques nearly fail to identify the latent structure buried in spaces of high dimensionality. TabClusterNet is a novel deep clustering model designed specifically for tasks related to tabular data analysis. The self-supervised learning encoder-decoder from TabNet is combined with the deep clustering framework of Deep Embedding Clustering (DEC). By the high feature-extracting power of TabNet and the high clustering ability of DEC, TabClusterNet achieves far superior performance than the conventional method in feature extraction towards efficient clustering. Our proposed novel deep clustering architecture has been extensively validated over various public datasets for its great performance over different evaluation metrics. A closer look at the model shows that it preserves the structure of the data. TabClusterNet has been demonstrated to achieve substantially improved clustering accuracy and not only offer insights useful for data analytics and decision support, but also enable data scientists and researchers to glean deeper insights from complex datasets.*

***Keywords:*** *Deep Clustering, Tabular Data, Self-supervised Learning*

## 1. Introduction

Unsupervised clustering is a basic task both in data science and in machine learning. The most popular clustering methods, K-means and Gaussian Mixture Models [1], classify data based on inherent attributes or similarities in manually crafted features. In many cases, however, similarity measures for high-dimensional input feature spaces become much less reliable. One way to address this issue and still proceed with clustering is to reduce data dimensionality prior to clustering. Principal Component Analysis (PCA) can be performed for this purpose. With the advent of deep learning, feature transformations can now be performed by Deep Neural Networks (DNNs), a technique known as deep clustering.

Recent deep clustering approaches leave many questions unanswered. Initial work in this area focused on feature learning to retain specific data properties by incorporating prior knowledge [2-3]. These approaches typically involve two stages: feature transformation and clustering. Later, joint algorithms like DEC [4] emerged, which combine both stages. The DEC algorithm defines an objective in a self-supervised manner, using clustering loss to update transformation network parameters and cluster centers simultaneously. Cluster assignments are incorporated as soft labels. However, recent research has shown that deep clustering methods based on image datasets do not universally apply to all data types [5-6], particularly tabular data. Experiments have demonstrated that traditional clustering for tabular data ranks second among eight methods, outperforming most deep embedding clustering methods [7].

To address this issue, we developed TabClusterNet, an advanced deep learning model designed for tabular data clustering. This model innovatively combines the self-supervised learning encoder and decoder of TabNet with the deep clustering framework of DEC, providing an enhanced solution for clustering tabular data. The greatest advantage of TabClusterNet is that it transcends the limitations of traditional feature extraction by incorporating an attention mechanism within its encoder and decoder for advanced clustering. This approach improves the quality of clustering and provides deeper insights through more nuanced techniques for data analysis and decision-making processes.

The main contributions of this paper are threefold: (1) the introduction of a new deep clustering architecture specifically designed for high-dimensional tabular data; (2) demonstrating that

TabClusterNet achieves superior clustering performance over existing baselines across a multitude of metrics in several benchmark datasets; and (3) providing comprehensive and thorough analyses of the model across different datasets, proving its ability to preserve data structure.

## 2. Related work

### *2.1 Autoencoders*

Autoencoders are known to compress data well and learn features; they can further improve clustering significantly. These neural architectures compress input data into a latent space and then reconstruct it, retaining essential characteristics but keeping redundancy at a minimum. Clustering in the latent space, rather than in the original space, produces clearer groupings. On the other hand, the introduction of a probabilistic layer in the encoding process provides a mechanism for modeling clusters as distributions in latent space with Variational Autoencoders [8]. This procedure enhances flexibility and robustness in clustering, providing deeper insights into data categorization and highlighting the significant potential of integrating Autoencoders within clustering algorithms.

### *2.2 Deep Clustering*

Popular deep clustering algorithms explicitly define a clustering loss, analogous to classification errors in supervised deep learning. DEC employs a deep neural network to map data from the observed space to a low-dimensional latent space, learning feature representations and clustering assignments simultaneously. The key contribution of DEC is its clustering loss (target distribution P), which supervises high-confidence samples to make cluster distributions more concentrated. Drawing inspiration from t-SNE, DEC minimizes the Kullback-Leibler (KL) divergence by focusing on a centroid-based probability distribution, reducing KL divergence to an auxiliary target distribution. This method enhances clustering assignments and feature representations while reducing complexity to $O(nk)$, with $k$ being the number of centroids.

## 3. TabClusterNet

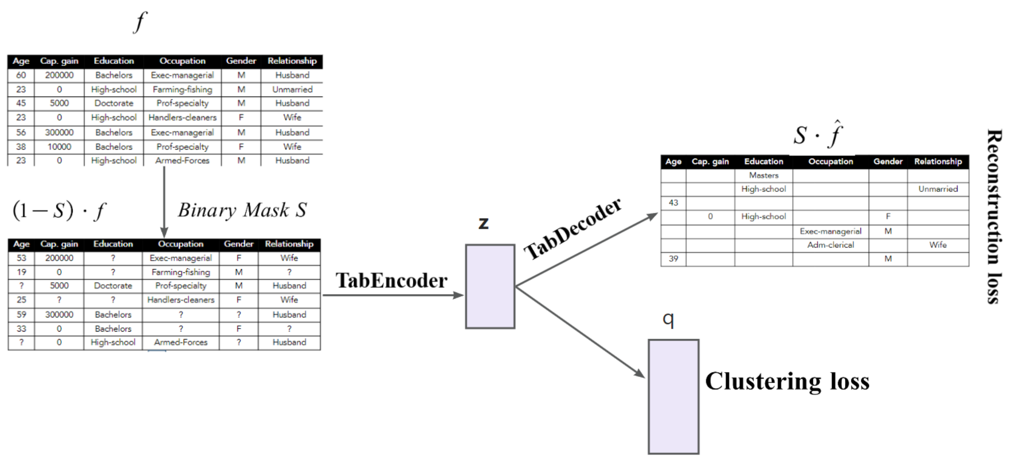
### *3.1 TabNet*

TabNet [9] was developed at Google in 2019 and later published at AAAI 2021. It offers a unique neural network structure explicitly designed for tabular data. The main novelty of the sequence-based attention mechanism is its ability to select features at each decision step, resulting in better representations and increased learning efficiency and model interpretability. TabNet enhances the performance of traditional Decision Trees (DT) while retaining most of their strengths through a sophisticated design, addressing: (i) sparse, per-instance feature selection derived from data; (ii) incrementally improved decision steps based on selected features and nonlinear processing, which enhances learning efficiency; and (iii) increased dimensionality and additional steps for simulating ensemble learning. TabNet's encoding is based on sequential multi-step processing, where each step uses information processed from the previous step to decide on feature usage and outputs the processed feature representation for the overall decision.

Currently, deep-learning methodologies like TabNet only perform well when applied to classification or regression tasks involving tabular data. Therefore, the investigation of its applicability as a proper framework for the implementation of clustering tasks came with an opportune moment useful in providing insight into the feature selection and representation learning ability of TabNet. This paper introduces a conceptual framework called TabClusterNet by incorporating the competency of the TabNet sequential multi-step feature selection mechanism along with its capability in self-supervised learning through deep clustering methods. This combination allows TabClusterNet to cluster tabular data effectively while still maintaining the interpretability of the data so that richer findings in complex structures within the data are returned.

### *3.2 Model Architecture*

TabClusterNet integrates the self-supervised learning encoder and decoder of TabNet into deep framework for clustering with DEC. It gives an answer to high-dimensional unsupervised tabular data clustering that marries the strength that TabNet has, in the limitation of the traditional feature extraction method in the quality of the feature representation and significant improvement in efficiency of clustering. TabClusterNet utilizes a multi-step architecture with the TabNet encoder, aiming to provide dynamic identification and processing of key features at each step, thus facilitating deeper understanding and analysis of data. The attention mechanism of the TabNet model employs an attentive transformer and a feature mask in this case, to help discover important characteristics at each step to use them appropriately in clustering tasks.



| Age | Cap. gain | Education | Occupation | Gender | Relationship |
|---|---|---|---|---|---|
| 60 | 200000 | Bachelors | Exec-managerial | M | Husband |
| 23 | 0 | High-school | Farming-fishing | M | Unmarried |
| 45 | 5000 | Doctorate | Prof-specialty | M | Husband |
| 23 | 0 | High-school | Handlers-cleaners | F | Wife |
| 56 | 300000 | Bachelors | Exec-managerial | M | Husband |
| 38 | 10000 | Bachelors | Prof-specialty | F | Wife |
| 23 | 0 | High-school | Armed-Forces | M | Husband |

| Age | Cap. gain | Education | Occupation | Gender | Relationship |
|---|---|---|---|---|---|
| 53 | 200000 | ? | Exec-managerial | F | Wife |
| 19 | 0 | ? | Farming-fishing | M | ? |
| ? | 5000 | Doctorate | Prof-specialty | M | Husband |
| 25 | ? | ? | Handlers-cleaners | F | Wife |
| 59 | 300000 | Bachelors | ? | ? | Husband |
| 33 | 0 | Bachelors | ? | F | ? |
| ? | 0 | High-school | Armed-Forces | ? | Husband |

| Age | Cap. gain | Education | Occupation | Gender | Relationship |
|---|---|---|---|---|---|
|  |  | Masters |  |  |  |
|  |  | High-school |  |  | Unmarried |
| 43 |  |  |  |  |  |
|  | 0 | High-school |  | F |  |
|  |  |  | Exec-managerial | M |  |
|  |  |  | Adm-clerical |  | Wife |
| 39 |  |  |  | M |  |

*Figure 1: The architecture of TabNet in self-supervised learning is an encoder–decoder structure. T The encoder, through a sequential multi-step process and attention mechanisms, extracts effective information from raw numerical features and processes categorical features using trainable embeddings. The features are encoded into an embedded representation, and the subsequent reconstruction by the TabNet decoder uses a binary mask to preserve the local structure of the initial data. The embedded representation is updated further on this round using clustering loss to ensure data points are well-distributed for good clustering.*

Figure 1 illustrates the TabClusterNet architecture for tabular data. We utilize TabNet's self-supervised learning encoder-decoder structure to replace the traditional autoencoder, thereby providing deeper feature representation. Features are encoded into an embedded representation and reconstructed through TabNet's decoder, with the reconstruction loss $L_r$ calculated to retain the original data's local structure. During reconstruction, a binary mask $S\in\{0,1\}^{B\times D}$ is used, where $B$ is the batch size and $D$ is the feature dimension, with $(1-S)\ f$ presenting masked features. The embedded representation $z$ is then used in the clustering loss $L_c$, guiding the embedding space adjustment to effectively disperse data points.

The overall optimization goal integrates reconstruction loss and clustering loss, expressed as $L=\alpha L_r+\gamma L_c$, where $\alpha$ and $\gamma$ are coefficients that balance the two losses, with $\alpha+\gamma=1$. When $\gamma$ increases, the clustering loss contributes more to the overall optimization. This balancing strategy allows TabClusterNet to boost clustering effectiveness while preserving the data's intrinsic properties, thereby achieving superior performance in tabular data clustering.

## *3.3 Feature Extraction Process*

The encoder in this process is sequential, with attention progressively at each decision level on the most informative features. Less important features are ignored, since the attention mechanism focuses on retaining only those features which are most relevant, based on which a compressed feature representation is formed. The latent features are then decoded to the original dimensionality by a decoder for the computation of reconstruction loss. The process learns such features that are not only effective at reconstructing the data but also have great discriminative ability for clustering. BN and feature transformers are applied directly on the original tabular data, which includes numeric features and categorical represented by trainable embeddings. The architecture for feature extraction is illustrated in Figure 2.

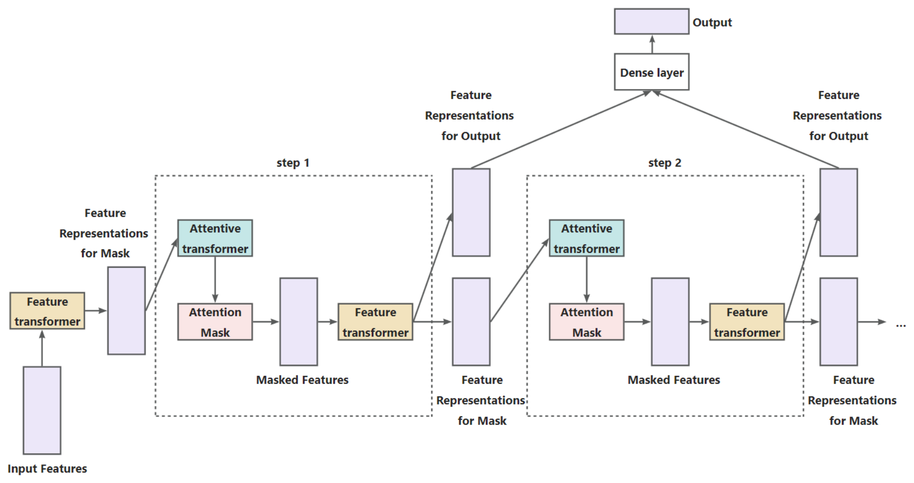


*Figure 2: The TabEncoder architecture in TabClusterNet for feature extraction processes data through a sequential multi-step approach, emphasizing the most significant features at each decision step. This attention mechanism retains the key features to create a compressed feature representation. This representation is then reconstructed back to its original space using the TabDecoder to calculate reconstruction loss, aiming to learn features that effectively reconstruct the original data while providing strong discriminative power for clustering.*

**Feature Selection:** Feature selection is implemented through a learnable mask $M[i] \in \mathbb{R}^{B \times D}$ for soft selection of the most significant features. This mask, used multiplicatively $M[i] \cdot f$, optimizes the model's parameter efficiency by sparsely selecting the most significant features at each decision step. Sparsemax normalization guides this process, encouraging sparsity and enabling the model to focus on a subset of relevant features within the high-dimensional input. The attentive transformer (Figure 3) utilizes features from the previous step $a[i-1]$ to create the mask: $M[i] = sparse\max\left(P[i-1] \cdot h_i\left(a[i-1]\right)\right)$, where $\sum_{j=1}^{D} M[i]_{b,j} = 1$, and $h_i$ is a trainable function, as shown in the FC (fully connected) +BN layer in Figure 3. $P[i]$ represents the prior scale, indicating previous usage of a specific feature: $P[i] = \prod_{j=1}^{i}(\Upsilon - M[j])$, where $\Upsilon$ is a relaxation parameter ensuring a feature is used only in one decision step when $\Upsilon = 1$. $P[0]$ is initialized as all ones, $1^{B \times D}$, imposing no prior on the mask features.

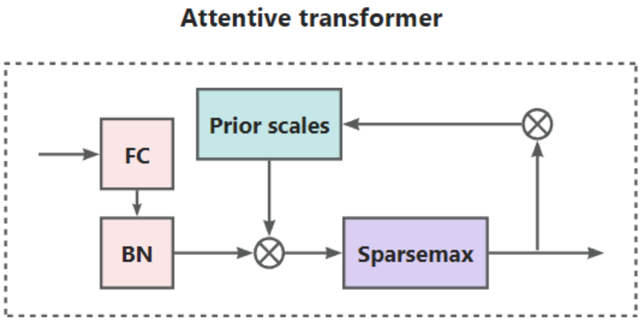


*Figure 3: An example of the attentive transformer block utilized in TabClusterNet demonstrates feature selection implemented through a learnable mask for the soft selection of the most significant features. This method optimizes parameter efficiency by sparsely selecting the relevant features at each decision step.*

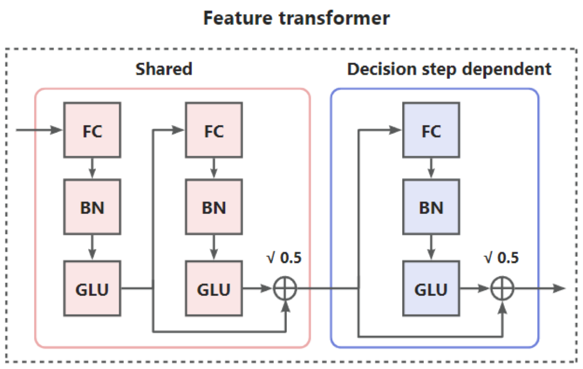


*Figure 4: An example of the feature transformer block in TabClusterNet demonstrates how filtered features pass through the transformer and are divided into outputs for the decision step and information for subsequent steps. The feature transformer includes shared layers across all decision steps and specific layers for each decision step. Each FC layer is followed by BN and gated linear units (GLU), leading to a normalized residual connection.*

**Feature Processing:** The filtered features pass through a feature transformer and are divided into outputs $d[i]$ for the decision step and information $a[i]$ for subsequent steps, $[d[i],a[i]] = f_i(M[i]\cdot f)$, where $d[i]\in\mathbb{R}^{B\times N_d}$ and $a[i]\in\mathbb{R}^{B\times N_a}$. For parameter-efficient and robust learning, the feature transformer includes layers shared across all decision steps (since the same features are input at different steps) and decision step-specific layers. Figure 4 illustrates this as a combination of two shared layers and one step-specific layer.

Normalizing by $\sqrt{0.5}$ stabilizes learning by keeping the variance consistent throughout the network [10]. Inspired by decision trees, the overall decision embedding is constructed as $d_{out}=\sum_{i=1}^{N_{steps}}\mathrm{Re}LU(d[i])$, which is then applied to a linear mapping to produce the output.

**Data Reconstruction:** The core of data reconstruction lies in the decoder architecture illustrated in Figure 5. The decoder includes a feature transformer followed by FC layers at each decision step. The outputs are summed to achieve the reconstructed features. A binary mask $S\in\{0,1\}^{B\times D}$ encodes the reconstructed features $(1-S)\cdot f$ input to the encoder, and the reconstructed features output $S\cdot\hat{f}$ by the decoder. Consider the reconstruction loss:

$$L_r=\sum_{b=1}^{B}\sum_{j=1}^{D}\left|\frac{\left(\hat{f}_{b,j}-f_{b,j}\right)\cdot S_{b,j}}{\sqrt{\sum_{b=1}^{B}\left(f_{b,j}-1/B\sum_{b=1}^{B}f_{b,j}\right)^2}}\right|^2$$

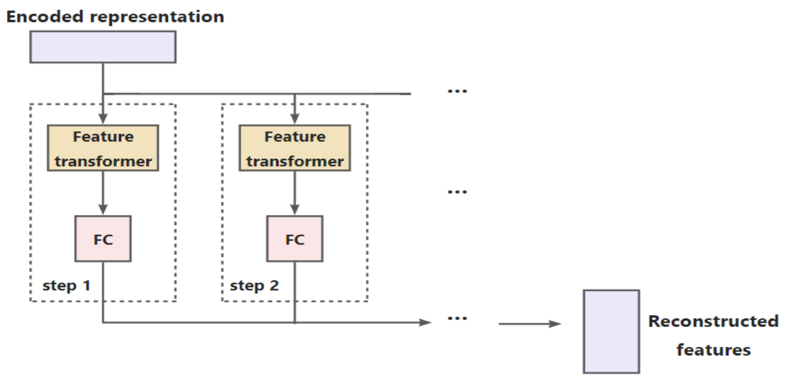


*Figure 5: The architecture of the TabDecoder in TabClusterNet includes a feature transformer followed by FC layers at each decision step, with the outputs summed to achieve the reconstructed features. A binary mask is used to encode the reconstructed features input to the encoder and the reconstructed features output by the decoder.*

### *3.4 Clustering Process*

DEC proposed a clustering loss based on the Kullback-Leibler (KL) divergence between distributions $P$ and $Q$. $Q$ is the soft label distribution measured by the Student's t-distribution, while $P$ is the target distribution derived from $Q$. The clustering loss is formulated as:

$$L_c = KL(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

The KL divergence measures the asymmetric difference between two probability distributions, $P$ and $Q$. $q_{ij}$ represents the similarity between the embedded point $z_i$ and the cluster center $\mu_j$, measured by the Student's t-distribution [11]:

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

The target distribution $p_{ij}$ is defined as:

$$p_{ij} = \frac{q_{ij}^2 / \sum_i q_{ij}}{\sum_j \left( q_{ij}^2 / \sum_i q_{ij} \right)}$$

The target distribution $P$ is defined by $Q$, so minimizing $L_c$ serves as self-training [12]. Let $f_W$ be the encoder mapping, $z_i = f_W(f_i)$, where $f_i$ is a sample from tabular data $f$. We can use $f_W$ to get embeddings $z_i$, apply k-means on $z_i$ to determine initial cluster centers $\mu_j$, and proceed with $L_c$ calculations as per the previous formula.

### *3.5 Optimization and Training Strategies*

This paper uses mini-batch Stochastic Gradient Descent (SGD) and backpropagation for optimization. Specifically, three sets of parameters are optimized: the encoder weights, the cluster centers, and the target distribution $P$.

**Update the weights of the encoder and the cluster centers:** The gradient of $L_c$ relative to the embedding point $z_i$ and the cluster center $\mu_j$ is calculated as:

$$\frac{\partial L_c}{\partial z_i} = 2\sum_{j=1}^{K} (1+\|z_i-\mu_j\|^2)^{-1} (p_{ij}-q_{ij})(z_i-\mu_i)$$

$$\frac{\partial L_c}{\partial \mu_j} = 2\sum_{i=1}^{n} (1+\|z_i-\mu_j\|^2)^{-1} (q_{ij}-p_{ij})(z_i-\mu_i)$$

The derivation above is from DEC. For a mini-batch with $m$ samples and a learning rate $\lambda$, the update of $\mu_j$ is done using the formula:

$$\mu_j = \mu_j - \frac{\lambda}{m} \sum_{i=1}^{m} \frac{\partial L_c}{\partial \mu_j}$$

The weights of the decoder are updated using the following formula:

$$W' = W' - \frac{\lambda}{m} \sum_{i=1}^{m} \frac{\partial L_r}{\partial W'}$$

The weights of the encoder are updated using the following formula:

$$W = W - \frac{\lambda}{m}\sum_{i=1}^{m}\left(\frac{\partial L_r}{\partial W} + \gamma\frac{\partial L_c}{\partial W}\right)$$

**Update target distribution:** The target distribution serves as "ground truth" soft labels but also depends on predicted soft labels. To prevent instability, the target distribution $P$ should not be updated using only one data batch per iteration. When updating the target distribution, the label assigned to $f_i$ is derived from, where $q_{ij}$ is computed from the previous formula:

$$s_i = arg\ \max_j q_{ij}$$

## 4. Experiments

### *4.1 Dataset*

**Dataset:** We utilized three public datasets, each with a single train-validation-test split to maintain consistency for all algorithms. The selected datasets include Helena (HE, anonymized) [13], Epsilon (EP, simulated physics experiments) [14], and Covertype (CO, forest characteristics) [15]. Table 1 provides a summary of the dataset attributes.

*Table 1: A summary of the attributes of the three public datasets used in this study, including sample size, dimensions, and the number of classes.*

| Data set | Sample size | Dimensions | Classes |
|---|---|---|---|
| HE | 65196 | 27 | 100 |
| EP | 500000 | 2000 | 2 |
| CO | 581012 | 54 | 7 |

### *4.2 Baseline*

To comprehensively evaluate the clustering performance of the developed TabClusterNet model, we proposed several baselines for comparison.

**K-means:** The algorithm iteratively calculates the distances between data points and centroids, reallocating points to the nearest clusters, so that the within-cluster sum of squared distances is minimized. In our experiments with different initializations, K-means was run 20 times, and the best result was selected.

**PCA+K-means:** Principal Component Analysis (PCA) reduces the dataset's dimensionality, followed by K-means clustering in the lower-dimensional space. As a result of this preliminary treatment, PCA constructs a clearer clustering structure by discarding noisy and redundant features.

**Autoencoder (AE)+K-means:** This approach employs an autoencoder to learn a low-dimensional dense representation of the data, followed by K-means clustering on these representations. The feature extraction capabilities of autoencoders provide more compact and meaningful data representations.

**DEC:** DEC unifies feature learning and clustering by iteratively refining a clustering objective to enhance the low-dimensional data representation, thereby improving clustering quality. The encoder and decoder are structured as $d-512-2048-16-2048-512-d$, where $d$ is the input data dimension, and the embedding dimension is 16.

### *4.3 Experiment Setup*

**Parameter settings:** Drawing from the TabNet and DEC models, the initial parameters are configured as follows: dimension of the prediction layer and dimension of the attention layer are both set to 2, number of steps is set to 2, scaling factor for attention updates is 1.3, number of independent GLU layers and number of shared GLU layers are both 2. In the clustering layer, the clustering loss coefficient $\gamma$ is set to 0.81 (identified through grid search from {0.1, 0.19, 0.27, 0.34, 0.41, 0.61, 0.86, 1.0}), and the batch size across all datasets is 512. The initial learning rate $\lambda$ for the Adam optimizer [16] is set at 0.001.

**Evaluation Metric:** To compare the algorithms, we use the Fowlkes-Mallows Index, Normalized Mutual Information, Silhouette Coefficient, and Davies-Bouldin Index for evaluation. For each algorithm, the number of clusters was set to the actual number of classes.

### *4.4 Performance on Different Datasets*

Tables 2, 3, and 4 provide a performance comparison of five clustering methods across different datasets, revealing valuable insights:

For the Helena dataset (Table 2), characterized by a moderate sample size, numerous categories, and low dimensionality, K-means and PCA+K-means perform robustly. However, TabClusterNet excels in the Silhouette Coefficient (SC) and the Davies-Bouldin Index (DBI). This indicates that TabClusterNet is particularly proficient at maintaining the data's intrinsic structure and achieving well-separated clusters.

For the Epsilon dataset (Table 3), which features a large sample size and high dimensionality with fewer categories, TabClusterNet's FMI is slightly lower than that of the top-performing method. However, the comparison shows that TabClusterNet's superior performance in NMI and SC measures, compared to the best results of other methods, underlines its strong robustness in capturing intrinsic data structures in high-dimensional data.

The Covertype dataset (Table 4) has the largest dimensionality, with a moderate number of categories. TabClusterNet achieves competitive FMI and SC scores while significantly outperforming in the DBI index. This further confirms TabClusterNet's methodological robustness in maintaining clustering quality, particularly in terms of cohesion and separation.

Overall, it can be observed that although TabClusterNet is not always the best in FMI scores, it tops or equals all other methods with respect to NMI, SC, and DBI. Therefore, this confirms TabClusterNet's ability to capture and exploit the underlying data structure for deeper insights in data analytics and decision support. TabClusterNet is very effective on large, complex, and high-dimensional datasets, making it highly useful for practical applications.

*Table 2: A comparison of five clustering methods on the Helena dataset shows that TabClusterNet performs exceptionally well according to both the Silhouette Coefficient and Davies–Bouldin Index. This indicates that this method effectively preserves the intrinsic structure of the data.*

| | **FMI** | **NMI** | **SC** | **DBI** |
|---|---|---|---|---|
| **K-means** | 0.271 | 0.249 | 0.288 | 1.857 |
| **PCA+K-means** | 0.295 | 0.166 | 0.331 | 1.066 |
| **AE+K-means** | 0.273 | 0.259 | 0.282 | 1.911 |
| **DEC** | 0.335 | 0.152 | 0.289 | 1.496 |
| **TabClusterNet** | 0.350 | 0.180 | 0.447 | 0.791 |

*Table 3: A performance comparison of the five clustering methods was conducted on the Epsilon dataset, showing that TabClusterNet has an FMI only slightly lower than the benchmarked best-performing method but mainly better or rather good results in terms of NMI, SC, and the DBI score against the scores from other approaches. This demonstrates that TabClusterNet effectively captures intrinsic data structures under high-dimensional settings.*

| | **FMI** | **NMI** | **SC** | **DBI** |
|---|---|---|---|---|
| **K-means** | 0.506 | 0.0045 | 0.364 | 1.156 |
| **PCA+K-means** | 0.506 | 0.0045 | 0.483 | 0.802 |
| **AE+K-means** | 0.504 | 0.0266 | 0.475 | 0.825 |
| **DEC** | 0.629 | 0.0345 | 0.238 | 0.955 |
| **TabClusterNet** | 0.685 | 0.0141 | 0.774 | 0.467 |

*Table 4: Through comparison of five clustering methods' performance on Covertype, it shows that TabClusterNet achieves a score that is competitive in FMI and SC and outperforms in DBI, thus further validating its robustness in maintaining clustering quality.*

| | **FMI** | **NMI** | **SC** | **DBI** |
|---|---|---|---|---|
| **K-means** | 0.294 | 0.243 | 0.281 | 1.766 |
| **PCA+K-means** | 0.574 | 0.126 | 0.146 | 0.789 |
| **AE+K-means** | 0.335 | 0.269 | 0.243 | 1.624 |
| **DEC** | 0.580 | 0.215 | 0.219 | 1.157 |
| **TabClusterNet** | 0.539 | 0.267 | 0.557 | 0.605 |

## 5. Conclusion

This paper introduces TabClusterNet, an advanced deep learning model designed for clustering tabular data. Comprehensive experiments on three different datasets, namely Helena, Epsilon and Covertype, show that TabClusterNet is robust in its clustering capabilities across various data contexts.

In the Helena dataset, TabClusterNet performs strongly, especially with Silhouette Coefficient and Davies-Bouldin Index values, which point to good capturing of the intrinsic structure in the data. The ability of TabClusterNet to preserve the data's characteristics and quality of the clustering is obvious, even though it does not score the best in the Fowlkes-Mallows Index and Normalized Mutual Information. For the high-dimensional Epsilon dataset, TabClusterNet demonstrates good effectiveness in preserving the intrinsic data structure while it reveals a stable clustering performance. The stability in the quality of clustering was also evident on the Covertype dataset, with TabClusterNet achieving significantly better performance over all measures, mainly for SC and DBI.

In conclusion, performance in different datasets and scenarios stands out due to its integrations, innovative in nature, with self-supervised learning TabNet encoders with decoders and deep clustering framework of DEC. Future research will include exploration over more data sets, optimization of the model architecture, and validation on real-world problems. We believe that TabClusterNet will provide a highly robust solution for clustering of tabular data and unlock deep insights for data scientists and researchers.

## References

*[1] Bishop, C. M., & Nasrabadi, N. M. (2006). Pattern recognition and machine learning . New York: springer. (Vol. 4, No. 4, p. 738)*
*[2] Tian, F., Gao, B., Cui, Q., Chen, E., & Liu, T. Y. (2014, June). Learning deep representations for graph clustering. In Proceedings of the AAAI conference on artificial intelligence (Vol. 28, No. 1, p56).*
*[3] Peng, X., Xiao, S., Feng, J., Yau, W. Y., & Yi, Z. (2016, July). Deep subspace clustering with sparsity prior. In IJCAI (pp. 1925-1931).*
*[4] Xie, J., Girshick, R., & Farhadi, A. (2016, June). Unsupervised deep embedding for clustering analysis. In International conference on machine learning (pp. 478-487).*
*[5] Shwartz-Ziv, R., & Armon, A. (2022). Tabular data: Deep learning is not all you need. Information Fusion, 81, 84-90.*
*[6] Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., & Kasneci, G. (2022). Deep neural networks and tabular data: A survey. IEEE Transactions on Neural Networks and Learning Systems.*
*[7] Abrar, S., & Samad, M. D. (2022). Are Deep Image Embedding Clustering Methods Effective for Heterogeneous Tabular Data. arXiv preprint arXiv:22, 12:14111.*
*[8] Kingma D P, Welling M .Auto-Encoding Variational Bayes [J].arXiv.org, 2014.DOI:10. 48550/arXiv. 1312.6114.*
*[9] Arik, S. Ö., & Pfister, T. (2021, May). Tabnet: Attentive interpretable tabular learning. In Proceedings of the AAAI conference on artificial intelligence (Vol. 35, No. 8, pp. 6679-6687).*
*[10] Gehring, J., Auli, M., Grangier, D., Yarats, D., & Dauphin, Y. N. (2017, July). Convolutional sequence to sequence learning. In International conference on machine learning (pp. 1243-1252).*
*[11] Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. Journal of machine learning research, 9(11).*
*[12] Nigam, K., & Ghani, R. (2000, November). Analyzing the effectiveness and applicability of co-training. In Proceedings of the ninth international conference on Information and knowledge management (pp. 86-93).*
*[13] Guyon, I., Sun-Hosoya, L., Boullé, M., Escalante, H. J., Escalera, S., Liu, Z., ... & Viegas, E. (2019). Analysis of the AutoML challenge series. Automated Machine Learning, 177.*
*[14] Yuan, G. X., Ho, C. H., & Lin, C. J. (2011, August). An improved glmnet for l1-regularized logistic regression. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 33-41).*
*[15] Blackard, J. A., & Dean, D. J. (1999). Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. Computers and electronics in agriculture, 24(3), 131-151.*
*[16] Nie, F., Zeng, Z., Tsang, I. W., Xu, D., & Zhang, C. (2011). Spectral embedded clustering: A framework for in-sample and out-of-sample spectral clustering. IEEE Transactions on Neural Networks, 22(11), 1796-1808.*