Interpretable Deep Clustering for Tabular Data

Jonathan Svirsky [1] **Ofir Lindenbaum** [1]

Abstract

Clustering is a fundamental learning task widely used as a first step in data analysis. For example, biologists use cluster assignments to analyze genome sequences, medical records, or images. Since downstream analysis is typically performed at the cluster level, practitioners seek reliable and interpretable clustering models. We propose a new deep-learning framework for general domain tabular data that predicts interpretable cluster assignments at the instance and cluster levels. First, we present a self-supervised procedure to identify the subset of the most informative features from each data point. Then, we design a model that predicts cluster assignments and a gate matrix that provides cluster-level feature selection. Overall, our model provides cluster assignments with an indication of the driving feature for each sample and each cluster. We show that the proposed method can reliably predict cluster assignments in biological, text, image, and physics tabular datasets. Furthermore, using previously proposed metrics, we verify that our model leads to interpretable results at a sample and cluster level. Our code is available on Github.

1. Introduction

Clustering is a crucial task in data science that helps researchers uncover and study latent structures in complex data. By grouping related data points into clusters, researchers can gain insights into the underlying characteristics of the data and identify relationships between samples and variables. Clustering is used in various scientific fields, including biology (Reddy et al., 2018), physics (Mikuni & Canelli, 2021), and social sciences (Varghese et al., 2010). For instance, in biology, clustering can identify different disease subtypes based on molecular or genetic data. In psychology, based on survey data, clustering can identify different types of behavior or personality traits.

Clustering is a common technique used in bio-medicine to analyze gene expression data. It involves identifying groups of genes that have similar expression patterns across different samples. Scientists often cluster high-dimensional points corresponding to individual cells to recover known cell populations and discover new, potentially rare cell types. However, bio-med gene expression data is generally represented in a tabular, high-dimensional format, making it difficult to obtain accurate clusters with meaningful structures. In addition, interpretability is a crucial requirement for real-world bio-med datasets since it is essential to understand the biological meaning behind the identified clusters. As a result, there is an increasing demand in bio-medicine for clustering models that offer interpretability for tabular data.

In bio-medicine, clustering is widely used to analyze gene expression data, where cluster assignments can identify groups of genes with similar expression patterns across different samples (Armingol et al., 2021). When applied to single-cell omics data (Wang & Bodovitz, 2010), clustering recovers known cell populations while discovering new and perhaps rare cell types (Deprez et al., 2020). Unfortunately, such bio-med gene expression data types are generally represented in a tabular, high-dimensional format, making it difficult to obtain accurate clusters with meaningful structures. In addition, interpretability is a crucial requirement for real-world bio-med datasets since it is essential to understand the biological meaning behind the identified clusters (Yang et al., 2021). As a result, there is an increasing demand in bio-medicine for clustering models that offer interpretability for tabular data.

Interpretability in machine learning refers to the ability to understand and explain the predictions and decisions made by predictive models. It is critical for the proper deployment of machine learning systems, especially in applications where transparency and accountability are essential. Interpretability can take different forms, including interpretable model structure, identification of feature importance for model predictions, visualization of data, and generation of explanations for the prediction. In this work,

[*] Equal contribution [1] Department of Engineering, Bar Ilan University, Ramat-Gan, Israel. Correspondence to: Jonathan Svirsky <svirskj@biu.ac.il>, Ofir Lindenbaum <ofir.lindenbaum@biu.ac.il>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

we aim to design a model that achieves interpretability by sample-wise feature selection and generating cluster-level interpretations of model results. This type of interpretability is crucial for biomedical applications, for example, when seeking marker genes that are "typical" for different clusters in high-dimensional biological measurements.

In recent years, the use of deep learning models for clustering has been gaining interest (Shen et al., 2021; Li et al., 2022; Cai et al., 2022; Niu et al., 2021). These models offer better clustering capabilities by providing an improved embedding of data points. However, most existing schemes focus on image data, require domain-specific augmentations, and are not interpretable. Interpretability has also been gaining attention in deep learning, but most models focus on supervised learning (Alvarez Melis & Jaakkola, 2018; Yoon et al., 2019; Yang et al., 2022). We aim to extend these ideas to unsupervised learning by developing a deep clustering model that is interpretable by design and can be applied to general domain data. In this context, interpretability means being able to identify variables that *drive* the formation of clusters in the data (Bertsimas et al., 2021). We demonstrate the effectiveness of our method in terms of clustering accuracy and interpretability in biomed, physics, text, and image data represented in tabular format.

This work introduces **I**nterpretable **D**eep **C**lustering (**IDC**), an unsupervised two-stage clustering method for tabular data. The method first selects informative features for reliable data reconstruction, a task that was demonstrated to be correlated with clustering capabilities (Song et al., 2013; Han et al., 2018). Then, using the sparsified data, we learn the cluster assignments by optimizing neural network parameters subject to a clustering objective function (Yu et al., 2020). In addition, the method provides both instance-level and cluster-level explanations represented by the selected feature set. The model learns instance-level *local gates* that select a subset of features using an autoencoder (AE) trained to reconstruct the original sample. The *global gates* for cluster-level interpretability are derived from the cluster label assignments learned by the model. To ensure sample-level interpretations, the gates are encouraged by the recently proposed discrimination constraint denoted as the total coding rate. Using synthetic data and MNIST, we demonstrate the interpretability quality of our model. Then, we use real-world data from biomed, images, text, and physics to demonstrate that our model can find meaningful clusters while using only a small subset of informative features. In the following sections, we provide a detailed description of our approach.

2. Related Work

Unsupervised Feature Selection The problem of unsupervised feature selection (UFS) involves identifying variables useful for downstream tasks such as data clustering. Towards this goal, several works have exploited regularized AEs (Han et al., 2018; Lee et al., 2022; Sokar et al., 2022; Balın et al., 2019), which identify a global subset of features that are sufficient for data reconstruction. Another line of UFS schemes relies on the graph Laplacian (Von Luxburg, 2007) to identify subsets of smooth features with respect to the core structure in the data (He et al., 2005; Zhao & Liu, 2012; Lindenbaum et al., 2021; Shaham et al., 2022). Both types of UFS frameworks can help improve downstream clustering capabilities; however, existing global schemes do not provide sample-level or cluster-level interpretability. Although there are recent works on supervised local feature selection (Yoon et al., 2019; Yang et al., 2022) that provide interpretability, we are not aware of any sample-level unsupervised feature selection schemes. Therefore, we present, for the first time, an end-to-end clustering scheme with local feature selection capabilities.

Interpretable Clustering Guan et al. (2011) presented a pioneering work for simultaneous unsupervised feature selection and clustering. The authors proposed a probabilistic model that performs feature selection by using beta-Bernoulli prior in the context of a Dirichlet process mixture for clustering. However, their model can only select cluster-level and dataset-level informative features, whereas our approach offers interpretability with sample-level granularity. In (Frost et al., 2020), the authors proposed tree-based K-means clustering as a part of other works in the same direction (Lawless et al., 2022; Gabidolla & Carreira-Perpiñán, 2022; Cohen, 2023; Kauffmann et al., 2022). However, the explanations are global only for a given dataset and rely on the whole set of points. In contrast, our approach learns local gates for each sample in the dataset by optimizing a neural network that performs local feature selection, thus producing sample-level interpretations. Since our method is fully parametric, it also offers enhanced generalization capabilities compared to existing schemes. Precisely, our model can predict cluster assignments and informative features for samples not seen during training.

Deep clustering Recently, several methods were proposed for NN-based clustering, to name a few: (Gao et al., 2020; Niu et al., 2021; Li et al., 2022; Shaham et al., 2018; Cai et al., 2022; Lv et al., 2021; Shen et al., 2021; Peng et al., 2022). However, these methods primarily focus on vision and rely on domain-specific augmentations and, therefore, can not be applied to tabular datasets. Therefore, we introduce an interpretable NN model for general high-dimensional datasets, such as biomedical tabular data.

Li et al. (2022) have generalized the maximum coding rate reduction loss (MCR^2) (Yu et al., 2020) for embedding and clustering. The model aims to separate clusters while mak-

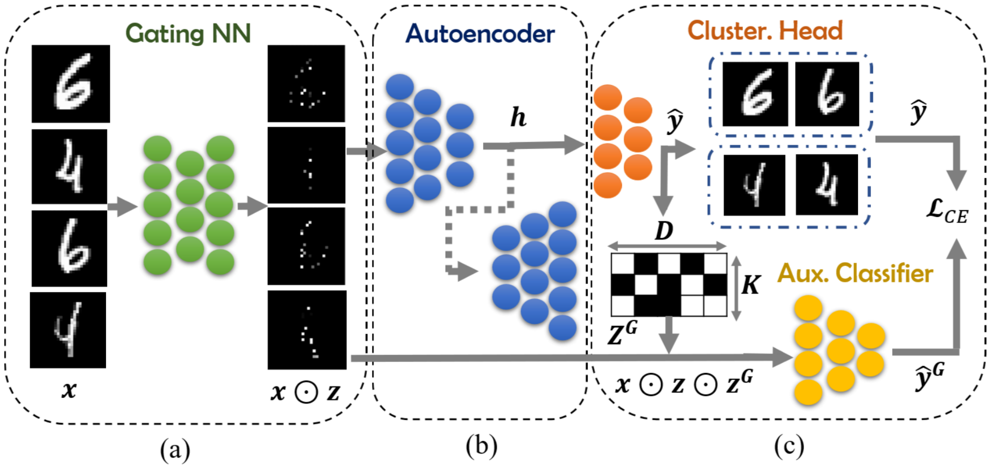

Figure 1. Illustration of the proposed model. The first step involves self-supervision for learning a meaningful latent representation and sample-level informative features. During this stage, we optimize the parameters of the Gating Network (green) and the autoencoder (blue) that reconstructs $\hat{x}$ from latent embedding h. The gating Network learns a sample-specific sparse gate vector z for input sample x such that $x \odot z$ is sufficient for reconstruction via an autoencoder. We train a clustering head (orange) to predict cluster assignments $\hat{y}$ from the latent embedding h by minimizing the mean cluster coding rate loss (see Eq. 4). This loss is designed to push clusters apart while making each cluster more compact. The Auxiliary Classifier (yellow) is trained on sparse representations $x \odot z \odot Z^G$ to predict cluster labels and optimizes the cluster level gating matrix Z^G.

ing them denser. However, this scheme also requires domain knowledge to design semantic preserving augmentations to train the contrastive loss. While such augmentations could be designed effectively for the visual domain, creating such semantic preserving augmentations of tabular data remains an open challenge (Qian et al., 2023). To overcome this limitation, we propose a two-step procedure that involves (i) self-supervision with locally sparse reconstruction that improves interpretability and induces a spectral bias that enhances performance on tabular data, and (ii) an adapted MCR^2 (Li et al., 2022) objective for identifying clusters based on diverse features.

3. Problem Setup

We want to cluster a set of data points $\mathbf{X} = \{\boldsymbol{x}_i\}_{i=1}^N$ into K clusters. Each data point $\mathbf{x}_i \in R^D$ is a D-dimensional vector-valued observation of general type, meaning it does not follow a particular feature structure. Our goal is to learn an interpretable clustering model defined by the tuple $\langle f_\Theta, \mathcal{S}^{glob} \rangle$ such that $f_\Theta(\boldsymbol{x}_i) = \{\hat{y}_i, \mathcal{S}_i^{loc}\}$ where $\hat{y}_i \in \{1, 2, ..., K\}$ is an accurate clustering assignment, and $\mathcal{S}_i^{loc}$ is a local feature importance set for sample i and defined by $\mathcal{S}_i^{loc} = \{s_i^j \in [0,1]\}_{j=1}^D$. $\mathcal{S}^{glob} \in [0,1]^{K \times D}$ is a global feature importance matrix where each gating vector of size D is learned for K clusters. By forcing $|\mathcal{S}_i^{loc}| << D$, we can attenuate nuisance features and identify (sample-specific) subsets of informative features, thus improving the interpretability of the clustering model.

Our research is driven by the critical task in biomedicine of cell clustering and identifying marker genes that are associated with each cluster (Kiselev et al., 2017). This task involves clustering a high-dimensional dataset while identifying genes with unique patterns in each cluster. Therefore, our goal is to develop an interpretable model that performs well in the clustering task, i.e., can identify groups with semantically related samples while focusing on local subsets of features. We want to highlight that in the supervised setting, this kind of local feature selection has been linked to interpretability by various authors (Alvarez Melis & Jaakkola, 2018; Yoon et al., 2019; Yang et al., 2022).

We have extended the concept of supervised interpretability to the unsupervised domain by introducing a new model that combines clustering and local feature selection. This model identifies and eliminates irrelevant input features that do not contribute to the clustering task. Moreover, the model generates unique explanations for each sample and cluster-level interpretations, which can help comprehend the predictions. We have used various metrics to demonstrate that our model enhances interpretability and produces accurate clustering results.

4. Method

We suggest a two-step approach involving training self-supervised *local* (sample-specific) gates and latent representations to enhance cluster separation. This approach provides interpretability through local feature selection, which improves performance on tabular data by allowing for the learning of high-frequency prediction functions using Neural Networks (NNs). In contrast to image or audio data, tabular data requires a prediction function with higher frequency components (Beyazit et al., 2023). However, the implicit bias of NNs towards learning low-frequency functions, as described in (Basri et al., 2020), hinders the performance of vanilla fully connected models on tabular data. We empirically demonstrate that *local* sparsification (gates) can aid NNs in learning high-frequency functions and improving predictive capabilities, as shown in Section 6.3.

In the second step of our proposed method, we identify the cluster assignments and *global* gates which focus on the driving features at the cluster level. The *global* gates are learned based on the clustering assignments, which are determined by minimizing the *coding rate* (Eq. 4) for each cluster. The result of this step is the clustering assignments for each sample, local gates for sample-level interpretation, and *global* gates for cluster-level explanations. The following subsection will present the proposed framework and architecture design in more detail.

4.1. Local Self Supervised Feature Selection

Given unlabeled observations $\{\boldsymbol{x}_i\}_{i=1}^N$, where $\boldsymbol{x_i} \in \mathbb{R}^D$, our goal is to learn a prediction function f_θ (parametrized using a NN) and a feature importance vectors $\boldsymbol{z}_i \in [0,1]^D$. The feature importance vector will “highlight” which subset of variables the model should rely on for clustering each sample $\boldsymbol{x}_i$. This will allow the model to use fewer features for each sample and reduce overfitting when predicting the clusters of unseen samples.

Towards our goal, we expand upon the recently proposed stochastic gates (Lindenbaum et al., 2021; Yamada et al., 2020) by allowing them to learn the importance vector locally (for each sample). These stochastic gates are continuously relaxed Bernoulli variables defined based on the hard thresholding function for feature d and sample i:

$$z_i^d = \max(0, \min(1, 0.5 + \mu_i^d + \epsilon_i^d)), \tag{1}$$

where ϵ_i^d is drawn from $\mathcal{N}(0,\sigma^2)$. The value μ_i^d is the logit output of the network before being passed through the hard thresholding function (in Eq. 1). In our model, σ is fixed to 0.5 throughout training, as suggested in (Yamada et al., 2020). The injected noise is controlled by σ and helps push the converged values of z_i^d towards 0 or 1. For more information, please see (Yamada et al., 2020) and (Yang et al., 2022). The parameters $\boldsymbol{\mu}_i \in \mathbb{R}^D, i = 1, ..., N$ are specific to each sample and are predicted using a gating network f_{θ_1}. In other words, $\boldsymbol{\mu}_i = f(\boldsymbol{x}_i|\theta_1)$, where θ_1 are the weights of the gating network. These weights are learned simultaneously with the weights of the prediction network θ_2 by minimizing the following loss:

$$\mathcal{L}_{\text{sparse}} = \mathbb{E}_{\boldsymbol{z}_i,\boldsymbol{x}_i}\left[\mathcal{L}(f_{\theta_2}(\boldsymbol{x}_i \odot \boldsymbol{z}_i)) + \lambda \cdot \mathcal{L}_{\text{reg}}(\boldsymbol{z}_i)\right],$$

where $\mathcal{L}$ is a desired prediction loss, for example, a clustering objective function or reconstruction error. The Hadamard product is simply an element-wise multiplication and is denoted by $\odot$. We compute the empirical expectation over $\boldsymbol{x}_i$ and $\boldsymbol{z}_i$, where i is an index that ranges over a dataset of size N. The term $\mathcal{L}_{\text{reg}}(\boldsymbol{z}_i)$ is a regularizer designed to sparsify the gates. It is defined as follows: $\mathcal{L}_{\text{reg}} = \|\boldsymbol{z}_i\|_0$. After taking the expectation over $\boldsymbol{z}_i$ and the samples $\boldsymbol{x}_i$, $\mathbb{E}[\mathcal{L}_{\text{reg}}]$ can be rewritten using a double sum in terms of the Gaussian error function (erf):

$$\mathcal{L}_{\text{reg}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{d=1}^{D}\left(\frac{1}{2} - \frac{1}{2}\,\text{erf}\left(-\frac{\mu_i^d + 0.5}{\sqrt{2}\sigma}\right)\right), \tag{2}$$

here, we calculate the expectation using the parametric definition of $\boldsymbol{z}_i$.

We have chosen a denoising autoencoder (Vincent et al., 2008) for our prediction network. It aims to identify and select only the informative features required for reconstruction by disregarding nuisance features. The network is trained through self-supervision using a reconstruction loss with domain-agnostic augmentations. This helps the network learn a latent embedding of the input sample and prompts the gating network to open only the gates needed to reconstruct data. The model consists of the following components:

- *Gating Network*: $f_{\theta_G}(\boldsymbol{x}_i) = \boldsymbol{z}_i$, is a hypernetwork that predicts the gates $\boldsymbol{z}_i$ vector for sample $\boldsymbol{x}_i$, where $\boldsymbol{z}_i \in [0,1]^D$ (each element is defined based on Eq.1).
- *Encoder*: $f_{\theta_E}(\boldsymbol{x}_i') = \boldsymbol{h}_i$, is a mapping function that learns an embedding $\boldsymbol{h}_i$ based on the element-wise gated sample $\boldsymbol{x}_i' = \boldsymbol{x}_i \odot \boldsymbol{z}_i$.
- *Decoder*: $f_{\theta_D}(\boldsymbol{h}_i) = \hat{\boldsymbol{x}}_i$, that reconstructs $\boldsymbol{x}_i$ based on the embedding $\boldsymbol{h}_i$.

We utilize an autoencoder with parameters $\theta_E \cup \theta_D$ and employ gated input reconstruction loss $\mathcal{L}_{\text{recon}}(f_{\theta_D}(f_{\theta_E}(f_{\theta_G}(\boldsymbol{x}_i) \odot \boldsymbol{x}_i)), \boldsymbol{x}_i)$ to measure the deviation between estimated $\hat{\boldsymbol{x}}_i = f_{\theta_D}(f_{\theta_E}(f_{\theta_G}(\boldsymbol{x}_i) \odot \boldsymbol{x}_i))$ and input sample $\boldsymbol{x}_i$. We introduce input (Vincent et al., 2008) and latent data augmentations (Doi et al., 2007) to learn semantically informative features. Additional details about these augmentations are in Appendix F. Additionally, we introduce an extra *gates total coding rate loss*, $\mathcal{L}_{\text{gtcr}}(\boldsymbol{Z})$ that encourages the selection of unique gates for each sample. The following equation defines this loss:

$$\mathcal{L}_{\text{gtcr}} = -\frac{1}{2} \cdot \text{logdet}(\mathbf{I} + \frac{D}{N_B \cdot \epsilon} \cdot (\boldsymbol{Z}^T \boldsymbol{Z})),$$

which is approximately the negative Shannon coding rate of a multivariate Gaussian distribution (Yu et al., 2020) up to precision ϵ, and is defined for a mini-batch of size N_B of normalized local gates $\boldsymbol{Z} \subseteq \{\boldsymbol{z}_i\}_1^N$. This component, which is inspired by (Li et al., 2022), aims to decrease the correlation between gate vectors $\boldsymbol{z}_i$.

The final loss for predicting sparse data is calculated as follows:

$$\mathcal{L}_{\text{sparse}} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{gtcr}} + \lambda \cdot \mathcal{L}_{\text{reg}}. \tag{3}$$

The weight term for regularization, represented by λ, is gradually increased in training using a cosine function scheduler. The autoencoder's weights are initially set through standard reconstruction objective training. In Section 6.4, we conduct an ablation study to validate the importance of each component of our loss.

Our model is designed to minimize the number of features required for data reconstruction. This is achieved by sparsifying the input samples. Additionally, the gated input reconstruction loss helps the model learn local masks, which are used to attenuate noisy features and improve interpretability. The gates' total coding rate loss encourages the selection of diverse gates across samples, ensuring that informative features are not constant across the data.

4.2. Cluster assignments with Global Interpretations

Clustering Head During the clustering phase, our goal is to compress the learned representations of gated samples of $\mathbf{X}$ into K clusters and predict cluster-level gates. To accomplish this, we train a clustering head $f_{\theta_C}(\boldsymbol{h}_i) = \hat{y}_i$, which learns cluster one-hot assignments $\hat{y}_i \subset \{1, ..., K\}$. The model outputs logits values $\boldsymbol{\pi}_i^{1\times K}$ for each sample. These logits values are then converted to cluster assignment probabilities $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ by using a Gumbel-Softmax (Jang et al., 2016) reparameterization:

$$\hat{y}_i^k = \frac{\exp((\log(\pi_i^k) + g_i^k)/\tau)}{\sum_{j=1}^K \exp((\log(\pi_i^k) + g_i^k)/\tau)}, \qquad \text{for } k = 1, ..., K.$$

where $\boldsymbol{g}_i^{1\times K}$ are i.i.d. samples drawn from a Gumbel(0, 1) distribution. The variable τ represents a temperature hyperparameter, where larger values of τ produce a uniform distribution of $\hat{\boldsymbol{y}}$. On the other hand, decreasing the temperature leads to one-hot vectors. This neural network is optimized with the following loss

$$\mathcal{L}_{\text{head}} = \sum_{k=1}^K \frac{1}{2} \cdot \text{logdet}\big[\mathbf{I} + \frac{d_h}{N_B \cdot \epsilon} \cdot (\boldsymbol{H}_k^T \boldsymbol{H}_k)\big], \tag{4}$$

where $\mathbf{H}_k = \{\boldsymbol{h}_i : f_{\theta_C}(\boldsymbol{h}_i) = k\} \subseteq \{\boldsymbol{h}_i = f_{\theta_E}(\boldsymbol{x}_i \odot \boldsymbol{z}_i)\}_{i=1}^N$ is a mini-bach of size N_B embedding vectors for all samples $\boldsymbol{x}_i$'s which were assigned to cluster k. d_h is the embedding dimension and ϵ is the hyperparameter that denotes the desired precision of coding rate (Yu et al., 2020). The aim is to decrease the average coding rate for each cluster of embeddings $\mathbf{H}_k$ to make them more compact.

Global Interpretations To provide global cluster-level interpretations, we develop a gate matrix called $\mathbf{Z}_G \in \{0, 1\}^{K\times D}$. Each row in the matrix corresponds to a cluster and each column to an input variable. We use an *auxiliary classifier*, $f_{\theta_{AC}}$, to train this gate matrix. The classifier accepts a gated representation of $\boldsymbol{x}_i$, defined by $\boldsymbol{x}_i \odot \boldsymbol{z}_i \odot \boldsymbol{z}_G^k$, where $\boldsymbol{z}_G^k$ is a global gates vector learned for cluster $k = \text{argmax}_{1,...,K}\hat{\boldsymbol{y}}_i$. The locally sparse samples $\boldsymbol{x}_i \odot \boldsymbol{z}_i$ learned with the autoencoder are multiplied by global gates $\mathbf{Z}_G$ and passed through a single-hidden-layer classifier. This classifier is trained to output cluster assignments $\hat{\boldsymbol{y}}_i$ identical to those predicted with clustering head f_{θ_C} using a cross-entropy loss. During inference, only $\mathbf{Z}_G$ is used, and $f_{\theta_{AC}}$ is not needed.

In our method, we use a regularization loss term called $\mathcal{L}_{\text{reg}}$ with an increasing weight λ_g to sparsify the gates in the global gates matrix. This is added to the clustering loss term $\mathcal{L}_{clust}$, which is the sum of the clustering-head term $\mathcal{L}_{\text{head}}$ and the cross-entropy loss term $\mathcal{L}_{CE}$. To summarize, we train the clustering head to predict assignments along with global gates. We use an auxiliary classifier to optimize the global gates while being trained in a self-supervised manner on the pseudo labels predicted by the clustering head.

5. Interpretability

Practitioners may need interpretability at different levels of granularity. At a coarser level, it is useful to identify which features are common to a group of semantically related samples (or clusters) (Guan et al., 2011). At a finer level of detail, we aim to find unique explanations for each data point. Specifically, we want to know which features drive the model to make specific predictions (Alvarez Melis & Jaakkola, 2018).

Although the internal workings of a model may remain a black box to the user, the relationship between the input features and the model's predictions can provide some insight into its interpretability. Recently, several criteria have been proposed to assess the interpretability of supervised models (Alvarez Melis & Jaakkola, 2018; Yang et al., 2022), as outlined in the following paragraphs. We will also discuss how these criteria can be adapted for an unsupervised setting.

Diversity We expect a good interpretability model to identify different sets of variables as driving factors for explaining distinct clusters. This is measured by the *diversity* metric, which is calculated by finding the negative mean Jaccard similarity between cluster-level informative features across all pairs of clusters. In simple terms, given a set of indices $S_{c_i} \subset 1, ..., D$ that indicate the selected informative variables of cluster c_i, where $i = 1, ..., K$, the *diversity* is defined as $1 - \sum_{i\neq j} \frac{J(S_{c_i}, S_{c_j})}{K\cdot(K-1)/2}$. Here, J is the Jaccard similarity between two sets. A score of 1 is obtained when there is no overlap between cluster-level features, indicating perfect diversity.

Faithfulness An interpretation is faithful if it accurately represents the reasoning behind the model's prediction function. To evaluate this quantity, *faithfulness* measures the correlation between the predictivity of the model and the feature importance. To calculate this, we first compute a feature's importance value, for instance, the value of our corresponding predicted gate averaged over all samples. We then sort the vector of importance values, remove features individually starting from the most important one, and measure the clustering model's performance. If the model's performance decreases in a monotonic manner with the importance of the removed features, we will get a high correlation, indicating that the model's prediction is faithful to the learned feature importance values. An example of this metric is presented in Fig. 3.

Uniqueness Since we are interested in sample-level interpretation, we extend the *diversity* to a metric that compares samples instead of clusters. Specifically, we propose to measure the *uniqueness* of the selected features for similar samples, or in other words, how granular our explanations are. We define *uniqueness* of an explanation $\boldsymbol{z}_i$ by: $\frac{1}{|I_i|}\sum_{\ell \in I_i} \min \frac{\|\boldsymbol{z}_i - \boldsymbol{z}_\ell\|_2}{\|\boldsymbol{x}_i - \boldsymbol{x}_\ell\|_2}$, where I_i is the set of indices of the r nearest neighbors of $\boldsymbol{x}_i$, and $\boldsymbol{z}_i, \boldsymbol{z}_\ell$ are feature weights for samples $\boldsymbol{x}_i, \boldsymbol{x}_\ell$. The smaller this value is, the less sample-specific the interpretation of the model is. Therefore, we want our model to obtain high *uniqueness* values. In case the uniqueness is not a desired property and could raise confusion in the produces interpretations, the loss term $\mathcal{L}_{\text{gtcr}}$ in Eq. 3 should be removed.

Generalizability We want the interpretation of the prediction to generalize to other simple prediction models. Specifically, if the selected features also lead to high predictive capabilities across different models, this may indicate that the interpretability is not an artifact of a specific model architecture or training instance. To evaluate this quantity, termed *generalizability*, we follow (Yang et al., 2022) and measure the accuracy of SVM when applied to the data, masked by the most informative features as identified by each method.

6. Experiments

We performed six different types of experiments in our study. Firstly, we tested our method's clustering and local feature selection capabilities on a synthetic dataset. Then, we evaluated the interpretability of our model using MNIST. The main experiment focused on assessing our method's ability to cluster real-world tabular datasets, including those with small sample sizes, high-dimensional biomed datasets, and physics and text datasets. It's worth noting that the real-world datasets we used are still considered challenging by several clustering studies that have used tabular data (Shaham et al., 2022; Xu et al., 2023).

In the fourth experiment, we demonstrated that our model can also perform well on image data without needing domain-specific augmentations. The datasets we used for our experiments have been summarized in Table E in the Appendix. In the fifth experiment, we showed that our local gates help the network learn a prediction function that includes high-frequency components. Finally, we conducted an ablation study to demonstrate the importance of each component in our model.

6.1. Evaluation

Interpretability Quality We use the metrics detailed in Section 5 to compare the interpretability of our method to the popular SHAP feature importance detection method (Lundberg & Lee, 2017) implemented here [1]. In addition, we compare Gradient SHAP [2] and Integrated Gradients (Sundararajan et al., 2017), both trained on IDC clustering model predictions. Additionally, both Gradient SHAP and Integrated Gradients explainers are evaluated while being applied on the TELL (Peng et al., 2022) model predictions. For uniqueness, we set the number of neighbors to 2 ($r = 2$).

Clustering Accuracy We use three popular clustering evaluation metrics: Clustering Accuracy (ACC), The adjusted Rand index score (ARI), and Normalized Mutual Information (NMI).

6.2. Results

Synthetic Dataset We started by testing our model's performance with a synthetic dataset inspired by (Armanfard et al., 2015). The dataset includes three informative features, denoted as $X[j] \in [-1, 1], j = 1, .., 3$, in which the samples are generated using Gaussian blobs [3]. We added ten background features, resulting in a total of 13 features.

The samples are equally distributed between 4 clusters, with ~ 800 samples in each cluster. Given the first two dimensions $\{X[1], X[2]\}$, only 3 clusters are separable, and the same property holds for dimensions pair $\{X[1], X[3]\}$. A visualization of these leading features appears in Figure 2. A good interpretable clustering model should correctly identify the four clusters while selecting the informative features for each cluster. For a more detailed explanation of how the dataset was generated, please refer to Appendix H.

We compute clustering accuracy (ACC) and F1-score to

[1] https://github.com/slundberg/shap

[2] https://captum.ai/api/gradient_shap.html

[3] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_blobs.html

Table 1. Evaluating the interpretability quality of our model on the MNIST$_{10K}$ data. Our IDC model improved clustering accuracy compared to baselines that do not use semantic-preserving image augmentations. We focus on the $|\mathcal{S}| = 15$ most informative features as provided by each interpretation model. We compare the interpretability of our model, as predicted by the gates (IDC), to (i) the top features explained by SHAP trained based on a K-means model, (ii) Integrated Gradients, and (iii) Gradient SHAP applied as explainers to our model and TELL. Our local gating network is generalizable (88.5) by selecting faithful (0.96) and unique (0.69) features while providing comparable diversity values (94.8).

Method	ACC ↑	$\lvert\mathcal{S}\rvert$ ↓	Uniqueness ↑	Diversity ↑	Faithfulness ↑	Generalizability ↑
K-means + SHAP	53.34	15	0.12	**100.0**	0.79	29.1
TELL + IntegGrads	74.79	15	0.03	89.1	0.67	75.0
TELL + GradSHAP	74.79	15	0.15	92.5	0.63	78.9
IDC w/o gates + IntegGrads	82.32	15	0.02	95.8	0.78	80.3
IDC w/o gates + GradSHAP	82.32	15	0.08	**100.0**	0.86	59.9
IDC + IntegGrads	**83.45**	15	0.01	95.3	0.94	63.7
IDC + GradSHAP	**83.45**	15	0.02	97.0	0.93	66.0
IDC	**83.45**	15	**0.69**	94.8	**0.96**	**88.5**

assess the model's effectiveness, which measures the feature selection quality. Since we know which are the informative features for each sample, we can determine the precision and recall for gate-level feature selection. We compare our results with those of K-means with informative features identified by SHAP. Our method leads to 99.91% clustering accuracy, thus significantly outperforming K-means with 25.72%. Moreover, IDC yields an average F1-score of 88.95 (compared to 49.65 by K-means), demonstrating its ability to identify the correct informative features.

Interpretability Results on MNIST$_{10K}$ Our next interpretability evaluation involves MNIST$_{10K}$ images, with $1K$ images for each category. We train our model with and without (w/o) gates and evaluate its clustering accuracy. These are compared to TELL (Peng et al., 2022), an interpretable deep clustering model, and to K-means. As demonstrated in Table 1, our model with the gates leads to improved clustering capabilities. It is important to note that we do not use any domain-specific image augmentations and that existing schemes that do use will typically lead to higher clustering accuracies on this data.

In terms of interpretability, we compare the most informative features identified by each method based on the metrics described in Section 5. As indicated in Table 1, IDC with our gates (bottom row) improves *faithfulness*, *uniqueness*, and *generalizability* while maintaining competitive *diversity*. In the left panel of Fig. 3, we plot the model's accuracy against the feature importance. Providing a more detailed view on the *faithfulness* of the interpretation. Finally, in the right panel of this figure, the top 15 features selected by each model. It can be seen that **IDC** selects more informative features that are local for each sample.

Real Tabular Data We conducted a benchmark of our method using 11 real tabular datasets. Most of these datasets are from different biological domains, such as Tox-171, ALLAML, PROSTATE, SRBCT, BIASE, INTESTINE, and PBMC-2. These datasets usually have more features than samples, making it challenging for clustering models to predict cluster assignments accurately. Additionally, we used image (MFEATZ), text (CNAE-9, ALBERT) and physical measurements (MiniMBooNE) datasets, all of which were treated as tables.

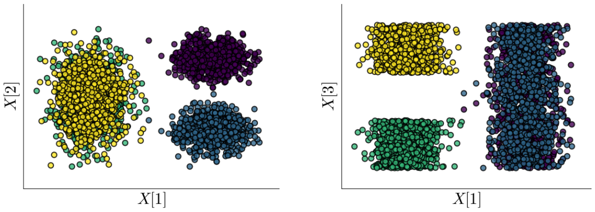

Figure 2. Visualization of the synthetic dataset. To separate between clusters, the model should select one of the pairs $\{X[1], X[2]\}$ or $\{X[1], X[3]\}$ of non-background features.

Since most datasets are tabular and high dimensional, we compare our clustering capabilities to several unsupervised feature selection (UFS) models, which have been demonstrated to lead to state-of-the-art clustering results on these datasets (Lindenbaum et al., 2021). Here, we follow the evaluation protocol of (Lindenbaum et al., 2021) when using the UFS methods. Table 2 presents clustering accuracy of different methods: K-means on the full set of features (KM), TELL an interpretable deep clustering model (Peng et al., 2022), variational autoencoder based k-DVAE (Caciularu & Goldberger, 2023) and different leading UFS models followed by K-means. These include SRCFS (Huang et al., 2019), Concrete Autoencoders (CAE) (Abid et al., 2019), DUFS (Lindenbaum et al., 2021), and our model IDC. This table shows that our model leads to the best or second-best clustering accuracy across all datasets. We run every model 10 times for each dataset and report the mean accuracy with standard deviation.

Extension to Image Domain We conducted usability tests on our model using popular image datasets. Initially, we

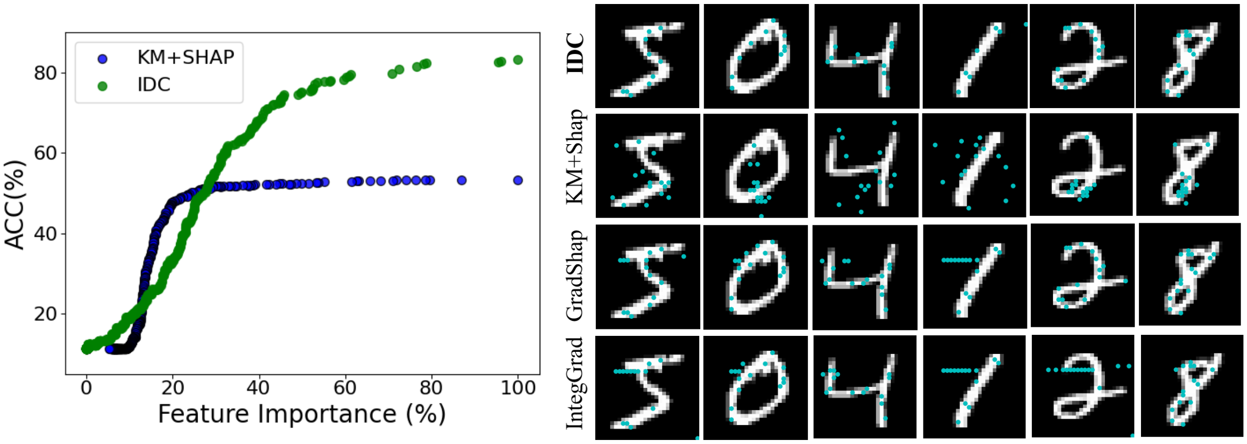

Figure 3. **Left**: Faithfulness plot of the proposed method (green) and K-Means+SHAP (blue) on MNIST$_{10K}$ subset. Accuracy drop and feature importance are well correlated for our approach 0.96 (see green dots) while less correlated for SHAP features with K-means clustering 0.79 (see blue dots). Furthermore, notice that K-means accuracy only reaches 53% while our method 83%. **Right**: Features selected by different interpretability model using MNIST$_{10K}$. The features are learned during clustering training as proposed by our approach (top), features selected by SHAP with K-means predictor (KM-SHAP), features obtained from Gradient-SHAP (GradSHAP), and features selected by Integrated Gradients explainer (bottom).

Table 2. Clustering Accuracy on Real Datasets. We compare the proposed method against leading unsupervised feature selection methods followed by K-means clustering, the interpretable deep clustering model TELL (Peng et al., 2022) and recently published k-DVAE (Caciularu & Goldberger, 2023). Our model produces superior results on 9 of 11 datasets and second-rank accuracy on two datasets.

Method/Data	TOX-171	ALLAML	PROSTATE	SRBCT	BIASE	INTESTINE	PBMC-2	CNAE-9	MFEATZ	MiniBooNE	ALBERT
KM	41.5 ± 2	67.3 ± 3	58.1 ± 0	39.6 ± 3	41.8 ± 8	54.8 ± 3	52.4 ± 0	44.9 ± 5	56.2 ± 0	71.6 ± 0	55.7 ± 0
LS	47.5 ± 1	73.2 ± 0	58.6 ± 0	41.1 ± 3	83.8 ± 0	43.2 ± 3	57.2 ± 0	35.9 ± 3	56.6 ± 3	71.9 ± 0	64.2 ± 0
MCFS	42.5 ± 3	72.9 ± 2	57.3 ± 0	43.7 ± 3	95.5 ± 3	48.2 ± 4	60.6 ± 12	43.4 ± 9	39.4 ± 9	71.6 ± 0	60.9 ± 0
SRCFS	45.8 ± 6	67.7 ± 6	60.6 ± 2	33.5 ± 5	50.8 ± 5	58.1 ± 10	58.5 ± 0	34.3 ± 3	58.5 ± 1	71.6 ± 0	56.8 ± 0
CAE	47.7 ± 1	73.5 ± 0	56.9 ± 0	$\mathbf{62.6} \pm 7$	85.1 ± 2	51.9 ± 3	59.1 ± 6	45.3 ± 2	70.0 ± 0	71.6 ± 0	64.1 ± 0
DUFS	49.1 ± 3	74.5 ± 1	64.7 ± 0	51.7 ± 1	$\mathbf{100} \pm 0$	71.9 ± 7	57.6 ± 9	46.3 ± 0	57.3 ± 9	71.6 ± 0	62.3 ± 0
TELL	28.7 ± 3	66.7 ± 14	63.6 ± 0	38.9 ± 8	84.6 ± 2	52.1 ± 28	52.1 ± 1	11.1 ± 0	63.8 ± 36	75.6 ± 5	60.6 ± 3
k-DVAE	49.1 ± 5	72.6 ± 5	59.5 ± 2	49.4 ± 12	39.3 ± 6	55.7 ± 7	50.1 ± 0	62.8 ± 7	66.5 ± 5	71.7 ± 0	51.5 ± 1
IDC	$\mathbf{50.6} \pm 3$	$\mathbf{77.5} \pm 6$	$\mathbf{65.3} \pm 3$	55.4 ± 5	95.7 ± 1	$\mathbf{74.2} \pm 2$	$\mathbf{65.1} \pm 5$	$\mathbf{66.0} \pm 7$	$\mathbf{86.81} \pm 4$	$\mathbf{77.0} \pm 4$	$\mathbf{64.2} \pm 4$

assessed the performance of IDC both with and without the gates on MNIST and FashionMNIST. The results, as presented in Table 3, indicate that the gating network improves the model's ability to cluster data accurately and select only a few relevant features, thereby enhancing interpretability.

In our second experiment, we utilized CIFAR10 and compared our model to other deep clustering models that do not utilize image augmentations. These models include TELL (Peng et al., 2022), VaDE (Jiang et al., 2016), and DEC (Xie et al., 2016). The results of the experiment are presented in Table 4. Our model produced competitive results while only selecting approximately 20% of the input features. It is worth noting that while these results are not state-of-the-art in deep image clustering, they were achieved without using strong image augmentations like resizing, translation, or rotations. Such augmentations are incompatible with our gating procedure, as the informative features may vary from one augmentation to the next. Additionally, these augmentations do not apply to tabular data..

6.3. Sparsity and Inductive Frequency Bias

According to a study by (Beyazit et al., 2023), tabular datasets generally require higher frequency target functions than images. However, several authors have shown that neural networks tend to learn low-frequency functions faster than higher ones. This might explain why tree-based models often outperform neural networks. For more information, please refer to the appendix section on inductive learning (Section I).

In this section, we analyze whether our local gates, which sparsify the weights of our learned prediction function, can provide an inductive bias that helps the network learn high-frequency functions. To test this, we trained our model on the tabular ALLAML data and evaluated the absolute value of Fourier amplitudes of predictions of our model $|\text{NUDFT}(f_{\text{IDC}})|$ at different frequencies $|k|$. In Figure 4, we compare these amplitudes to those learned without the gates (IDC$_{\text{w/o\_gates}}$) and to those learned by the interpretable

Table 3. Clustering evaluation and the number of features selected (last column). IDC with the proposed gates improves clustering accuracy by using only $\sim$ 16 from MNIST$_{60K}$ and $\sim$ 69 features from FashionMNIST datasets.

Dataset	Method	ACC ↑	ARI ↑	NMI ↑	$\lvert\mathcal{S}\rvert$ ↓
MNIST$_{60K}$	IDC (w/o gates)	81.1	75.9	80.3	784
	IDC	**87.9**	**82.8**	**85.1**	**15.81**
FashionMNIST	IDC (w/o gates)	61.0	**49.3**	62.7	784
	IDC	**61.9**	49.1	**63.3**	**68.6**

Table 4. Clustering performance on CIFAR10 dataset. IDC model selects 586 features (on average) out of 3,072.

Model	ACC ↑	ARI ↑	NMI ↑
TELL	**25.65**	5.96	10.41
VaDE	20.87	3.95	7.20
DEC	18.09	2.47	4.56
IDC	25.01	**6.16**	**11.96**

Table 5. Ablation study on MNIST$_{60K}$ dataset

Model	ACC ↑	ARI ↑	NMI ↑
IDC	**87.9**	**82.8**	**85.1**
IDC w/o $\mathcal{L}_{\text{reg}}$	85.9 (-2.0)	81.2(-1.6)	84.7 (-0.4)
IDC w/o latent denoising	86.5 (-1.4)	80.9 (-1.9)	83.2 (-1.9)
IDC w/o input denoising	84.3 (-3.6)	80.0 (-2.8)	83.9 (-1.2)
IDC features + K-Means	65.5 (-22.4)	49.3 (-33.5)	57.6 (-27.5)
IDC w/o $\mathcal{L}_{\text{recon}}$	18.0 (-69.9)	2.6 (-80.2)	4.3 (-80.8)

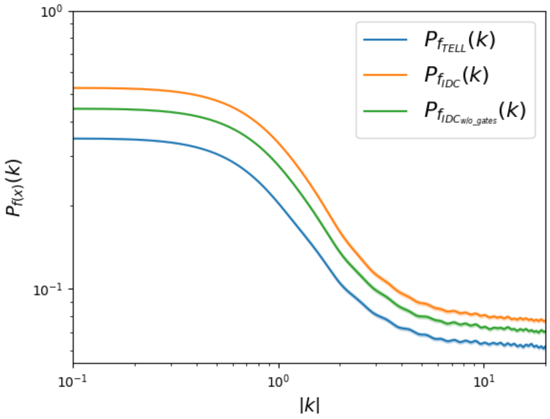

Figure 4. Spectral properties of the learned predictive function using ALLAML dataset. The model trained with the gating network (IDC) has higher Fourier amplitudes at all frequency levels than without gates (IDC$_{w/o\_gates}$) the baseline (TELL). This suggests that IDC can better handle the inductive bias of tabular data.

deep clustering model TELL (Peng et al., 2022). This figure demonstrates that the sparsification helps the model learn high-frequency components. The bias induced by the gating network makes our model more accurate on tabular data while being interpretable.

6.4. Ablation Study

We conducted an ablation study to determine if all the components of the loss are necessary for the model to converge. The evaluation was carried out on the MNIST$_{60K}$ dataset, and we ran each experiment 10 times. The results are presented in Table 4. We train our model without regularization term in Eq.2 (row 2), without latent or input augmentations (rows 3-4), then we replace the clustering head with K-Means (row 5) and finally train without reconstruction term $\mathcal{L}_{\text{recon}}$. Our findings indicate that all the proposed components significantly contribute to the performance of the model.

7. Conclusions

We present a deep clustering model that accurately assigns clusters on tabular data and predicts informative features at both the sample and cluster levels, providing interpretability. We tested our model on 15 datasets including synthetic datasets, high-dimensional tabular datasets, and several image datasets (treated as tables). A main limitation of our scheme is dealing with correlated variables. This is a known caveat of the reconstruction loss (Abid et al., 2019). One way to alleviate this limitation is to incorporate a group sparsity loss as presented by (Imrie et al., 2022). Our method also struggles to handle datasets with a large number of clusters. This problem could be mitigated using weak supervision or by introducing several clustering heads to prevent collapse to a subset of clusters. We hope our work will be beneficial to scientists in the biomedical field.

Impact Statement

The proposed framework for deep learning clustering in general domain tabular data has wide applications in scientific research, healthcare, and biomedicine. It addresses a critical need in data analysis across diverse domains by delivering reliable and interpretable cluster assignments. Biologists can use the framework to analyze genome sequences, medical records, and images with precision. Our hope is that it will increase trust in deep learning-based clustering models for multidisciplinary datasets, such as biological, text, image, and physics tabular datasets. Nonetheless, the framework can suffer from fairness biases at a societal level since it is fully unsupervised and could reflect data biases. We note that this concern is not different from any other unsupervised model, and we leave this as an open issue for future work.

Acknowledgements

We thank the anonymous reviewers for constructive comments during the reviewing process. In addition, we would like to express our deepest gratitude to Uri Shaham (Bar Ilan University) for insightful and constructive discussions significantly enriched this research.

References

GitHub - zalandoresearch/fashion-mnist: A MNIST-like fashion product database. Benchmark — github.com. https://github.com/zalandoresearch/fashion-mnist.

Datasets | Feature Selection @ ASU — jundongl.github.io. https://jundongl.github.io/scikit-feature/datasets.html.

MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges — yann.lecun.com. http://yann.lecun.com/exdb/mnist/.

Abid, A., Balin, M. F., and Zou, J. Concrete autoencoders for differentiable feature selection and reconstruction. *arXiv preprint arXiv:1901.09346*, 2019.

Alvarez Melis, D. and Jaakkola, T. Towards robust interpretability with self-explaining neural networks. *Advances in neural information processing systems*, 31, 2018.

Armanfard, N., Reilly, J. P., and Komeili, M. Local feature selection for data classification. *IEEE transactions on pattern analysis and machine intelligence*, 38(6):1217–1227, 2015.

Armingol, E., Officer, A., Harismendy, O., and Lewis, N. E. Deciphering cell–cell interactions and communication from gene expression. *Nature Reviews Genetics*, 22(2): 71–88, 2021.

Balın, M. F., Abid, A., and Zou, J. Concrete autoencoders: Differentiable feature selection and reconstruction. In *International conference on machine learning*, pp. 444–453. PMLR, 2019.

Basri, R., Galun, M., Geifman, A., Jacobs, D., Kasten, Y., and Kritchman, S. Frequency bias in neural networks for input of non-uniform density. In *International Conference on Machine Learning*, pp. 685–694. PMLR, 2020.

Bertsimas, D., Orfanoudaki, A., and Wiberg, H. Interpretable clustering: an optimization approach. *Machine Learning*, 110:89–138, 2021.

Beyazit, E., Kozaczuk, J., Li, B., Wallace, V., and Fadlallah, B. H. An inductive bias for tabular deep learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Biase, F. H., Cao, X., and Zhong, S. Cell fate inclination within 2-cell and 4-cell mouse embryos revealed by single-cell rna sequencing. *Genome research*, 24(11): 1787–1796, 2014.

Bischl, B., Casalicchio, G., Feurer, M., Gijsbers, P., Hutter, F., Lang, M., Mantovani, R. G., van Rijn, J. N., and Vanschoren, J. Openml benchmarking suites. *arXiv preprint arXiv:1708.03731*, 2017.

Caciularu, A. and Goldberger, J. An entangled mixture of variational autoencoders approach to deep clustering. *Neurocomputing*, 2023.

Cai, J., Fan, J., Guo, W., Wang, S., Zhang, Y., and Zhang, Z. Efficient deep embedded subspace clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1–10, 2022.

Ciarelli, P. M. and Oliveira, E. Agglomeration and elimination of terms for dimensionality reduction. In *2009 Ninth International Conference on Intelligent Systems Design and Applications*, pp. 547–552. IEEE, 2009.

Cohen, E. Interpretable clustering via soft clustering trees. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pp. 281–298. Springer, 2023.

Deprez, M., Zaragosi, L.-E., Truchi, M., Becavin, C., Ruiz García, S., Arguel, M.-J., Plaisant, M., Magnone, V., Lebrigand, K., Abelanet, S., et al. A single-cell atlas of the human healthy airways. *American journal of respiratory and critical care medicine*, 202(12):1636–1645, 2020.

Doi, E., Balcan, D. C., and Lewicki, M. S. Robust coding over noisy overcomplete channels. *IEEE Transactions on Image Processing*, 16(2):442–452, 2007.

Frost, N., Moshkovitz, M., and Rashtchian, C. Exkmc: Expanding explainable k-means clustering. *arXiv preprint arXiv:2006.02399*, 2020.

Gabidolla, M. and Carreira-Perpiñán, M. Á. Optimal interpretable clustering using oblique decision trees. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 400–410, 2022.

Gao, B., Yang, Y., Gouk, H., and Hospedales, T. M. Deep clusteringwith concrete k-means. In *Icassp 2020-2020 ieee international conference on acoustics, speech and signal processing (icassp)*, pp. 4252–4256. IEEE, 2020.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., et al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, 286(5439):531–537, 1999.

Guan, Y., Jordan, M. I., and Dy, J. G. A unified probabilistic model for global and local unsupervised feature selection.

In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 1073–1080, 2011.

Guyon, I., Sun-Hosoya, L., Boullé, M., Escalante, H. J., Escalera, S., Liu, Z., Jajetic, D., Ray, B., Saeed, M., Sebag, M., Statnikov, A., Tu, W., and Viegas, E. Analysis of the automl challenge series 2015-2018. In *AutoML*, Springer series on Challenges in Machine Learning, 2019. URL https://www.automl.org/wp-content/uploads/2018/09/chapter10-challenge.pdf.

Han, K., Wang, Y., Zhang, C., Li, C., and Xu, C. Autoencoder inspired unsupervised feature selection. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 2941–2945. IEEE, 2018.

He, X., Cai, D., and Niyogi, P. Laplacian score for feature selection. *Advances in neural information processing systems*, 18, 2005.

Huang, D., Cai, X., and Wang, C.-D. Unsupervised feature selection with multi-subspace randomization and collaboration. *Knowledge-Based Systems*, 182:104856, 2019.

Imrie, F., Norcliffe, A., Liò, P., and van der Schaar, M. Composite feature selection using deep ensembles. *Advances in Neural Information Processing Systems*, 35: 36142–36160, 2022.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Jiang, Z., Zheng, Y., Tan, H., Tang, B., and Zhou, H. Variational deep embedding: An unsupervised and generative approach to clustering. *arXiv preprint arXiv:1611.05148*, 2016.

Kar, S., Sharma, K. D., and Maitra, M. Gene selection from microarray gene expression data for classification of cancer subgroups employing pso and adaptive k-nearest neighborhood technique. *Expert Systems with Applications*, 42(1):612–627, 2015.

Kauffmann, J., Esders, M., Ruff, L., Montavon, G., Samek, W., and Müller, K.-R. From clustering to cluster explanations via neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Khan, J., Wei, J. S., Ringner, M., Saal, L. H., Ladanyi, M., Westermann, F., Berthold, F., Schwab, M., Antonescu, C. R., Peterson, C., et al. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature medicine*, 7(6):673–679, 2001.

Kiselev, V. Y., Kirschner, K., Schaub, M. T., Andrews, T., Yiu, A., Chandra, T., Natarajan, K. N., Reik, W., Barahona, M., Green, A. R., et al. Sc3: consensus clustering of single-cell rna-seq data. *Nature methods*, 14(5):483–486, 2017.

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

Lawless, C., Kalagnanam, J., Nguyen, L. M., Phan, D., and Reddy, C. Interpretable clustering via multi-polytope machines. In *Proceedings of the aaai conference on artificial intelligence*, volume 36, pp. 7309–7316, 2022.

Lee, C., Imrie, F., and van der Schaar, M. Self-supervision enhanced feature selection with correlated gates. In *International Conference on Learning Representations*, 2022.

Li, Z., Chen, Y., LeCun, Y., and Sommer, F. T. Neural manifold clustering and embedding. *arXiv preprint arXiv:2201.10000*, 2022.

Lindenbaum, O., Shaham, U., Peterfreund, E., Svirsky, J., Casey, N., and Kluger, Y. Differentiable unsupervised feature selection based on a gated laplacian. *Advances in Neural Information Processing Systems*, 34:1530–1542, 2021.

Lundberg, S. M. and Lee, S.-I. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

Lv, J., Kang, Z., Lu, X., and Xu, Z. Pseudo-supervised deep subspace clustering. *IEEE Transactions on Image Processing*, 30:5252–5263, 2021.

Mikuni, V. and Canelli, F. Unsupervised clustering for collider physics. *Physical Review D*, 103(9):092007, 2021.

Nie, F., Huang, H., Cai, X., and Ding, C. Efficient and robust feature selection via joint l2, 1-norms minimization. *Advances in neural information processing systems*, 23, 2010.

Niu, C., Shan, H., and Wang, G. Spice: Semantic pseudo-labeling for image clustering. *arXiv preprint arXiv:2103.09382*, 2021.

Peng, X., Li, Y., Tsang, I. W., Zhu, H., Lv, J., and Zhou, J. T. Xai beyond classification: Interpretable neural clustering. *The Journal of Machine Learning Research*, 23(1):227–254, 2022.

Piloto, S. and Schilling, T. F. Ovo1 links wnt signaling with n-cadherin localization during neural crest migration. *Development*, 137(12):1981–1990, 2010.

Qian, Z., Cebere, B.-C., and van der Schaar, M. Synthcity: facilitating innovative use cases of synthetic data in different data modalities. *arXiv preprint arXiv:2301.07573*, 2023.

Reddy, C. K., Al Hasan, M., and Zaki, M. J. Clustering biological data. *Data Clustering*, pp. 381–414, 2018.

Roe, B. P., Yang, H.-J., Zhu, J., Liu, Y., Stancu, I., and McGregor, G. Boosted decision trees as an alternative to artificial neural networks for particle identification. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 543(2-3):577–584, 2005.

Sato, T., Vries, R. G., Snippert, H. J., Van De Wetering, M., Barker, N., Stange, D. E., Van Es, J. H., Abo, A., Kujala, P., Peters, P. J., et al. Single lgr5 stem cells build crypt-villus structures in vitro without a mesenchymal niche. *Nature*, 459(7244):262–265, 2009.

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

Shaham, U., Lindenbaum, O., Svirsky, J., and Kluger, Y. Deep unsupervised feature selection by discarding nuisance and correlated features. *Neural Networks*, 152: 34–43, 2022.

Shen, Y., Shen, Z., Wang, M., Qin, J., Torr, P., and Shao, L. You never cluster alone. *Advances in Neural Information Processing Systems*, 34:27734–27746, 2021.

Singh, D., Febbo, P. G., Ross, K., Jackson, D. G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A. A., D'Amico, A. V., Richie, J. P., et al. Gene expression correlates of clinical prostate cancer behavior. *Cancer cell*, 1(2):203–209, 2002.

Sokar, G., Atashgahi, Z., Pechenizkiy, M., and Mocanu, D. C. Where to pay attention in sparse training for feature selection? *arXiv preprint arXiv:2211.14627*, 2022.

Song, C., Liu, F., Huang, Y., Wang, L., and Tan, T. Autoencoder based data clustering. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications: 18th Iberoamerican Congress, CIARP 2013, Havana, Cuba, November 20-23, 2013, Proceedings, Part I 18*, pp. 117–124. Springer, 2013.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. In *International conference on machine learning*, pp. 3319–3328. PMLR, 2017.

van Breukelen, M., Duin, R. P., Tax, D. M., and Den Hartog, J. Handwritten digit recognition by combined classifiers. *Kybernetika*, 34(4):381–386, 1998.

Varghese, B. M., Unnikrishnan, A., Sciencist, G., Kochi, N., and Kochi, C. Clustering student data to characterize performance patterns. *Int. J. Adv. Comput. Sci. Appl*, 2: 138–140, 2010.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103, 2008.

Von Luxburg, U. A tutorial on spectral clustering. *Statistics and computing*, 17:395–416, 2007.

Wang, D. and Bodovitz, S. Single cell analysis: the new frontier in 'omics'. *Trends in biotechnology*, 28(6):281–290, 2010.

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

Xu, L., Wang, R., Nie, F., and Li, X. Efficient top-k feature selection using coordinate descent method. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 10594–10601, 2023.

Yamada, Y., Lindenbaum, O., Negahban, S., and Kluger, Y. Feature selection using stochastic gates. In *International Conference on Machine Learning*, pp. 10648–10659. PMLR, 2020.

Yang, J., Lindenbaum, O., and Kluger, Y. Locally sparse neural networks for tabular biomedical data. In *International Conference on Machine Learning*, pp. 25123–25153. PMLR, 2022.

Yang, P., Huang, H., and Liu, C. Feature selection revisited in the single-cell era. *Genome Biology*, 22:1–17, 2021.

Yoon, J., Jordon, J., and van der Schaar, M. Invase: Instance-wise variable selection using neural networks. In *International Conference on Learning Representations*, 2019.

Yu, Y., Chan, K. H. R., You, C., Song, C., and Ma, Y. Learning diverse and discriminative representations via the principle of maximal coding rate reduction. *Advances in Neural Information Processing Systems*, 33:9422–9434, 2020.

Zhao, Z. A. and Liu, H. *Spectral feature selection for data mining*. Taylor & Francis, 2012.

Zheng, G. X., Terry, J. M., Belgrader, P., Ryvkin, P., Bent, Z. W., Wilson, R., Ziraldo, S. B., Wheeler, T. D., McDermott, G. P., Zhu, J., et al. Massively parallel digital transcriptional profiling of single cells. *Nature communications*, 8(1):14049, 2017.

A. Implementation details

We implement our model in Pytorch and run experiments on Nvidia A100 GPU server with Intel(R) Xeon(R) Gold 6338 CPU @ 2.00GHz.

B. Model Architecture

The model is trained with a single hidden layer in the Clustering Head and the Gating NN. We use up to 4 hidden layers for the Encoder, and The Decoder is a mirrored version of the encoder. The dimensions of hidden layers are detailed in the supplementary included code and will be released to GitHub.

C. Training Setup

We train all models with a two-stage approach - we train Encoder, Decoder, and Gating NN in the first stage and then train Clustering Head in the second stage. For interpretabiltiy experiments, we train K-means [4] and TELL (Peng et al., 2022) [5]. Both methods are trained without additional augmentations for fair comparison to our method with the provided default experimental parameters. For small-sample datasets we increase the number of epochs in TELL model up to 10K for fair comparison. In method k-DVAE [6] (Caciularu & Goldberger, 2023) is trained with default parameters provided in the shared code with only incresed number of epochs to 200 (initial step) and to 100 (other steps) for small-sample datasets.

Table 6. The number of epochs and batch size for different datasets.

Dataset	Epochs Stage 1	Epochs Stage 2	Batch size
Synthetic	50	2000	800
MNIST$_{60K}$	300	600	256
MNIST$_{10K}$	300	700	100
FashionMNIST	100	500	256
TOX-171	1000	1000	16
ALLAML	1000	1000	36
PROSTATE	1000	1000	102
SRBCT	2000	1000	83
BIASE	10000	1000	56
INTESTINE	5000	1000	238
PBMC-2	100	100	256
CNAE-9	1000	1000	500
MFEATZ	1000	1000	500
MiniBooNE	20	30	512
ALBERT	10	40	1024
CIFAR-10	600	700	256

D. Regularization Term

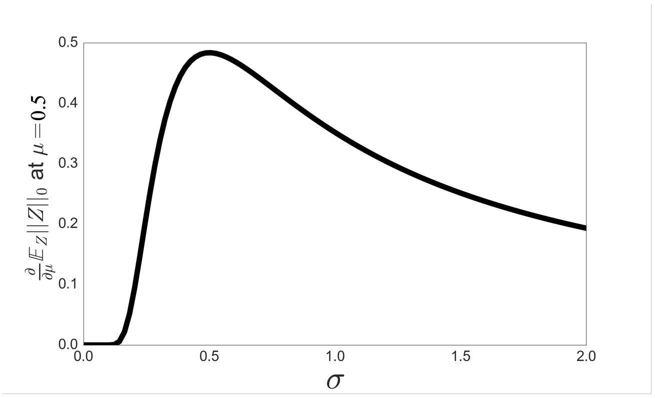

Figure 5. The value of $\frac{\partial}{\partial\mu}\mathbb{E}_Z||\boldsymbol{Z}||_0|_{\mu=0.5} = \frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{1}{8\sigma^2}}$ for $\sigma = [0.001, 2]$.

The leading term in our regularizer is expressed by :

[4] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

[5] The implementation was found here: https://github.com/XLearning-SCU/2022-JMLR-TELL/tree/main, accessed on 2023-09-28.

[6] https://github.com/aviclu/k-DVAE

$$\begin{aligned}
\mathbb{E}_Z||\boldsymbol{Z}||_0 &= \sum_{d=1}^{D}\mathbb{P}[z_d > 0] = \sum_{d=1}^{D}\mathbb{P}[\mu_d + \sigma\epsilon_d + 0.5 > 0] \\
&= \sum_{d=1}^{D}\{1 - \mathbb{P}[\mu_d + \sigma\epsilon_d + 0.5 \leq 0]\} \\
&= \sum_{d=1}^{D}\{1 - \Phi(\frac{-\mu_d - 0.5}{\sigma})\} \\
&= \sum_{d=1}^{D}\Phi\left(\frac{\mu_d + 0.5}{\sigma}\right) \\
&= \sum_{d=1}^{D}\left(\frac{1}{2} - \frac{1}{2}\operatorname{erf}\left(-\frac{\mu_d + 0.5}{\sqrt{2}\sigma}\right)\right)
\end{aligned}$$

To tune σ, we follow the suggestion in (Yamada et al., 2020). Specifically, the effect of σ can be understood by looking at the value of $\frac{\partial}{\partial\mu_d}\mathbb{E}_Z||\boldsymbol{Z}||_0$. In the first training step, μ_d is 0. Therefore, at initial training phase, $\frac{\partial}{\partial\mu_d}\mathbb{E}_Z||\boldsymbol{Z}||_0$ is close to $\frac{1}{\sqrt{2\pi\sigma_d^2}}e^{-\frac{1}{8\sigma_d^2}}$. To enable sparsification, this term (multiplied by the regularization parameter λ) has to be greater than the derivative of the loss with respect to μ_d because otherwise μ_d is updated in the incorrect direction. To encourage such behavior, we tune σ to the value that maximizes the gradient of the regularization term. As demonstrated in Fig. 5 this is obtained when $\sigma = 0.5$. Therefore, we keep $\sigma = 0.5$ throughout our experiments unless specifically noted.

E. Datasets Properties and References

In Table E we add the references of the datasets used in the experiments. We provide here a short description for each dataset:

- **MNIST$_{10K}$ and MNIST$_{60K}$** are the subsets of MNIST (lec) dataset, the first one of size 10,000 samples and the second one is the full trainset. Additionally, we use the MNIST test set split for evaluations of the model on unseen data. The images include hand-written digits from 10 categories.
- **FashionMNIST$_{60K}$** is a train set of (git). The images include 10 categories of clothers.
- **TOX-171** (Piloto & Schilling, 2010) dataset is an example of the use of toxicology to integrate diverse biological data, such as clinical chemistry, expression, and other types of data. The database contains the profiles resulting from the three toxicants: alpha-naphthyl-isothiocyanate, dimethylnitrosamine, and N-methylformamide administered to rats. The classification task is to identify whether the samples are toxic, non toxic or control.
- **ALLAML** dataset (Golub et al., 1999) consists of gene expression profiles of two acute cases of leukemia: acute lymphoblastic leukemia (ALL) and acute myeloblastic leukemia (AML). The ALL part of the dataset comes from two types, B-cell and T-cell, while the AML part is split into two types, bone marrow samples and peripheral blood samples. It contains in total 72 samples in 2 classes, ALL and AML, which have 47 and 25 samples, respectively. Every sample contains 7,129 gene expression values.
- **PROSTATE** dataset (Singh et al., 2002) has in total 102 samples in two classes tumor and normal, which have 52 and 50 samples, respectively. After preprocessing described in (Nie et al., 2010), a data set contains 102 samples and 5966 genes.
- **SRBCT** dataset (Kar et al., 2015) consists of four classes of cancers in 83 samples. These four classes were ewing sarcoma (EWS), non-Hodgkin lymphoma (NHL), neuroblastoma (NB), and rhabdomyosarcoma (RMS).
- **BIASE** dataset (Biase et al., 2014) is bimodal mRNA expressions to embryonic genome activation and it contains 56 samples each one of dimension 25,683.
- INTESTINE (Sato et al., 2009) is a mouse intestine dataset with 238 samples.
- **PBMC-2** dataset is a binary-class subset of the original PBMC (Zheng et al., 2017) dataset. We select two categories that have the most number of samples in the original set. In addition, we remove all zero columns from the data resulting in 17,126 featurees × 20,742 samples size.
- **CIFAR10** dataset (Krizhevsky et al., 2009) contains 60,000 small images of size 32 × 32 with 3 color channels. In total there are 3,072 features.

- **cnae-9** [7] (Bischl et al., 2017; Ciarelli & Oliveira, 2009) contains 1080 documents of free text business descriptions of Brazilian companies categorized into a subset of 9 categories. The original texts were preprocessed to obtain the current data set: initially, it was kept only letters and then it was removed prepositions of the texts. Next, the words were transformed to their canonical form. Finally, each document was represented as a vector, where the weight of each word is its frequency in the document. This data set is highly sparse (99.22% of the matrix is filled with zeros).
- **MFEATZ** is known as mfeat-zernike [8] (van Breukelen et al., 1998; Bischl et al., 2017) dataset describes features of handwritten numerals (0 - 9) extracted from a collection of Dutch utility maps. Corresponding patterns in different datasets correspond to the same original character. 200 instances per class (for a total of 2,000 instances) have been digitized in binary images. These digits are represented in terms of 47 Zernike moments.
- **MiniBooNE** is a physical dataset for particle identification task (Roe et al., 2005).
- **ALBERT** is a text dataset from AutoML challenge (Guyon et al., 2019). Since the dataset comes with mnostly categorical variables, we follow the work (Beyazit et al., 2023) and preprocess the dataset with target-encoder[9].

Table 7. Properties and references for the dataset used in the experiments.

Dataset	Features	Samples	Clusters	Reference
MNIST$_{10K}$	784	10,000	10	(lec)
MNIST$_{60K}$	784	60,000	10	(lec)
FashionMNIST$_{60K}$	784	60,000	10	(git)
TOX-171	5,748	171	4	(jun)
ALLAML	7,192	72	2	(jun)
PROSTATE	5,966	102	2	(jun)
SRBCT	2,308	83	4	(Khan et al., 2001)
BIASE	25,683	56	4	(Biase et al., 2014)
INTESTINE	3,775	238	13	(Sato et al., 2009)
PBMC-2	17,126	20,742	2	(Zheng et al., 2017)
CIFAR10	3,072	60,000	10	(Krizhevsky et al., 2009)
CNAE-9	857	1080	9	(Ciarelli & Oliveira, 2009; Bischl et al., 2017)
MFEATZ	48	2000	2	(van Breukelen et al., 1998; Bischl et al., 2017)
MiniBooNE	50	130064	2	(Roe et al., 2005; Bischl et al., 2017)
ALBERT	78	425240	2	(Guyon et al., 2019; Bischl et al., 2017)

F. Train Loss Augmentations

In addition to the loss presented in Section 4.2 we exploit the next dataset-agnostic augmentations during model training. The first one is the standard reconstruction loss that is calculated between input samples and reconstructed samples. Input denoising is based on (Vincent et al., 2008) and latent denoising on (Doi et al., 2007):

- Clean reconstruction loss, $||f_{\theta_D} \circ f_{\theta_E}(\boldsymbol{x}_i) - \boldsymbol{x}_i||_1$, which measures the deviation of estimated $\hat{\boldsymbol{x}}_i$ from the input sample $\boldsymbol{x}_i$.
- Denoising reconstruction loss (Vincent et al., 2008), $||f_{\theta_D} \circ f_{\theta_E}(\boldsymbol{x}_i \odot m_{rand}) - \boldsymbol{x}_i||_1$, where $m_{rand} \in \{0, 1\}^D$ is a random binary mask generated for each sample $\boldsymbol{x}_i$. We generate a mask such that about $m_{rand}\%$ of the input features are multiplied by zero value, which indicates that the gate is closed. The loss pushes the method to pay less attention to unnecessary features for the reconstruction.
- Latent denoising reconstruction loss, $||f_{\theta_D}(\boldsymbol{h}_i \odot \boldsymbol{h}_{noise}) - \boldsymbol{x}_i||_1$, where $h_{noise} \sim \mathcal{N}(1, \sigma_h)$ is a noise generated from a normal distribution with mean one and scale σ_h which is a dataset-specific hyperparameter (Doi et al., 2007). This term aims to improve latent embedding representation by small perturbation augmentation to treat small sample-size datasets.

G. Model Training Scalability

In Figure 9 we show the training time as a function of number of data samples. It could be seen that training time scales linearly with an increase in dataset length. In addition, in Table 8 we present the total training time (in seconds) measured for the baselines and our method. The experiments were done on a single A100 GPU. The measurements were done for different dataset (MNIST) sizes and averaged over 5 trials with different random seeds (700 epochs, batch size 128, time in seconds).

[7] https://www.openml.org/search?type=data&sort=runs&id=1468. Accessed on 2024-01-23

[8] https://www.openml.org/search?type=data&status=active&id=22. Accessed on 2024-01-23

[9] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.TargetEncoder.html

Table 8. Training time in seconds for different MNIST subset sizes.

MNIST subset	IDC	TELL	k-DVAE
100	181.4	1.8	487.1
1000	401.3	69.3	1647.9
10000	2016.2	735.5	9705.6

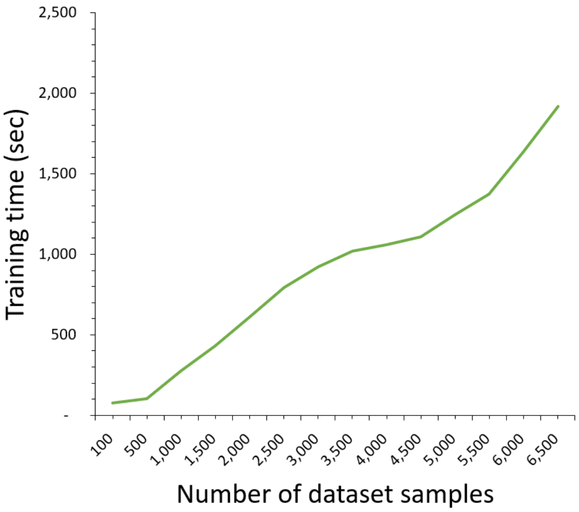

Table 9. Training time in seconds measured for different numbers of samples.

H. Synthetic Dataset Generation

The dataset consists of three informative features $x_i[j] \in [-1, 1], j = 1, .., 3$ for each sample $\boldsymbol{x}_i$ and is generated as isotropic Gaussian blobs [10] with standard deviation of each cluster of 0.5. The detailed description of the dataset generation could be found in H. Then we add ten nuisance background features with values drawn from $\mathcal{N}(0, \sigma_n^2)$ (with $\sigma_n = 0.1$) resulting in 13 total features. The samples are equally distributed between 4 clusters, with ~ 800 samples in each cluster. Given the first two dimensions $\{x[1], x[2]\}$, only 3 clusters are separable, and the same property holds for dimensions pair $\{x[1], x[3]\}$.

I. Indactive Bias Analysis

We present the original plot from the paper (Beyazit et al., 2023) that emphasizes the difference in target labels Fourier amplitudes distribution P_y across different frequencies. The authors claim that distributiuon of Foueirer amplitueds obtaned on ground truth targets on tabular datasets has higher values than those of image datasets. We support this claim by testing it from the learned function - does our model learns a bias for tabular domain by using gating network. To produce the anaylsis plot we predict $\hat{y}_i^j = argmax_{j\in[1,...,K]}(\hat{\boldsymbol{p}}_i)$ where $\hat{\boldsymbol{p}}_i = f_{\theta_C}(f_{\theta_E}(f_G(\boldsymbol{x}_i)))$ and is calculated for each sample $\boldsymbol{x}_i$ and for each feature value $x_i^d \in \boldsymbol{x}_i$. Then we calculate the non uniform discrete fourier transform (NUDFT) for 1000 frequencies values in range [0.1, 20] which accepts a vector of N values of x_i^d, $i = 1, ...N$ and corresponding binary prediction values y_i. NUDFT is calculated by the pytorch code:

```
def spectrum_NUDFT(x, y, kmax=20, nk=1000):
    kvals = np.linspace(0.1, kmax, nk+1)
    nufft = (1 / len(x)) * nfft_adjoint(-(x * kmax / nk), y, 2 * (nk + 1))[nk + 1:]
    return [kvals, np.array(nufft, dtype="complex_")]
```

where nfft_adjoint function is part of the **nfft** package [11]. Then we take the absolute value of NUDFT amplitudes. By repeating the process for each feature d we obtain D vectors with $|NUDFT|$ values and we plot them with 1000 frequency steps $|\mathbf{k}|$ in logarithmic scale axes. In this way we obtain the plot in Figure 4.

J. The number of selected features

Table 10. Selected and total features for various datasets

Dataset	TOX-171	ALLAML	PROSTATE	SRBCT	BIASE	INTESTINE	PBMC-2	CNAE-9	MFEATZ	ALBERT	MiniBoonE
Selected	49.6	527.6	170.9	46.7	210	65	137.8	360.5	26.2	59.4	29.4
Total	5748	7192	5966	2308	25683	3775	17126	856	47	78	50

[10] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_blobs.html

[11] https://github.com/jakevdp/nfft

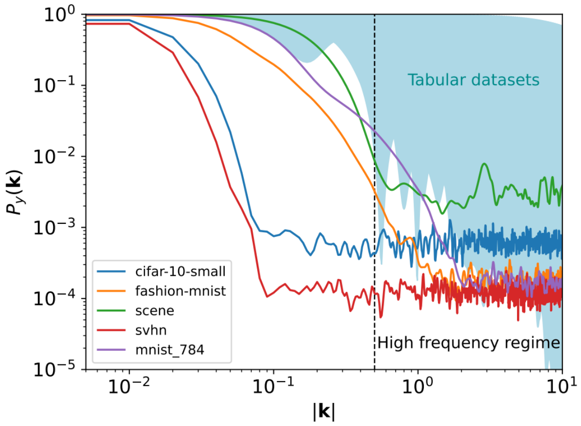

Figure 6. (Beyazit et al., 2023): Due to their heterogeneous nature, tabular datasets tend to describe higher frequency target functions compared to images. The spectra corresponding to image datasets (curves in color) tend to feature lower Fourier amplitudes at higher frequencies than hetergoneous tabular datasets (cyan region).

K. Sensitivity analysis of hyper parameters

We present the sensitivity analysis of the clustering as a function of hyperparameter changes. The analysis is done on MNIST subset (10K samples), we present the mean values obtained from 5 trials for each parameter change. Please, note that in same cases we get even better results than were presented in the main part of this paper:

Table 11. Sensitivity to ϵ_{gtcr} values

ϵ_{gtcr}	**ACC**	**ARI**	**NMI**	**# open gates**
100	0.8318	0.7533	0.7856	16
10	0.7732	0.6897	0.7386	17
1	0.834	0.7479	0.7807	18
0.1	0.8067	0.7251	0.7641	18
0.01	0.8081	0.7256	0.7667	18

Table 12. Sensitivity to σ_h values

σ_h	**ACC**	**ARI**	**NMI**	**# open gates**
0.01	0.8104	0.7249	0.7683	18
0.1	0.8183	0.7377	0.7736	18
0.2	**0.8531**	**0.7646**	**0.7841**	18
0.5	0.6158	0.4707	0.5661	18
1	0.7336	0.6115	0.6745	18

Table 13. Sensitivity to ϵ_{head} values

ϵ_{head}	**ACC**	**ARI**	**NMI**	**# open gates**
100	0.6645	0.5339	0.6294	18
10	0.6025	0.503	0.6206	18
1	0.7941	0.6842	0.7298	18
0.1	0.834	0.7479	0.7807	18
0.01	0.708	0.6123	0.7122	18

Table 14. Sensitivity to λ_{reg} values

λ_{reg}	**ACC**	**ARI**	**NMI**	**# open gates**
0	**0.8643**	**0.7803**	**0.8021**	435
0.1	0.8227	0.7471	0.7804	111
1	0.834	0.7479	0.7807	18
10	0.7795	0.6921	0.7421	4
100	0.1374	0.0028	0.053	0

Table 15. Sensitivity to m_{rand} values

m_{rand}	**ACC**	**ARI**	**NMI**	**# open gates**
0%	0.7969	0.7131	0.7585	21
10%	0.8381	0.7539	0.7864	20
30%	**0.8812**	**0.7915**	**0.8049**	19
50%	0.8334	0.7491	0.7844	19
70%	0.8188	0.7464	0.7864	18
90%	0.8104	0.7249	0.7683	18