TabDeco: A Comprehensive Contrastive Framework for Decoupled Representations in Tabular Data

Suiyao Chen
Buyer Abuse Prevention
suiyaoc@amazon.com

Jing Wu
WWPS Solution Architecture
jingwua@amazon.com

Yunxiao Wang, Cheng Ji, Tianpei Xie
Buyer Abuse Prevention
{yunxiaow, cjiamzn, lukexie}@amazon.com

Daniel Cociorva[1], **Michael Sharps**[2], **Cecile Levasseur**[1], **Hakan Brunzell**[1]
Buyer Abuse Prevention[1], WWPS Solution Architecture[2]
{cociorva, sharpsm, cecilele, brunzell}@amazon.com

Abstract

Representation learning is a fundamental aspect of modern artificial intelligence, driving substantial improvements across diverse applications. While self-supervised contrastive learning has led to significant advancements in fields like computer vision and natural language processing, its adaptation to tabular data presents unique challenges. Traditional approaches often prioritize optimizing model architecture and loss functions but may overlook the crucial task of constructing meaningful positive and negative sample pairs from various perspectives like feature interactions, instance-level patterns and batch-specific contexts. To address these challenges, we introduce TabDeco, a novel method that leverages attention-based encoding strategies across both rows and columns and employs contrastive learning framework to effectively disentangle feature representations at multiple levels, including features, instances and data batches. With the innovative feature decoupling hierarchies, TabDeco consistently surpasses existing deep learning methods and leading gradient boosting algorithms, including XGBoost, CatBoost, and LightGBM, across various benchmark tasks, underscoring its effectiveness in advancing tabular data representation learning.

1 Introduction

Self-supervised learning methods, particularly contrastive learning, have gained popularity in response to the challenges of acquiring labeled data. This approach often rivals or even surpasses supervised learning, especially in computer vision and natural language processing, by creating embeddings that distinctly separate similar and dissimilar data points. In visual tasks, techniques like image rotation or puzzle assembly enhance positive pair similarities, while token masking in text processing helps capture robust, invariant features. However, when applied to tabular data, contrastive learning encounters unique challenges in constructing meaningful positive and negative samples. Traditional methods frequently alter individual features or use entirely different samples, lacking a deeper exploration of feature interactions, instance-level patterns and batch-specific contexts, which could be essential for enhancing the learning process.

Table Representation Learning Workshop at NeurIPS 2024

To produce more structured representations for tabular data, one notable attempt is SwitchTab [34], which employs an asymmetric encoder-decoder framework to decouple shared and unique features within data pairs. This feature decoupling process could naturally construct meaningful positive and negative samples, thus facilitating the contrastive learning process. However, this decoupling process faces notable difficulties. The linear projector used in SwitchTab often struggles to effectively delineate feature boundaries, leading to embeddings that are poorly organized and difficult to interpret, ultimately limiting the robustness of the approach. Another recent innovation, Self-Attention and Intersample Attention Transformer (SAINT) [29], which has enhanced feature learning by integrating attention mechanisms at both column and row levels, however, may struggle with dataset complexity, especially when faced with high-dimensional, heterogeneous, and noisy data.

The complementary strengths and limitations of SwitchTab and SAINT present an opportunity to develop a more robust framework for tabular data representation learning, focusing on effective feature decoupling to generate meaningful positive and negative sample pairs for contrastive learning, as well as capturing the data complexities to ehance model performance. Therefore, we propose TabDeco, an innovative contrastive learning framework that utilizes attention mechanisms to achieve finer-grained decoupling of local and global features across multiple levels in tabular data. TabDeco constructs positive and negative pairs from multiple perspectives, including local and global contrasts, feature-level and instance-level contrasts, enabling a more refined separation of feature hierarchies. By integrating attention-based encoding and feature decoupling strategies, TabDeco enhances the model's capacity to isolate and emphasize relevant features and instances, surpassing the limitations of existing approaches and enhancing the representation learning performance.

Our contributions can be summarized as follows:

- We propose TabDeco, a novel contrastive learning framework for decoupled representation learning for tabular data. To the best of our knowledge, this is the first attempt to explore and explicitly facilitate structured embeddings learning through contrasting for tabular data.
- By integrating the strengths from feature decoupling and attention-based learning, we demonstrate out method to achieve competitive results across extensive datasets and benchmarks.
- We develop the comprehensive framework for contrastive learning on decoupled features and construct positive and negative pairs from diverse perspectives, enhancing the exploration of feature interactions at local, global, and instance levels.

2 Related Work

2.1 Classical vs Deep Learning Models

For tasks like classification and regression in tabular data, traditional machine learning methods remain effective. Logistic Regression (LR) [33] and Generalized Linear Models (GLM) [12] are commonly used for modeling linear relationships. In contrast, tree-based models such as Decision Trees (DT) [5], XGBoost [7], Random Forest [4], CatBoost [27], and LightGBM [16] are preferred for their ability to handle complex non-linear relationships. These models are noted for their interpretability and effectiveness in dealing with diverse feature types, including missing values and categorical features.

Recent trends in the tabular data domain have seen the adoption of deep learning models designed to enhance performance. These include a variety of neural architectures such as ResNet [14], SNN [20], AutoInt [30], and DCN V2 [31], which are primarily supervised methods. Additionally, hybrid approaches combine decision trees with neural networks for end-to-end training, examples of which include NODE [26], GrowNet [2], TabNN [18], and DeepGBM [17]. There are also emerging focuses on representation learning methods that utilize self- and semi-supervised learning for effective information extraction, such as VIME [37], SCARF [3], and Recontab [6].

2.2 Transformer-based Models

In the realm of tabular data, transformer-based methods have become increasingly prominent, leveraging attention mechanisms to discern relationships across features and data samples. Notable models in this category include TabNet [1], TabTransformer [15], FT-Transformer [10], and SAINT [29]. These models exemplify the integration of transformer technology in tabular learning, offering enhanced modeling capabilities through attention-driven feature interactions.

2.3 Contrastive Representation Learning

Contrastive Representation Learning (CRL) has been substantively developed through significant contributions across various fields. The introduction of MoCo, a dynamic dictionary technique for unsupervised visual representation learning, marked a major advancement in the efficiency of learning algorithms [13]. Following this, the SimCLR framework simplified and improved the process by applying a straightforward contrastive loss to visual representations [8]. In the natural language processing arena, enhancements in sentence embeddings have been demonstrated through an efficient learning framework that applies CRL principles [22]. Furthermore, the application of CRL to supervised learning scenarios has shown substantial improvements in classifier robustness and accuracy, broadening the potential uses of this approach [19]. Recently, efforts inspired by CRL have been extended to the tabular domain. However, these initiatives primarily focus on instance-level contrast, which may limit their broader applicability [6, 29, 37].

2.4 Feature Decoupling

Feature decoupling is integral to advancing machine learning models, enhancing both interpretability and performance by separating complex, intertwined data elements. In computer vision, techniques such as unsupervised domain-specific deblurring leverage disentangled representations to improve image clarity by effectively isolating content from blur features [23]. The introduction of disentangled non-local neural networks demonstrates significant improvements in context modeling, benefiting tasks like semantic segmentation and object detection [36]. In multimodal transformers, decoupling strategies have been particularly effective, as demonstrated in zero-shot semantic segmentation, where the separation of components allows for better utilization of vision-language pre-trained models [9]. Moreover, in the tabular data domain, SwitchTab [34] showcases feature decoupling to enhance self-supervised learning by effectively isolating mutual and salient features to improve decision-making and model robustness in downstream tasks. This work has inspired us to propose a more comprehensive contrastive learning framework to enhance the decoupled feature learning.

3 Method

In this section, we introduce TabDeco, our comprehensive framework for contrastive learning tailored to tabular data representation. We begin with outlining the supervised training process of TabDeco, setting the foundation for effective feature extraction, decoupling and contrastive learning approaches. We then introduce the core component of TabDeco, the global-local feature decoupling mechanisms, to enhance the model's adaptability and robustness across diverse datasets. Furthermore, we explore the integration of various contrastive loss combinations, illustrating how each tailored loss function uniquely contributes to improving the model's performance by optimizing representation learning with better separation and alignment in feature space. Finally, we summarize the supervised learning algorithm and discuss the training strategies.

3.1 Supervised Training Framework

The supervised training architecture in our framework integrates column and row attention blocks with feature decoupling and contrastive learning, specifically tailored for tabular data. The attention blocks are from the state-of-the-art transformer-based model architectures considering both column-wise attentions for feature representation [15, 10] and row-wise attentions for intersample representation [29]. The feature decoupling module is inspired by the breakthrough idea in SwitchTab [34], further enhanced by the comprehensive contrastive learning process. Meanwhile, the natural integration of highly-resolved attention mechanism and fine-grained feature distinction could systematically enhance the model's ability to capture complex interactions among features and instances, making the most out of the unique characteristics of tabular data.

Given a tabular dataset represented by $\mathcal{D} = \{(\mathbf{x_i}, y_i)\}_{i=1}^{N}$, where each $\mathbf{x_i}$ is a m-dimensional feature vector with y_i as its associated label. N is the total number of samples. As shown in Figure 1, we append a $[\mathbf{CLS}]$ token with a learned embedding to each data sample, making $\mathbf{x_i} = [[\mathbf{CLS}], f_i^1, f_i^2, \cdots, f_i^m]$ be the single data point with categorical or numerical features. $\mathbf{E}$ is the embedding layer using different embedding functions to embed each feature into a d-dimensional space, i.e., $\mathbf{x_i} \in \mathbb{R}^{(m+1)} \rightarrow E(\mathbf{x_i}) \in \mathbb{R}^{(m+1)\times d}$. The encoded representations are then passed

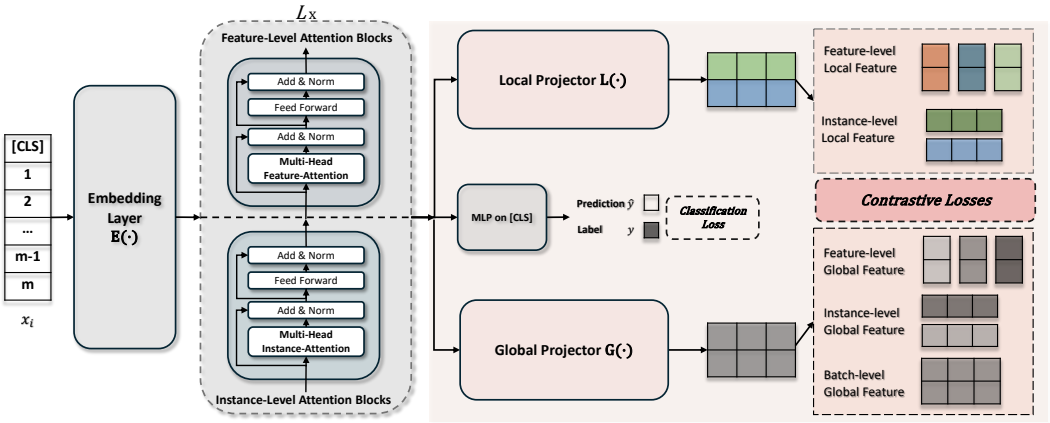

Figure 1: An overview of the TabDeco Framework.

through L-layer attention blocks composed of feature-level and instance-level attention mechanisms. Feature-level attention focuses on refining relationships among features (columns), while instance-level attention blocks handle the interactions among instances (rows). This dual attention mechanism ensures the model's adaptability to complex feature interactions and intersample relationships.

The outputs from these attention blocks are projected into local and global spaces via respective projectors: the Local Projector $\mathbf{L}(\cdot)$ generates a local feature vector to represent both feature-level and instance-level local features, whereas the Global Projector $\mathbf{G}(\cdot)$ produces a global feature vector which could simutaneously represent feature-level, instance-level, and batch-level global features. The local and global feature vectors are then used to compute various contrastive losses (e.g., InfoNCE) to enhance the selected distinctions among feature, instance and/or global patterns. Additionally, supervised loss (e.g., Cross Entropy), is integrated to directly guide the learning process towards label prediction. This structured approach allows the model to capture fine-grained feature relationships and improve predictive performance across different data distributions and complexities.

3.2 Global-Local Decoupling

We introduce the global-local decoupling mechanism to effectively capture information at multiple levels. Specifically, each feature vector $\mathbf{x_i}$ is decomposed into two distinct representations: a global feature vector $\mathbf{g_i}$ and a local feature vector $\mathbf{l_i}$. The global vector $\mathbf{g_i}$ captures broad patterns and shared characteristics across the entire dataset, focusing on the correlations among features and interrelationships across different samples. In contrast, the local vector $\mathbf{l_i}$ captures instance-specific variations that highlights the individual distinctions within each sample as well as the feature-specific uniqueness to distinguish one feature from the others. Mathematically, we define the global and local feature mappings as:

$$\mathbf{g_i} = \mathbf{G}(\mathbf{x_i}; \theta_g), \quad \mathbf{l_i} = \mathbf{L}(\mathbf{x_i}; \theta_l) \tag{1}$$

where $\mathbf{G}(\cdot)$ and $\mathbf{L}(\cdot)$ are learnable functions parameterized by θ_g and θ_l respectively.

3.3 Contrastive Losses

Given the decoupled global and local feature vectors $\mathbf{g_i}$ and $\mathbf{l_i}$, we are constructing positive and negative pairs to distinguish between broader feature distributions and fine-grained individual characteristics. In general, we are encouraging global features to be similar to other global features, and discouraging local features from being similar to other local features. In addition, global features are discouraged from being similar to local features. The loss functions are demoted as

$$\mathcal{L}_{\text{global}} = -\log \frac{\exp\left(\mathbf{sim}(\mathbf{g_i}, \mathbf{g_j})/\tau\right)}{\sum_k \exp\left(\mathbf{sim}(\mathbf{g_i}, \mathbf{g_k})/\tau\right)}, \tag{2}$$

$$\mathcal{L}_{\text{local}} = -\log \frac{\exp\left(-\mathbf{sim}(\mathbf{l_i}, \mathbf{l_j})/\tau\right)}{\sum_k \exp\left(-\mathbf{sim}(\mathbf{l_i}, \mathbf{l_k})/\tau\right)}, \tag{3}$$

$$\mathcal{L}_{\text{cross}} = -\log \frac{\exp\left(-\mathbf{sim}(\mathbf{g_i}, \mathbf{l_j})/\tau\right)}{\sum_k \exp\left(-\mathbf{sim}(\mathbf{g_i}, \mathbf{l_k})/\tau\right)}, \tag{4}$$

where τ is the temperature parameter used to scale the similarity values and k is the index across the entire set of features to compute the normalization factor for the similarity scores. $\mathbf{sim}(\cdot)$ represents the similarity measure between two vectors, e.g., cosine similarity.

Depending on the level of distinctiveness to be achieved, we proposed six types of contrastive losses, as shown in Table 1. Each loss function consists of the three loss components shown in Equation (2-4) with its corresponding similarity measure aggregating from the decoupled global and local feature vectors $\mathbf{g_i}$ and $\mathbf{l_i}$. The major difference lies in the similarity measures with different summation logic to distinguish aspects of global and local features. Specifically, $\mathcal{L}_{\text{all}}$ aims to contrast each feature of each instance across the entire batch, creating a comprehensive similarity matrix of size $(b*m, b*m)$. It ensures that every data point is contrasted against all others, enforcing a comprehensive alignment that helps the model to learn subtle distinctions between different features and instances. Intuitively, this broad-level contrasting pushes the model to learn detailed, high-granularity representations by maximizing the separation between different feature-instance combinations. $\mathcal{L}_{\text{gg}}$ compares each batch of features to a shared, global feature representation randomly initialized with dimension (m, d), producing a (b, m, m) similarity structure. By contrasting batch-level features against a global standard, this loss encourages the model to align local features with global patterns, improving consistency across batches. The intuition here is to synchronize individual batches with a common global feature representation, promoting a unified feature understanding across diverse instances.

The feature-level loss $\mathcal{L}_{\text{f}}$ contrasts each feature across all instances within the same batch and captures relationships between different features within the same batch, encouraging the model to differentiate features that frequently occur together while still learning their distinct roles. The intuition is to refine feature boundaries, making the model more adept at identifying individual feature influences within complex interactions. $\mathcal{L}_{\text{fs}}$ further contrasts each feature for each single instance across the batch and targets the fine-grained relationships of specific features of individual instances and fosters detailed understanding of how particular features vary between samples. Similarly, this sample-level loss $\mathcal{L}_{\text{s}}$ contrasts each instance by comparing all features within that instance against others in the batch. This approach focuses on enhancing the discriminative ability of the model at the instance level, making it better at distinguishing between different samples based on their overall feature profiles. The intuitive aim is to fine-tune how the model distinguishes between individual data points, aiding in more accurate instance-level classification. Furthermore, $\mathcal{L}_{\text{sf}}$ contrasts instances one feature at a time to emphasize on unique features with more differentiating power. The intuition is to enhance the model's sensitivity to how each feature contributes to instance-level differences, sharpening the model's ability to make precise comparisons between instances. Algorithm 1 shows the details on how to integrate the global-local decoupling with contrastive learning through different types of losses and positive and negative pairs generation. In practices, different types of contrastive losses can be combined together to learn the feature and instance representations with different granularities. Figure 2 further visualize the different levels of contrastive learning across global and local features.

Loss	Similarity Summation	Description
$\mathcal{L}_{\text{all}}$	$(b*m, d), (b*m, d) \rightarrow (b*m, b*m)$	Contrasting each feature of each instance across the batch
$\mathcal{L}_{\text{gg}}$	$(b, m, d), (m, d) \rightarrow (b, m, m)$	Contrasting each batch with the cross-batch feature-level global representation
$\mathcal{L}_{\text{f}}$	$(b, m, d), (b, m, d) \rightarrow (m, m)$	Contrasting each feature of all instances in the batch
$\mathcal{L}_{\text{s}}$	$(b, m, d), (b, m, d) \rightarrow (b, b)$	Contrasting each instance of all features in the batch
$\mathcal{L}_{\text{fs}}$	$(b, m, d), (b, m, d) \rightarrow (b, m, m)$	Contrasting each feature for each instance across the batch
$\mathcal{L}_{\text{sf}}$	$(b, m, d), (b, m, d) \rightarrow (b, b, m)$	Contrasting each instance across the batch for each feature

Table 1: Contrastive losses

4 Experiments and Results

4.1 Datasets, Setup, and Baselines

4.1.1 Datasets

We evaluate the proposed method using a selection of widely recognized datasets, as utilized in recent studies [29]. These datasets include Bank (BK) [24], Blastchar (BC) [25], Shoppers (SH) [28],

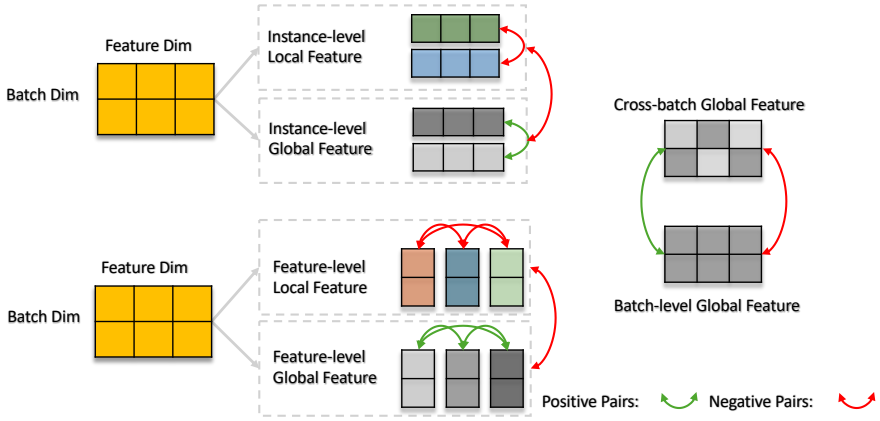

Figure 2: A demonstration of different levels of contrasts

Algorithm 1 Supervised Learning with TabDeco

Require: training data with label $\mathcal{D} = \{(\mathbf{x_i}, y_i)\}_{i=1}^{N}$, number of features m, batch size b, embedding layer $\mathbf{E}$, embedding size d, attention blocks $\mathbf{A}$, projector for global feature $\mathbf{G}$, projector for local feature $\mathbf{L}$, multi-layer perceptron (MLP), cross-entropy (CE), mean squared error (MSE).

1: **for** one mini-batch $\{(\mathbf{x_i}, y_i)\}_{i=1}^{b} \subseteq \mathcal{D}$, $\mathbf{x_i} \in \mathbb{R}^{(m)}$ **do**
2: for each sample $\mathbf{x_i}$, embed each feature into d-dimensional embeddings:
$\breve{\mathbf{x}}_\mathbf{i} = \mathbf{E}(\mathbf{x_i}) \in \mathbb{R}^{m \times d}$
3: for the batch of samples $\mathcal{X}^b = \{\breve{\mathbf{x}}_\mathbf{i}\}_{i=1}^{b}$, embeddings are learned through the attention blocks:
$\mathcal{Z}^b = \mathbf{A}(\mathcal{X}^b), \mathcal{Z}^b \in \mathbb{R}^{b \times m \times d}$
4: Decouple the embedding sample into global and local vectors:
$\mathbf{g}^b = \mathbf{G}(\mathcal{Z}^b), \quad \mathbf{l}^b = \mathbf{L}(\mathcal{Z}^b), \quad \mathbf{g}^b, \mathbf{l}^b \in \mathbb{R}^{b \times m \times d}$
5: Supervised learning prediction:
$\hat{y}^b = \text{MLP}(\mathcal{Z}^b, \mathbf{g}^b, \mathbf{l}^b), \quad$ where $\hat{y}^b = \{\hat{y}_i\}_{i=1}^{b}$
6: Supervised loss computation:
$\mathcal{L}_{\text{sup}} = \text{CE}(y_i, \hat{y}_i)$ for classification or $\mathcal{L}_{sup} = \text{MSE}(y_i, \hat{y}_i)$ for regression
7: Contrastive loss computation:
(1) select the loss types, e.g., $\mathcal{L}_{\text{all}}$ or choose multiple ones $\mathcal{L}_{\text{f}}$, $\mathcal{L}_{\text{s}}$
(2) for each loss type, compute the loss components from Equation (2-4), e.g.,
$\mathcal{L}_{\text{all}} = \mathcal{L}_{\text{global}} + \mathcal{L}_{\text{local}} + \mathcal{L}_{\text{cross}}$
(3) get the total contrastive loss
$\mathcal{L}_{\text{contrast}} = \mathcal{L}_{\text{all}}$ or $\mathcal{L}_{\text{contrast}} = \mathcal{L}_{\text{f}} + \mathcal{L}_{\text{s}}$
8: combine supervised loss with contrastive losses to get the total
$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{sup}} + \alpha \mathcal{L}_{\text{contrastive}}$ where α is the weight balancing the contributions of each loss.
9: update embedding layer $\mathbf{E}$, attention blocks $\mathbf{A}$, projectors $\mathbf{G}$ and $\mathbf{L}$, and minimize $\mathcal{L}_{total}$ through backpropagation.
10: **end for**

Volkert (VO) [11] and MNIST (MN) [35], among others. The datasets are diverse, ranging in size from 200 to 495,141 samples and spanning 8 to 784 features, incorporating both categorical and continuous variables. Some datasets contain missing data, while others are complete; similarly, some are well-balanced, whereas others exhibit highly skewed class distributions. All of these datasets are publicly accessible as shown in table 4.

4.1.2 Model variants

The TabDeco architecture discussed above follows similar design in SAINT to define three variants of model structures with different choices of attention block. TabDeco has the attention transformer encoder stacking both feature-level and instance-level attention blocks. TabDeco-s has only instance-level attention (e.g., SAINT-s) while TabDeco-f has only feature-level attention (e.g., SAINT-f).

4.1.3 Training Details

We train all models, including those with pre-training, using the AdamW optimizer with parameters set to $\beta_1 = 0.9$, $\beta_2 = 0.999$, a weight decay of 0.01, and a learning rate of 0.0001. The batch size is set to 128, except for datasets with a large number of features, such as MNIST and Volkert, where we use smaller batch sizes. The data is split into 65% for training, 15% for validation, and 20% for testing. For the contrastive losses, we use temperature τ = 0.5. In each of our experiments, we use one NVIDIA A10G Tensor Core GPU. Individual training runs take between 5 minutes and 10 hours. For most of the datasets, we use embedding size d = 32. Due to the memory constraints of a single GPU, we use d = 4 for MNIST and d = 8 for Volkert. We used L = 6 layers and h = 8 attention heads in all datasets except MNIST and Volkert, where we reduce to L = 2 and h = 4. We leverage the implementations of various methods in TALENT toolbox [21] for training and comparisons.

4.1.4 Metrics

Since most of the tasks in our analysis focus on binary classification, we primarily use the Area Under the Receiver Operating Characteristic curve (AUROC) to evaluate performance. AUROC provides a robust measure of the model's ability to differentiate between the two classes within the dataset. For the two multi-class datasets, Volkert (VO) and MNIST (MN), we instead use accuracy on the test set as the performance metric. This approach allows us to appropriately assess the model's effectiveness across different types of classification tasks.

4.1.5 Baselines

Following the previous paradigm [29, 34], we conduct a comprehensive comparison of our model against a range of established techniques, including traditional algorithms like logistic regression and random forests, as well as advanced boosting frameworks such as XGBoost, LightGBM, and CatBoost. Additionally, we evaluate our model's performance alongside state-of-the-art deep learning models, including multi-layer perceptrons, VIME[37], TabNet[1], TabTransformer[32] and the three variants of SAINT [29]. SAINT by itself include both feature-level attention block and instance-level attention block in each stage. For the two variants, SAINT-f has only feature-level attention and SAINT-s has only intersample attention. For models that incorporate unsupervised pre-training, We focus on comparing the supervised performance and only considering the classification tasks.

4.2 Main Results

We report performance comparisons on the set of public datasets, with the AUROC results summarized in Table 2 for model predicting power. The corresponding standard errors are reported in Appendix in Table 5 for model robustness. Each number reported in the two tables represents the mean of 10 trials with different random seeds.

In Table 2, TabDeco demonstrates significant predicting power ranking the best or second best across all datasets. Specifically, in 7 out of 11 datasets, one of the TabDeco variants outperforms all baseline models. In the remaining 4 datasets, TabDeco achieves the second best twice in Bank, outperformed by LightGBM and SAINT-f respectively. Income and Spambase datasets are also getting inferior performance from TabDeco, for which boosting methods have dominating predicting power. Meanwhile, TabDeco variants have shown the consistency to outperform their corresponding SAINT variants and SwitchTab for 10 datasets except Blastchar, which indicates the comprehensive enhancement from feature decoupling and constrastive learning. Table 5 further demonstrates the robustness of model performance for TabDeco, which exhibits better consistency than boosting methods and SAINT variants, especially for the datasets with better predicting power, such as Shoppers, HTRU2, etc.

4.3 Ablation Studies

In this section, we conduct experiments on different contrastive loss combinations to evaluate how they could performance differently for various datasets. We test adding the loss combination to the total loss for each TabDeco variant and report the best performance for each combination from all variants. The baseline noted by "-" is without any contrastive losses or feature decoupling, deteriorating to the SAINT variants for classification-only tasks. To determine whether the loss

Dataset size	45,211	7,043	12,330	32,561	17,898	10,000	4,601	1,055	495,141	58,310	60,000
Feature size	16	27	17	14	8	11	57	41	49	147	784
Method/Dataset	**Bank**	**Blastchar**	**Shoppers**	**Income**	**HTRU2**	**Shrutime**	**Spambase**	**QSARBio**	**Forest**	**Volkert†**	**MNIST†**
Logistic Reg.	90.73	82.34	87.03	92.12	98.23	83.37	92.77	84.06	84.79	53.87	89.89
Random Forest	89.12	80.63	89.87	88.04	96.41	80.87	98.02	91.49	98.80	66.25	93.75
XGBoost	92.96	81.78	92.51	92.31	97.81	83.59	98.91	92.70	95.53	68.95	94.13
LightGBM	**93.39**	83.17	93.20	**92.57**	98.10	85.36	**99.01**	92.97	93.29	67.91	95.20
CatBoost	90.47	84.77	93.12	90.80	97.85	85.44	98.47	93.05	85.36	66.37	96.60
MLP	91.47	59.63	84.71	92.08	98.35	73.70	66.74	79.66	96.81	63.02	93.87
VIME	76.64	50.08	74.37	88.98	97.02	70.24	69.24	81.04	75.06	64.28	95.77
TabNet	91.76	79.61	91.38	90.72	97.58	75.24	97.93	67.55	96.37	56.83	96.79
TabTransormer	91.34	81.67	92.70	90.60	96.56	85.60	98.50	91.80	84.96	57.98	88.74
SwitchTab	91.80	83.56	91.20	89.79	97.15	84.89	96.06	93.85	97.66	68.90	96.80
SAINT	93.12	84.58	93.22	90.99	98.28	85.79	97.66	93.34	99.06	68.87	97.48
SAINT-s	92.84	83.57	92.31	90.59	98.58	85.43	97.83	92.48	98.85	69.61	97.45
SAINT-f	92.68	**85.18**	93.10	91.23	98.54	85.09	97.40	91.91	98.96	58.76	91.25
TabDeco	93.34	84.96	**93.53**	91.17	98.57	**86.14**	97.79	93.80	**99.15**	69.46	**97.85**
TabDeco-s	93.00	84.16	92.66	90.95	**98.60**	85.93	97.97	**94.87**	99.06	**69.91**	97.71
TabDeco-f	92.48	85.00	93.05	91.26	**98.60**	85.00	97.36	92.08	99.00	58.52	95.56

Table 2: Mean AUROC scores averaged over 10 trials for 11 datasets on classification tasks. Baseline results are quoted from original papers when possible and reproduced otherwise. We highlight best result in bold and second best in underline. Columns denoted by † are multi-class problems, and we report accuracy.

combinations are useful, we count the ones outperforming baseline for each dataset. Similarly, we count the outperforming ones for each loss combination across the datasets to check the consistency. As shown in Table‘3, out of the 11 datasets, losses for more comprehensive feature-level (e.g., $\mathcal{L}_{\text{fs}}$) and/or instance-level (e.g., $\mathcal{L}_{\text{sf}}$) and/or cross-batch contrasting (e.g., $\mathcal{L}_{\text{gg}}$) and their combinations are demonstrated to be significantly superior on getting better performance, outperforming the baseline for at least 6 and up to 11 datasets. For each datasets, the performance may vary across different contrastive losses. Out of the 14 combinations we test, the counts that outperform baseline varies from 1 to 10. Specifically the simplified feature-level or instance-level losses $\mathcal{L}_{\text{f}}$ and $\mathcal{L}_{\text{s}}$ can hardly get better performance. The global contrastive loss $\mathcal{L}_{\text{gg}}$ as well as the one capturing more granular level feature and/or instance characteristics generally perform better.

Dataset/Loss	–	$\mathcal{L}_{\text{f}}$	$\mathcal{L}_{\text{s}}$	$\mathcal{L}_{\text{f+s}}$	$\mathcal{L}_{\text{fs}}$	$\mathcal{L}_{\text{sf}}$	$\mathcal{L}_{\text{fs+sf}}$	$\mathcal{L}_{\text{all}}$	$\mathcal{L}_{\text{gg}}$	$\mathcal{L}_{\text{fs+gg}}$	$\mathcal{L}_{\text{sf+gg}}$	$\mathcal{L}_{\text{all+gg}}$	$\mathcal{L}_{\text{sf+fs+gg}}$	Counts by Row (14)
Bank	93.12	92.56	93.08	92.69	93.29	93.21	93.22	93.20	93.13	93.34	93.23	92.99	93.15	8
Blastchar	84.58	82.61	84.56	84.51	84.75	84.96	84.51	84.79	84.86	85.00	84.71	84.96	85.00	8
Shoppers	93.22	90.19	92.95	92.18	93.32	93.15	93.24	93.29	93.24	93.29	93.09	93.06	93.53	6
Income	91.23	91.00	90.70	90.73	91.15	91.19	91.05	91.14	91.18	91.16	91.20	91.17	91.26	1
HTRU2	98.58	98.07	98.59	98.21	98.60	98.56	98.57	98.49	98.57	98.58	98.56	98.46	98.60	5
Shrutime	85.79	78.32	85.45	77.19	85.75	86.14	85.67	85.84	86.08	85.79	85.94	86.03	85.82	7
Spambase	97.83	93.00	97.97	91.96	97.77	97.96	97.94	97.83	97.83	97.82	97.91	97.89	97.93	7
QSARBio	93.34	84.27	94.87	83.14	94.33	93.79	94.50	94.34	93.45	94.13	93.71	94.27	93.80	10
Forest	99.06	94.35	97.89	97.50	98.99	99.15	99.10	96.18	99.06	98.96	99.10	98.26	99.08	5
Volkert†	69.61	62.27	64.86	61.28	69.88	69.50	69.70	69.91	69.58	69.78	69.88	69.58	69.67	6
MNIST†	97.48	91.02	93.96	63.58	97.39	97.13	97.58	96.96	97.58	97.44	97.85	97.37	97.83	4
Counts by Column (11)	–	0	3	0	6	6	6	7	8	7	8	4	11	

Table 3: Comparison of the best performance for each contrastive loss combination across TabDeco variants. "-" corresponds to the SAINT variants without feature decoupling or contrastive learning. We count the results by rows across datasets or by columns across lossess as long as it is better than "-". Rows denoted by † are multi-class problems, and we report accuracy rather than AUROC.

5 Conclusion

In conclusion, the complexities inherent in tabular data demand innovative approaches that go beyond traditional deep learning and tree-based models. While methods like SwitchTab and SAINT have made strides in enhancing representation learning through feature decoupling and attention mechanisms, they fall short in fully addressing the challenges of dataset complexity and the generation of meaningful sample pairs. Our proposed framework, TabDeco, synthesizes the strengths of these methods by integrating attention-based contrastive learning with feature decoupling, enabling a deeper exploration of local and global interactions within the data. Extensive experiments demonstrate that TabDeco consistently outperforms existing models, including leading gradient boosting algorithms, across various benchmark tasks, underscoring its effectiveness and adaptability. By advancing the construction of positive and negative samples through multi-perspective contrastive learning, TabDeco sets a new standard for tabular data representation, offering a robust and interpretable solution for complex tabular scenarios. This work not only addresses current limitations but also paves the way for future innovations in contrastive learning for tabular data.

References

[1] Sercan Ö Arik and Tomas Pfister. Tabnet: Attentive interpretable tabular learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 6679–6687, 2021.

[2] Sarkhan Badirli, Xuanqing Liu, Zhengming Xing, Avradeep Bhowmik, Khoa Doan, and Sathiya S Keerthi. Gradient boosting neural networks: Grownet. *arXiv preprint arXiv:2002.07971*, 2020.

[3] Dara Bahri, Heinrich Jiang, Yi Tay, and Donald Metzler. Scarf: Self-supervised contrastive learning using random feature corruption. *arXiv preprint arXiv:2106.15147*, 2021.

[4] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[5] Leo Breiman. *Classification and regression trees*. Routledge, 2017.

[6] Suiyao Chen, Jing Wu, Naira Hovakimyan, and Handong Yao. Recontab: Regularized contrastive representation learning for tabular data. *arXiv preprint arXiv:2310.18541*, 2023.

[7] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[9] Jian Ding, Nan Xue, Gui-Song Xia, and Dengxin Dai. Decoupling zero-shot semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11583–11592, 2022.

[10] Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34:18932–18943, 2021.

[11] Isabelle Guyon, Lisheng Sun-Hosoya, Marc Boullé, Hugo Jair Escalante, Sergio Escalera, Zhengying Liu, Damir Jajetic, Bisakha Ray, Mehreen Saeed, Michéle Sebag, Alexander Statnikov, WeiWei Tu, and Evelyne Viegas. Analysis of the automl challenge series 2015-2018. In *AutoML*, Springer series on Challenges in Machine Learning, 2019.

[12] Trevor J Hastie and Daryl Pregibon. Generalized linear models. In *Statistical models in S*, pages 195–247. Routledge, 2017.

[13] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] Xin Huang, Ashish Khetan, Milan Cvitkovic, and Zohar Karnin. Tabtransformer: Tabular data modeling using contextual embeddings. *arXiv preprint arXiv:2012.06678*, 2020.

[16] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

[17] Guolin Ke, Zhenhui Xu, Jia Zhang, Jiang Bian, and Tie-Yan Liu. Deepgbm: A deep learning framework distilled by gbdt for online prediction tasks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 384–394, 2019.

[18] Guolin Ke, Jia Zhang, Zhenhui Xu, Jiang Bian, and Tie-Yan Liu. Tabnn: A universal neural network solution for tabular data. 2018.

[19] Prannay Khosla, Piotr Teterwak, Chen Wang, Abhinav Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020.

[20] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. *Advances in neural information processing systems*, 30, 2017.

[21] Si-Yang Liu, Hao-Run Cai, Qi-Le Zhou, and Han-Jia Ye. Talent: A tabular analytics and learning toolbox. *arXiv preprint arXiv:2407.04057*, 2024.

[22] Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. In *International Conference on Learning Representations*, 2018.

[23] Boyu Lu, Jun-Cheng Chen, and Rama Chellappa. Unsupervised domain-specific deblurring via disentangled representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10225–10234, 2019.

[24] Sérgio Moro, Paulo Cortez, and Paulo Rita. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62:22–31, 2014.

[25] James Ouk, David Dada, and Kyung Tae Kang. Telco customer churn. 2018.

[26] Sergei Popov, Stanislav Morozov, and Artem Babenko. Neural oblivious decision ensembles for deep learning on tabular data. *arXiv preprint arXiv:1909.06312*, 2019.

[27] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

[28] C Okan Sakar, S Olcay Polat, Mete Katircioglu, and Yomi Kastro. Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and lstm recurrent neural networks. *Neural Computing and Applications*, 31:6893–6908, 2019.

[29] Gowthami Somepalli, Micah Goldblum, Avi Schwarzschild, C Bayan Bruss, and Tom Goldstein. Saint: Improved neural networks for tabular data via row attention and contrastive pre-training. *arXiv preprint arXiv:2106.01342*, 2021.

[30] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1161–1170, 2019.

[31] Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. Dcn v2: Improved deep & cross network and practical lessons for web-scale learning to rank systems. In *Proceedings of the web conference 2021*, pages 1785–1797, 2021.

[32] Zifeng Wang and Jimeng Sun. Transtab: Learning transferable tabular transformers across tables. *Advances in Neural Information Processing Systems*, 35:2902–2915, 2022.

[33] Raymond E Wright. Logistic regression. 1995.

[34] Jing Wu, Suiyao Chen, Qi Zhao, Renat Sergazinov, Chen Li, Shengjie Liu, Chongchao Zhao, Tianpei Xie, Hanqing Guo, Cheng Ji, et al. Switchtab: Switched autoencoders are effective tabular learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 15924–15933, 2024.

[35] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[36] Minghao Yin, Zhuliang Yao, Yue Cao, Xiu Li, Zheng Zhang, Stephen Lin, and Han Hu. Disentangled non-local neural networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, pages 191–207. Springer, 2020.

[37] Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela van der Schaar. Vime: Extending the success of self-and semi-supervised learning to tabular domain. *Advances in Neural Information Processing Systems*, 33:11033–11043, 2020.

A Appendix / supplemental material

A.1 Datasets Details

Dataset	Source Link
Bank	https://archive.ics.uci.edu/ml/datasets/bank+marketing
Blastchar	https://www.kaggle.com/blastchar/telco-customer-churn
Shoppers	https://archive.ics.uci.edu/ml/datasets/Online+Shoppers+Purchasing+Intention+Dataset
Income	https://www.kaggle.com/lodetomasi1995/income-classification
HTRU2	https://archive.ics.uci.edu/ml/datasets/HTRU2
Shrutime	https://www.kaggle.com/shrutimechlearn/churn-modelling
Spambase	https://archive.ics.uci.edu/ml/datasets/Spambase
QSARBio	https://archive.ics.uci.edu/ml/datasets/QSAR+biodegradation
Forest	https://kdd.ics.uci.edu/databases/covertype
Volkert	http://automl.chalearn.org/data
MNIST	http://yann.lecun.com/exdb/mnist/

Table 4: Dataset links

A.2 Additional Results

Method/Dataset	Bank	Blastchar	Shoppers	Income	HTRU2	Shrutime	Spambase	QSARBio	Forest	Volkert†	MNIST†
Logistic Reg.	0.25	0.20	0.41	6.34	0.26	0.53	0.12	0.70	0.11	1.33	3.19
Random Forest	0.27	0.70	0.60	0.30	0.25	0.38	0.27	0.80	0.01	1.27	4.59
XGBoost	0.15	0.34	0.50	0.15	0.10	0.39	0.08	0.45	0.01	0.51	1.98
LightGBM	0.21	0.34	0.48	0.13	0.13	0.58	0.05	0.67	0.01	0.64	3.78
CatBoost	0.17	0.19	0.41	0.15	0.23	0.41	0.11	0.79	0.01	1.17	1.66
MLP	0.21	0.32	0.60	2.74	0.31	1.65	0.15	1.00	0.68	1.56	3.74
VIME	2.03	0.26	2.74	5.10	2.52	1.15	3.03	0.71	6.91	6.67	8.15
TabNet	0.33	0.30	0.68	0.17	0.29	5.12	0.15	2.67	0.01	1.47	2.22
TabTransormer	0.34	0.30	0.69	0.17	0.29	5.18	0.15	2.70	0.01	1.48	2.24
SwitchTab	0.54	0.39	0.65	0.47	0.45	0.76	0.44	1.55	0.03	0.38	0.54
SAINT	0.12	0.35	0.49	0.21	0.12	0.20	0.31	0.26	0.01	0.37	0.36
SAINT-s	0.19	0.20	0.49	0.49	0.06	0.56	0.32	0.97	0.02	0.42	0.11
SAINT-f	0.22	0.38	0.37	0.07	0.06	0.36	0.15	1.09	0.00	0.44	0.47
TabDeco	0.15	0.24	0.26	0.08	0.04	0.31	0.27	0.41	0.01	0.48	0.18
TabDeco-s	0.11	0.33	0.24	0.12	0.09	0.32	0.15	0.24	0.01	0.27	0.11
TabDeco-f	0.52	0.27	0.33	0.07	0.04	0.40	0.10	0.99	0.00	0.53	0.26

Table 5: Standard deviations on AUROC scores computed over 10 trials from Table 2. Baseline results are quoted from original papers when possible and reproduced otherwise. Columns denoted by † are multi-class problems, and we report standard errors on accuracy rather than AUROC.