Tab-PET: Graph-Based Positional Encodings for Tabular Transformers

Yunze Leng[1], **Rohan Ghosh**[1], **Mehul Motani**[1, 2]

[1]Department of Electrical and Computer Engineering, College of Design and Engineering, National University of Singapore
[2]N.1 Institute for Health, Institute for Digital Medicine (WisDM), Institute of Data Science, National University of Singapore
yleng@u.nus.edu, rghosh92@gmail.com, motani@nus.edu.sg

Abstract

Supervised learning with tabular data presents unique challenges, including low data sizes, the absence of structural cues, and heterogeneous features spanning both categorical and continuous domains. Unlike vision and language tasks, where models can exploit inductive biases in the data, tabular data lacks inherent positional structure, hindering the effectiveness of self-attention mechanisms. While recent transformer-based models like TabTransformer, SAINT, and FT-Transformer (which we refer to as 3T) have shown promise on tabular data, they typically operate without leveraging structural cues such as positional encodings (PEs), as no prior structural information is usually available. In this work, we find both theoretically and empirically that structural cues, specifically PEs can be a useful tool to improve generalization performance for tabular transformers. We find that PEs impart the ability to reduce the effective rank (a form of intrinsic dimensionality) of the features, effectively simplifying the task by reducing the dimensionality of the problem, yielding improved generalization. To that end, we propose Tab-PET (PEs for Tabular Transformers), a graph-based framework for estimating and inculcating PEs into embeddings. Inspired by approaches that derive PEs from graph topology, we explore two paradigms for graph estimation: association-based and causality-based. We empirically demonstrate that graph-derived PEs significantly improve performance across 50 classification and regression datasets for 3T. Notably, association-based graphs consistently yield more stable and pronounced gains compared to causality-driven ones. Our work highlights an unexpected role of PEs in tabular transformers, revealing how they can be harnessed to improve generalization.

Code — https://github.com/kentridgeai/Tab-PET
Extended version — https://arxiv.org/abs/2511.13338

1 Introduction

Tabular data remains one of the most prevalent formats in applied machine learning, spanning domains from healthcare to finance and recommender systems. Yet, learning from tabular data presents unique challenges that distinguish it from vision, language, and audio modalities. First, tabular datasets often suffer from scarcity of data: many real-world problems provide only hundreds to thousands of training examples, limiting model capacity and generalization (Shavitt and Segal 2018). Second, high dimensionality exacerbates the problem: each sample may contain dozens or hundreds of heterogeneous features (especially after one-hot encoding the categorical variables), making interactions sparse and difficult to model. Compounding this is the heterogeneous nature of the features: categorical and continuous variables coexist, yet require distinct treatment in both preprocessing and architectural design (Huang et al. 2020). Crucially, tabular data lacks the inductive biases that underpin recent deep learning breakthroughs: unlike images it has no spatial locality, unlike language it has no sequential ordering, and unlike audio it has no spectral coherence. This absence of structure directly impacts sample efficiency, as models must learn dependency structure from scratch, making the problems of data scarcity and high dimensionality even more pronounced. As a result, tabular learning remains an active area of research (Gorishniy et al. 2021; Kadra et al. 2021).

Modalities like vision and audition benefit from inherent structural priors (spatial locality/temporal continuity). In vision, CNNs exploit translational symmetry by applying shared filters across spatial patches, embedding a strong inductive bias toward local structure. Even transformer-based models in vision and audition, such as ViT and AST, preserve this bias by tokenizing inputs into fixed-size patches or frames, thereby retaining partial spatial or temporal invariance (Dosovitskiy et al. 2020; Gong, Chung, and Glass 2021). In contrast, language models introduce structure into the self-attention mechanism via positional encoding (PE), which enables the model to distinguish token order and learn position-sensitive dependencies (Vaswani et al. 2017). This technique has been adapted to ViTs, where positional embeddings help recover spatial relationships lost during patch flattening (Chu et al. 2021; Xu et al. 2021). These architectural strategies demonstrate that injecting structure, whether through convolution, patching, or PEs, can be integral for sample-efficient learning in domains where raw data lacks explicit structural organization.

A foundational limitation of self-attention in transformers is that it operates over unordered inputs. Without extra information, the mechanism treats all tokens or features as equally exchangeable, encoding no preference for proximity or sequence. This lack of structural bias does not align with

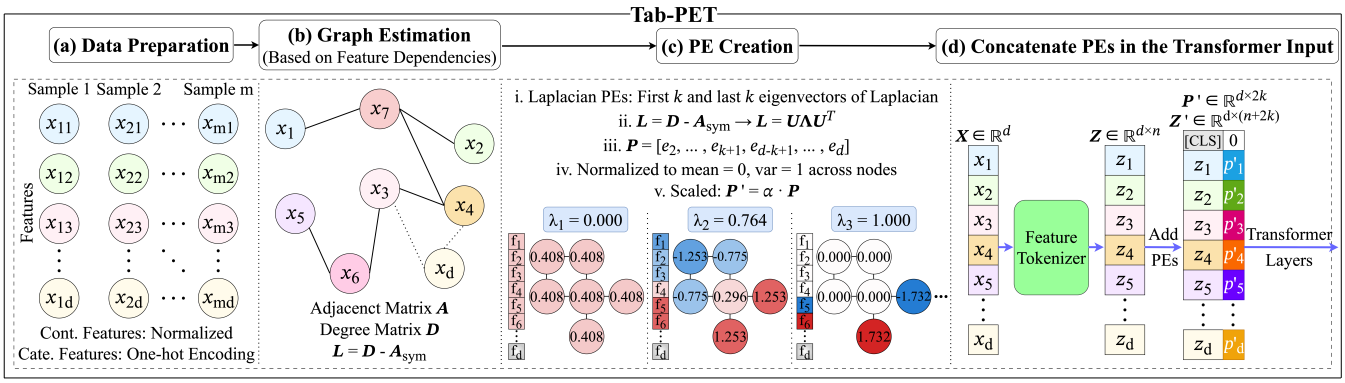

Figure 1: Tab-PET framework for integrating PEs in tabular transformers. (a) Categorical features are one-hot encoded and continuous features are normalized. (b) A feature-wise graph is estimated based on intra-feature dependencies, capturing relational structure among dimensions. (c) Graph Laplacian eigenvectors (examples shown) are extracted to form fixed PEs and scaled using the hyperparameter α to emphasize the degree of importance. (d) These encodings are concatenated with standard embeddings and fed into transformer layers.

natural language, which is inherently organized: grammatical relations and syntactic dependencies rely on the relative positions of words and phrases (Vaswani et al. 2017).

To introduce this bias directly into the model's computation, PEs modify each token by appending a position-specific vector that conveys its location in the input. These vectors, whether fixed or learned, alter the dot product between queries and keys, reshaping the attention pattern. Tokens close in sequence typically carry similar position vectors, which causes the self-attention mechanism to favor interactions among them. This preference allows the model to construct higher-order representations while discouraging irrelevant pairings unless they are statistically warranted. The outcome is a soft inductive bias that recovers local structure otherwise missing from the architecture.

Despite growing interest in PEs across sequence and graph-based domains, their application to tabular data remains largely unexplored. This stems from the lack of inherent structural priors in tabular inputs, i.e., feature order is arbitrary and structural relationships among features are rarely specified upfront. Yet, given the challenges facing tabular classification and regression tasks—limited data, high dimensionality, and feature heterogeneity—the question of whether we can impose meaningful inductive bias via PEs arises. The current consensus in literature is that PEs cannot benefit tabular transformers because tabular data is structure-less and has inputs of different types, in contrast to vision and language (Somepalli et al. 2021). In this work, however, we show that PEs can serve to control learning complexity, reducing the dimensionality of features extracted by self-attention layers. We estimate graphical structures that capture inter-feature associations from the data, and then derive PEs from the eigenvectors of the graph Laplacians. These graph-derived encodings are used to augment the original embeddings, enabling a structured inductive bias where none existed natively. We refer to this as Positional Encodings for Tabular Transformers (Tab-PET).

2 Contributions

Our work has the following contributions:

1. **Tab-PET:** We propose a principled method for constructing PEs in tabular domains. The first step involves learning a feature graph that captures inter-feature associations. We explore two approaches: causality-based and association-based. The second step involves generating feature-wise encodings using the Laplacian eigenvectors of the learned feature graph. These embeddings are then used to augment input embeddings in the transformer.
2. **Theoretical Motivation via Rank Analysis:** We find that the use of PEs imparts the ability to reduce the effective rank of the embeddings within transformer architectures, and more so when they are aligned with the structure of the data. Empirical tests confirm these findings.
3. **Empirical Evaluation Across Benchmarks:** We apply our proposed approach to leading tabular transformer models, including TabTransformer, SAINT, and FT-Transformer. Tab-PET demonstrates consistent improvements across 50 classification and regression datasets.
4. **Ablation Studies and Performance Analysis:** We conduct ablations, statistical significance testing and comparisons with learnable PEs to demonstrate the effectiveness of our approach.

3 Background

3.1 Tabular Transformers

The application of neural networks to tabular data has made strides in recent years. Architectures such as ResNet-like (He et al. 2016), NODE (Popov, Morozov, and Babenko 2019), and SNN (Klambauer et al. 2017) have demonstrated strong performance despite a fundamental challenge: tabular datasets tend to be small in size and lack the rich structural priors present in language or vision domains.

Given the widespread success of transformer architectures in language and vision domains (Vaswani et al. 2017; Dosovitskiy et al. 2020), there has been growing interest in

adapting self-attention mechanisms to tabular data. Models such as TabTransformer (Huang et al. 2020) and FT-Transformer (Gorishniy et al. 2021) attempt this adaptation by embedding features into a higher-dimensional space via projection layers. FT-Transformer, for instance, utilizes a feature-tokenizer to create learnable embeddings for both categorical and continuous features. TabTransformer applies tokenization only to categorical features, treating continuous values collectively through concatenated representations. These embedding transformations enable application of multi-head self-attention followed by classification heads.

Architectures like SAINT (Somepalli et al. 2021) extend attention across intra-sample and inter-sample domains, yielding strong performance. Meanwhile, language-guided models designed for spreadsheet-style inputs (e.g., TAPAS (Herzig et al. 2020)) incorporate semantic cues from column headers and contextual metadata. In our work, we focus exclusively on scenarios where only feature names and raw values are provided, discarding external linguistic context.

3.2 Graph Estimation Approaches

A variety of methods have been proposed to estimate graphical structures over tabular data, where each feature dimension is treated as a node in a graph. Broadly, these approaches fall into two categories: *causality-based* and *association-based*.

Causality-Based Graphs: Causality-based methods aim to infer directed edges between features that reflect underlying generative mechanisms: i.e., an edge from feature i to feature j implies that i is a cause of j. These models often assume a linear structural equation model (SEM) of the form: $\mathbf{x} = \mathbf{W}\mathbf{x} + \boldsymbol{\epsilon}$, where $\mathbf{W}$ is a weighted adjacency matrix encoding causal relationships, and $\boldsymbol{\epsilon}$ is a noise vector. Classical approaches such as LiNGAM (Shimizu et al. 2006) rely on non-Gaussianity assumptions to identify causal directions. More recent methods like NOTEARS and its variants (Zheng et al. 2018; Lee et al. 2019) reformulate structure learning as a continuous optimization problem, minimizing reconstruction error while enforcing acyclicity constraints via smooth penalties. These approaches have enabled scalable and differentiable learning of directed acyclic graphs (DAGs) from observational data.

Association-Based Graphs: Association-based methods construct graphs by quantifying statistical dependence between feature pairs. A common formulation sets the edge weight w_{ij} between features x_i and x_j as:

$$w_{ij} = \rho(x_i, x_j), \tag{1}$$

where ρ is a measure of association, such as Pearson correlation, Spearman correlation, mutual information (MI), or distance. The Chow-Liu algorithm (Chow and Liu 1968) is a canonical example, which uses pairwise MI to construct a maximum-weight spanning tree that approximates the joint distribution. Notably, while Chow-Liu ensures the resulting graph is a tree-structured DAG, it does not model generative mechanisms explicitly. In general, association-based graphs prioritize statistical dependence over causal interpretability.

3.3 Positional Encoding Integration with Graphs

Our objective in this work is to infer PEs for each feature based on the underlying structure of the data, which we estimate via graph estimation (see part (b) of Figure 1). Graph-based PEs have been widely studied in the literature, particularly in the context of graph neural networks and graph transformers. These methods can be broadly categorized into two families: *fixed* and *learnable* PEs.

Fixed Positional Encodings: Fixed PEs typically leverage the spectral properties of the graph Laplacian. The most common strategy involves computing its eigenvectors and using them to encode node positions: (1) *First k eigenvectors*: Dwivedi and Bresson (Dwivedi and Bresson 2020) propose using the lowest-frequency components of the Laplacian spectrum, which capture global graph structure; (2) *All eigenvectors*: Ito et al. (Ito et al. 2025) argue that using the full spectrum preserves all structural information, avoiding frequency truncation; (3) *First and last k eigenvectors*: The same work (Ito et al. 2025) shows that only combining low- and high-frequency components yields robust encodings for both homophilous and heterophilous graphs.

Learnable Positional Encodings: Learnable PEs aim to optimize the encoding process by adapting to task-specific signals. Two notable approaches include: (1) *Elastic PEs*: Liu et al. (Cantürk et al. 2023) propose learning linear projections of Laplacian eigenvectors, enabling flexible adaptation to graph structure; (2) *Eigenvector weighting*: Ito et al. (Ito et al. 2025) introduce a method that learns the importance of each eigenvector via backpropagation, allowing the model to emphasize relevant spectral components.

Our Approach: In this work, we adopt the fixed PE strategy using the first and last k eigenvectors of the graph Laplacian. This choice avoids increasing parametric complexity and also ensures fair comparisons across architectures. Furthermore, empirically, we find that fixed PEs outperform learnable PEs (from scratch) on tabular datasets. Moreover, the use of both low- and high-frequency components has been shown to improve expressiveness across graph types, particularly in heterophilous settings (Ito et al. 2025).

4 Motivation

4.1 Theoretical Results: PEs and Effective Rank

In this section, we theoretically study the ability of PEs to control the intrinsic dimensionality of the learning problem. There have been many significant works that find that lower rank (i.e. low intrinsic dimensionality) of features often leads to better generalization performance (Ghosh and Motani 2023; Huh et al. 2021; Arora et al. 2019). A significant study that explores the link between intrinsic dimension and generalization finds that the intrinsic dimensionality of the data controls both the approximation error (training fit) and the generalization error (Nakada and Imaizumi 2020). Thus, the ability of an architecture to reduce the dimensionality of the learning task via low rank features is a positive sign from a generalization perspective.

In what follows, we provide results that show the ability PEs to directly reduce the effective rank (Roy and

Vetterli 2007) of the CLS output embeddings within FT-Transformers. Note that the CLS output eventually is used for the final prediction via fully connected layers. Proofs are provided in technical appendix I (Leng, Ghosh, and Motani 2025). We reiterate the definition of effective rank below.

Definition 1 (Effective Rank)**.** *For a matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ representing CLS embeddings from n samples, the effective rank is defined as:*

$$r_{\mathit{eff}}(\mathbf{X}) = \exp\left(-\sum_{i=1}^{r} \tilde{\sigma}_i \log \tilde{\sigma}_i\right), \tag{2}$$

where $\tilde{\sigma}_i = \sigma_i / \sum_{j=1}^{r} \sigma_j$ are the normalized singular values obtained from SVD decomposition $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, and r is the rank of $\mathbf{X}$. This formulation captures the intrinsic dimensionality of the learned representations through the Shannon entropy of the singular value distribution.

First, we provide a result in the general case where the input dimensions are i.i.d.

Theorem 1. *[Effective Rank under Random Inputs] Let $x \in \mathbb{R}^d$ be an input vector to a single-layer, single-head FT-Transformer with components $x_i \sim$ i.i.d. and $x_i \in (0, 1)$. Let d_T denote the token dimension (inclusive of concatenated position encodings). Let $q \in \mathbb{R}^{d_T}$ denote the learnable CLS token embedding, and $p_i \in \mathbb{R}^{d_p}$ be the positional encodings for each input dimension. Assume the scaled positional encodings $p_i' = \alpha p_i$ are used, where $\alpha > 0$. Given the query, key and value matrices Q, K, V, where K can be decomposed as $[K_x; K_p]$, where $K_x \in \mathbb{R}^{d\times d_T}, K_p \in \mathbb{R}^{d_p \times d_T}$, and similarly for V. Suppose the following conditions hold: $\max_i \langle Q^T q, K_p^T p_i\rangle - \max_{j\neq i}\langle Q^T q, K_p^T p_j\rangle = \tau$, and the norm of the query-key matrices Q, K and CLS embedding q are all bounded by c_Q, c_K and c_q respectively. Lastly, assume that the tokenizer weights w_i have the same norm and the value matrix V is norm preserving and satisfies $V_p = 0$. Define*

$$C_\alpha = \exp\left(\frac{\alpha\tau - 2c_K c_Q c_q}{\sqrt{d_T}}\right). \tag{3}$$

Then the effective rank r_{eff} of the CLS token output after self-attention satisfies

$$r_{\text{eff}} \leq (C_\alpha + d)\cdot \exp\left(-\frac{C_\alpha}{C_\alpha + d}\cdot \log C_\alpha\right). \tag{4}$$

In the regime where $C_\alpha \gg d$, this simplifies to $r_{\text{eff}} \approx 1 + \frac{d}{C_\alpha}$.

Remark 1. *Note that when $C_\alpha \gg d$, $r_{\text{eff}} \approx 1 + Ce^{-\alpha\tau/\sqrt{d_T}}$. Thus, the effective rank can be significantly reduced when the PEs are weighted higher using larger α, but only when $\tau > 0$. When not using any PEs however, one obtains $\tau = 0$ as $p_i = [0, 0, ..0]$. Thus without PEs, effective rank can be significantly larger.*

We next discuss the case where the input data is not i.i.d dimension-wise but has some underlying structure.

Theorem 2. *[Effective Rank under Structured Inputs] Consider the same setting as in Theorem 1, except that the input vector $x \in \mathbb{R}^d$ is structured as follows: d is even, and*

$$x_i = \begin{cases} \theta & \text{for } i \leq d/2, \\ \theta' & \text{for } i > d/2, \end{cases} \tag{5}$$

with shared latent variables $\theta, \theta' \in (0, 1)$, and coefficients $\beta_i, \gamma_i \in \mathbb{R}$. Then the effective rank r_{eff} of the CLS token output after self-attention satisfies:

- ***(a) Random positional encodings:***

$$r_{\text{eff}} \leq (2C_\alpha + d)\cdot \exp\left(-\frac{2C_\alpha \log(2C_\alpha) + d\log d}{2C_\alpha + d}\right), \tag{6}$$

  which simplifies to $r_{\text{eff}} \approx 1 + \frac{d}{2C_\alpha}$ when $C_\alpha \gg d$.
- ***(b) Shared positional encodings within groups:** If p_i is fixed for all $i \leq d/2$, and is a different fixed vector for $i > d/2$, then*

$$r_{\text{eff}} \leq (C_\alpha + 1)\cdot \exp\left(-\frac{C_\alpha}{C_\alpha + 1}\cdot \log C_\alpha\right), \tag{7}$$

  which simplifies to $r_{\text{eff}} \approx 1 + \frac{1}{C_\alpha}$ for large C_α.

Remark 2. *The above result clearly indicates that the effective rank of the CLS token output of the FT-Transformer depends on whether the PEs have adapted to the structure of the underlying data. When some input dimensions are similar to each other (i.e. Theorem 2), assigning the same PE to the similar dimensions can significantly reduce the effective rank of the CLS output. Thus, choosing appropriate PEs that follow data structure can significantly reduce the dimensionality of the learning problem, enhancing generalization performance. But this carries some limitations as well. Tasks which intrinsically require larger effective rank to appropriately address, may not benefit from the inclusion of PEs.*

5 Methodology

In this section, we outline the four main steps involved in estimating and integrating PEs in transformer-based architectures for tabular data. We summarize the following steps in Figure 1 as well.

5.1 Data Preparation

We begin by applying one-hot encoding to all categorical variables. This helps eliminate any implicit structural bias induced by their native ordering based representation. A side effect of this transformation is that the feature dimensionality increases, which impacts the size of the graph estimated in subsequent sections. For continuous variables, we normalize each to have zero mean and unit variance.

5.2 Graph Estimation

After preprocessing, we denote each input sample by the d-dimensional feature vector:

$$\mathbf{x}^{(j)} = [x_1^{(j)}, x_2^{(j)}, \ldots, x_d^{(j)}]^\top, \quad j = 1, \ldots, m \tag{8}$$

where m is the number of samples, and $x_i^{(j)}$ denotes the individual feature dimensions of the processed input. Each feature x_i corresponds to a node in graph, and edges represent statistical or causal dependencies between features.

We explore two primary paradigms for graph learning: *causality-based methods* and *association-based methods*. In the former, we assume a linear structural causal model given

by $\mathbf{x} = \mathbf{A}\mathbf{x} + \boldsymbol{\epsilon}$, where $\mathbf{A}$ is a weighted adjacency matrix representing causal relationships, and $\boldsymbol{\epsilon}$ is an independent noise vector. We apply algorithms such as LiNGAM and NOTEARS to learn this causal graph, resulting in DAGs.

As mentioned earlier, in association-based approaches, we can define the edge weight w_{ij} between nodes x_i and x_j as a function of their statistical dependency: $w_{ij} = \rho(x_i, x_j)$. Here, we choose ρ from Pearson correlation, Spearman rank correlation, or MI. For MI-based graph estimation, we employ the Chow-Liu algorithm to ensure the resulting structure remains a DAG.

5.3 Positional Encoding Creation

Given the estimated graph, we first symmetrize the adjacency matrix to produce an undirected version: $\mathbf{A}_{\text{sym}} = \frac{1}{2}(\mathbf{A} + \mathbf{A}^\top)$. We then compute the graph Laplacian: $\mathbf{L} = \mathbf{D} - \mathbf{A}_{\text{sym}}$, where $\mathbf{D}$ is the degree matrix. The top-k and bottom-k eigenvectors of $\mathbf{L}$ are selected (excluding the first eigenvector, which is constant valued), normalized to have zero mean and unit variance across nodes, and concatenated to form the PE matrix: $\mathbf{P} = [\mathbf{e}_2, \ldots, \mathbf{e}_{k+1}, \mathbf{e}_{d-k+1}, \ldots, \mathbf{e}_d]$. To modulate the influence of these encodings, we scale them using a hyperparameter α:

$$\mathbf{P}' = \alpha \cdot \mathbf{P}. \tag{9}$$

For categorical features with multiple one-hot encoded nodes, we average the individual encodings to yield a consolidated PE vector for the feature.

5.4 Positional Encoding Integration

In transformer-based architectures for tabular data, each feature undergoes tokenization to obtain an n-dimensional embedding vector. We integrate our estimated PEs $\mathbf{P}'$ by concatenating them with the corresponding feature embedding:

$$\mathbf{z}'_i = [\mathbf{z}_i; \mathbf{p}'_i] \in \mathbb{R}^{n+2k}, \tag{10}$$

where $\mathbf{z}_i$ is the original embedding for x_i and $\mathbf{p}'_i$ is the scaled PE for x_i. Subsequently, these modified embeddings serve as inputs to the self-attention layers during training.

6 Synthetic Experiments: Evaluating Structure-Sensitive Positional Encodings

In this section, we design synthetic experiments to evaluate whether the benefits of PEs are inherently tied to the presence of structural relationships within tabular data. Transformers rely on PEs to guide self-attention, yet in tabular domains, structural cues are typically absent. We hypothesize that when feature correlations exist, PEs derived from such associations can improve learning. Conversely, when features are independent, PEs may have limited impact. To explore this, we introduce a synthetic framework where the structure of the dataset can be controlled parametrically.

6.1 Definition of Structure

We define structure as the degree of association between features. If all features are independent, no meaningful pairwise relationships exist, and all features are equally unrelated.

Algorithm 1: Structure-Controlled Tabular Data Generation

Input: Feature dimension d, number of partitions k, number of samples n
Output: Synthetic dataset $X \in \mathbb{R}^{n \times d}$, output variable $y \in \mathbb{R}^n$
Group Assignment: Partition features into k disjoint groups $\{G_1, G_2, \ldots, G_k\}$
Sample fixed weights $w_f \sim \mathcal{U}(-1, 1)$ for all features $f \in \{1, \ldots, d\}$
Generative Process:
for each sample $t = 1$ to n **do**
  for each group $g = 1$ to k **do**
    Sample group latent variable $\theta_g^{(t)} \sim \mathcal{U}(-2, 2)$
    for each feature $f \in G_g$ **do**
      $x_f^{(t)} = \theta_g^{(t)} \cdot w_f + \varepsilon_f^{(t)}, \quad \varepsilon_f^{(t)} \sim \mathcal{N}(0, 0.01)$
    end for
  end for
  Target Outputs Generation:
  Select fixed group index $g^* \in \{1, \ldots, k\}$ (shared across all samples)
  Sample $w_t, b \sim \mathcal{U}(-1, 1)$
  Compute $y^{(t)} = w_t \cdot \theta_{g^*}^{(t)} + b$
end for

To simulate a controllable structure, we partition the feature space into k groups where intra-group features share latent correlations while inter-group features remain independent.

6.2 Data Generation Algorithm

Given input dimensionality d and partition count k, the synthetic data generation is detailed in Algorithm 1. We note that as k increases, most features end up in their own group, leading to independence and minimal structure. When k is lower, many features share common generative variables, increasing structure. We consider the scenario where the generated dataset embodies a regression problem, where the underlying function is a linear function of one of the partitions.

6.3 Experimental Setup and Results

We evaluate FT-Transformer on synthetic datasets with input dimensionality fixed at $d = 30$, using Spearman-based graph-derived PEs. Each feature embedding is concatenated with PE and scaled by a hyperparameter α to modulate its influence. To analyze the impact of structural variation, we partition features into k groups and categorize results into three regimes: (a) high structure ($k \leq 8$), (b) moderate structure ($10 \leq k \leq 22$), and (c) low structure ($k > 22$). Accuracy is assessed across varying α values and partition counts.

Results Figure 2 shows performance across structural regimes. As anticipated, datasets with stronger internal associations benefit more from graph-derived PEs, yielding larger improvements as the positional signal is amplified via higher α values. This empirically confirms that PEs are most useful when the data exhibits meaningful structure.

Interestingly, even in highly unstructured settings, we observe minor but consistent gains from PEs. This is because

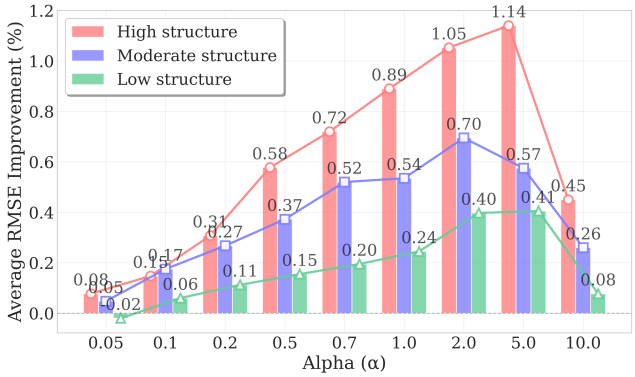

Figure 2: RMSE performance comparison with low, moderate, and high structure synthetic datasets across varying α.

PEs can reduce the effective rank of the learning problem, even when the inputs are unstructured (Theorem 1), and the generative structure is a simple linear function that doesn't require high-rank features.

Lastly, Figure 2 shows that excessively amplifying the contribution of PEs, by increasing α to 10, can degrade model performance. This is intuitive, as a large α disproportionately weights the positional signal, potentially overshadowing the original input content encoded within the value vectors of the query-key-value decomposition.

7 Experiment on Real Datasets

In this section, we discuss the results on real datasets. We present the details regarding our experimental setup, the main results, and ablation as well as analytical studies on Tab-PET. All experiments were conducted on 3 NVIDIA A100 SXM4 80GB GPUs and 3 NVIDIA RTX 6000 Ada Generation GPUs.

7.1 Experimental Setup

Datasets: We run experiments on a comprehensive collection of 50 tabular datasets sourced from OpenML (https://www.openml.org), comprising 25 classification and 25 regression tasks. These datasets span diverse domains with varying characteristics in terms of sample sizes, feature dimensionalities, and categorical ratios. The complete list of datasets with detailed properties is provided in technical appendix B.1. To preserve the statistical properties of each dataset, we employ stratified sampling for classification tasks that maintains the exact class distribution of the original dataset. During training, we split each dataset into a 60:20:20 train-validation-test split. We outline the detailed data preprocessing steps in technical appendix B.2.

Graph Estimation: We found that some graph estimation approaches are computationally expensive. NO-TEARS was particularly expensive for larger datasets. Association-based graphs use weights from pairwise measures in Eq. (1). Chow-Liu requires an additional step to ensure the graph is a DAG. The NO-TEARS and LinGAM approaches for graph estimation have tunable hyperparameters, which are outlined in technical appendix C.

PE Creation: When generating PEs, we design an automatic k selection algorithm that adaptively determines the optimal number of low-frequency and high-frequency eigenvectors based on spectral gap analysis. The k selection algorithm is based on thresholding the normalized eigenvalues, which are a proxy of the effective frequency of the eigenvector, from both sides. We outline this in technical appendix D. The hyperparameter α (Fig. 1 (c) and Eq. (9)), which is chosen from a set of 9 elements from 0.05 to 10, is optimized via the validation set using a greedy approach.

Training and Evaluation: Following numerous previous studies (Ye et al. 2024; Gorishniy et al. 2021) that use RMSE and accuracy, we report RMSE for regression tasks and balanced accuracy for classification tasks. Balanced accuracy presents an unbiased estimate of performance that is independent of class imbalance, and following best practices, we train our models using balanced cross-entropy loss. Each result is reported after averaging across five random seeds for all approaches tested in our work. We use early-stopping for all approaches during the training process. Details are provided in technical appendix E.

7.2 Baselines for Comparison

We compare Tab-PET with two categories of baselines:

- **Tree-based:** We compare Tab-PET against two state-of-the-art (SOTA) gradient boosting methods that show superior performance on tabular data: XGBoost (Chen and Guestrin 2016) and CatBoost (Prokhorenkova et al. 2018). For these methods, we use Optuna-driven hyperparameter optimization (Akiba et al. 2019) with 100 trials per dataset. Detailed hyperparameter spaces can be found in technical appendix F.1.
- **Transformer-based:** We compare Tab-PET against three SOTA transformer-based methods designed for tabular data: TabTransformer (Huang et al. 2020), SAINT (Somepalli et al. 2021), and FT-Transformer (Gorishniy et al. 2021). Lastly, for fair comparison, we keep batch size, epochs, learning rate, dimensionality of the feature tokenizer output, and other such hyperparameters fixed when comparing PE and non-PE approaches for each approach. Complete hyperparameter configurations and fair comparison details are provided in technical appendix F.2 and F.3.

7.3 Graph Estimation Approaches Analysis

Performance: To evaluate how different graph estimation approaches influence downstream performance, we compare five representative approaches on 50 datasets using FT-Transformer as the backbone. As shown in Table 1, association-based approaches consistently outperform causality-based ones across both tasks. Spearman correlation achieves the highest average improvement, followed closely by Pearson. Additionally, Spearman exhibits the most consistent positive gains, rarely showing performance degradation. In contrast, causal discovery approaches NOTEARS and LiNGAM exhibit relatively weaker performance improvement. Chow-Liu, which constructs tree-structured dependency graphs, doesn't perform well on classification. Detailed results for all approaches are provided in technical appendix G.1.

Method	CA	AS	DA	NL	TR	C Improv. ↑	R Improv. ↑	Time (min)
NOTEARS	✓		✓			1.36%	3.64%	76.83
LiNGAM	✓		✓			1.41%	3.97%	10.96
Pearson		✓				1.61%	4.16%	0.78
Spearman		✓		✓		**1.72%**	**4.34%**	0.79
Chow-Liu		✓	✓	✓	✓	1.17%	4.29%	**0.38**

Table 1: Average performance improvement of graph estimation approaches with Tab-PET. Results show improvement percentage over baseline. Time (min) represents the average extra computational time introduced by Tab-PET for graph estimation and PE creation. CA = Causal, AS = Association, DA = Directed Acyclic, NL = Nonlinear, TR = Tree.

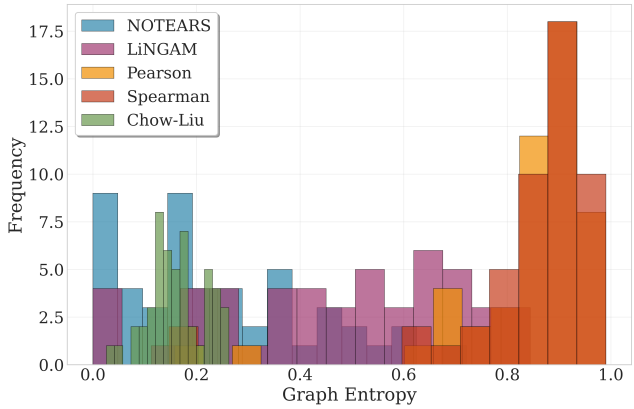

Figure 3: Graph entropy distributions across five graph estimation approaches. Varying bar widths reflect the different value ranges spanned by each approach.

Computational Cost: Addressing computational efficiency concerns, we report the average computational time (in minutes) for graph estimation and PE creation across all 50 datasets in Table 1. The introduced computational overhead is minimal; for instance, Spearman graphs add only 0.79 minutes on average. Detailed timing results are provided in technical appendix G.2.

Graph Entropy: To better understand why association-based approaches outperform causality-based ones, we analyze structural properties via graph entropy, a measure quantifying the uniformity of edge weight distributions. For a graph with n feature nodes, we compute the normalized entropy for each node i. Given the edge weights w_{ij} to other nodes, the normalized entropy H_i is defined as: $H_i = -\frac{\sum_{j \neq i} p_{ij} \ln p_{ij}}{\ln(n-1)}$, where $p_{ij} = \frac{w_{ij}}{\sum_{j \neq i} w_{ij}}$. The overall graph entropy is then obtained by averaging H_i across all nodes. Higher entropy indicates more uniform, densely connected graphs, while lower entropy suggests sparse, highly structured graphs with concentrated edge weights.

Figure 3 reveals a clear pattern between graph entropy and downstream performance. For all 50 datasets tested here, causal methods (NOTEARS and LiNGAM) concentrate in the low graph entropy region, producing sparse, highly constrained graphs. Spearman and Pearson correlations generate graphs with high entropy. These denser structures align with the strongest performance gains, suggesting that PEs yield more useful information when derived from dense feature dependencies rather than sparse causal structures. Additional causal–association comparison analyses are provided in technical appendix H.1.

7.4 Classification and Regression: Main Results

Based on Table 1, we select the best-performing approach (Spearman) for integrating PEs into transformer architectures. Table 2 compares performance across tree-based and transformer-based models on 50 datasets, with detailed results with standard deviations in technical appendix G.3. Metrics include balanced accuracy for classification and RMSE for regression, based on five-fold cross-validation. For TabTransformer, we use only datasets with multiple categorical variables, since TabTransformer's architecture only applies embeddings (and thus PEs) to categorical features, while continuous features bypass the embedding layer entirely. Overall, Tab-PET consistently improves performance across multiple transformer architectures and tasks.

To understand performance better, following common practice (Gorishniy et al. 2021; Gorishniy, Rubachev, and Babenko 2022; Gorishniy et al. 2023; McElfresh et al. 2023; Ye et al. 2025), we report mean ranks in Table 2 and find that Tab-PET methods achieve the best overall performance, surpassing GBDTs and baseline transformers in both classification and regression tasks, with Tab-PET variants of FT-Transformer and SAINT ranking as the top two methods overall. Notably, Tab-PET shows statistically significant improvements against no-PE baselines across all transformer architectures ($p < 0.05$, Wilcoxon signed-rank tests), with detailed p-values provided in technical appendix H.6.

7.5 Comparing with Learnable Positional Encodings

A key consideration in our study is whether fixed PEs, derived from the intrinsic structure of tabular data, outperform learnable PEs that adapt based on data input. This question echoes ongoing debates in NLP where it's widely accepted that learnable PEs can perform well when ample data is available, but may falter under low-data regimes (Radford et al. 2018).

To explore this in the tabular domain, we compare performance improvements between learnable PEs and those generated via Tab-PET across all classification and regression datasets. These results, detailed in Table 3, reveal that Tab-PET consistently achieves higher average gains than learnable alternatives. This suggests that for tabular datasets, which are typically smaller in size, incorporating graph-structured positional information as Tab-PET does provide a substantial advantage. Detailed results are provided in technical appendix G.4, while further analysis and ablation studies on Tab-PET are shown in technical appendix H.

Model	Rank	AU↑	GE↑	SA↑	BL↑	CHU↑	CM↑	CR↑	DI↑	DN↑	EY↑	FI↑	HE↑	JA↑	KC↑	KR↑	MA↑	PH↑	QSA↑	ST↑	SY↑	TI↑	VE↑	WI↑	WIN↑	YE↑
XGB	3.40	0.837‡	0.614	0.746	0.725	0.893	0.571†	0.722†	0.721	0.966†	0.696	0.524†	0.711	0.811‡	0.704	0.997†	0.861†	0.845	0.858†	0.806	0.934	0.978	0.759	0.914	0.396†	0.582†
CB	3.76	0.830	0.635	0.776	0.705	0.894	0.562	0.681	0.710	0.964‡	0.726	0.502‡	0.722	0.819†	0.719‡	0.992	0.856‡	0.860	0.842	0.810‡	0.953†	0.981†	0.747	0.930	0.273	0.567‡
TT	7.33	0.822	—	—	—	0.831	0.485	0.661	—	0.956	0.595	—	—	0.785	—	0.993	—	—	—	—	—	0.973	—	—	—	—
+PET	**5.33**	**0.836**	—	—	—	**0.840**	**0.488**	**0.683**	—	**0.960**	**0.598**	—	—	**0.791**	—	**0.994**	—	—	—	—	—	**0.980**‡	—	—	—	—
ST	4.52	**0.826**	**0.663**†	0.814‡	0.733	0.855	0.557	**0.709**‡	0.730	0.961	0.687	0.471	0.725	0.797	0.715	0.992	**0.809**	0.852	0.841	0.798	0.940	0.969	0.762	0.977‡	0.350	0.545
+PET	**3.28**‡	0.826	0.660‡	**0.862**†	**0.756**‡	**0.878**	**0.563**	0.709	**0.740**	**0.964**	**0.730**	**0.474**	**0.728**‡	**0.807**	**0.718**	**0.995**	0.805	**0.854**	**0.854**‡	**0.805**	**0.943**	**0.973**	**0.785**	**0.982**†	**0.351**	**0.551**
FT	4.44	0.828	0.641	0.779	0.748	0.908‡	0.545	0.644	0.769‡	0.953	0.748‡	0.460	0.724	0.805	0.714	**0.996**‡	0.718	0.861‡	0.831	0.795	0.947	0.972	0.786‡	0.957	0.363	0.545
+PET	**2.44**†	**0.838**†	**0.650**	**0.792**	**0.758**†	**0.917**†	**0.571**‡	**0.682**	**0.771**†	**0.962**	**0.750**†	**0.470**	**0.730**†	**0.806**	**0.731**†	0.995	**0.740**	**0.866**†	**0.844**	**0.811**†	**0.948**‡	**0.977**	**0.808**†	**0.970**	**0.376**‡	**0.562**

Model	Rank	QS↓	AB↓	AI↓	BO↓	BOS↓	CA↓	CH↓	CL↓	CO↓	CP↓	CPU↓	DIA↓	EN↓	FR↓	GR↓	KI↓	LI↓	MU↓	PL↓	SE↓	SO↓	SP↓	STO↓	TE↓	WIS↓
XGB	5.20	1.047	2.333	2.247	3.424	3.545	0.169	0.593	0.300‡	5.050	6.367	3.112	1157	0.476	0.980	0.015	0.153	2.858†	30.15	238.0	0.646†	24.62	0.169	1.811	6.530	39.75
CB	2.96	1.000	2.243	1.445‡	0.587	3.159	0.133†	0.540‡	0.272†	4.305†	2.288†	2.713†	529.4†	0.433†	0.987	0.007	0.093	3.011	10.87‡	227.5	0.676	24.22	0.108	0.689‡	1.544†	36.69
TT	7.14	—	—	—	—	5.203	—	—	1.369	—	—	—	1039	—	—	—	—	—	153.5	227.9	0.801	28.83	—	—	—	—
+PET	**5.71**	—	—	—	—	**5.173**	—	—	**1.327**	—	—	—	**1025**	—	—	—	—	—	**153.0**	**226.9**	**0.713**	**28.08**	—	—	—	—
ST	3.64	0.939‡	2.187	**1.957**	1.060	3.194	**0.145**	0.575	0.491	5.169	2.339	**2.791**‡	537.5	**0.547**	**0.947**	0.006	0.067	**2.940**	12.40	224.6‡	0.717	17.53‡	**0.100**†	0.994	2.485	35.38‡
+PET	**2.84**†	**0.932**†	**2.160**†	2.005	**1.039**	**3.090**‡	0.146	**0.547**	**0.376**	**4.984**‡	**2.332**‡	2.807	**535.5**‡	0.550	0.950	**0.006**‡	**0.064**†	2.940	**9.717**†	**224.1**†	**0.715**	**17.51**†	0.100†	**0.989**	**1.818**‡	**35.20**†
FT	4.08	0.971	2.200	1.466	0.327‡	3.232	**0.141**‡	**0.540**†	0.484	5.079	2.358	2.856	2468	0.484	0.739‡	0.006	0.070	2.936	25.85	592.0	0.691	17.85	0.106	0.695	**4.113**	38.01
+PET	**2.88**‡	**0.961**	**2.168**‡	**1.310**†	**0.277**†	**2.993**†	0.141	0.540	**0.315**	**5.045**	**2.339**	**2.827**	**2447**	**0.472**‡	**0.716**†	**0.005**†	**0.067**‡	**2.880**‡	**22.98**	**591.4**	**0.666**‡	**17.83**	**0.104**	**0.676**†	4.123	**37.92**

Table 2: Performance comparison on classification and regression datasets. Each result is averaged over 5 random seeds. ↑ indicates balanced accuracy, ↓ indicates RMSE. Rank = the mean rank across all datasets (lower is better). **Bold** = better between baseline transformer and its Tab-PET variant. † = best across all methods, ‡ = second-best across all methods. Models: XGB = XGBoost, CB = CatBoost, TT = TabTransformer, ST = SAINT, FT = FT-Transformer, **+PET** = corresponding Tab-PET variant. The dataset names corresponding to the abbreviations are provided in technical appendix B.1.

Method	Avg Improv.%		Median Improv.%		Min Improv.%		Win Rate%	
	C	R	C	R	C	R	C	R
Learnable	0.04	0.62	-0.08	0.05	-3.1	-8.6	12	8
Tab-PET	**1.72**	**4.34**	**1.39**	**1.92**	**-0.1**	**-0.2**	**88**	**92**

Table 3: Comparison between Tab-PET and Learnable PE. C = Classification, R = Regression.

7.6 PEs and Effective Rank on Real Datasets

To empirically validate our theoretical results, we conduct experiments across 15 real-world tabular datasets where we measure the effective rank of features, comparing three conditions: (1) *Baseline* without PEs ($\alpha = 0$); (2) *Tab-PET* with graph-derived PEs; (3) *Random PE* with the same dimensionality and statistical properties.

Experimental Setup: We use an FT-Transformer architecture with a single layer and single attention head to isolate the effect of PEs on effective rank. For each PE type, we vary the scaling parameter $\alpha \in \{1, 2, 3, \ldots, 30\}$ and compute the effective rank of CLS token embeddings before the final prediction via fully connected layers. Complete experimental details are provided in technical appendix A.1.

Observations: Figure 4 presents our main findings, which strongly align with Theorems 1 and 2. Specifically, as α increases we see: (1) Both Tab-PET and random PE reduce effective rank compared to baseline across all α values, confirming that PEs enable the architecture to lower representation complexity when needed; (2) Tab-PET's effective rank is significantly lower than random PE, with the gap widening as α increases initially, before converging again; (3) The exponential-like decay in effective rank aligns with our theoretical takeaways. Further analysis is provided in technical appendix A.2.

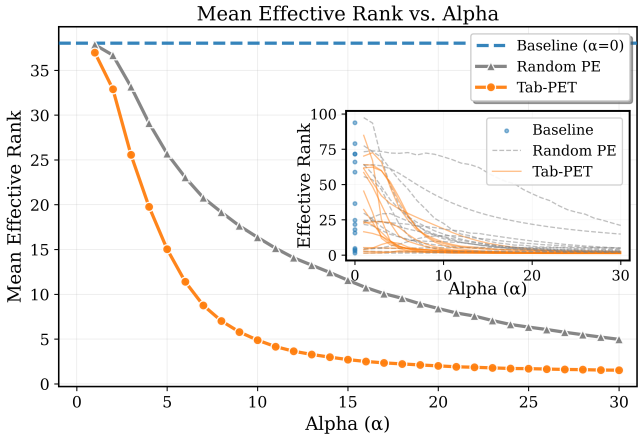

Figure 4: Empirical validation of effective rank reduction. Main plot shows the mean effective rank vs. α for Tab-PET, Random PE, and baseline. Inset shows per-dataset trends.

8 Conclusion

PEs, traditionally overlooked in tabular learning, can greatly enhance transformer performance by incorporating graph-derived structural priors. Evaluation on 50 datasets across multiple baselines establishes that Tab-PET (particularly Spearman graph estimation) extensions show best performance overall when compared with gradient boosting approaches and the non-PE baselines. Our theoretical analysis confirms that PEs can reduce effective rank of embeddings, and even more so when meaningful structure exists. Our synthetic studies highlight the significant performance improvements when the data has underlying structure. Overall, our work highlights the importance of actively incorporating structural inductive biases in tabular data learning.

Acknowledgments

This research is supported by A*STAR, CISCO Systems (USA) Pte. Ltd and the National University of Singapore under its Cisco-NUS Accelerated Digital Economy Corporate Laboratory (Award I21001E0002).

References

Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; and Koyama, M. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2623–2631.

Arora, S.; Cohen, N.; Hu, W.; and Luo, Y. 2019. Implicit regularization in deep matrix factorization. *Advances in Neural Information Processing Systems*, 32.

Cantürk, S.; Liu, R.; Lapointe-Gagné, O.; Létourneau, V.; Wolf, G.; Beaini, D.; and Rampášek, L. 2023. Graph positional and structural encoder. *arXiv preprint arXiv:2307.07107*.

Chen, T.; and Guestrin, C. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM sigkdd international conference on knowledge discovery and data mining*, 785–794.

Chow, C.; and Liu, C. 1968. Approximating discrete probability distributions with dependence trees. *IEEE transactions on Information Theory*, 14(3): 462–467.

Chu, X.; Zhang, B.; Tian, Z.; Wei, X.; and Xia, H. 2021. Do we really need explicit position encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 3(8).

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Dwivedi, V. P.; and Bresson, X. 2020. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*.

Ghosh, R.; and Motani, M. 2023. Local intrinsic dimensional entropy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 7714–7721.

Gong, Y.; Chung, Y.-A.; and Glass, J. 2021. Ast: Audio spectrogram transformer. *arXiv preprint arXiv:2104.01778*.

Gorishniy, Y.; Rubachev, I.; and Babenko, A. 2022. On embeddings for numerical features in tabular deep learning. *Advances in Neural Information Processing Systems*, 35: 24991–25004.

Gorishniy, Y.; Rubachev, I.; Kartashev, N.; Shlenskii, D.; Kotelnikov, A.; and Babenko, A. 2023. Tabr: Tabular deep learning meets nearest neighbors in 2023. *arXiv preprint arXiv:2307.14338*.

Gorishniy, Y.; Rubachev, I.; Khrulkov, V.; and Babenko, A. 2021. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34: 18932–18943.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Identity mappings in deep residual networks. In *European conference on computer vision*, 630–645. Springer.

Herzig, J.; Nowak, P. K.; Müller, T.; Piccinno, F.; and Eisenschlos, J. M. 2020. TaPas: Weakly supervised table parsing via pre-training. *arXiv preprint arXiv:2004.02349*.

Huang, X.; Khetan, A.; Cvitkovic, M.; and Karnin, Z. 2020. Tabtransformer: Tabular data modeling using contextual embeddings. *arXiv preprint arXiv:2012.06678*.

Huh, M.; Mobahi, H.; Zhang, R.; Cheung, B.; Agrawal, P.; and Isola, P. 2021. The low-rank simplicity bias in deep networks. *arXiv preprint arXiv:2103.10427*.

Ito, M.; Zhu, J.; Chen, D.; Koutra, D.; and Wiens, J. 2025. Learning laplacian positional encodings for heterophilous graphs. *arXiv preprint arXiv:2504.20430*.

Kadra, A.; Lindauer, M.; Hutter, F.; and Grabocka, J. 2021. Well-tuned simple nets excel on tabular datasets. *Advances in Neural Information Processing Systems*, 34: 23928–23941.

Klambauer, G.; Unterthiner, T.; Mayr, A.; and Hochreiter, S. 2017. Self-normalizing neural networks. *Advances in Neural Information Processing Systems*, 30.

Lee, H.-C.; Danieletto, M.; Miotto, R.; Cherng, S. T.; and Dudley, J. T. 2019. Scaling structural learning with NOBEARS to infer causal transcriptome networks. In *Pacific Symposium on Biocomputing 2020*, 391–402. World Scientific.

Leng, Y.; Ghosh, R.; and Motani, M. 2025. Tab-PET: Graph-Based Positional Encoding for Tabular Transformers. *https://github.com/kentridgeai/Tab-PET/blob/main/Tab-PET-Arxiv.pdf?raw=true*.

McElfresh, D.; Khandagale, S.; Valverde, J.; Prasad C, V.; Ramakrishnan, G.; Goldblum, M.; and White, C. 2023. When do neural nets outperform boosted trees on tabular data? *Advances in Neural Information Processing Systems*, 36: 76336–76369.

Nakada, R.; and Imaizumi, M. 2020. Adaptive approximation and generalization of deep neural network with intrinsic dimensionality. *J. Mach. Learn. Res.*, 21(1).

Popov, S.; Morozov, S.; and Babenko, A. 2019. Neural oblivious decision ensembles for deep learning on tabular data. *arXiv preprint arXiv:1909.06312*.

Prokhorenkova, L.; Gusev, G.; Vorobev, A.; Dorogush, A. V.; and Gulin, A. 2018. CatBoost: unbiased boosting with categorical features. *Advances in Neural Information Processing Systems*, 31.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving Language Understanding by Generative Pre-Training. *OpenAI Technical Report*.

Roy, O.; and Vetterli, M. 2007. The effective rank of a matrix: A measure of effective dimensionality. *European Signal Processing Conference*, 606–610.

Shavitt, I.; and Segal, E. 2018. Regularization learning networks: deep learning for tabular datasets. *Advances in Neural Information Processing Systems*, 31.

Shimizu, S.; Hoyer, P. O.; Hyvärinen, A.; Kerminen, A.; and Jordan, M. 2006. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7(10).

Somepalli, G.; Goldblum, M.; Schwarzschild, A.; Bruss, C. B.; and Goldstein, T. 2021. Saint: Improved neural networks for tabular data via row attention and contrastive pretraining. *arXiv preprint arXiv:2106.01342*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

Xu, Y.; Zhang, Q.; Zhang, J.; and Tao, D. 2021. Vitae: Vision transformer advanced by exploring intrinsic inductive bias. *Advances in Neural Information Processing Systems*, 34: 28522–28535.

Ye, H.; Fan, W.; Song, X.; Zheng, S.; Zhao, H.; Guo, D.; and Chang, Y. 2024. Ptarl: Prototype-based tabular representation learning via space calibration. *arXiv preprint arXiv:2407.05364*.

Ye, J.; Tan, Z.; Hu, Y.; Yang, X.; Cheng, G.; and Huang, K. 2025. Disentangling Tabular Data Towards Better One-Class Anomaly Detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 13061–13068.

Zheng, X.; Aragam, B.; Ravikumar, P. K.; and Xing, E. P. 2018. Dags with no tears: Continuous optimization for structure learning. *Advances in Neural Information Processing Systems*, 31.