

LimiX: Unleashing Structured-Data Modeling Capability for Generalist Intelligence

LimiX Team

Stable AI & Tsinghua University

-  <https://github.com/limix-ldm/LimiX/>
-  <https://huggingface.co/stableai-org/>
-  <https://modelscope.cn/organization/stable-ai/>

Abstract

We argue that progress toward general intelligence requires complementary foundation models grounded in language, the physical world, and structured data. This report presents LimiX-16M and LimiX-2M, two instantiations of our large structured-data models (LDMs). Both models treat structured data as a joint distribution over variables and missingness, thus capable of addressing a wide range of tabular tasks through query-based conditional prediction via a single model. They are pretrained using masked joint-distribution modeling with an episodic, context-conditional objective, supporting rapid, training-free adaptation at inference. We evaluate LimiX models across 11 large structured-data benchmarks with broad regimes of sample size, feature dimensionality, class number, categorical-to-numerical feature ratio, missingness and sample-to-feature ratios. LimiX-16M consistently surpasses strong baselines, as shown in [Figure 1](#) and [Figure 2](#). The superiority holds across a wide range of tasks, such as classification, regression, missing value imputation, and data generation, often by substantial margins, while avoiding task-specific architectures or bespoke training per task. Notably, LimiX-2M delivers strong results under tight compute and memory budgets. We also present the first scaling law study for LDMs, revealing how data and model scaling jointly influence downstream performance and offering quantitative guidance for tabular foundation modeling. All LimiX models are publicly accessible under Apache 2.0.

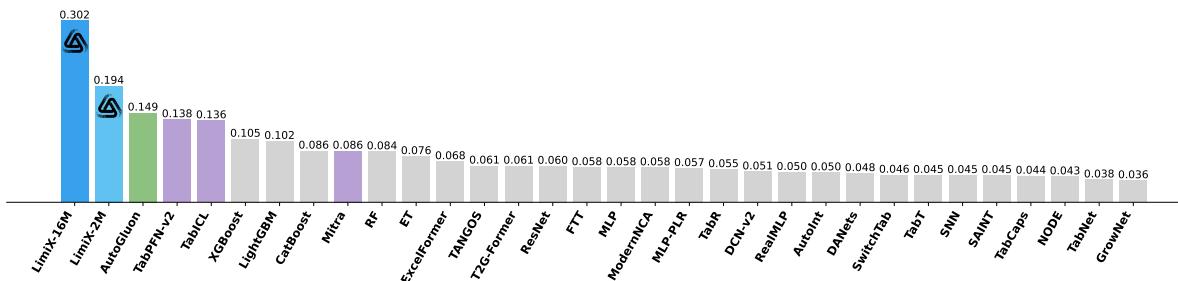


Figure 1: Performance comparison on the averaged reciprocal of the ranks, where the rank is that of the corresponding model on ROC AUC. Higher values indicate stronger average ranking performance across all the classification benchmarks.

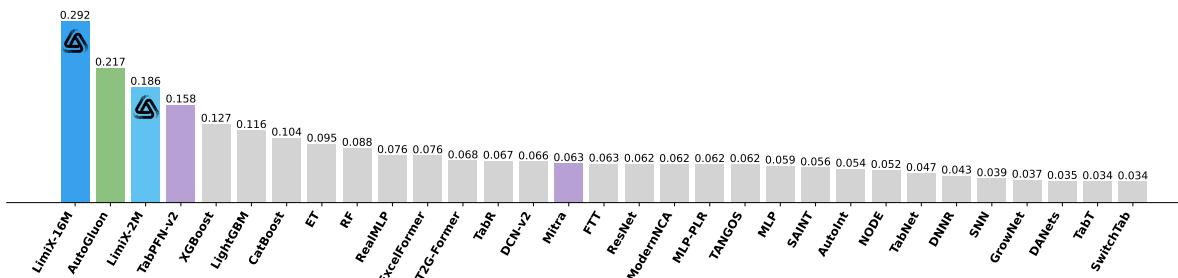


Figure 2: Performance comparison on the averaged reciprocal of the ranks, where the rank is that of the corresponding model on R^2 across all the regression benchmarks.

Contents

| | | |
|-----------|---|-----------|
| 1 | Introduction | 3 |
| 2 | Architecture | 4 |
| 2.1 | Embedding of Tabular Data | 4 |
| 2.2 | Discriminative Feature Encoding | 4 |
| 2.3 | Model Structure | 5 |
| 3 | Pretraining | 5 |
| 3.1 | Context-Conditional Masked Modeling for Joint Distribution Learning | 5 |
| 3.2 | Mask Pattern Design | 6 |
| 3.3 | Mask Embedding | 6 |
| 4 | Pretraining Data Generation | 6 |
| 4.1 | DAG Generation based on Hierarchical SCMs | 7 |
| 4.2 | Data Sampling | 8 |
| 4.3 | Task Adaptation | 8 |
| 5 | Retrieval-based Ensemble | 8 |
| 6 | Theoretical Analysis on Context-Conditional Masked Modeling | 10 |
| 6.1 | Modeling the Joint Conditional with Random Masks | 10 |
| 6.2 | The Choice of Mask Number | 11 |
| 7 | Evaluation | 11 |
| 7.1 | Classification | 11 |
| 7.2 | Regression | 26 |
| 7.3 | Missing Value Imputation | 36 |
| 7.4 | Robustness | 37 |
| 7.5 | Embedding | 38 |
| 7.6 | Fine-tuning | 39 |
| 7.7 | Data Generation | 42 |
| 7.8 | Out-of-Distribution Generalization | 43 |
| 8 | Scaling law | 43 |
| 8.1 | Scaling with Dataset Size and Model Parameters | 44 |
| 8.2 | Conclusions and Insights | 45 |
| 9 | Conclusion | 46 |
| 10 | Contribution | 47 |
| A | Experimental Details | 54 |
| A.1 | Details of Datasets with Distribution Shifts | 54 |
| A.2 | Hyperparameter Search Space for Baselines | 54 |
| B | Omitted Details in Section 6 | 55 |
| B.1 | Omitted Details in Section 6.1 | 55 |
| B.2 | Omitted Details in Section 6.2 | 56 |

1 Introduction

We posit that progress toward general intelligence is best organized around three complementary spaces: language, physical-world, and structured data, each anchored to a distinct data modality and set of inductive biases. In the language space, large language models (LLMs) provide a universal interface for natural and programming languages and have rapidly advanced instruction following, tool use, and explicit reasoning over token sequences (Achiam et al., 2023; Touvron et al., 2023; Team et al., 2023; Bai et al., 2023). In the physical-world space, recent foundation models ground knowledge in space perception and embodied reasoning. They emphasize spatial intelligence through structured scene understanding, controllable scene generation, and neural radiance field reconstruction (Mildenhall et al., 2020; Ke et al., 2025; Xiang et al., 2025; Li et al., 2024a; Yi et al., 2024). These models also include self-supervised video world models such as V-JEPA, which learn predictive abstractions from large-scale videos and support downstream planning and control (Bardes et al., 2024; Assran et al., 2025).

On the other hand, structured data serves as the foundational bedrock for evidence-based decision-making across a multitude of critical domains, including finance, healthcare, logistics, and public policy (Fuster et al., 2022; Johnson et al., 2016; Yu et al., 2021; Krafft et al., 2020). The structural consistency and inherent order of structured data provide a robust framework for quantitative analysis and reliable operations (Ramakrishnan et al., 2003; Silberschatz et al., 2011; Stonebraker & Çetintemel, 2018), enabling precise prediction, automated reasoning, and rigorous causal inference (Pearl, 2009; Hernán & Robins, 2010; Little & Rubin, 2019). While the emergence of unstructured data has captured considerable attention, the analytical power and operational reliability of structured data remain unparalleled for a vast array of real-world applications (Fang et al., 2024; Van Breugel & Van Der Schaar, 2024). Consequently, advancements in structured-data prediction, analysis, and reasoning are not merely an academic pursuit but a critical enabler for efficiency, innovation, and accuracy in modern data-driven systems. Structured data is not subsumed by language models or embodied intelligence. Converting tables into free text discards metric geometry, physical units, and patterns of missingness that are central to reliable prediction, while models focusing on perception and control in three-dimensional physical environments do not capture discrete interventions, business rules, or causal heterogeneity across environments. Empirical surveys also document current limitations of language models on tabular prediction without bespoke adaptation (Fang et al., 2024; Sui et al., 2024).

Traditionally, practitioners deploy pipelines of specialized models with gradient-boosting trees and automated ensembles such as XGBoost (Chen & Guestrin, 2016), LightGBM (Ke et al., 2017), CatBoost (Dorogush et al., 2018), and AutoGluon (Erickson et al., 2020)—that are trained separately for each dataset and task. These systems excel at supervised prediction but require full retraining on every new dataset, prolonging deployment and preventing reuse of knowledge across domains. Recent deep approaches for tables have improved accuracy on mixed-type data (Huang et al., 2020; Yoon et al., 2020; Gorishniy et al., 2021; Somepalli et al., 2022; Bahri et al., 2022). However, they are still typically trained per dataset and do not provide a single model that transfers across objectives and constraints.

These limitations motivate the pursuit of a foundation model for structured data, i.e. models trained on large families of datasets to perform in-context learning (ICL) without per-task fine-tuning. Notably, TabPFN (Hollmann et al., 2022) and its successor TabPFN-v2 (Hollmann et al., 2025) demonstrate state-of-the-art performance and speed on small-to-medium-scale tables via prior-data fitting over diverse generative processes, and community efforts increasingly argue for tabular foundation models as a distinct paradigm (Van Breugel & Van Der Schaar, 2024). Concurrent works such as TabDPT (Ma et al., 2024) and TabICL (Qu et al., 2025) explore scaling these ideas to real and larger datasets, narrowing the gap between tabular foundation models and classical methods under the scenarios of larger sample sizes.

Despite the progress, these are still in the early stages of foundation model development and remain limited in generality and performance. Most models are developed and evaluated primarily for supervised prediction (classification or regression), and typically require per-task models, adapters, or external pipelines to address other objectives. In practice, one still needs to assemble separate components for classification, regression, missing value imputation, data generation, and sample selection for interpretability, with different training losses and hyperparameters, so the resulting system is not a single reusable learner that delivers all of these capabilities end-to-end while maintaining reliable performance. This gap motivates a large structured-data model (LDM) that treats structured data as a joint distribution over variables and missingness, enabling multiple tasks to be posed as queries to one model.

In this work, we present LimiX, a unified family of our LDM series, and release its first two variants, LimiX-16M and LimiX-2M. LimiX models aim to push generality further: a single model capable of classification, regression, missing value imputation, data generation, and sample selection for interpretability under one training and inference recipe, shifting the paradigm from bespoke pipelines to unified, foundation-style tabular learning. LimiX models adopt a lightweight, scalable architecture that represents structured data as a set of sample-feature embeddings and learns dependencies across two dimensions: across features

(columns) and across samples (rows). To make the attention module explicitly column-aware without inflating parameters, we introduce a low-rank discriminative feature encoding that encodes feature identities. Pretraining of LimiX models follows a masked joint-distribution objective and an episodic, context-conditional formulation: For each dataset, an in-context subset establishes dataset-specific priors, and the model is trained to predict masked entries in a disjoint query subset, enabling per-dataset adaptation without fine-tuning at inference. The pretraining corpus consists of data synthesized from hierarchical structural causal models (SCMs). Within the synthesis pipeline, we employ graph-aware sampling to obey the causal structure and solvability-aware sampling to accommodate the data quantity of various downstream tasks, improving coverage of local patterns and generalization. At inference, attention-guided retrieval provides an efficient, optional ensemble and fine-tuning mechanism. LimiX models retrieve informative samples and features using their own attention scores, aggregate predictions across a handful of lightweight pipelines, and deliver calibrated outputs for various downstream tasks, all through a unified conditional-inference interface and without task-specific architectures or bespoke per-dataset training.

We evaluate LimiX-16M and LimiX-2M across 11 large structured-data benchmarks with broad regimes of sample size, feature dimensionality, class number, categorical-to-numerical feature ratio, missingness and sample-to-feature ratios. With a single model and a unified inference interface, LimiX-16M surpasses competitive baselines including gradient-boosting trees, deep tabular networks, recent tabular foundation models, and automated ensemble methods. Across classification, regression, missing value imputation, data generation, and out-of-distribution prediction, LimiX-16M delivers consistent gains, often by large margins, while avoiding task-specific model architectures, customized ensembles, or per-dataset training. Notably, LimiX-2M deliver strong results under tight compute and memory budgets.

On most benchmarks, such as OpenML-CC18 (Bischl et al., 2017), TabArena (Erickson et al., 2025), TALENT (Liu et al., 2024), LimiX-16M is the only model that consistently outperforms AutoGluon, which is considered an outstanding baseline across various tabular-data tasks.

This work introduces the first scaling law for LDMs, providing quantitative insights into how data and model scaling shape downstream performance and inform principled design of tabular foundation models.

2 Architecture

We consider a dataset $\mathcal{D} = \{(\mathbf{x}^R, \mathbf{y}^R)\}$ of d features and an outcome variable, where $\mathbf{x}^R = \{\mathbf{x}_i^R\}_{i=1}^m$ and $\mathbf{y}^R = \{\mathbf{y}_i^R\}_{i=1}^m$; the superscript R indicates the raw input. Here, $\mathbf{x}_i^R \in \mathbb{R}^d$ and $\mathbf{y}_i^R \in \mathbb{R}$ correspond to the i^{th} sample, while $\mathbf{x}^R \in \mathbb{R}^{m \times d}$ and $\mathbf{y}^R \in \mathbb{R}^m$ correspond to the 2D tabular data. For in-context samples and test samples, we use subscripts to distinguish between them. For example, $\mathbf{x}_{ct}^R \in \mathbb{R}^{m_{ct} \times d}$ denotes the features of in-context samples, while $\mathbf{x}_{te}^R \in \mathbb{R}^{m_{te} \times d}$ denotes those of the test samples.

2.1 Embedding of Tabular Data

To ensure compatibility with modern architectures such as Transformers (Vaswani et al., 2017), each cell of the 2D tabular input $x_{i,j}^R \in \mathbb{R}$ is first projected into a latent embedding space \mathbb{R}^p . Specifically, $\mathbf{x}^R \in \mathbb{R}^{m \times d}$ is transformed into $\mathbf{x} \in \mathbb{R}^{m \times d \times p}$ and $\mathbf{y}^R \in \mathbb{R}^m$ is transformed into $\mathbf{y} \in \mathbb{R}^{m \times p}$. All subsequent attention operations are then conducted within this latent embedding space. Such a high-dimensional embedding space could strengthen the expressivity of the model. Concretely, we employ a two-layer MLP with LayerNorm (Ba et al., 2016) as the embedding module, i.e. $\mathbf{x}_{i,j} = \sigma(LN(\sigma(LN(\mathbf{x}_{i,j}^R \mathbf{W}^{(1)} + \mathbf{b}^{(1)})) \mathbf{W}^{(2)} + \mathbf{b}^{(2)}))$, where LN is LayerNorm and σ is the GELU activation function (Hendrycks & Gimpel, 2016). Separate embedding modules are used for \mathbf{x}^R and \mathbf{y}^R .

2.2 Discriminative Feature Encoding

The attention score between feature j and j' for sample i is $\frac{1}{\sqrt{p}} \langle \mathbf{x}_{i,j} \mathbf{W}_Q, \mathbf{x}_{i,j'} \mathbf{W}_K \rangle$. This score depends solely on the interactions between the embeddings of cell values and imposes no explicit prior on feature (column) identity. As a result, the model cannot reliably infer the column from which a cell value originates. Inspired by Hollmann et al. (2025), we introduce a learnable low-rank column identifier termed discriminative feature encoding (DFE) whose design philosophy includes two objectives: (i) Encodings of different features should be well separated to ensure discriminability; (ii) Encodings in the embedding space admit a low effective rank so that the model can express column identities compactly and share statistical strength across features.

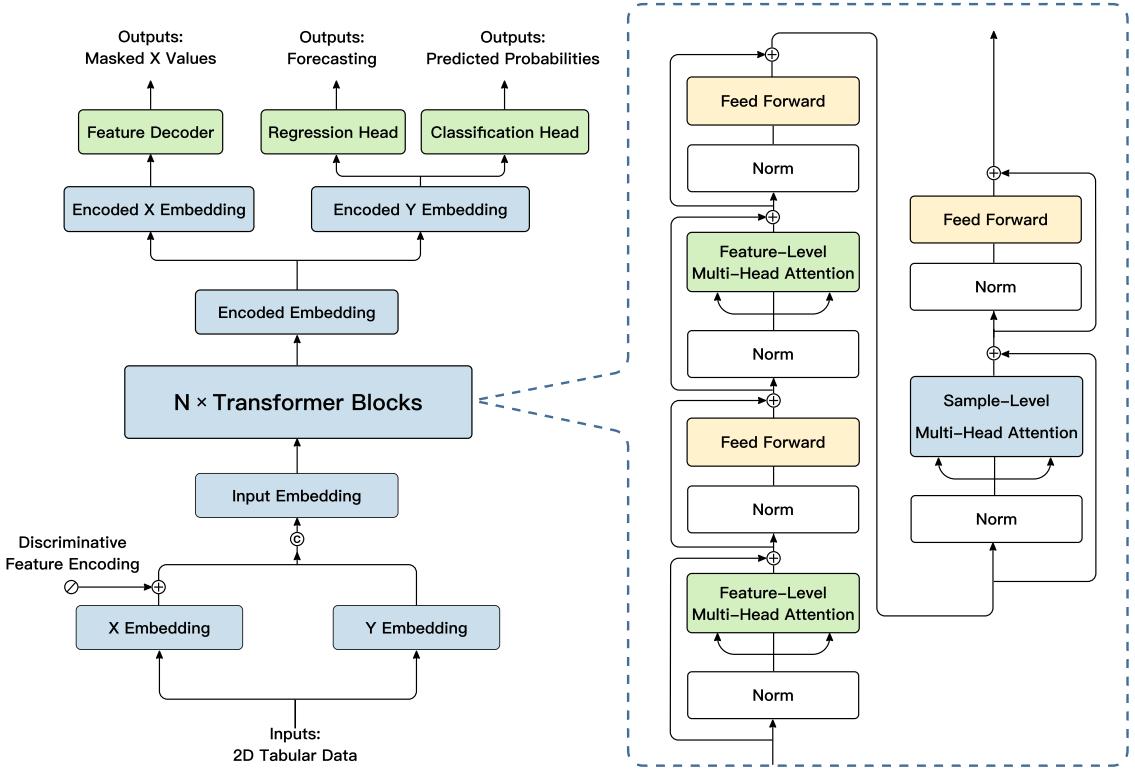


Figure 3: The overall model structure of LimiX-16M.

For implementation, let s denote the rank of DFE, and $s \ll p$ compared with the dimension of the embedding space p . We initialize a matrix $\mathbf{u} \in \mathbb{R}^{d \times s}$ whose j^{th} row vector $\mathbf{u}_j \in \mathbb{R}^s$ is a low-dimensional code of the column identifier corresponding to feature (column) j . Rows are initialized to be approximately orthogonal and normalized. Then a linear transformation $\mathbf{E} \in \mathbb{R}^{s \times p}$ lifts the code from the low-dimensional code space to the high-dimensional embedding space, i.e. $\mathbf{e}_j = \mathbf{u}_j \mathbf{E} \in \mathbb{R}^p$, serving as the DFE for feature j . Finally, the embedding of cell (i, j) is augmented additively as $\tilde{\mathbf{x}}_{i,j} = \mathbf{x}_{i,j} + \mathbf{e}_j$, which is analogous to absolute positional encodings but is applied along the feature axis. Empirically, we set $s = p/4$ by default.

2.3 Model Structure

As shown in Figure 3, LimiX models comprise 12 transformer blocks. Each block performs axis-wise self-attention along the feature and sample axes, incorporating position-wise feed-forward networks (FFNs). In LimiX-16M, we employ an asymmetric configuration with two feature-level passes and one sample-level pass, each followed by an FFN, as ablation studies demonstrate that equal numbers of feature and sample passes underrepresent feature interactions, while increased feature-axis attention enhances modeling capacity for heterogeneous schemas with negligible overhead. All sublayers use pre-normalized LayerNorm (Ba et al., 2016) to stabilize optimization and support deeper scaling.

3 Pretraining

3.1 Context-Conditional Masked Modeling for Joint Distribution Learning

We pretrain LimiX models by randomly masking cells in each row and training the model to recover the hidden entries from the visible context. Exposing the model to various mask patterns forces it to master a wide spectrum of conditional dependencies among variables. When these conditionals are learned properly, they effectively define a single joint model of the data. This joint model can then be queried for diverse tasks with one mechanism: Treat a chosen column as the target to perform regression or classification, fill in missing values by predicting the masked cells from the observed ones, and generate

new samples by iteratively masking and refilling subsets of features.

To better align pretraining with inference, we adopt a Context-Conditional Masked Modeling (CCMM) objective. For each dataset, an episode splits rows into a context subset and a query subset. The model encodes the context and conditions predictions for the query rows on this context through attention or feature modulation, learning to answer queries of the form “predict masked entries in the query given the observed query features and the context”. This episodic formulation enables rapid and label-free adaptation to new datasets at inference time: A handful of context rows establish dataset-specific priors such as category frequencies, marginal scales, and cross-feature couplings, while a single parametric model serves all schemas and tasks.

Unlike BERT-style masked modeling (Devlin et al., 2019), where adaptation is implicit in parameters accumulated during pretraining, CCMM considers context as a non-parametric memory that the model can consult during inference. This yields per-dataset calibration, better handling of rare categories, and improved robustness to distribution shift without gradient updates, aligning with principles of in-context learning and meta-learning for mixed-type tabular data.

We also find that CCMM improves modeling of the underlying dependency structure compared with recent tabular foundation models (Hollmann et al., 2025; Qu et al., 2025). By demanding consistency across numerous conditional predictions of all features rather than optimizing the loss of a single conditional prediction of a prefixed feature, the model is compelled to capture stable variable–variable relations instead of brittle decision boundaries. As the coverage of masks becomes richer, these cross-conditional constraints get tightened, yielding more reliable recovery of the joint distribution of all the features and more stable estimates in downstream usages. Please refer to Section 6 for details.

3.2 Mask Pattern Design

We employ a heterogeneous schedule that interleaves cell-wise, column-wise, and block masks, enabling the model to practise recovering both isolated entries and coordinated subsets of variables. Cell-wise masks refine local conditional predictions; column-wise masks force the model to treat an entire feature as missing and infer it from the remaining attributes; block masks target higher-order dependencies across semantically related groups (e.g., demographics with outcomes, laboratory panels with diagnoses). Masking rates are stratified by variable type, prevalence, and dispersion to avoid overfitting to common categories and to limit the influence of high-variance continuous features, and we exclude degenerate patterns that remove nearly all informative context. This diversified schedule provides broad coverage of conditional relationships and prevents the model from specializing to a narrow reconstruction regime, yielding a more faithful approximation of the joint distribution. In practice, we sample the mask ratio in [0.1, 0.4] for LimiX models.

3.3 Mask Embedding

To model masked cells, we introduce learnable mask embeddings that explicitly mark positions to be predicted. For each masked entry, its embedding is replaced by a trainable mask vector that is combined with the column embedding, so the encoder can condition on what is missing as well as where. We align the training masks introduced by the objective with naturally missing/structurally unavailable values observed in real data by using the shared mask embedding and calibrating the masking schedule to match empirical missingness patterns, thereby minimizing distribution shift between synthetic and real scenarios.

The mask embeddings flow through the same attention blocks as observed cells, enabling the model to request information from relevant columns and to produce calibrated distributional predictions at the output head. To reduce the mismatch between pretraining and fine-tuning, we condition the network on the corruption level of each dataset using a mask density feature, defined as the proportion of cells that are masked in the dataset. This scalar is encoded via a lightweight statistics token whose embedding is conditioned on the mask density with a small MLP module. The mask ratio embedding is included alongside the data tokens and participates in self-attention. Providing this cue regularizes the model across a range of masking rates and reduces pretraining to inference mismatch, which improves calibration when the inference pattern, such as masking a single target column, differs from the heavier masks used during pretraining.

4 Pretraining Data Generation

Performance of foundation models largely depends on the diversity and quality of pretraining data. To obtain a well-generalized foundation model for tabular data, we generate synthetic datasets using

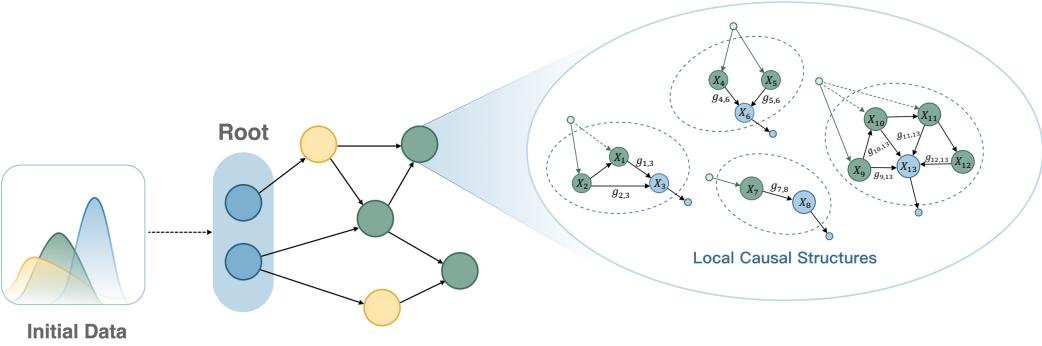


Figure 4: An example of the generated DAG, where $g_{i,j}$ is an edge function defining the relationship between a parent node X_i and its child node X_j in the local causal structures.

Directed Acyclic Graphs (DAGs), enabling the creation of datasets with diverse characteristics. The data generation process consists of three stages: DAG generation, data sampling, and task adaptation. First, a DAG is constructed to represent complex causal dependencies among variables. Then, a subset of these variables is sampled to define a specific problem, allowing LimiX models to develop causal reasoning capabilities. Finally, the sampled data is processed to align with various downstream tasks. Compared with the data generation methods used in current foundation models (Hollmann et al., 2025; Qu et al., 2025), we adopt a hierarchical generation paradigm that establishes causal dependencies in a more controllable and interpretable manner, while the sampling strategies further enhance the ability to generate datasets with diverse solvability and characteristics.

4.1 DAG Generation based on Hierarchical SCMs

The theoretical foundation of data generation lies in structural causal models (SCMs). As illustrated in Figure 4, within a DAG, initial data for each root node is independently sampled from an assigned distribution, with the distributions themselves chosen from a collection parameterized by randomized hyperparameters. Beginning at the root nodes, the initial data propagates through the DAG along its directed edges, wherein each encountered node represents a distinct local causal structure (LCS). As previously mentioned, the diversity of synthetic data is crucial for effective pretraining. To ensure that DAG generation is both diverse and well-structured, we adopt a hierarchical generation scheme rather than generating DAGs directly, thereby allowing for more fine-grained controls over the generation process. Within the LCS, the data input first propagates to the connected parent nodes, and the data of the child node can be obtained through $X_i = f(\{g_{i,k}(X_k)_{k \in PA(X_i)}, \epsilon_i\})$ where $g_{i,k}(\cdot)$ represents the edge function from X_k to X_i , $PA(X_i)$ denotes the set of parent nodes of X_i , $f(\cdot)$ is a parameterized aggregation function, and ϵ_i is an observational noise. This framework allows us to capture diverse local causal dependencies. For example, if an LCS contains only one parent node, that node is the direct cause of the child, making the dependency clear and straightforward. However, when multiple parent nodes are included in the LCS, the causal relationship becomes more complex due to the presence of potential confounding variables within the network.

In addition to the structural properties of LCSs, edge functions and the aggregation function also play crucial roles in shaping the dependencies. In this work, we use three types of edge functions:

- **Multilayer perceptrons (MLPs):** To determine the architecture of these neural networks, we uniformly sample properties such as the number of linear layers and their associated activation functions. When multiple layers are included, the MLP introduces complex nonlinear dependencies along the edge, whereas with only a single layer, it degenerates into a simple transformation. For weight initialization, we randomly choose from Xavier initialization (Glorot & Bengio, 2010) and He initialization (He et al., 2015). The activation functions are uniformly sampled from identity mapping, hyperbolic tangent, sigmoid, logarithm, absolute value, sine, squaring, modulo operation, heaviside step function, and GELU.
- **Convolutional layers:** For node-level tabular data, the convolution operation is applied along the sample dimension, serving as a local information mixer. The choices of weight initialization and activation functions follow the same procedure as those employed for MLPs.
- **Decision trees:** These are employed to introduce rule-based mappings and can take the form of either classification or regression trees. Unlike some approaches that fit decision trees on

random data, we argue that model fitting is neither necessary nor computationally efficient for the purpose of introducing rule-based dependencies. In this work, decision trees are fixed once constructed, with their hyperparameters sampled independently for each edge.

Once the child node receives the mapped values, they are aggregated using an aggregation function selected from one of three options: simple average, weighted average, or MLP-based aggregation. With respect to observational noise introduced at each edge, rather than adding noise of a fixed magnitude, we generate noise whose magnitude is scaled according to the distribution of each feature. To construct the DAG from all LCSs, we begin with a single-LCS DAG. Each time a new LCS is incorporated, it may connect to one or multiple existing LCSs in the DAG, whereby the child node of the new LCS is linked to the parent nodes of the target LCSs. This process is repeated until no LCSs remain to be added. In the resulting structure, all nodes with an in-degree of zero serve as the root nodes.

4.2 Data Sampling

Determining how to sample a subset of the generated data as the training set is also a critical challenge. Although random sampling is possible, the resulting training data often deviates significantly from being truly representative or good for training models. Thus it is necessary to devise an efficient strategy to sample high-quality training data for pretraining. We employ two sampling strategies: graph-aware sampling and solvability-aware sampling. Graph-aware sampling can be regarded as a more advanced variant of random sampling, where the key difference lies in its consideration of the graphical distribution of the sampled training data, which constrains the sampling space and renders it significantly smaller than in the case of random sampling. In contrast to graph-aware sampling, solvability-aware sampling aims to provide training data with varying degrees of solvability, thereby encouraging the model to achieve better generalization. From this point, we divide the subsampled problems into three classes: high-solvability, moderate-solvability and low-solvability problems. We ensure that the sampling ratios of these classes follow a categorical distribution, each of whose parameters is sampled from a distinct Gaussian distribution. In practice, we alternate between the two sampling strategies according to a predefined probability.

4.3 Task Adaptation

During the data generation process, the sampled data may be intended for either classification or regression tasks, and the major difference is the processing of the target variable y , which is initially sampled as a continuous variable. For classification tasks, it is subsequently discretized into categorical variables. For regression tasks, the procedure differs slightly. If the sampled y is already a discrete variable, a regression task will not be constructed; In cases where y is continuous but clusters closely around a limited set of distinct values, an in-order transformation can be applied to achieve a more uniform distribution across the magnitude scale.

5 Retrieval-based Ensemble

We adopt an inference-time and retrieval-based ensemble strategy that leverages LimiX’s learned attention scores to upweight and select representative in-context samples and features without any additional training, so that we can further improve the performance of LimiX models.

In terms of ensemble, we run multiple inference pipelines per dataset and aggregate the results. In each pipeline, we (i) randomly permute the feature columns and reorder the labels for categorical features or outcomes, and (ii) augment a subset of features with simple, schema-preserving transformations like quantile normalization, log-normal transformation, and high-energy SVD components. For classification, the number of inference pipelines is set to 4. For regression, it is set to 8.

In terms of retrieval, we perform two forward passes of LimiX models for each pipeline. The first pass is performed based on all in-context samples and is employed for retrieval. The second pass is performed based on the customized in-context samples retrieved in the first pass.

The procedure of the first pass is as follows. First, last-layer feature-level attention provides importance scores over features, which can be used as feature weights for subsequent sample selection. Concretely, for each test sample, we calculate $\mathbf{a}_f \in \mathbb{R}^{d+1}$, which is the feature-level attention score between the outcome y_{te} and the concatenation of F features and outcome $(\mathbf{x}_{te}, y_{te})$. Then, the module of last-layer sample-level cross-attention between test samples and in-context samples induces importance scores over in-context samples. Concretely, we calculate $\mathbf{a}_s \in \mathbb{R}^{m_{ct} \times (d+1)}$, which is the sample-level cross-attention scores between the test sample $(\mathbf{x}_{te}, y_{te})$ and in-context samples $(\mathbf{x}_{ct}, y_{ct})$. Finally, for a given test sample,

we retrieve in-context samples with highest reweighted attention scores as customized in-context samples for this test sample, and perform a forward propagation again for prediction. Concretely, we calculate $\mathbf{a}_{sf} = \mathbf{a}_s \mathbf{a}_f \in \mathbb{R}^{m_{ct}}$, a weighted average of \mathbf{a}_s along the feature dimension with \mathbf{a}_f as the feature weights.

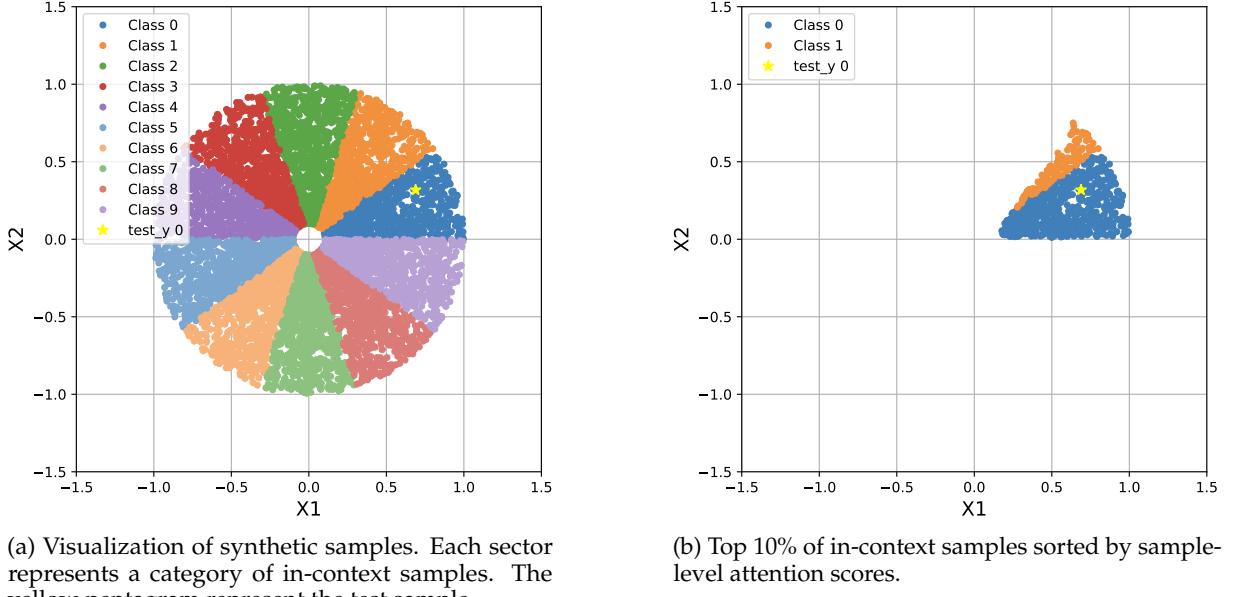


Figure 5: Toy example of sample-level attention. In-context samples that share the same category as the query sample are assigned higher scores through the attention module of LimiX-16M.

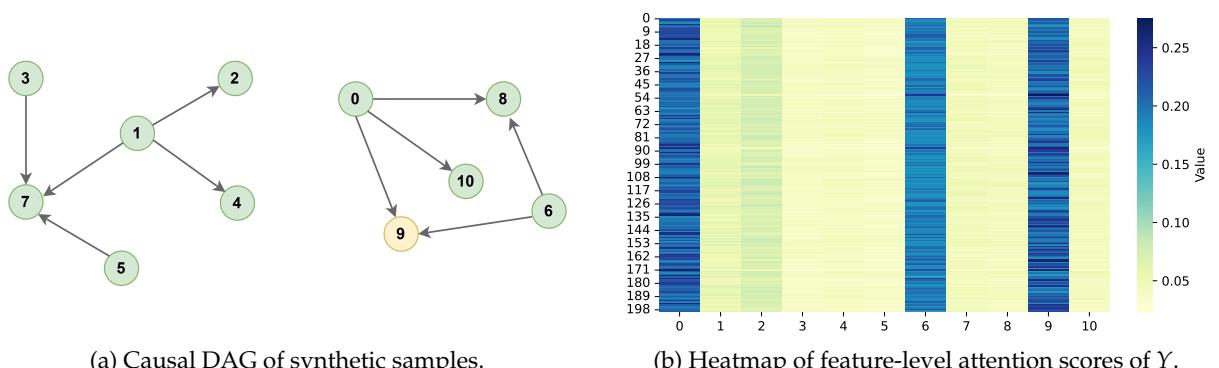


Figure 6: Toy example of feature-level attention for the outcome variable. Direct causes of the outcome and the outcome itself are assigned almost all the attention weights in LimiX-16M.

Toy examples. We present the effect of bi-level attention-based retrieval via toy examples.

- **Sample-level retrieval.** We generate 2D synthetic data of 10 classes, where each class of data points occupy a sector of a circle in Figure 5a. Since attention depicts sample similarity in the latent embedding space and it also takes the dependency between input features and the category label into account, it is capable of capturing more complex dependencies than naively using Euclidean distance in the original 2D space of features. From Figure 5b, we can see that the attention module predominantly assigns large weights to in-context instances sharing the same category label as the query instance. This indicates that the model leverages class-consistent contextual information from in-context samples to assist its prediction.
- **Feature-level retrieval.** We generate synthetic data via the SCM in Figure 6a, where each green node denotes an observed feature and the yellow node denotes the outcome. Each edge is a two-layer MLP using ReLU as the activation function added with Gaussian noise. From Figure 6b, we can see that the attention module assigns most weights to a subset of features, which is exactly the set of direct causes of the outcome in the SCM. This suggests that feature-level attention could help to focus on causal features and reduce reliance on spurious correlations.

6 Theoretical Analysis on Context-Conditional Masked Modeling

We begin by introducing the mathematical formulation underlying our theoretical analysis.

Notations. Let Ω denote the space for each feature in the table. We represent the m in-context samples, each with d features, by the random matrix $\mathbf{X}^{\text{ct}} \in \Omega^{m \times d}$. We use the random vector $\mathbf{X}^{\text{te}} = (X_1^{\text{te}}, \dots, X_d^{\text{te}}) \in \Omega^d$ to denote the test sample¹. Meanwhile, we do not explicitly introduce a target label y in this section for simplicity; instead, we treat it as a particular dimension of both the in-context and test samples. Consequently, one dimension of \mathbf{X}^{te} is unknown to the model during the test phase.

For any $\pi \subseteq [d]$, let $\mathbf{X}_{\pi}^{\text{te}} = (X_j^{\text{te}})_{j \in \pi}$ and $\mathbf{X}_{-\pi}^{\text{te}} = (X_j^{\text{te}})_{j \notin \pi}$ denote the subvectors on π and its complement, respectively. In particular, for $\pi = \{j\}$ we use $\mathbf{X}_{-j}^{\text{te}}$ as shorthand for $\mathbf{X}_{-\{j\}}^{\text{te}}$.

Training and test procedures. We are given n i.i.d. samples $\{(\mathbf{x}^{\text{ct},(i)}, \mathbf{x}^{\text{te},(i)})\}_{i=1}^n$ drawn from $p(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}})$, given in [Section 4](#). The model is trained using CCMM, a masked pre-training method ([Section 3](#)). Let $\Pi \subseteq 2^{[d]}$ be a set of masks and $\text{Unif}(\Pi)$ the uniform distribution over Π . In practice, we take $\Pi = \{\pi \subseteq [d] : |\pi| \in [0.1d, 0.4d]\}$, while for theory we simplify to masks of fixed size k :

$$\Pi_k := \{\pi \subseteq [d] : |\pi| = k\}. \quad (1)$$

It is easy to verify that our theoretical insights extend naturally to settings with variable mask sizes.

For each sample $(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}})$, we draw $\pi \sim \text{Unif}(\Pi_k)$ and train $q_{\theta}(\mathbf{X}_{\pi}^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})$, where $\theta \in \Theta$ and Θ denotes the parameter space of the model, to reconstruct the masked features. Let π_i be the mask for $(\mathbf{x}^{\text{ct},(i)}, \mathbf{x}^{\text{te},(i)})$. The empirical loss $\hat{L}_k(\theta)$ and estimator $\hat{\theta}_{k,n}$ are

$$\hat{L}_k(\theta) = \frac{1}{n} \sum_{i=1}^n -\log q_{\theta} \left(\mathbf{x}_{\pi_i}^{\text{te},(i)} | \mathbf{x}_{-\pi_i}^{\text{te},(i)}, \mathbf{x}^{\text{ct},(i)} \right), \quad \hat{\theta}_{k,n} = \arg \min_{\theta \in \Theta} \hat{L}_k(\theta). \quad (2)$$

As $n \rightarrow \infty$, this converges to the population-level loss $L_k(\theta)$ and the corresponding solution θ_k^* :

$$L_k(\theta) = \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_k)} [-\log q_{\theta}(\mathbf{X}_{\pi}^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})], \quad \theta_k^* = \arg \min_{\theta \in \Theta} L_k(\theta). \quad (3)$$

At test time, all features in \mathbf{X}^{ct} are observed, while one feature X_j^{te} of \mathbf{X}^{te} is missing and must be inferred. The goal is to output $p(X_j^{\text{te}} | \mathbf{X}_{-j}^{\text{te}}, \mathbf{X}^{\text{ct}})$.

6.1 Modeling the Joint Conditional with Random Masks

We now explain the necessity of randomly sampling masks from $\text{Unif}(\Pi_k)$. Our result is based on the following proposition.

Proposition 6.1 (Informal; See [Theorem B.1](#)). *Under mild assumptions, for any $k \in [d]$, there is a one-to-one correspondence between the distribution $p(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$ and the family of conditionals $\{p(\mathbf{X}_{\pi}^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}}) : \forall \pi \in \Pi_k\}$.*

At test time, the target variable y may correspond to any feature X_j^{te} of the test sample \mathbf{X}^{te} , so the model must be able to estimate $p(X_j^{\text{te}} | \mathbf{X}_{-j}^{\text{te}}, \mathbf{X}^{\text{ct}})$ for every $j \in [d]$. As shown in [Theorem 6.1](#), this requirement is equivalent to learning the joint conditional distribution $p(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$. Hence, a model can achieve strong predictive performance at test time if and only if it learns $p(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$. Moreover, if each $p(\mathbf{X}_{\pi}^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})$ can be well approximated by $q_{\theta}(\mathbf{X}_{\pi}^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})$ for all $\pi \in \Pi_k$, then the model can recover the joint conditional distribution and thereby generalize effectively. By contrast, [Example B.1](#) shows that knowledge of $p(\mathbf{X}_{\pi}^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})$ for only a subset of $\pi \in \Pi_k$ may be insufficient to recover $p(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$, preventing the model from generalizing effectively at test time.

Moreover, the target distribution $p(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$ is closely connected to the underlying structural causal models (SCMs). Let S denote a random variable corresponding to a structural causal model (SCM). By the data-generating process, we have $\mathbf{X}^{\text{ct}} \perp \mathbf{X}^{\text{te}} | S$, which yields

$$p(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}}) = \int_S p(\mathbf{X}^{\text{te}}, S | \mathbf{X}^{\text{ct}}) dS = \int_S p(S | \mathbf{X}^{\text{ct}}) p(\mathbf{X}^{\text{te}} | S, \mathbf{X}^{\text{ct}}) dS = \int_S p(S | \mathbf{X}^{\text{ct}}) p(\mathbf{X}^{\text{te}} | S) dS. \quad (4)$$

Intuitively, [Equation \(4\)](#) suggests that one efficient way to learn $p(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$ is through two components: (i) $p(S | \mathbf{X}^{\text{ct}})$, the posterior distribution over SCMs given the context, and (ii) $p(\mathbf{X}^{\text{te}} | S)$, the likelihood of the test sample under a given SCM.

¹For simplicity, we assume (i) all features share the same space Ω , though the results generalize to distinct spaces; (ii) we use m instead of m_{ct} in [Section 2](#); and (iii) there is only one test sample, i.e., $m_{te} = 1$ in [Section 2](#).

6.2 The Choice of Mask Number

In this subsection, we demonstrate that the choice k in [Equation \(1\)](#) has great impact on models' performances and we choose $k > 1$ due to both sample efficiency and generalization considerations. Our result is an extension from the analysis by [Li et al. \(2024c\)](#) on the properties of masked sequence prediction.

Based on [Theorem 6.1](#), we will henceforth use $q_\theta(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$ to denote the distributions induced by the learned family of conditional probabilities $\{q_\theta(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}}) : \pi \in \Pi_k\}$.

Sample efficiency. Theorem below shows that larger k yields lower estimation uncertainty.

Theorem 6.2 (Informal; see [Theorem B.2](#)). *Suppose there exists $\theta^* \in \Theta$ such that $q_{\theta^*}(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}}) = p(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})$ for all $\mathbf{X}^{\text{ct}} \in \Omega^{m \times d}$, $\mathbf{X}^{\text{te}} \in \Omega^d$, and $\pi \subseteq [d]$, and that the minimizer of $L_k(\theta)$ is unique for every k . Then, under mild regularity conditions, as $n \rightarrow \infty$,*

$$\sqrt{n} (\hat{\theta}_{k,n} - \theta^*) \xrightarrow{d} \mathcal{N}(0, \Gamma_k),$$

where Γ_k does not depend on n and satisfies $\Gamma_{k+1} \preceq \Gamma_k$.

The assumption on θ^* states that the optimal solution of the model can approximate any conditional distribution, which is reasonable given the expressive power of transformer-based architectures. This theorem further shows that for sufficiently large and fixed n , if $k_1 > k_2$ then $\Gamma_{k_1} \preceq \Gamma_{k_2}$, implying that $\hat{\theta}_{k_1,n}$ has lower estimation uncertainty than $\hat{\theta}_{k_2,n}$. Equivalently, when k is larger, fewer samples are required to achieve the same uncertainty, leading to greater sample efficiency.

Generalization for joint distribution learning. Masked pretraining empowers the model to infer the joint distribution of data from the context of training samples. We further show that an increase in the density of random masks enables a more accurate reconstruction of the joint distribution.

Theorem 6.3 (Informal; See [Theorem B.5](#)). *Under regularity conditions on q_θ , with high probability, for any $\theta \in \Theta$, the expected total variation between $q_\theta(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$ and the true conditional distribution $p(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$ is at most:*

$$\sqrt{\frac{1}{2} C_k(q_\theta)(\hat{L}_k(\theta) + \text{complexity terms}) + \mathcal{O}(n^{-1/2})},$$

where constants $C_k(q_\theta)$ depend only on q_θ and k . Furthermore, $C_{k+1}(q_\theta) \leq C_k(q_\theta)$ for any $\theta \in \Theta$.

The theorem establishes an upper bound for the generalization error of the estimated conditional joint distribution $q_\theta(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$ compared to the true joint distribution $p(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$, and shows that the upper bound decreases monotonically with respect to the number of masked cells.

7 Evaluation

7.1 Classification

Benchmarks. For the quantitative evaluation of classification performance, we utilize multiple benchmarks, including TALENT-CLS ([Liu et al., 2024](#)), OpenML-CC18 ([Bischl et al., 2017](#)), PFN-CLS ([Hollmann et al., 2025](#)), TabZilla ([McElfresh et al., 2023](#)), and TabArena-CLS ([Erickson et al., 2025](#)). Among these benchmarks, the datasets containing more than 50,000 training samples (The number of testing samples is not constrained), 10,000 features, or 10 target categories were excluded. This selection process resulted in a final collection of 179 datasets from TALENT-CLS, 62 from OpenML-CC18, 29 from PFN-CLS, 27 from TabZilla, and 33 from TabArena-CLS.

Furthermore, we introduce **Balanced Comprehensive Challenging Omni-domain (BCCO) Benchmark**, comprising BCCO-CLS and BCCO-REG, for the evaluation of LimiX models and baseline models. The BCCO benchmark is constructed from extensive open-source structured-data corpora, meticulously deduplicated and cleaned. It presents a significant challenge due to several intrinsic characteristics: the distribution of dataset attributes (such as the ratio of categorical features), and the diversity of real-world prediction targets. Unlike previous benchmarks, nearly one-third of the datasets in our BCCO benchmark contain missing values. The BCCO benchmarks are publicly available at https://huggingface.co/datasets/stableai-org/bcco_cls and https://huggingface.co/datasets/stableai-org/bcco_reg.

The BCCO-CLS benchmark comprising 106 datasets. The collection spans diverse sources, domains, and scales, and is designed to cover a wide range of problem characteristics, including the number of samples, number of features, number of classes, categorical-to-numerical feature ratio, sample-to-feature ratio,

and proportion of missing values. This diversity allows for uniform binning along these dimensions, enabling results to be reported within bins and macro-averaged across bins, thereby providing a nearly unbiased assessment of model performance across heterogeneous task regimes. Additionally, datasets containing more than 50,000 training samples (The number of testing samples is not constrained), 10,000 features, or 10 target categories were excluded.

For each dataset in these benchmarks, we use the provided train-test split when available. If no predefined test set exists, the data are partitioned into a 70% training set and a 30% test set using stratified sampling to preserve the label distribution.

Baselines. We compare LimiX models with a range of state-of-the-art baseline models, categorized into tree-based models, neural networks (NN), and recent ICL-based approaches.

- **Tree-based approaches.** We include XGBoost (Chen & Guestrin, 2016), LightGBM (Ke et al., 2017), CatBoost (Dorogush et al., 2018), Random Forest (RF) (Breiman, 2001), and Extra Trees (ET) (Geurts et al., 2006). All models are optimized using the Optuna (Akiba et al., 2019) framework via 5-fold stratified cross-validation, with hyperparameters sampled from the ranges specified in Section A.2. Additionally, for AutoGluon-Tabular (Erickson et al., 2020), which automates workflows of model searching and ensemble, we use the default search space and set a default 600s time constraint for hyperparameter searching for each dataset.
- **NN-based approaches.** We evaluate against SNN (Klambauer et al., 2017), AutoInt (Song et al., 2019), NODE (Popov et al., 2020), TabTransformer (Huang et al., 2020), GrowNet (Badirlı et al., 2020), TabNet (Arik & Pfister, 2021), DCN-v2 (Wang et al., 2021), FT-Transformer (Gorishniy et al., 2021), MLP (Goodfellow et al., 2016; Gorishniy et al., 2021), ResNet (He et al., 2016; Gorishniy et al., 2021), SAINT (Somepalli et al., 2022), MLP-PLR (Gorishniy et al., 2022), DANets (Chen et al., 2022), TANGOS (Jeffares et al., 2023), T2G-Former (Yan et al., 2023), ExcelFormer (Chen et al., 2023b), Trompt (Chen et al., 2023c), TabR (Gorishniy et al., 2024), TabCaps (Chen et al., 2023a), RealMLP (Holzmüller et al., 2024), SwitchTab (Wu et al., 2024), and ModernNCA (Ye et al., 2025). These NN-based models are trained using the TALENT (Liu et al., 2024) Toolbox.
- **ICL-based models.** Recent baselines includes TabPFN-v2 (Hollmann et al., 2025), TabICL (Qu et al., 2025), and Mitra (Zhang & Danielle, 2025).

Since TabDPT (Ma et al., 2024), another recent ICL-based model, is pretrained on real data that has a large overlap with datasets in the benchmarks, we do not adopt it as a baseline in the evaluation of standard classification and regression for a fair comparison. However, we include it in our experiments of fine-tuning in Section 7.6 to show the superiority of LimiX models.

Metrics. To evaluate model performance, we employ ROC AUC (area under the receiver operating characteristic curve), accuracy, and F1 score for classification tasks. For multi-class classification, the One-vs-One strategy is applied to both ROC AUC and F1 score calculations.

The critical difference diagram is also employed to compare the performance differences between LimiX models and the baseline models. We conducted a Friedman test followed by a post-hoc Wilcoxon-Holm test, using a significance level of 0.05. In the diagram, the horizontal line indicates the range of ranks among which differences are not statistically significant.

Results. Figure 7 shows that LimiX-16M achieves the best performance among all methods on most datasets in BCCO-CLS, followed closely by LimiX-2M. The most competitive baselines include other ICL-based models TabICL and TabPFN-v2, and the AutoML ensemble framework AutoGluon. In all critical difference diagrams, i.e. Figures 8 to 13, we can see that LimiX-16M achieves the highest ranking in terms of all three evaluation metrics on BCCO-CLS, OpenML-CC18, and TALENT-CLS, where critical differences can be observed in most cases, indicating a significant margin. Despite operating under limited computational resources, LimiX-2M still delivers highly competitive results.

For detailed quantitative results listed in Tables 1, 3, 5, 7, 9 and 11, LimiX-16M outperforms all baselines on every benchmark in terms of both mean and rank of all three metrics. All these results clearly demonstrate that LimiX-16M achieves state-of-the-art performance in terms of tabular classification tasks, surpassing not only traditional ensemble methods but also advanced ICL-based models. Across all classification benchmarks, LimiX-2M also outperforms other strong baselines, despite operating under highly limited computational resources.

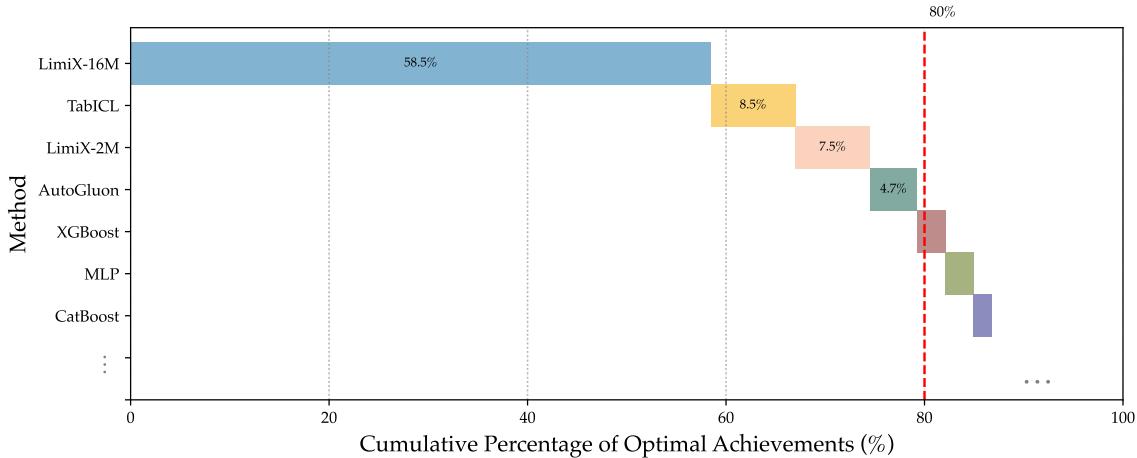


Figure 7: The proportion of models achieving the best AUC. The length of each bar represents the proportion of the 106 datasets in BCCO-CLS where a given method achieves the highest AUC.

Subgroup analysis. We use the sample subgroups when building BCCO-CLS to perform stratified analyses. Subgroups are defined based on the following criteria: the type of classification (binary or multi-class), the number of training samples, the ratio between the number of samples and features (length-to-width ratio), the proportion of categorical features, and the presence or absence of missing values. The number of training samples, categorical feature ratio, and length-to-width ratio are discretized into terciles (equal-frequency bins), ensuring that approximately the same number of datasets is allocated to each stratum and that the distribution of the dataset attribute which is not considered during analysis remains nearly uniform across strata.

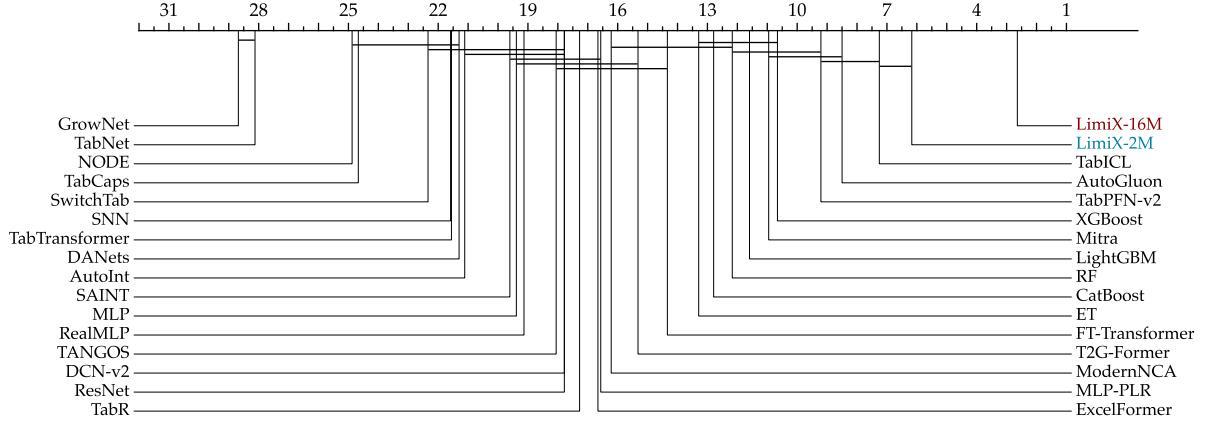
From Figure 20, we observe that LimiX-16M exhibits leading performance across all subgroups compared with other methods. Notably, after stratification, under some subgroups like the third subgroup in Figure 20c that indicates a larger training sample size, AutoGluon proves to be a strong competitor to ICL-based models. In these cases, LimiX-16M is the only ICL-based model that outperforms AutoGluon. Notably, Figure 20e shows that, as the proportion of categorical features increases, performance for most baselines drops rapidly, reflecting the sparsity and high-cardinality challenges. In contrast, LimiX-16M exhibits only modest degradation and remains comparatively stable across these regimes.

Table 1: Classification results on the BCCO-CLS benchmark. The best scores are shown in bold.

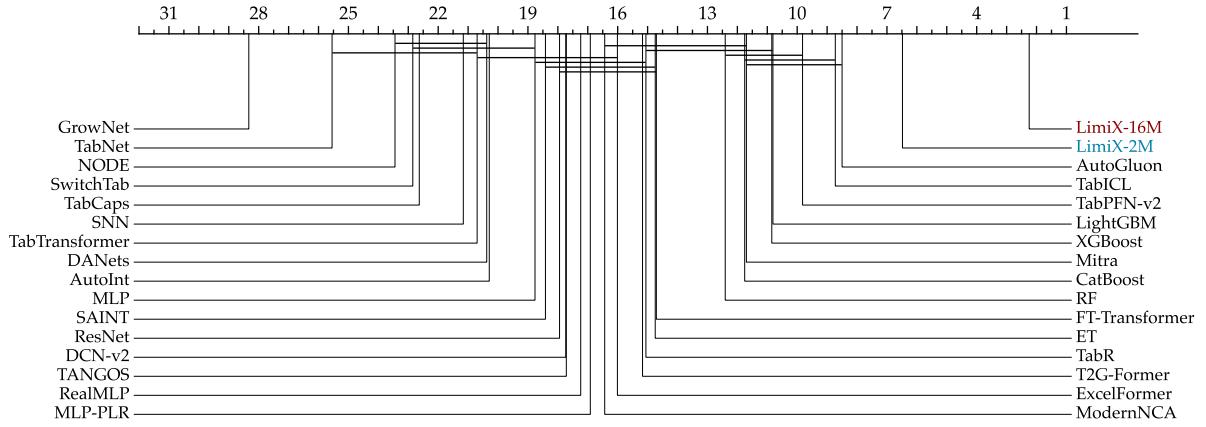
| Model | BCCO-CLS | | | | | |
|----------------|--------------------|---------------------|-------------------|----------------------|-----------------------|---------------------|
| | Mean | | | Rank | | |
| | AUC (\uparrow) | Acc. (\uparrow) | F1 (\uparrow) | AUC (\downarrow) | Acc. (\downarrow) | F1 (\downarrow) |
| Limix-16M | 0.871 | 0.804 | 0.731 | 2.642 | 2.245 | 2.887 |
| Limix-2M | 0.858 | 0.787 | 0.701 | 6.170 | 6.481 | 8.538 |
| TabICL | 0.847 | 0.768 | 0.672 | 7.255 | 8.726 | 10.783 |
| AutoGluon | 0.846 | 0.771 | 0.677 | 8.500 | 8.500 | 9.934 |
| TabPFN-v2 | 0.843 | 0.772 | 0.679 | 9.208 | 9.821 | 11.396 |
| XGBoost | 0.834 | 0.762 | 0.674 | 10.660 | 10.849 | 11.547 |
| Mitra | 0.836 | 0.764 | 0.664 | 10.953 | 11.698 | 13.585 |
| LightGBM | 0.832 | 0.763 | 0.678 | 11.594 | 10.811 | 11.406 |
| RF | 0.829 | 0.756 | 0.652 | 12.170 | 12.406 | 13.726 |
| CatBoost | 0.829 | 0.757 | 0.664 | 12.792 | 11.755 | 12.557 |
| ET | 0.825 | 0.745 | 0.618 | 13.292 | 14.745 | 17.585 |
| FT-Transformer | 0.813 | 0.744 | 0.642 | 14.340 | 14.708 | 15.104 |
| T2G-Former | 0.808 | 0.742 | 0.646 | 15.321 | 15.170 | 14.377 |
| ModernNCA | 0.815 | 0.752 | 0.658 | 16.217 | 16.434 | 16.047 |
| MLP-PLR | 0.804 | 0.733 | 0.635 | 16.566 | 16.915 | 16.113 |
| ExcelFormer | 0.810 | 0.742 | 0.655 | 16.660 | 16.009 | 14.566 |
| TabR | 0.809 | 0.750 | 0.657 | 17.274 | 15.057 | 14.104 |
| DCN-v2 | 0.794 | 0.725 | 0.618 | 17.783 | 17.736 | 17.453 |
| ResNet | 0.800 | 0.728 | 0.641 | 17.783 | 17.943 | 16.670 |
| TANGOS | 0.799 | 0.731 | 0.641 | 18.057 | 17.717 | 16.151 |
| RealMLP | 0.794 | 0.738 | 0.644 | 19.132 | 17.236 | 15.934 |
| MLP | 0.787 | 0.720 | 0.614 | 19.387 | 18.764 | 19.594 |
| SAINT | 0.791 | 0.726 | 0.623 | 19.604 | 18.415 | 17.066 |
| AutoInt | 0.779 | 0.718 | 0.601 | 21.113 | 20.292 | 20.660 |
| DANets | 0.771 | 0.705 | 0.601 | 21.302 | 20.377 | 20.708 |
| TabTransformer | 0.762 | 0.699 | 0.566 | 21.557 | 20.698 | 21.170 |
| SNN | 0.773 | 0.708 | 0.584 | 21.585 | 21.160 | 21.415 |
| SwitchTab | 0.766 | 0.700 | 0.590 | 22.340 | 22.849 | 21.764 |
| TabCaps | 0.744 | 0.701 | 0.580 | 24.670 | 22.632 | 23.019 |
| NODE | 0.754 | 0.695 | 0.531 | 24.877 | 23.443 | 26.198 |
| TabNet | 0.712 | 0.685 | 0.561 | 28.123 | 25.547 | 25.679 |
| GrowNet | 0.682 | 0.641 | 0.522 | 28.679 | 28.330 | 27.321 |

Table 2: Statistical profile of the benchmark BCCO-CLS, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively; for categorical feature statistics, we only consider features that are either string-typed or have fewer than 10 unique values.

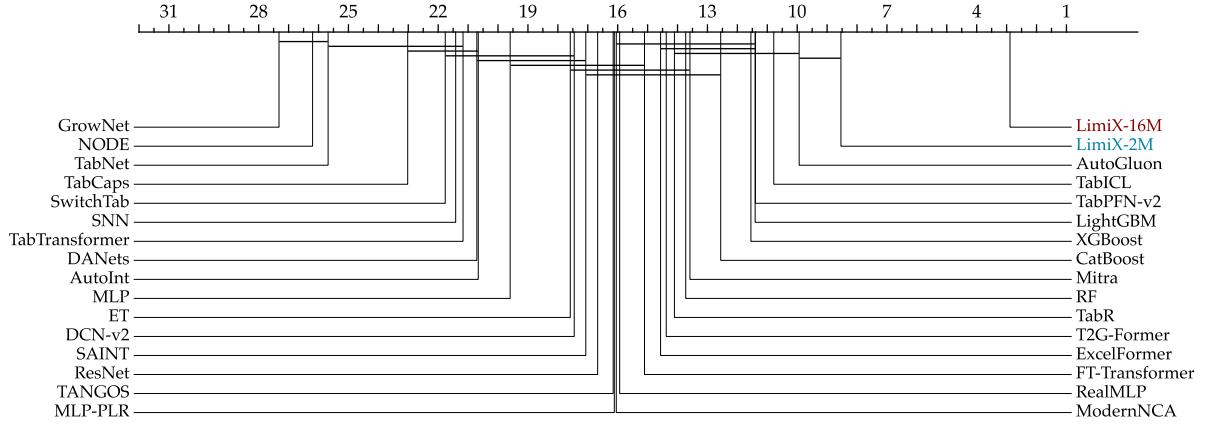
| Metric | Statistics | | | | | | |
|------------------------------------|------------|-------|-------|-------|-------|-----|-------|
| | Q10 | Q50 | Q90 | Mean | Std | Min | Max |
| # Features | 4 | 12 | 48 | 23 | 33 | 1 | 259 |
| # Classes | 2 | 2 | 5 | 3 | 2 | 2 | 10 |
| Missing Values (Ratio) | 0 | 0 | 0.069 | 0.024 | 0.068 | 0 | 0.403 |
| Categorical Features (Ratio) | 0 | 0.384 | 0.929 | 0.399 | 0.337 | 0 | 1 |
| Features w/ Missing Values (Ratio) | 0 | 0 | 0.523 | 0.126 | 0.241 | 0 | 0.909 |



(a) AUC on the BCCO-CLS benchmark.



(b) Accuracy on the BCCO-CLS benchmark.



(c) F1-score on the BCCO-CLS benchmark.

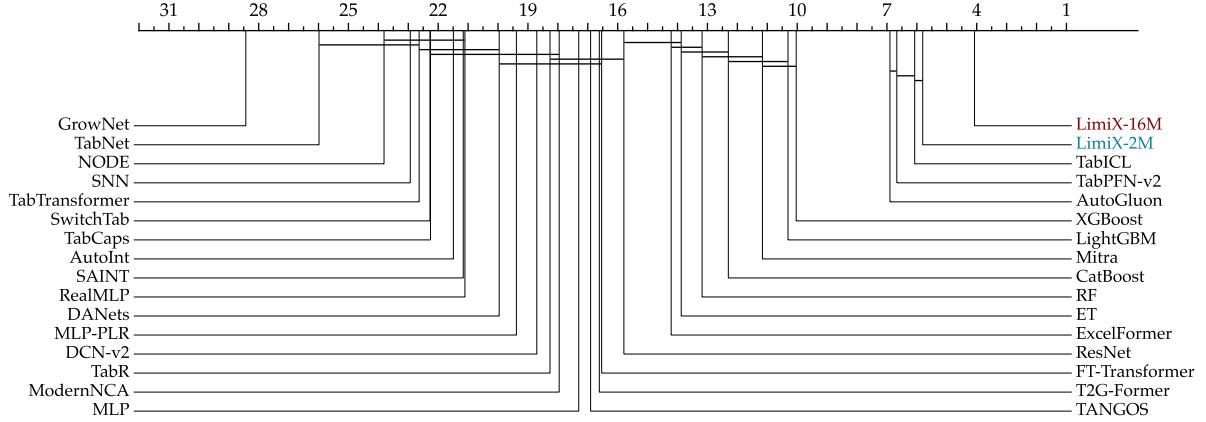
Figure 8: Critical difference diagrams on BCCO-CLS benchmark.

Table 3: Classification results on the TALENT-CLS benchmark. The best scores are shown in bold.

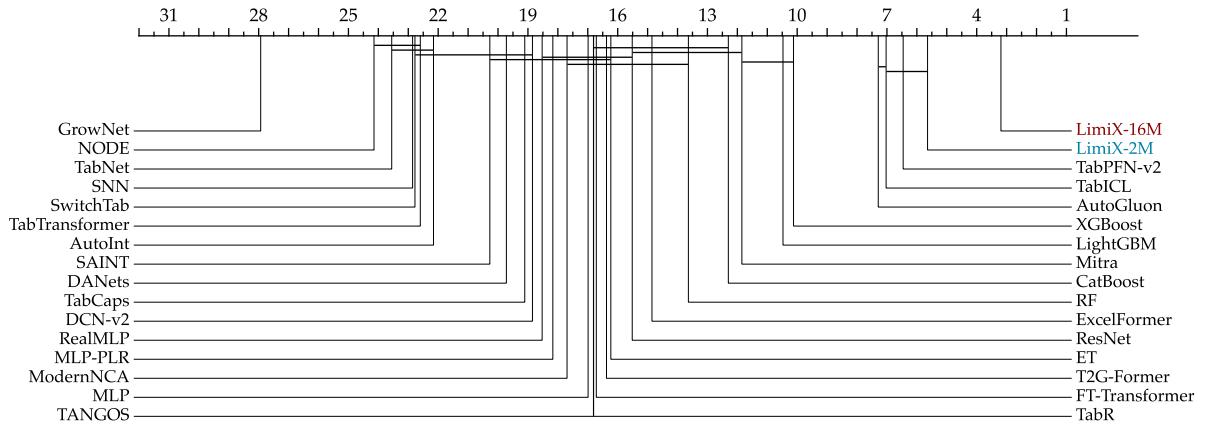
| Model | TALENT-CLS | | | | | |
|----------------|--------------------|---------------------|-------------------|----------------------|-----------------------|---------------------|
| | Mean | | | Rank | | |
| | AUC (\uparrow) | Acc. (\uparrow) | F1 (\uparrow) | AUC (\downarrow) | Acc. (\downarrow) | F1 (\downarrow) |
| Limix-16M | 0.903 | 0.861 | 0.752 | 4.073 | 3.190 | 4.246 |
| Limix-2M | 0.897 | 0.853 | 0.734 | 5.804 | 5.642 | 6.994 |
| TabICL | 0.894 | 0.845 | 0.715 | 6.073 | 7.028 | 8.514 |
| TabPFN-v2 | 0.895 | 0.850 | 0.727 | 6.670 | 6.458 | 8.296 |
| AutoGluon | 0.891 | 0.845 | 0.719 | 6.899 | 7.291 | 8.067 |
| XGBoost | 0.881 | 0.837 | 0.713 | 10.034 | 10.123 | 10.425 |
| LightGBM | 0.880 | 0.836 | 0.713 | 10.307 | 10.475 | 10.777 |
| Mitra | 0.882 | 0.834 | 0.689 | 11.162 | 11.849 | 13.916 |
| CatBoost | 0.876 | 0.828 | 0.704 | 12.302 | 12.302 | 12.508 |
| RF | 0.877 | 0.828 | 0.691 | 13.179 | 13.637 | 14.754 |
| ET | 0.875 | 0.821 | 0.662 | 13.877 | 16.229 | 17.788 |
| ExcelFormer | 0.870 | 0.826 | 0.699 | 14.212 | 14.855 | 13.749 |
| ResNet | 0.866 | 0.825 | 0.695 | 15.793 | 15.514 | 13.955 |
| FT-Transformer | 0.859 | 0.822 | 0.678 | 16.536 | 16.715 | 16.413 |
| T2G-Former | 0.858 | 0.823 | 0.683 | 16.615 | 16.374 | 15.933 |
| TANGOS | 0.861 | 0.818 | 0.684 | 16.905 | 16.810 | 15.542 |
| MLP | 0.862 | 0.817 | 0.675 | 17.302 | 16.989 | 17.162 |
| ModernNCA | 0.861 | 0.825 | 0.683 | 17.955 | 17.693 | 16.966 |
| TabR | 0.858 | 0.824 | 0.680 | 18.263 | 16.804 | 15.922 |
| DCN-v2 | 0.854 | 0.815 | 0.662 | 18.704 | 18.849 | 18.687 |
| MLP-PLR | 0.849 | 0.816 | 0.663 | 19.385 | 18.168 | 18.201 |
| DANets | 0.848 | 0.805 | 0.654 | 19.961 | 19.721 | 19.374 |
| RealMLP | 0.839 | 0.820 | 0.678 | 21.106 | 18.525 | 16.793 |
| SAINT | 0.813 | 0.781 | 0.630 | 21.162 | 20.274 | 19.715 |
| AutoInt | 0.842 | 0.803 | 0.646 | 21.492 | 22.156 | 21.587 |
| TabCaps | 0.834 | 0.813 | 0.654 | 22.257 | 19.106 | 19.570 |
| SwitchTab | 0.842 | 0.795 | 0.637 | 22.274 | 22.777 | 21.989 |
| TabTransformer | 0.832 | 0.790 | 0.627 | 22.637 | 22.598 | 21.760 |
| SNN | 0.836 | 0.796 | 0.625 | 22.933 | 22.855 | 22.972 |
| NODE | 0.830 | 0.779 | 0.570 | 23.810 | 24.145 | 25.726 |
| TabNet | 0.818 | 0.794 | 0.630 | 25.983 | 23.553 | 23.983 |
| GrowNet | 0.743 | 0.704 | 0.542 | 28.430 | 27.933 | 25.933 |

Table 4: Statistical profile of the benchmark TALENT-CLS, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively; for categorical feature statistics, we only consider features that are either string-typed or have fewer than 10 unique values.

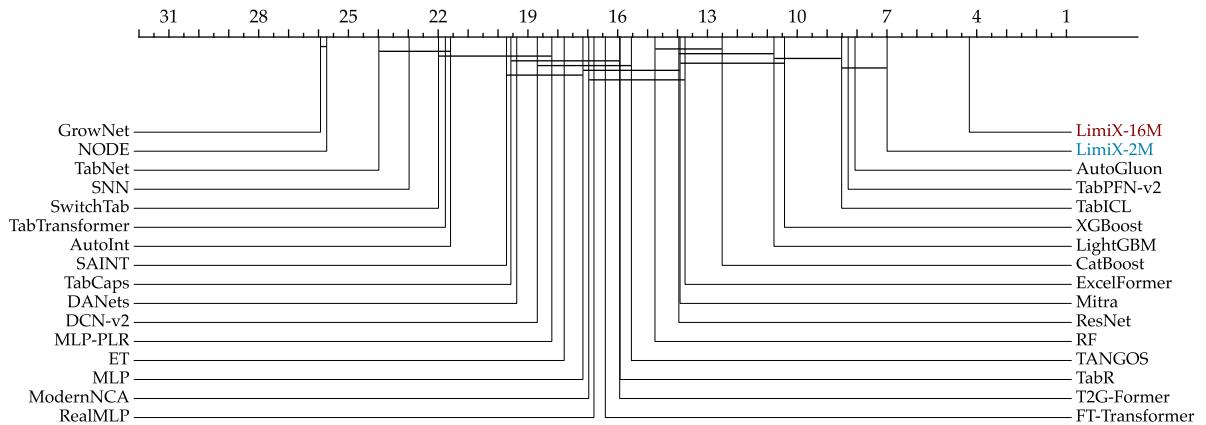
| Metric | Statistics | | | | | | |
|------------------------------------|------------|-------|-------|-------|-------|-----|-------|
| | Q10 | Q50 | Q90 | Mean | Std | Min | Max |
| # Features | 7 | 19 | 70 | 33 | 47 | 3 | 308 |
| # Classes | 2 | 2 | 10 | 6 | 14 | 2 | 100 |
| Missing Values (Ratio) | 0 | 0 | 0 | 0.001 | 0.008 | 0 | 0.1 |
| Categorical Features (Ratio) | 0 | 0.121 | 0.972 | 0.306 | 0.365 | 0 | 1 |
| Features w/ Missing Values (Ratio) | 0 | 0 | 0 | 0.025 | 0.121 | 0 | 0.979 |



(a) AUC on the TALENT-CLS benchmark.



(b) Accuracy on the TALENT-CLS benchmark.



(c) F1-score on the TALENT-CLS benchmark.

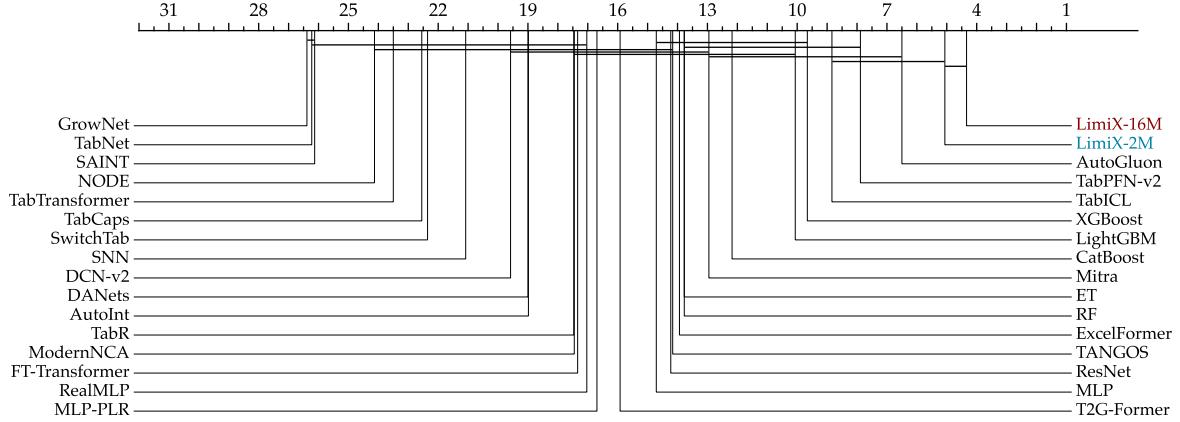
Figure 9: Critical difference diagram on the TALENT-CLS benchmark

Table 5: Classification results on the OpenML-CC18 benchmark. The best scores are shown in bold.

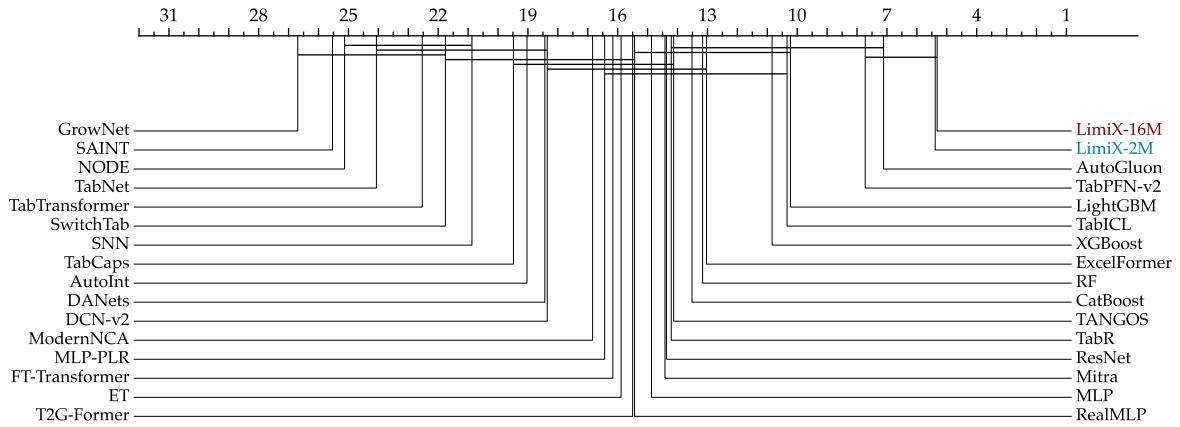
| Model | OpenML-cc18 | | | | | |
|----------------|--------------------|---------------------|-------------------|----------------------|-----------------------|---------------------|
| | Mean | | | Rank | | |
| | AUC (\uparrow) | Acc. (\uparrow) | F1 (\uparrow) | AUC (\downarrow) | Acc. (\downarrow) | F1 (\downarrow) |
| Limix-16M | 0.939 | 0.893 | 0.811 | 4.339 | 5.323 | 4.548 |
| Limix-2M | 0.936 | 0.891 | 0.803 | 5.065 | 5.387 | 5.919 |
| AutoGluon | 0.932 | 0.885 | 0.790 | 6.500 | 7.113 | 7.613 |
| TabPFN-v2 | 0.929 | 0.886 | 0.790 | 7.887 | 7.726 | 8.161 |
| TabICL | 0.927 | 0.875 | 0.782 | 8.839 | 10.339 | 10.306 |
| XGBoost | 0.929 | 0.879 | 0.775 | 9.661 | 10.839 | 10.790 |
| LightGBM | 0.927 | 0.879 | 0.775 | 10.065 | 10.226 | 10.323 |
| CatBoost | 0.926 | 0.870 | 0.770 | 12.177 | 13.516 | 13.419 |
| Mitra | 0.920 | 0.866 | 0.743 | 12.952 | 14.419 | 16.048 |
| ET | 0.922 | 0.861 | 0.721 | 13.774 | 15.887 | 17.774 |
| RF | 0.925 | 0.871 | 0.762 | 13.774 | 13.161 | 14.048 |
| ExcelFormer | 0.918 | 0.870 | 0.773 | 13.935 | 13.032 | 12.839 |
| TANGOS | 0.910 | 0.863 | 0.759 | 14.161 | 14.129 | 14.290 |
| ResNet | 0.913 | 0.860 | 0.764 | 14.226 | 14.371 | 13.581 |
| MLP | 0.908 | 0.857 | 0.743 | 14.710 | 14.871 | 15.871 |
| T2G-Former | 0.908 | 0.859 | 0.748 | 15.919 | 15.500 | 15.484 |
| MLP-PLR | 0.896 | 0.858 | 0.734 | 16.694 | 16.435 | 16.790 |
| RealMLP | 0.889 | 0.858 | 0.742 | 17.032 | 15.435 | 14.823 |
| FT-Transformer | 0.904 | 0.856 | 0.739 | 17.339 | 16.161 | 17.065 |
| ModernNCA | 0.906 | 0.858 | 0.747 | 17.452 | 16.839 | 16.371 |
| TabR | 0.900 | 0.863 | 0.757 | 17.468 | 14.210 | 13.790 |
| AutoInt | 0.889 | 0.835 | 0.717 | 18.984 | 19.032 | 19.129 |
| DANets | 0.900 | 0.840 | 0.715 | 19.000 | 18.435 | 19.435 |
| DCN-v2 | 0.899 | 0.851 | 0.729 | 19.581 | 18.355 | 18.968 |
| SNN | 0.896 | 0.835 | 0.698 | 21.081 | 20.871 | 21.661 |
| SwitchTab | 0.881 | 0.819 | 0.671 | 22.355 | 21.758 | 22.419 |
| TabCaps | 0.875 | 0.848 | 0.692 | 22.548 | 19.484 | 21.097 |
| TabTransformer | 0.829 | 0.776 | 0.616 | 23.500 | 22.532 | 23.274 |
| NODE | 0.889 | 0.809 | 0.626 | 24.129 | 25.129 | 27.161 |
| SAINT | 0.512 | 0.492 | 0.441 | 26.129 | 25.532 | 24.210 |
| TabNet | 0.863 | 0.816 | 0.666 | 26.226 | 24.065 | 25.823 |
| GrowNet | 0.816 | 0.748 | 0.581 | 26.387 | 26.694 | 25.903 |

Table 6: Statistical profile of the benchmark OpenML-CC18, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively; for categorical feature statistics, we only consider features that are either string-typed or have fewer than 10 unique values.

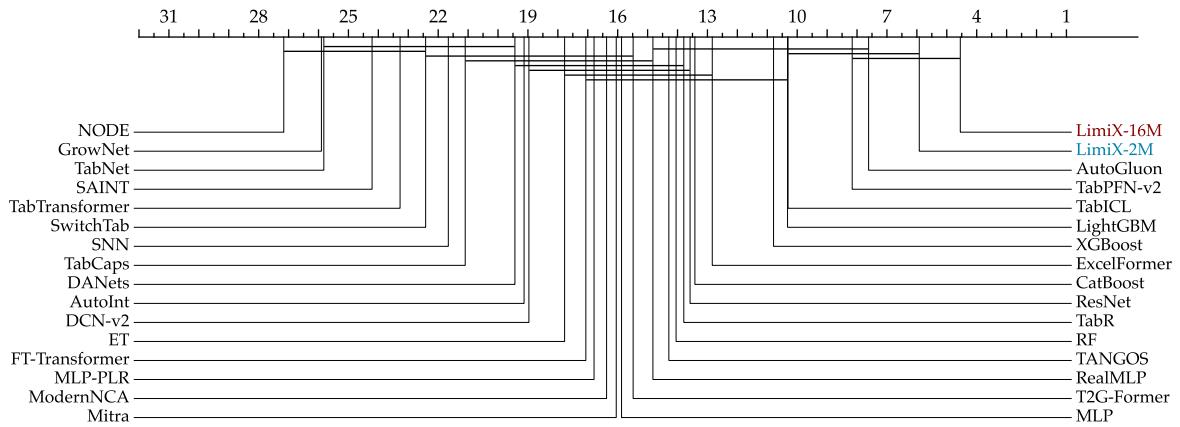
| Metric | Statistics | | | | | | |
|------------------------------------|------------|-------|-------|-------|-------|-----|-------|
| | Q10 | Q50 | Q90 | Mean | Std | Min | Max |
| # Features | 6 | 30 | 611 | 336 | 1344 | 4 | 10935 |
| # Classes | 2 | 3 | 10 | 6 | 7 | 2 | 46 |
| Missing Values (Ratio) | 0 | 0 | 0.002 | 0.004 | 0.019 | 0 | 0.139 |
| Categorical Features (Ratio) | 0 | 0.091 | 1 | 0.327 | 0.406 | 0 | 1 |
| Features w/ Missing Values (Ratio) | 0 | 0 | 0.204 | 0.054 | 0.165 | 0 | 0.757 |



(a) AUC on the OpenML-CC18 benchmark.



(b) Accuracy on the OpenML-CC18 benchmark.



(c) F1-score on the OpenML-CC18 benchmark.

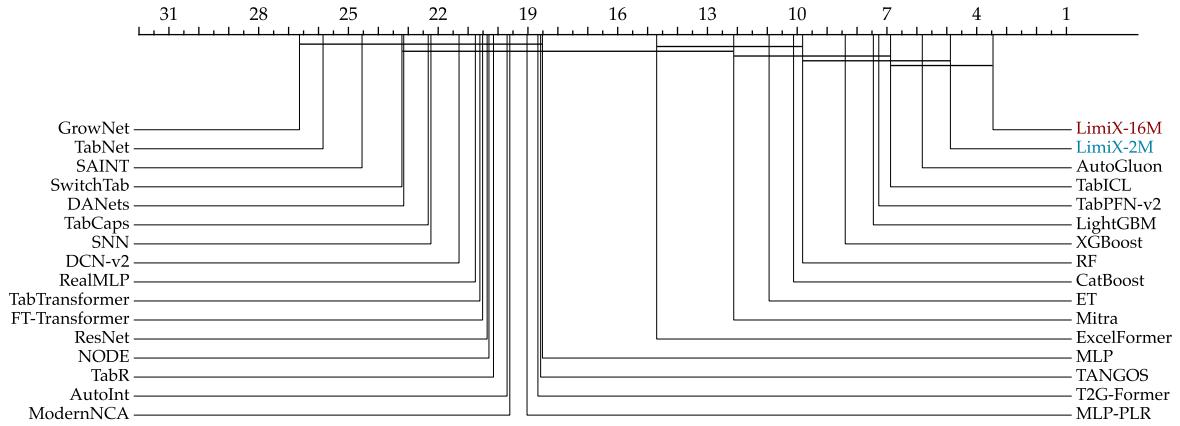
Figure 10: Critical difference diagrams on the OpenML-CC18 benchmark.

Table 7: Classification results on the TabArena-CLS benchmark. The best scores are shown in bold.

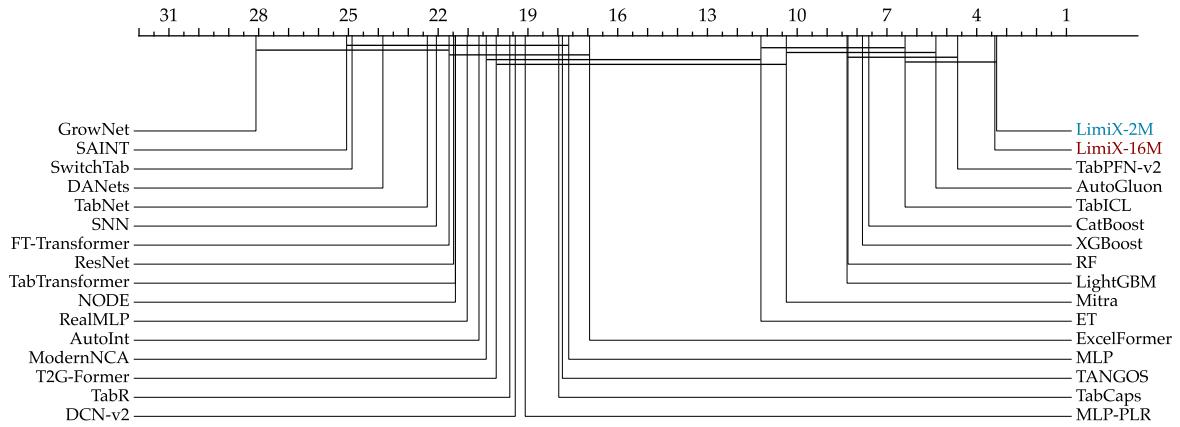
| Model | TabArena-CLS | | | | | |
|----------------|--------------------|---------------------|-------------------|----------------------|-----------------------|---------------------|
| | Mean | | | Rank | | |
| | AUC (\uparrow) | Acc. (\uparrow) | F1 (\uparrow) | AUC (\downarrow) | Acc. (\downarrow) | F1 (\downarrow) |
| Limix-16M | 0.849 | 0.877 | 0.597 | 3.455 | 3.394 | 7.000 |
| Limix-2M | 0.846 | 0.876 | 0.594 | 4.879 | 3.333 | 6.727 |
| AutoGluon | 0.844 | 0.870 | 0.574 | 5.818 | 5.364 | 9.091 |
| TabICL | 0.840 | 0.870 | 0.553 | 6.879 | 6.394 | 10.576 |
| TabPFN-v2 | 0.838 | 0.872 | 0.589 | 7.273 | 4.636 | 8.848 |
| LightGBM | 0.841 | 0.868 | 0.574 | 7.455 | 8.333 | 10.545 |
| XGBoost | 0.838 | 0.867 | 0.567 | 8.394 | 7.818 | 10.909 |
| RF | 0.837 | 0.864 | 0.558 | 9.818 | 8.303 | 12.061 |
| CatBoost | 0.835 | 0.867 | 0.574 | 10.121 | 7.606 | 9.970 |
| ET | 0.833 | 0.857 | 0.505 | 10.939 | 11.212 | 16.333 |
| Mitra | 0.815 | 0.862 | 0.533 | 12.121 | 10.364 | 14.970 |
| ExcelFormer | 0.810 | 0.849 | 0.555 | 14.697 | 16.939 | 14.970 |
| MLP | 0.772 | 0.822 | 0.459 | 18.515 | 17.636 | 19.576 |
| TANGOS | 0.791 | 0.844 | 0.522 | 18.576 | 17.848 | 16.000 |
| T2G-Former | 0.779 | 0.822 | 0.482 | 18.667 | 20.061 | 15.848 |
| MLP-PLR | 0.781 | 0.836 | 0.460 | 19.030 | 19.091 | 19.667 |
| ModernNCA | 0.783 | 0.846 | 0.511 | 19.606 | 20.394 | 16.970 |
| AutoInt | 0.769 | 0.826 | 0.474 | 19.697 | 20.636 | 18.576 |
| TabR | 0.785 | 0.842 | 0.510 | 20.152 | 19.606 | 16.091 |
| NODE | 0.769 | 0.792 | 0.352 | 20.303 | 21.424 | 24.333 |
| ResNet | 0.781 | 0.824 | 0.532 | 20.364 | 21.485 | 16.303 |
| FT-Transformer | 0.770 | 0.803 | 0.468 | 20.515 | 21.636 | 18.606 |
| TabTransformer | 0.739 | 0.781 | 0.438 | 20.606 | 21.424 | 19.091 |
| RealMLP | 0.777 | 0.822 | 0.512 | 20.758 | 21.030 | 14.424 |
| DCN-v2 | 0.769 | 0.833 | 0.482 | 21.303 | 19.424 | 18.212 |
| SNN | 0.755 | 0.818 | 0.442 | 22.242 | 22.061 | 22.000 |
| TabCaps | 0.742 | 0.837 | 0.471 | 22.333 | 17.970 | 19.667 |
| DANets | 0.749 | 0.776 | 0.453 | 23.152 | 23.848 | 18.333 |
| SwitchTab | 0.754 | 0.799 | 0.409 | 23.212 | 24.879 | 22.303 |
| SAINT | 0.694 | 0.739 | 0.437 | 24.545 | 25.061 | 21.515 |
| TabNet | 0.709 | 0.789 | 0.438 | 25.848 | 22.364 | 23.182 |
| GrowNet | 0.646 | 0.674 | 0.361 | 26.636 | 28.091 | 23.273 |

Table 8: Statistical profile of the benchmark TabArena, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively; for categorical feature statistics, we only consider features that are either string-typed or have fewer than 10 unique values.

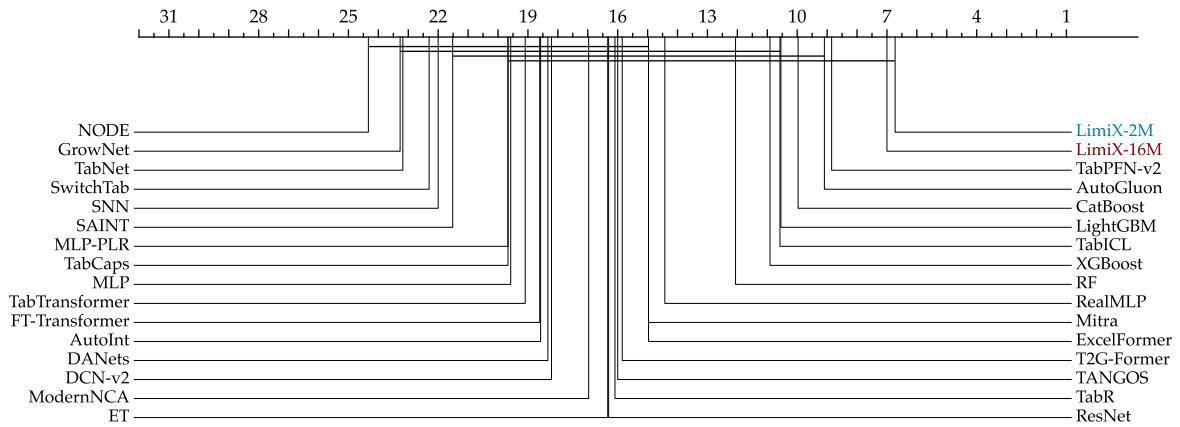
| Metric | Statistics | | | | | | |
|------------------------------------|------------|------|-------|-------|-------|-----|-------|
| | Q10 | Q50 | Q90 | Mean | Std | Min | Max |
| # Features | 9 | 21 | 129 | 125 | 374 | 4 | 1776 |
| # Classes | 2 | 2 | 3 | 262 | 1577 | 2 | 9856 |
| Missing Values (Ratio) | 0 | 0 | 0.056 | 0.027 | 0.109 | 0 | 0.672 |
| Categorical Features (Ratio) | 0 | 0.47 | 1 | 0.45 | 0.366 | 0 | 1 |
| Features w/ Missing Values (Ratio) | 0 | 0 | 0.696 | 0.139 | 0.294 | 0 | 0.994 |



(a) AUC on the TabArena-CLS benchmark.



(b) Accuracy on the TabArena-CLS benchmark.



(c) F1-score on the TabArena-CLS benchmark.

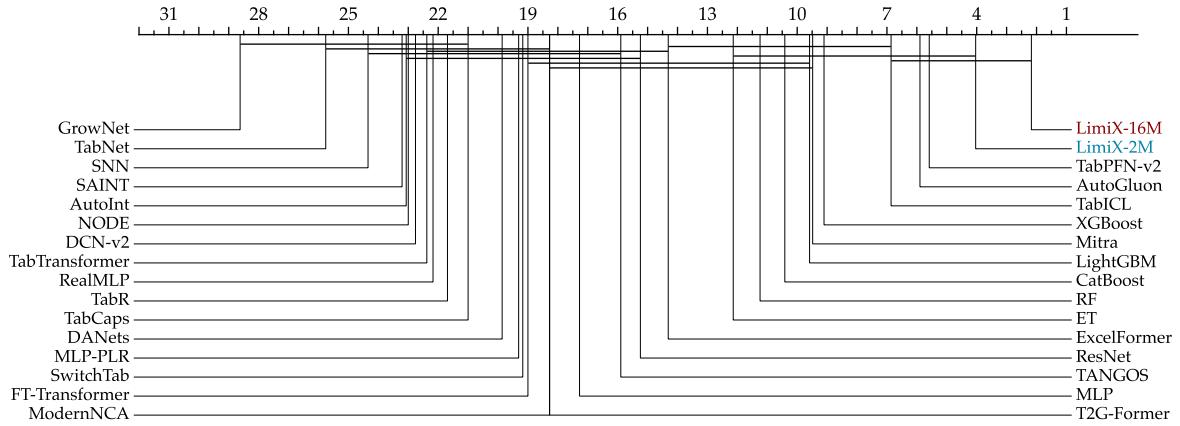
Figure 11: Critical difference diagrams on the TabArena-CLS benchmark.

Table 9: Classification results on the PFN-CLS benchmark. The best scores are shown in bold.

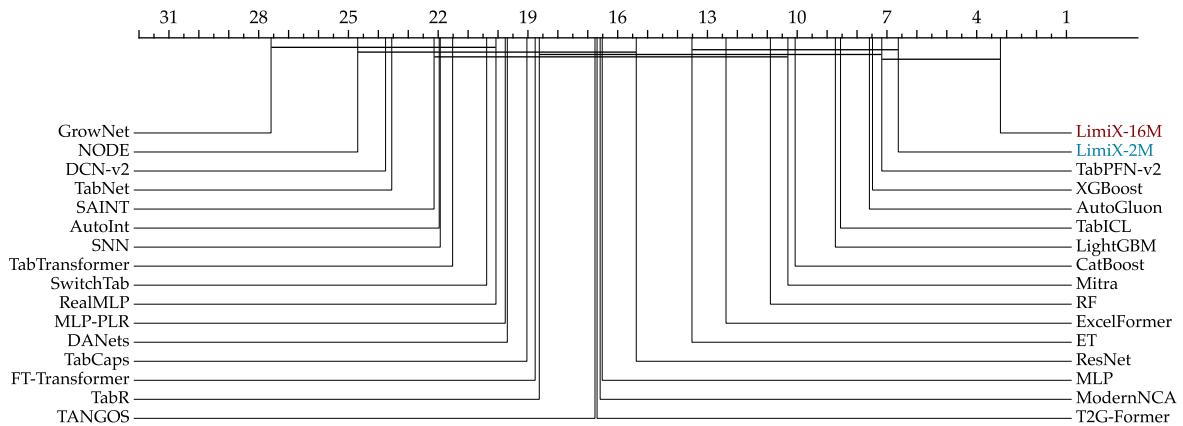
| Model | PFN-CLS | | | | | |
|----------------|--------------------|---------------------|-------------------|----------------------|-----------------------|---------------------|
| | Mean | | | Rank | | |
| | AUC (\uparrow) | Acc. (\uparrow) | F1 (\uparrow) | AUC (\downarrow) | Acc. (\downarrow) | F1 (\downarrow) |
| Limix-16M | 0.923 | 0.862 | 0.786 | 2.172 | 3.207 | 3.103 |
| Limix-2M | 0.913 | 0.848 | 0.766 | 4.034 | 6.621 | 5.897 |
| TabPFN-v2 | 0.910 | 0.845 | 0.756 | 5.586 | 7.172 | 7.138 |
| AutoGluon | 0.906 | 0.835 | 0.738 | 5.897 | 7.586 | 7.379 |
| TabICL | 0.903 | 0.832 | 0.742 | 6.862 | 8.552 | 9.034 |
| XGBoost | 0.898 | 0.831 | 0.733 | 9.103 | 7.483 | 8.448 |
| Mitra | 0.897 | 0.826 | 0.719 | 9.483 | 10.310 | 12.103 |
| LightGBM | 0.893 | 0.826 | 0.725 | 9.586 | 8.724 | 9.345 |
| CatBoost | 0.895 | 0.819 | 0.720 | 10.414 | 10.069 | 11.414 |
| RF | 0.896 | 0.822 | 0.721 | 11.241 | 10.897 | 12.103 |
| ET | 0.893 | 0.809 | 0.675 | 12.138 | 13.517 | 15.483 |
| ExcelFormer | 0.883 | 0.812 | 0.713 | 14.310 | 12.379 | 13.759 |
| ResNet | 0.869 | 0.801 | 0.700 | 15.241 | 15.379 | 14.828 |
| TANGOS | 0.865 | 0.797 | 0.698 | 15.897 | 16.759 | 15.207 |
| MLP | 0.866 | 0.795 | 0.695 | 17.276 | 16.517 | 16.138 |
| T2G-Former | 0.848 | 0.792 | 0.688 | 18.276 | 16.690 | 16.759 |
| ModernNCA | 0.860 | 0.798 | 0.692 | 18.276 | 16.586 | 15.586 |
| FT-Transformer | 0.849 | 0.789 | 0.683 | 19.000 | 18.759 | 17.448 |
| SwitchTab | 0.858 | 0.776 | 0.626 | 19.172 | 20.379 | 20.000 |
| MLP-PLR | 0.848 | 0.786 | 0.650 | 19.310 | 19.759 | 20.931 |
| DANets | 0.844 | 0.770 | 0.631 | 19.862 | 19.690 | 20.862 |
| TabCaps | 0.834 | 0.788 | 0.636 | 21.000 | 19.034 | 20.138 |
| TabR | 0.842 | 0.789 | 0.688 | 21.690 | 18.621 | 17.276 |
| RealMLP | 0.829 | 0.780 | 0.678 | 22.172 | 20.069 | 19.138 |
| TabTransformer | 0.821 | 0.761 | 0.604 | 22.379 | 21.517 | 23.310 |
| DCN-v2 | 0.846 | 0.771 | 0.633 | 22.759 | 23.759 | 24.069 |
| NODE | 0.844 | 0.754 | 0.535 | 23.000 | 24.690 | 26.966 |
| AutoInt | 0.838 | 0.772 | 0.640 | 23.069 | 21.966 | 22.034 |
| SAINT | 0.708 | 0.669 | 0.563 | 23.207 | 22.138 | 21.207 |
| SNN | 0.831 | 0.762 | 0.595 | 24.345 | 21.931 | 23.966 |
| TabNet | 0.825 | 0.768 | 0.631 | 25.759 | 23.552 | 24.759 |
| GrowNet | 0.756 | 0.689 | 0.497 | 28.621 | 27.586 | 27.138 |

Table 10: Statistical profile of the benchmark PFN-CLS, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively; for categorical feature statistics, we only consider features that are either string-typed or have fewer than 10 unique values.

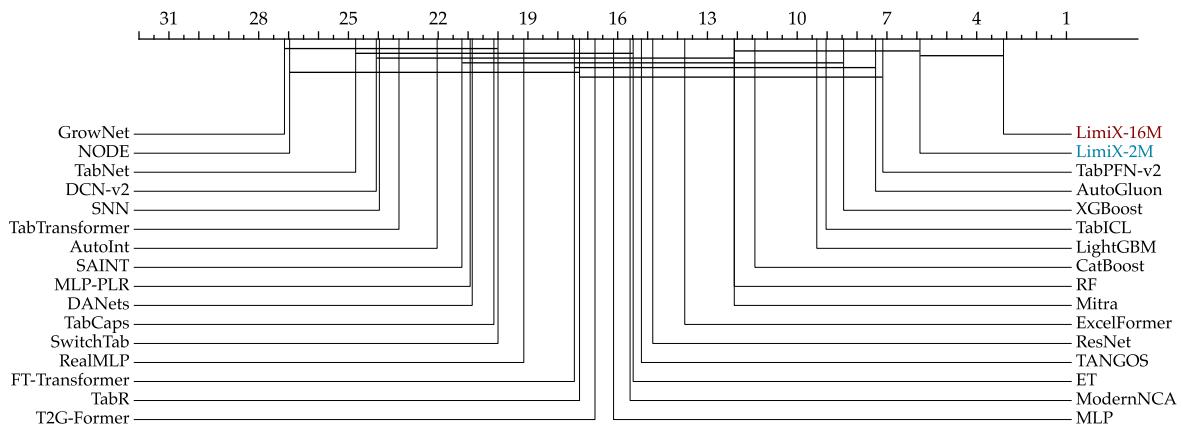
| Metric | Statistics | | | | | | |
|------------------------------------|------------|-------|-------|-------|-------|-----|-------|
| | Q10 | Q50 | Q90 | Mean | Std | Min | Max |
| # Features | 6 | 21 | 187 | 58 | 80 | 4 | 308 |
| # Classes | 2 | 2 | 7 | 4 | 2 | 2 | 10 |
| Missing Values (Ratio) | 0 | 0 | 0 | 0.001 | 0.006 | 0 | 0.032 |
| Categorical Features (Ratio) | 0 | 0.081 | 0.956 | 0.291 | 0.381 | 0 | 1 |
| Features w/ Missing Values (Ratio) | 0 | 0 | 0 | 0.016 | 0.086 | 0 | 0.474 |



(a) AUC on the PFN-CLS benchmark



(b) Accuracy on the PFN-CLS benchmark



(c) F1-score on the PFN-CLS benchmark

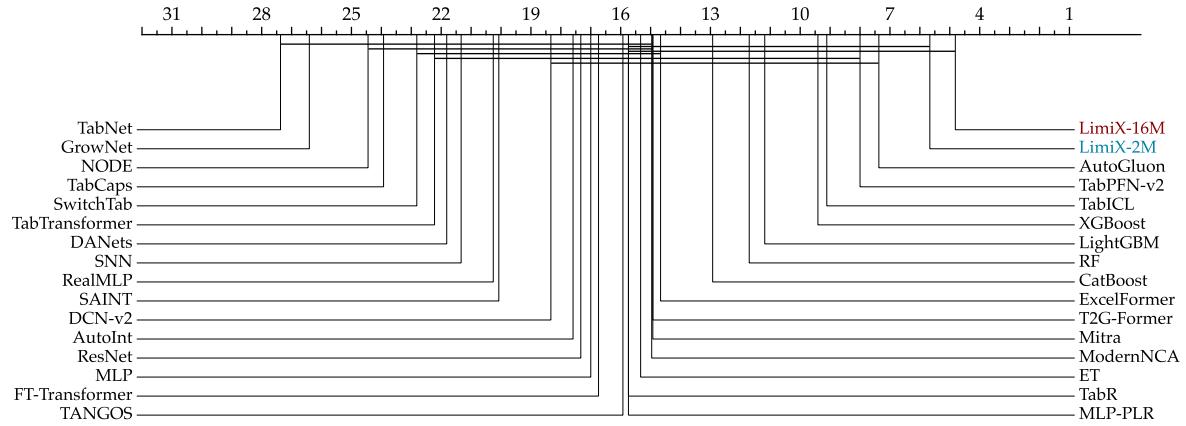
Figure 12: Critical difference diagrams on PFN-CLS benchmark

Table 11: Classification results on the TabZilla benchmark. The best scores are shown in bold.

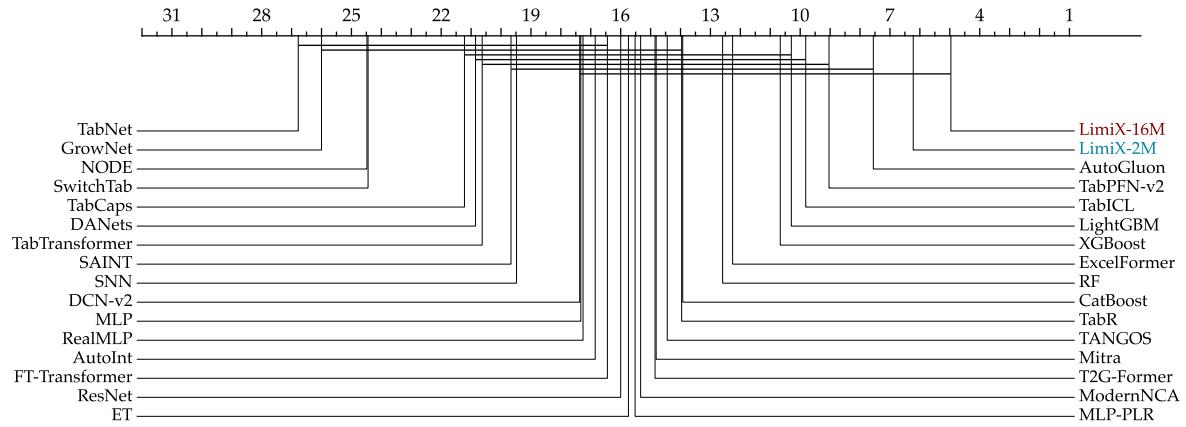
| Model | TabZilla | | | | | |
|----------------|--------------------|---------------------|-------------------|----------------------|-----------------------|---------------------|
| | Mean | | | Rank | | |
| | AUC (\uparrow) | Acc. (\uparrow) | F1 (\uparrow) | AUC (\downarrow) | Acc. (\downarrow) | F1 (\downarrow) |
| Limix-16M | 0.943 | 0.885 | 0.836 | 4.815 | 4.963 | 6.185 |
| Limix-2M | 0.938 | 0.883 | 0.832 | 5.667 | 6.222 | 6.630 |
| AutoGluon | 0.933 | 0.871 | 0.803 | 7.370 | 7.556 | 9.148 |
| TabPFN-v2 | 0.929 | 0.863 | 0.797 | 8.000 | 9.037 | 9.481 |
| TabICL | 0.933 | 0.864 | 0.803 | 9.111 | 9.815 | 10.815 |
| XGBoost | 0.929 | 0.863 | 0.789 | 9.407 | 10.667 | 11.926 |
| LightGBM | 0.927 | 0.863 | 0.796 | 11.185 | 10.296 | 10.852 |
| RF | 0.924 | 0.852 | 0.773 | 11.704 | 12.593 | 13.185 |
| CatBoost | 0.922 | 0.848 | 0.780 | 12.926 | 13.926 | 13.926 |
| ExcelFormer | 0.915 | 0.861 | 0.802 | 14.667 | 12.259 | 12.296 |
| Mitra | 0.915 | 0.841 | 0.758 | 14.926 | 14.815 | 15.778 |
| T2G-Former | 0.909 | 0.852 | 0.790 | 14.926 | 14.852 | 15.148 |
| ModernNCA | 0.907 | 0.850 | 0.794 | 14.963 | 15.333 | 14.630 |
| ET | 0.912 | 0.837 | 0.745 | 15.333 | 15.741 | 17.037 |
| TabR | 0.904 | 0.853 | 0.793 | 15.741 | 13.963 | 14.370 |
| MLP-PLR | 0.906 | 0.847 | 0.773 | 15.741 | 15.519 | 16.704 |
| TANGOS | 0.909 | 0.841 | 0.776 | 15.926 | 14.444 | 14.963 |
| FT-Transformer | 0.903 | 0.842 | 0.769 | 16.741 | 16.444 | 16.074 |
| MLP | 0.903 | 0.825 | 0.747 | 17.000 | 17.333 | 17.852 |
| ResNet | 0.908 | 0.834 | 0.769 | 17.333 | 16.000 | 15.556 |
| AutoInt | 0.896 | 0.833 | 0.748 | 17.593 | 16.852 | 17.370 |
| DCN-v2 | 0.904 | 0.844 | 0.781 | 18.333 | 17.370 | 17.889 |
| SAINT | 0.824 | 0.764 | 0.680 | 20.074 | 19.667 | 19.370 |
| RealMLP | 0.892 | 0.846 | 0.781 | 20.259 | 17.259 | 16.556 |
| SNN | 0.874 | 0.816 | 0.706 | 21.333 | 19.481 | 20.667 |
| DANets | 0.881 | 0.800 | 0.712 | 21.815 | 20.852 | 21.444 |
| TabTransformer | 0.814 | 0.759 | 0.659 | 22.222 | 20.630 | 21.593 |
| SwitchTab | 0.860 | 0.764 | 0.660 | 22.815 | 24.444 | 24.111 |
| TabCaps | 0.887 | 0.816 | 0.729 | 23.926 | 21.222 | 21.556 |
| NODE | 0.869 | 0.784 | 0.633 | 24.444 | 24.481 | 26.519 |
| GrowNet | 0.829 | 0.732 | 0.651 | 26.407 | 26.000 | 25.037 |
| TabNet | 0.860 | 0.771 | 0.668 | 27.370 | 26.778 | 26.407 |

Table 12: Statistical profile of the benchmark TabZilla, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively; for categorical feature statistics, we only consider features that are either string-typed or have fewer than 10 unique values.

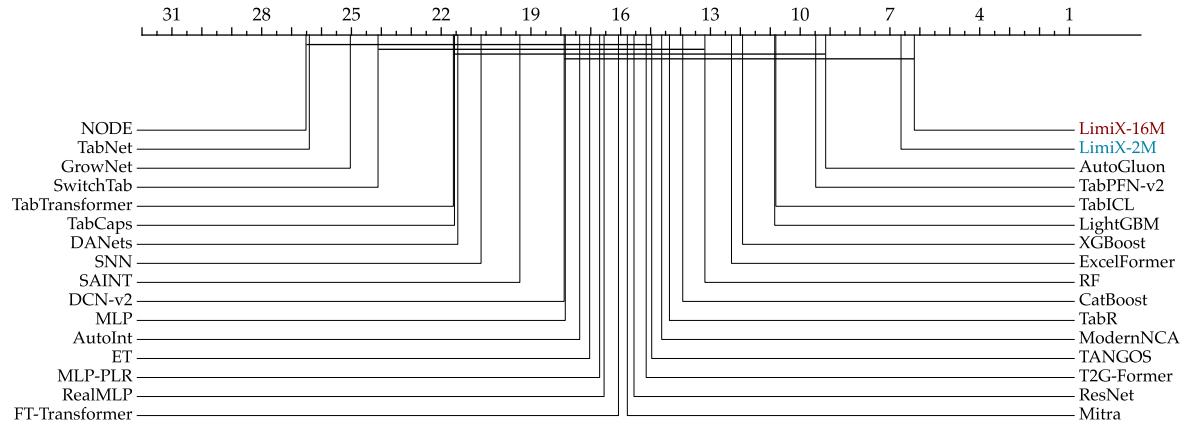
| Metric | Statistics | | | | | | |
|------------------------------------|------------|-------|-------|-------|-------|-----|-------|
| | Q10 | Q50 | Q90 | Mean | Std | Min | Max |
| # Features | 6 | 24 | 132 | 226 | 756 | 4 | 4296 |
| # Classes | 2 | 2 | 10 | 7 | 16 | 2 | 100 |
| Missing Values (Ratio) | 0 | 0 | 0.078 | 0.022 | 0.059 | 0 | 0.205 |
| Categorical Features (Ratio) | 0 | 0.528 | 1 | 0.474 | 0.39 | 0 | 1 |
| Features w/ Missing Values (Ratio) | 0 | 0 | 0.483 | 0.102 | 0.218 | 0 | 0.808 |



(a) AUC on the TabZilla benchmark.



(b) Accuracy on the TabZilla benchmark.



(c) F1-score on the TabZilla benchmark.

Figure 13: Critical difference diagrams on the TabZilla benchmark.

7.2 Regression

Benchmark. For the quantitative evaluation of regression performance, we leverage four benchmarks, including three open-source benchmark, TALENT-REG (Liu et al., 2024), PFN-REG (Hollmann et al., 2025), TabArena-REG (Erickson et al., 2025), and CTR23 (Fischer et al., 2023). After applying filters to exclude datasets containing more than 50,000 training samples or 10,000 features, we have 33 datasets from CTR23, 28 from PFN-REG, 13 from TabArena-REG, and 99 from TALENT-REG.

Similar to BCCO-CLS, we introduce BCCO-REG, a balanced regression benchmark comprising 50 datasets. For each dataset, we adopt the provided train-test split when available. If no predefined test set is available, we partition the data into a 70% training set and a 30% testing set randomly. Similar to the protocol employed in classification tasks, regression datasets with more than 50,000 training samples or 10,000 features are excluded.

Baselines. For comparison, we include tree-based methods, NN-based methods, AutoML frameworks, and ICL-based models, which are basically consistent with baselines employed in classification tasks described above. TabICL and TabCaps are excluded because it cannot handle regression tasks. We add another baseline, DNNR (Nader et al., 2022), which is specifically designed for regression. The model training procedure remains aligned with that employed in the classification experiments.

Metrics. To evaluate regression performance properly, we employ normalized RMSE and R^2 as the two evaluation metrics. We also analyze the ranks of models with respect to both metrics.

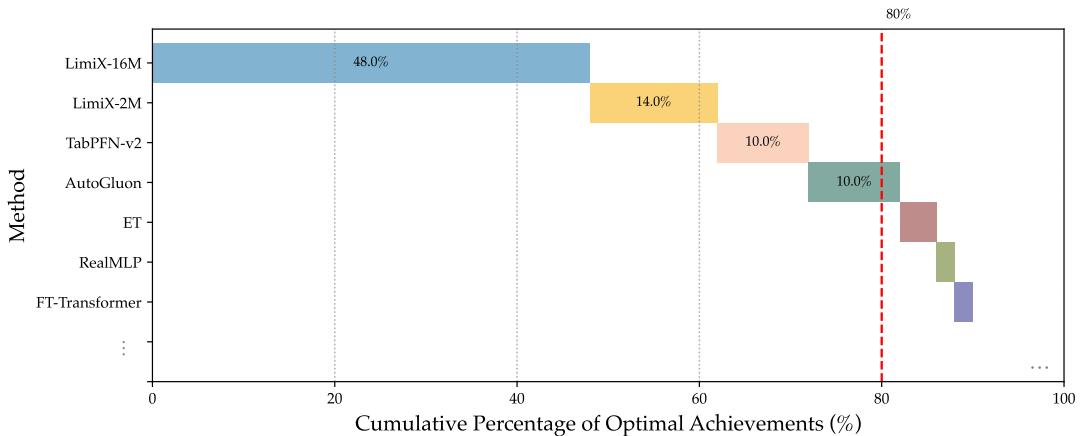


Figure 14: The proportion of models achieving the best R^2 . The length of each bar represents the proportion of 106 datasets in which a given method achieved the highest R^2 on the BCCO-REG benchmark.

Results. As shown by critical difference diagrams, i.e. Figures 15 to 19, LimiX-16M always achieves the best performance, outperforming strong baselines like AutoGluon, XGBoost, and TabPFN-v2. Quantitative results in Tables 13, 15, 17, 19 and 21 also confirms that LimiX-16M achieves state-of-the-art regression performance, leading in both normalized RMSE and R^2 . On the regression benchmarks, LimiX-2M exhibits predictive performance comparable to AutoGluon and TabPFN-v2, while requiring substantially less computational resources and runtime.

Subgroup analysis. Similar to the subgroup analysis in classification, for regression, we also use the sample subgroups when building BCCO-REG to perform stratified analyses, whose criteria include the number of training samples, the ratio between the number of samples and features (length-to-width ratio), and the proportion of categorical features. Figure 21 shows that LimiX-16M outperforms other methods across all subgroups compared with other methods. For Figures 21b to 21d, we can see that performance of Mitra substantially drops for the second subgroup, while LimiX-16M do not exhibit a large drop.

Table 13: Regression results on the BCCO-REG benchmark. The best scores are shown in bold.

| Model | BCCO-REG | | | |
|----------------|-------------------------------|-----------------------|---------------------------------|-----------------------|
| | Mean | | Rank | |
| | R ² (\uparrow) | RMSE (\downarrow) | R ² (\downarrow) | RMSE (\downarrow) |
| LimiX-16M | 0.794 | 0.386 | 3.340 | 4.120 |
| AutoGluon | 0.781 | 0.398 | 4.260 | 4.220 |
| TabPFN-v2 | 0.772 | 0.404 | 5.480 | 5.380 |
| LimiX-2M | 0.785 | 0.392 | 5.720 | 5.600 |
| XGBoost | 0.764 | 0.415 | 8.520 | 8.560 |
| LightGBM | 0.715 | 0.423 | 8.760 | 8.720 |
| ET | 0.757 | 0.431 | 10.720 | 10.660 |
| CatBoost | 0.741 | 0.427 | 11.320 | 11.160 |
| RF | 0.752 | 0.438 | 12.080 | 12.040 |
| ExcelFormer | 0.743 | 0.443 | 12.260 | 12.320 |
| RealMLP | 0.725 | 0.441 | 13.240 | 13.200 |
| T2G-Former | 0.743 | 0.442 | 13.680 | 13.640 |
| FT-Transformer | 0.737 | 0.448 | 14.240 | 14.220 |
| DCN-v2 | 0.739 | 0.448 | 14.600 | 14.600 |
| TabR | 0.733 | 0.448 | 15.300 | 15.360 |
| ResNet | 0.720 | 0.468 | 15.940 | 15.900 |
| ModernNCA | 0.598 | 0.471 | 16.060 | 15.960 |
| TANGOS | 0.719 | 0.468 | 16.620 | 16.620 |
| Mitra | 0.667 | 0.474 | 16.620 | 16.540 |
| MLP-PLR | 0.734 | 0.453 | 16.660 | 16.660 |
| SAINT | 0.701 | 0.481 | 17.440 | 17.460 |
| MLP | 0.701 | 0.487 | 17.860 | 17.920 |
| AutoInt | 0.724 | 0.465 | 18.120 | 18.040 |
| NODE | 0.643 | 0.543 | 21.000 | 20.920 |
| TabNet | 0.670 | 0.516 | 21.600 | 21.540 |
| DNNR | -2.152 | 1.329 | 21.780 | 21.800 |
| SNN | 0.434 | 0.720 | 26.200 | 26.200 |
| GrowNet | 0.201 | 0.864 | 27.840 | 27.820 |
| DANets | 0.005 | 0.979 | 29.440 | 29.480 |
| TabTransformer | -0.000 | 0.981 | 29.600 | 29.640 |
| SwitchTab | 0.001 | 0.981 | 29.700 | 29.700 |

Table 14: Statistical profile of the benchmark BCCO-REG, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively; for categorical feature statistics, we only consider features that are either string-typed or have fewer than 10 unique values.

| Metric | Statistics | | | | | | |
|------------------------------------|------------|------|------|-------|-------|-----|-------|
| | Q10 | Q50 | Q90 | Mean | Std | Min | Max |
| # Features | 6 | 12 | 33 | 16 | 14 | 4 | 81 |
| Missing Values (Ratio) | 0 | 0 | 0 | 0 | 0 | 0 | 0.001 |
| Categorical Features (Ratio) | 0 | 0.16 | 0.65 | 0.242 | 0.264 | 0 | 0.967 |
| Features w/ Missing Values (Ratio) | 0 | 0 | 0 | 0.008 | 0.051 | 0 | 0.333 |

Table 15: Regression results on the TALENT-REG benchmark. The best scores are shown in bold.

| Model | TALENT-REG | | | |
|----------------|-------------------------------|-----------------------|---------------------------------|-----------------------|
| | Mean | | Rank | |
| | R ² (\uparrow) | RMSE (\downarrow) | R ² (\downarrow) | RMSE (\downarrow) |
| Limix-16M | 0.735 | 0.433 | 2.808 | 3.364 |
| AutoGluon | 0.722 | 0.448 | 4.818 | 4.929 |
| TabPFN-v2 | 0.695 | 0.465 | 6.192 | 6.101 |
| Limix-2M | 0.721 | 0.451 | 6.263 | 6.182 |
| XGBoost | 0.710 | 0.462 | 7.424 | 7.354 |
| LightGBM | 0.707 | 0.461 | 8.182 | 8.152 |
| ET | 0.696 | 0.476 | 9.970 | 9.919 |
| CatBoost | 0.700 | 0.471 | 10.424 | 10.354 |
| RF | 0.697 | 0.474 | 10.556 | 10.556 |
| ExcelFormer | 0.653 | 0.512 | 13.313 | 13.354 |
| RealMLP | 0.656 | 0.510 | 13.444 | 13.444 |
| T2G-Former | 0.656 | 0.512 | 14.202 | 14.141 |
| TabR | 0.651 | 0.516 | 15.283 | 15.242 |
| DCN-v2 | -0.361 | 0.818 | 15.576 | 15.616 |
| FT-Transformer | 0.648 | 0.519 | 15.848 | 15.798 |
| Mitra | 0.602 | 0.547 | 15.848 | 15.848 |
| MLP-PLR | 0.653 | 0.521 | 16.101 | 16.061 |
| ResNet | 0.562 | 0.550 | 16.616 | 16.586 |
| ModernNCA | 0.633 | 0.530 | 16.646 | 16.626 |
| TANGOS | 0.592 | 0.547 | 17.131 | 17.101 |
| MLP | 0.556 | 0.564 | 17.596 | 17.636 |
| SAINT | -1.541 | 0.571 | 18.384 | 18.253 |
| AutoInt | 0.636 | 0.538 | 19.040 | 19.051 |
| NODE | 0.568 | 0.600 | 20.515 | 20.374 |
| TabNet | 0.576 | 0.586 | 21.747 | 21.687 |
| DNNR | -9.172 | 2.528 | 23.505 | 23.545 |
| SNN | 0.344 | 0.777 | 25.465 | 25.485 |
| GrowNet | -0.182 | 0.920 | 27.131 | 27.162 |
| DANets | 0.005 | 0.998 | 28.434 | 28.455 |
| TabTransformer | 0.001 | 1.001 | 28.576 | 28.626 |
| SwitchTab | -0.002 | 1.002 | 28.949 | 28.990 |

Table 16: Statistical profile of the benchmark TALENT-REG, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively; for categorical feature statistics, we only consider features that are either string-typed or have fewer than 10 unique values.

| Metric | Statistics | | | | | | |
|------------------------------------|------------|-------|-------|-------|-------|-----|-------|
| | Q10 | Q50 | Q90 | Mean | Std | Min | Max |
| # Features | 6 | 11 | 79 | 30 | 49 | 4 | 266 |
| Missing Values (Ratio) | 0 | 0 | 0 | 0.007 | 0.034 | 0 | 0.283 |
| Categorical Features (Ratio) | 0 | 0.131 | 0.667 | 0.238 | 0.283 | 0 | 1 |
| Features w/ Missing Values (Ratio) | 0 | 0 | 0.013 | 0.043 | 0.161 | 0 | 0.875 |

Table 17: Regression results on the CTR23 benchmark. The best scores are shown in bold.

| Model | CTR23 | | | |
|----------------|-------------------------------|-----------------------|---------------------------------|-----------------------|
| | Mean | | Rank | |
| | R ² (\uparrow) | RMSE (\downarrow) | R ² (\downarrow) | RMSE (\downarrow) |
| Limix-16M | 0.745 | 0.477 | 4.061 | 4.152 |
| AutoGluon | 0.725 | 0.497 | 5.303 | 5.212 |
| Limix-2M | 0.730 | 0.495 | 6.879 | 6.909 |
| TabPFN-v2 | 0.716 | 0.503 | 7.636 | 7.515 |
| XGBoost | 0.712 | 0.511 | 8.697 | 8.758 |
| LightGBM | 0.706 | 0.516 | 9.727 | 9.818 |
| ET | 0.697 | 0.535 | 10.515 | 10.576 |
| CatBoost | 0.700 | 0.528 | 11.212 | 11.212 |
| RF | 0.694 | 0.539 | 11.485 | 11.455 |
| ExcelFormer | 0.665 | 0.556 | 12.515 | 12.606 |
| T2G-Former | 0.674 | 0.544 | 13.879 | 13.848 |
| DCN-v2 | 0.670 | 0.545 | 14.515 | 14.576 |
| ResNet | 0.645 | 0.587 | 14.545 | 14.606 |
| RealMLP | 0.661 | 0.549 | 14.576 | 14.606 |
| FT-Transformer | 0.667 | 0.549 | 14.848 | 14.939 |
| TabR | 0.671 | 0.543 | 14.939 | 14.879 |
| MLP-PLR | 0.672 | 0.553 | 15.788 | 15.818 |
| MLP | 0.608 | 0.623 | 15.848 | 15.848 |
| ModernNCA | 0.667 | 0.550 | 15.909 | 15.576 |
| Mitra | 0.624 | 0.583 | 16.727 | 16.848 |
| TANGOS | 0.642 | 0.586 | 16.727 | 16.758 |
| SAINT | 0.654 | 0.561 | 16.758 | 16.667 |
| AutoInt | 0.655 | 0.568 | 17.848 | 17.818 |
| NODE | 0.568 | 0.666 | 20.545 | 20.394 |
| TabNet | 0.605 | 0.623 | 20.576 | 20.636 |
| DNNR | -2.969 | 1.651 | 22.636 | 22.667 |
| SNN | 0.369 | 0.834 | 25.667 | 25.697 |
| GrowNet | 0.185 | 0.944 | 27.606 | 27.576 |
| DANets | 0.001 | 1.052 | 28.970 | 28.970 |
| TabTransformer | 0.000 | 1.053 | 29.000 | 29.000 |
| SwitchTab | -0.006 | 1.057 | 30.061 | 30.061 |

Table 18: Statistical profile of the benchmark CTR23, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively; for categorical feature statistics, we only consider features that are either string-typed or have fewer than 10 unique values.

| Metric | Statistics | | | | | | |
|------------------------------------|------------|-------|-------|-------|-------|-----|-------|
| | Q10 | Q50 | Q90 | Mean | Std | Min | Max |
| # Features | 6 | 11 | 39 | 20 | 22 | 5 | 116 |
| Missing Values (Ratio) | 0 | 0 | 0 | 0.008 | 0.036 | 0 | 0.209 |
| Categorical Features (Ratio) | 0 | 0.143 | 0.842 | 0.312 | 0.345 | 0 | 1 |
| Features w/ Missing Values (Ratio) | 0 | 0 | 0 | 0.011 | 0.05 | 0 | 0.286 |

Table 19: Regression results on the PFN-REG benchmark. The best scores are shown in bold.

| Model | PFN-REG | | | |
|----------------|-------------------------------|-----------------------|---------------------------------|-----------------------|
| | Mean | | Rank | |
| | R ² (\uparrow) | RMSE (\downarrow) | R ² (\downarrow) | RMSE (\downarrow) |
| LimiX-16M | 0.692 | 0.461 | 4.607 | 5.464 |
| LimiX-2M | 0.696 | 0.458 | 5.250 | 5.250 |
| AutoGluon | 0.677 | 0.482 | 7.321 | 7.321 |
| TabPFN-v2 | 0.676 | 0.475 | 7.857 | 7.607 |
| XGBoost | 0.671 | 0.487 | 9.464 | 9.464 |
| RealMLP | 0.664 | 0.484 | 10.250 | 10.179 |
| LightGBM | 0.664 | 0.496 | 10.464 | 10.393 |
| CatBoost | 0.661 | 0.498 | 11.429 | 11.393 |
| ET | 0.652 | 0.515 | 11.500 | 11.429 |
| Mitra | 0.630 | 0.531 | 13.679 | 13.714 |
| RF | 0.643 | 0.524 | 13.821 | 13.893 |
| T2G-Former | 0.640 | 0.503 | 13.964 | 13.857 |
| ExcelFormer | 0.646 | 0.513 | 14.036 | 14.071 |
| FT-Transformer | 0.636 | 0.509 | 14.964 | 14.821 |
| MLP | 0.577 | 0.576 | 14.964 | 15.107 |
| ModernNCA | 0.645 | 0.508 | 15.107 | 15.286 |
| ResNet | 0.599 | 0.553 | 15.286 | 15.321 |
| MLP-PLR | 0.638 | 0.513 | 15.321 | 15.393 |
| TabR | 0.636 | 0.509 | 15.393 | 15.250 |
| TANGOS | 0.587 | 0.562 | 15.607 | 15.750 |
| DCN-v2 | 0.638 | 0.511 | 15.607 | 15.429 |
| SAINT | -8.088 | 0.612 | 17.214 | 16.821 |
| AutoInt | 0.610 | 0.539 | 19.143 | 19.036 |
| NODE | 0.496 | 0.666 | 20.964 | 20.786 |
| TabNet | 0.430 | 0.643 | 22.071 | 22.000 |
| DNNR | -7.850 | 2.224 | 23.286 | 23.429 |
| SNN | 0.378 | 0.756 | 25.179 | 25.214 |
| GrowNet | 0.096 | 0.942 | 26.607 | 26.571 |
| DANets | 0.001 | 0.997 | 27.893 | 27.929 |
| SwitchTab | -0.026 | 1.011 | 28.393 | 28.429 |
| TabTransformer | -0.021 | 1.007 | 29.357 | 29.393 |

Table 20: Statistical profile of the benchmark PFN-REG, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively; for categorical feature statistics, we only consider features that are either string-typed or have fewer than 10 unique values.

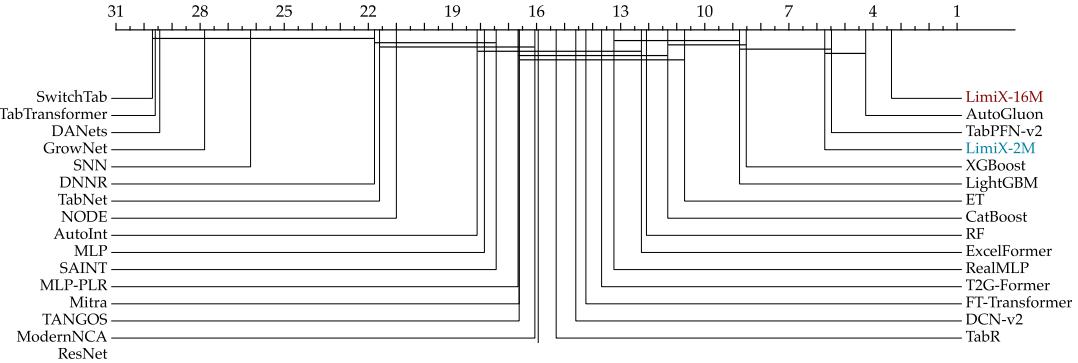
| Metric | Statistics | | | | | | |
|------------------------------------|------------|-------|-------|-------|-------|-----|-------|
| | Q10 | Q50 | Q90 | Mean | Std | Min | Max |
| # Features | 6 | 16 | 176 | 66 | 93 | 3 | 376 |
| Missing Values (Ratio) | 0 | 0 | 0.089 | 0.029 | 0.076 | 0 | 0.335 |
| Categorical Features (Ratio) | 0 | 0.209 | 0.949 | 0.355 | 0.381 | 0 | 1 |
| Features w/ Missing Values (Ratio) | 0 | 0 | 0.211 | 0.059 | 0.169 | 0 | 0.841 |

Table 21: Regression results on the TabArena-REG benchmark. The best scores are shown in bold.

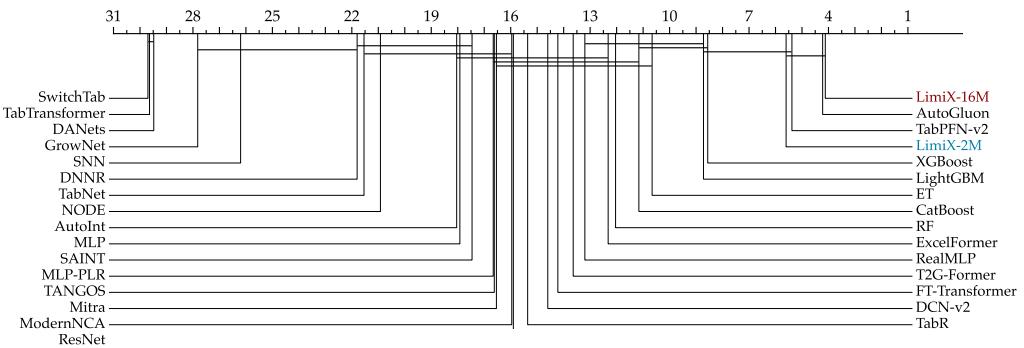
| Model | TabArena-REG | | | |
|----------------|-------------------------------|-----------------------|---------------------------------|-----------------------|
| | Mean | | Rank | |
| | R ² (\uparrow) | RMSE (\downarrow) | R ² (\downarrow) | RMSE (\downarrow) |
| LimiX-16M | 0.796 | 0.406 | 2.923 | 2.923 |
| AutoGluon | 0.791 | 0.414 | 3.154 | 3.154 |
| LimiX-2M | 0.788 | 0.413 | 3.846 | 3.846 |
| TabPFN-v2 | 0.777 | 0.422 | 5.385 | 5.385 |
| CatBoost | 0.774 | 0.431 | 6.231 | 6.231 |
| XGBoost | 0.778 | 0.430 | 6.231 | 6.231 |
| LightGBM | 0.771 | 0.435 | 6.769 | 6.769 |
| RF | 0.758 | 0.456 | 9.769 | 9.769 |
| ET | 0.746 | 0.464 | 10.231 | 10.154 |
| TabR | 0.729 | 0.476 | 13.692 | 13.692 |
| ExcelFormer | 0.681 | 0.521 | 13.923 | 13.923 |
| NODE | 0.665 | 0.542 | 14.538 | 14.538 |
| TANGOS | 0.673 | 0.534 | 15.077 | 15.077 |
| DCN-v2 | 0.718 | 0.486 | 15.385 | 15.462 |
| RealMLP | 0.719 | 0.487 | 15.385 | 15.385 |
| Mitra | 0.666 | 0.538 | 16.615 | 16.615 |
| ModernNCA | 0.712 | 0.496 | 17.231 | 17.231 |
| MLP-PLR | 0.714 | 0.496 | 17.231 | 17.231 |
| AutoInt | 0.705 | 0.503 | 17.923 | 17.923 |
| ResNet | 0.687 | 0.521 | 18.538 | 18.538 |
| MLP | 0.694 | 0.516 | 18.538 | 18.538 |
| SAINT | 0.712 | 0.498 | 19.000 | 19.000 |
| T2G-Former | 0.677 | 0.526 | 19.385 | 19.385 |
| TabNet | 0.641 | 0.564 | 19.615 | 19.615 |
| FT-Transformer | 0.626 | 0.572 | 22.077 | 22.077 |
| DNNR | -0.368 | 1.058 | 24.538 | 24.538 |
| SNN | 0.451 | 0.729 | 25.692 | 25.692 |
| GrowNet | 0.316 | 0.814 | 27.385 | 27.385 |
| TabTransformer | 0.016 | 0.993 | 29.333 | 29.333 |
| DANets | 0.012 | 0.998 | 29.462 | 29.462 |
| SwitchTab | 0.004 | 1.002 | 29.923 | 29.923 |

Table 22: Statistical profile of the benchmark TabArena-REG, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively; for categorical feature statistics, we only consider features that are either string-typed or have fewer than 10 unique values.

| Metric | Statistics | | | | | | |
|------------------------------------|------------|-------|-------|-------|--------|-----|------|
| | Q10 | Q50 | Q90 | Mean | Std | Min | Max |
| # Features | 6 | 9 | 67.8 | 92.23 | 280.67 | 5 | 1024 |
| Missing Values (Ratio) | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Categorical Features (Ratio) | 0 | 0.333 | 0.619 | 0.287 | 0.299 | 0 | 1 |
| Features w/ Missing Values (Ratio) | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

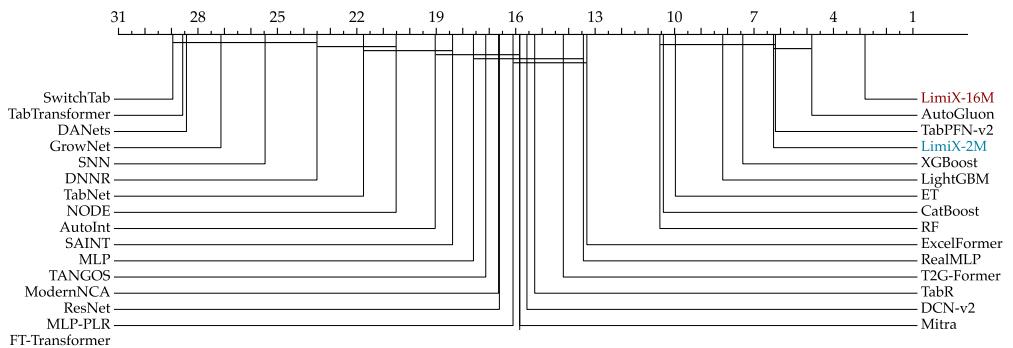


(a) R^2 on the BCCO-REG benchmark

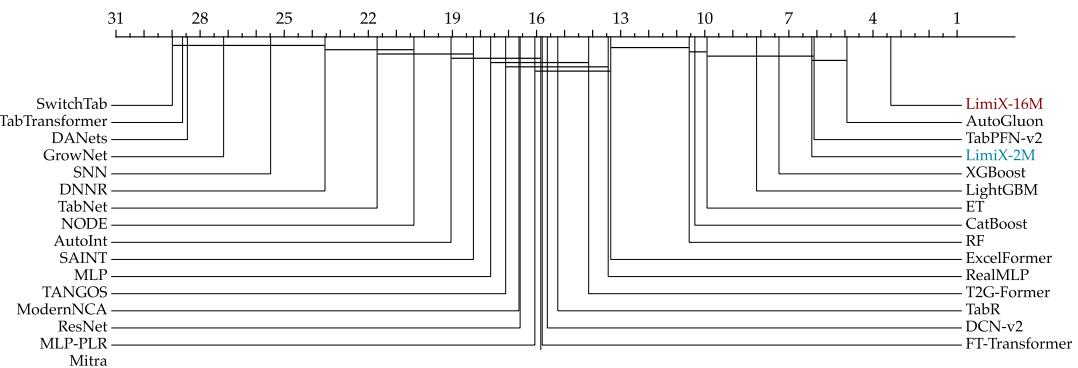


(b) RMSE on the BCCO-REG benchmark

Figure 15: Critical difference diagram on the BCCO-REG benchmark

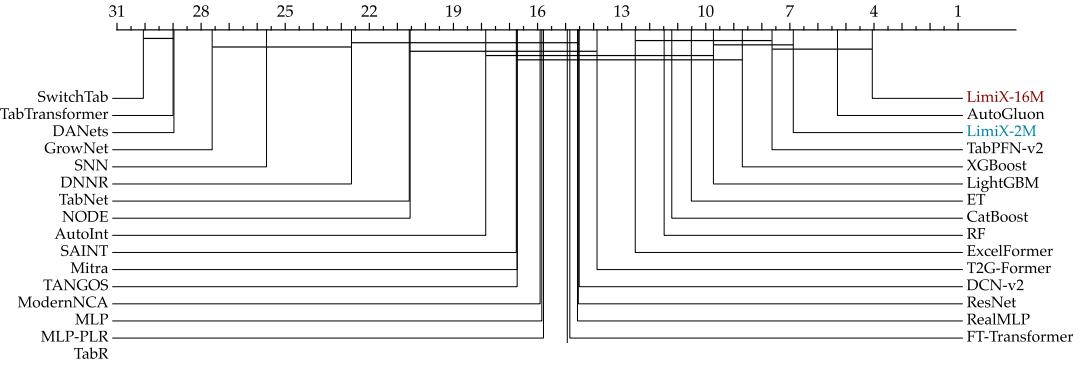


(a) R^2 on the TALENT-REG benchmark

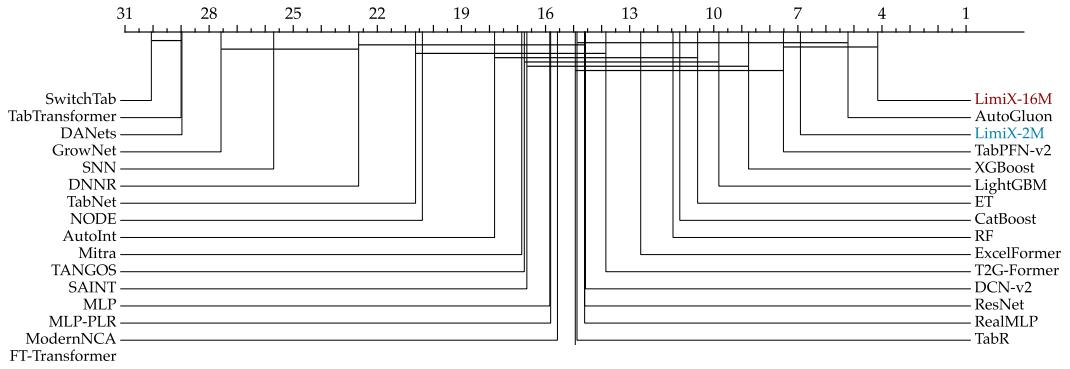


(b) RMSE on TALENT-REG benchmark

Figure 16: Critical difference diagram on the TALENT-REG benchmark

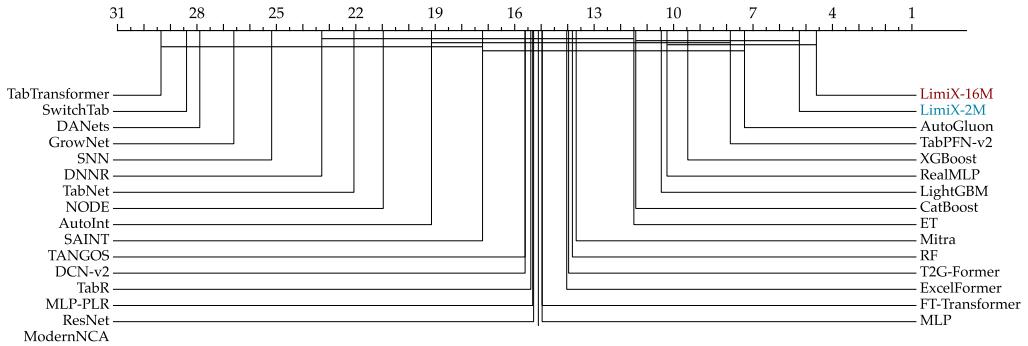


(a) R^2 on the CTR23 benchmark

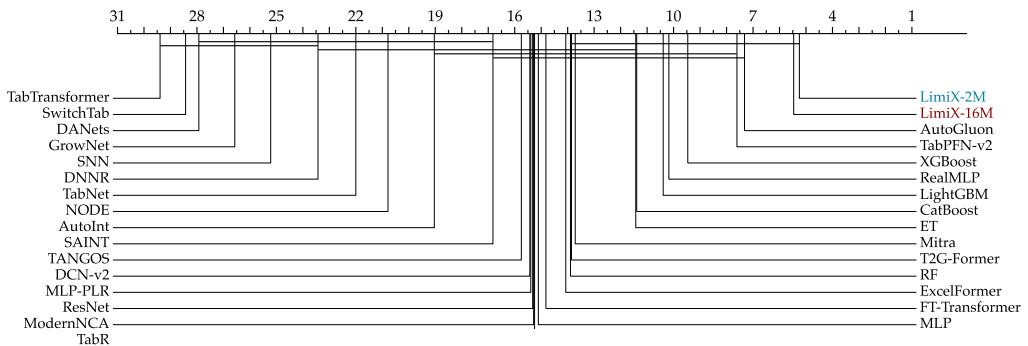


(b) RMSE on the CTR23 benchmark

Figure 17: Critical difference diagram on the CTR23 benchmark

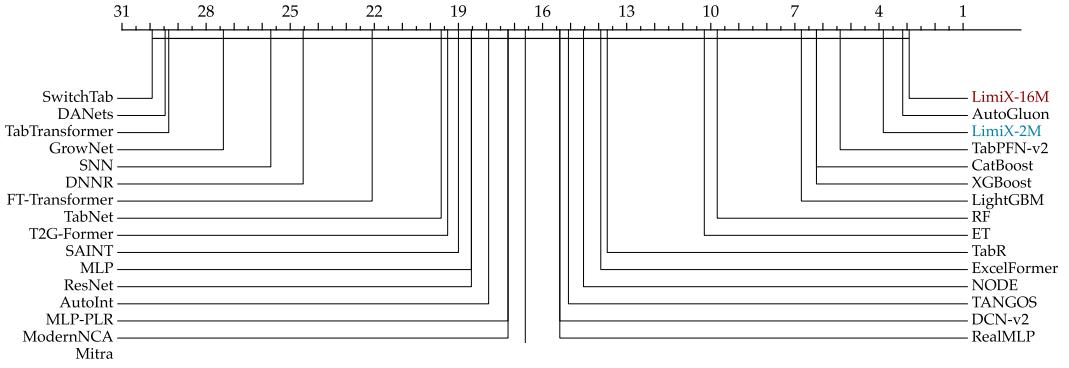


(a) R^2 on the PFN-REG benchmark

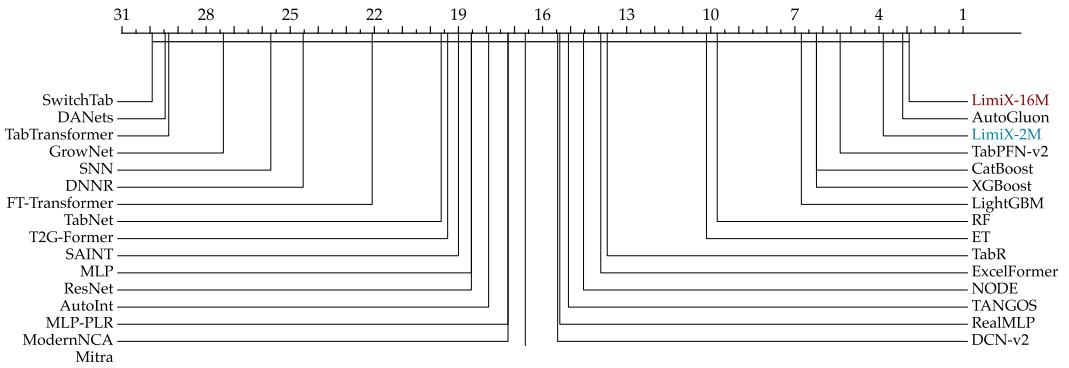


(b) RMSE on PFN-REG benchmark

Figure 18: Critical difference diagram on the PFN-REG benchmark



(a) R^2 on the TabArena-REG benchmark.



(b) RMSE on the TabArena-REG benchmark.

Figure 19: Critical difference diagrams on TabArena-REG benchmark.

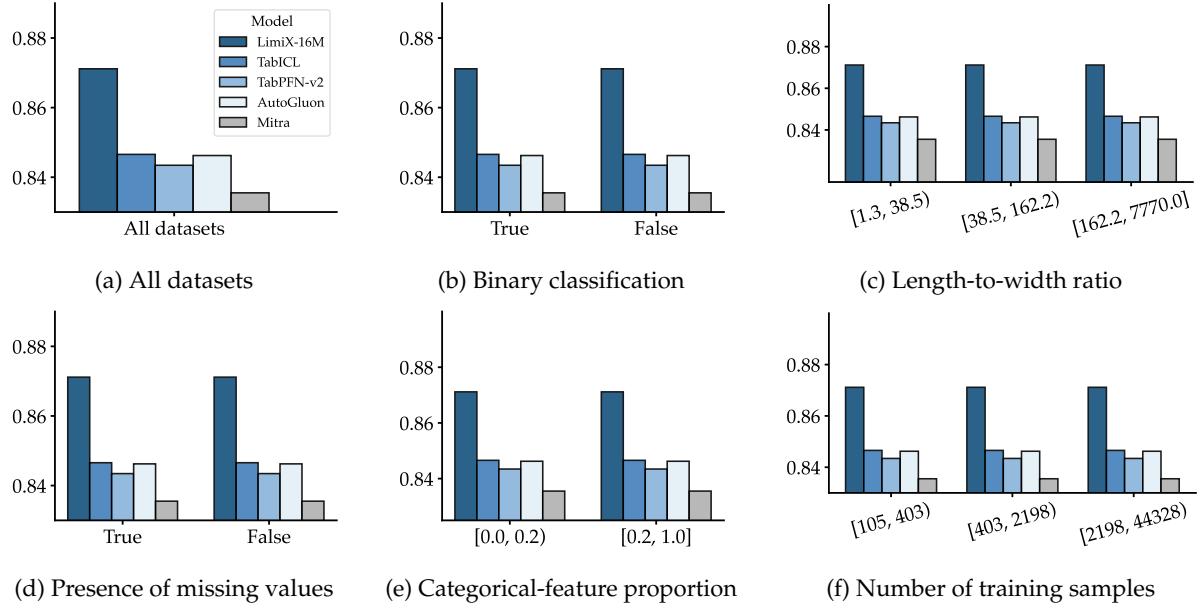


Figure 20: AUC on subsets with various sample size, number of classes, categorical–to-numerical feature ratios, missing values, and sample-to-feature ratios.

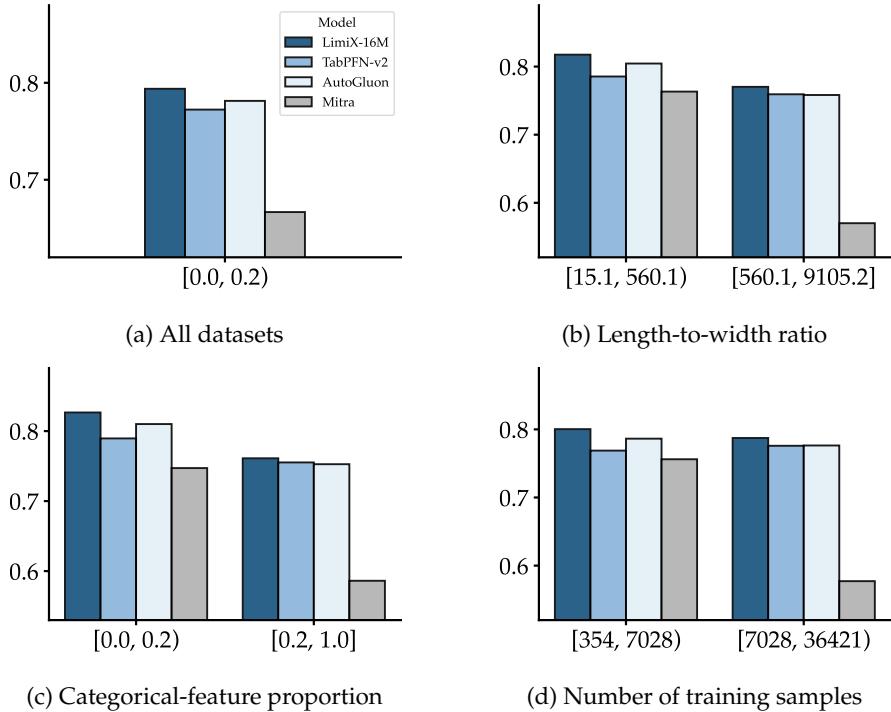


Figure 21: R^2 on subsets with various sample size, categorical–to-numerical feature ratios, and sample-to-feature ratios.

7.3 Missing Value Imputation

The ubiquity of missing values in tabular data is harmful to downstream tasks or statistical analyses (Lin & Tsai, 2020). Meanwhile, the problem of missing value imputation poses a bigger challenge compared with standard classification and regression tasks since it requires modeling of the joint distribution of $P(X, Y)$ instead of $P(Y|X)$ only. A noteworthy capability of LimiX models is missing value imputation, which naturally stems from the mask modeling during pretraining. Although there are existing deep learning approaches for missing value imputation (Jarrett et al., 2022; Zhang et al., 2025), they all require additional training on unseen datasets. In contrast, LimiX models are the first to perform missing value imputation on unseen datasets via in-context learning without additional training. This brings great convenience to the usage in downstream tasks.

We conduct experiments on the following datasets: Analcatdata BroadwayMult (Simonoff, 2003; Van-schoren, 2014a), Early Stage Diabetes 2020 (Islam et al., 2019; 2020), Forty Soybean Cultivars from Subsequent Harvests (de Oliveira et al., 2023; Rodrigues de Oliveira & Mario Zuffo, 2023), HCC survival (Santos et al., 2015b;a), Vehicle (Siebert, 1987; Mowforth & Shepherd, 1987), Eucalyptus (van Rijn, 2014), and Z-Alizadeh Sani (Alizadehsani et al., 2013). To simulate the scenario of missing value imputation, we randomly mask a fixed proportion α of cells in \mathbf{x}_{te} and try to recover values of these cells. In our experiments, we set $\alpha = 0.05$. Baselines include: Mean/Mode (use mean for continuous features and mode for categorical features), k-nearest neighbors (KNN), MICE (Van Buuren & Groothuis-Oudshoorn, 2011), MissForest (Stekhoven & Bühlmann, 2012), GAIN (Yoon et al., 2018), MIWAE (Mattei & Frellsen, 2019), and HyperImpute (Jarrett et al., 2022). For continuous features, we normalize each feature using MinMaxScaler and calculate RMSE between imputed values and ground truth values after normalization as the evaluation metric. For categorical features, the error rate is used as the evaluation metric. Table 23 shows that LimiX-16M consistently outperforms previous baselines, all of which require additional training or fitting.

Table 23: Evaluation of missing value imputation. For continuous features, we calculate RMSE. For categorical features, we calculate classification error. Our method outperforms previous missing value imputation methods.

| Metric | Regression RMSE (\downarrow) | | | | | | Classification Error (\downarrow) | |
|-------------|----------------------------------|----------------|--------------|--------------|--------------|--------------|---------------------------------------|--|
| Method | Analcatdata | Early Diabetes | Harvests | Hcc Survival | Vehicle | Eucalyptus | Z-Alizadeh Sani | |
| Mean/Mode | 0.321 | 0.235 | 0.179 | 0.351 | 0.209 | 0.804 | 0.200 | |
| KNN | 0.358 | 0.205 | 0.158 | 0.246 | 0.083 | 0.275 | 0.206 | |
| MICE | 0.294 | 0.244 | 0.155 | 0.234 | 0.102 | 0.627 | 0.181 | |
| MissForest | 0.203 | 0.223 | 0.136 | 0.215 | 0.107 | 0.137 | 0.156 | |
| GAIN | 0.299 | 0.328 | 0.196 | 0.413 | 0.102 | 0.647 | 0.175 | |
| MIWAE | 0.561 | 0.478 | 0.322 | 0.639 | 0.415 | 0.706 | 0.294 | |
| HyperImpute | 0.272 | 0.270 | 0.152 | 0.297 | 0.086 | 0.647 | 0.194 | |
| LimiX-16M | 0.194 | 0.161 | 0.104 | 0.188 | 0.064 | 0.118 | 0.131 | |

7.4 Robustness

Neural networks have been found to be vulnerable to various types of perturbations and attacks (Carlini & Wagner, 2017; Akhtar & Mian, 2018). In order to investigate the robustness of LimiX, following Hollmann et al. (2025), we conduct experiments under two types of controlled perturbations: adding uninformative features and outliers.

For uninformative features, we randomly select columns from the original dataset, shuffle each column, and concatenate them to the original dataset. For outliers, we multiply a outlier coefficient to each cell value in the original dataset with a probability of 2%. The outlier coefficient is randomly chosen between 0 and the outlier factor.

In Figure 22, we show Normalized AUC in classification tasks under the two types of perturbations. We compare LimiX with TabPFN-v2, TabICL, and CatBoost. The left figure shows that even when adding up to 90% uninformative features, normalized AUC of LimiX remains nearly unchanged. In contrast, the performance of TabICL and CatBoost drops significantly. This indicates that LimiX is much more robust than TabICL and CatBoost. The right figure shows that LimiX consistently outperforms other methods regardless of the outlier factor. In Figure 23, we show RMSE in regression tasks under perturbations. In the left figure of adding uninformative features, a similar phenomenon is observed as that in classification tasks. In the right figure of adding outliers, we find that RMSE of TabPFN-v2 rapidly increases when the outlier factor grows from 100 to 10000, while LimiX do not. This demonstrates the superior robustness of LimiX-16M compared with TabPFN-v2 in regression tasks. Overall, LimiX exhibits superior robustness relative to the baselines.

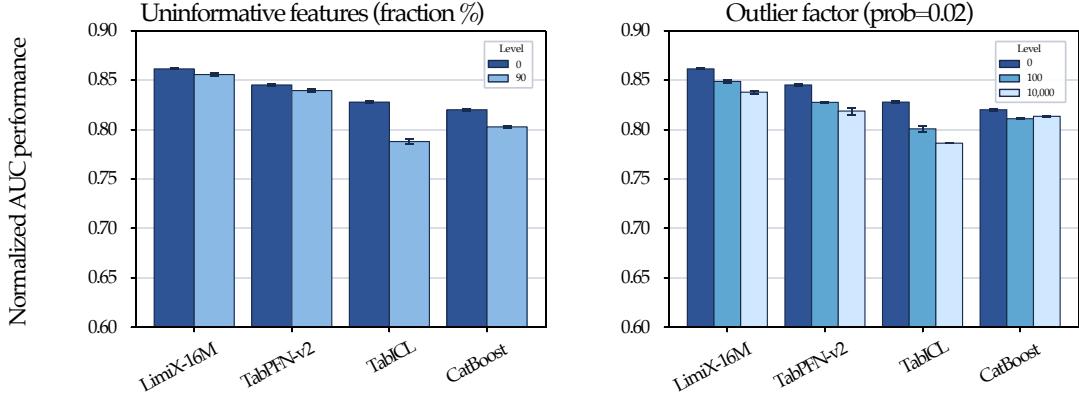


Figure 22: Robustness analysis in classification Tasks. LimiX consistently exhibits the best performance and superior robustness under perturbations.

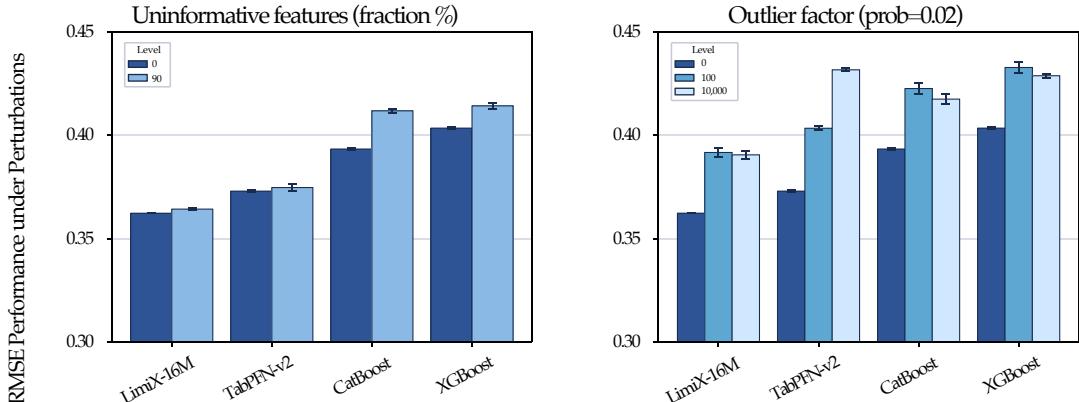


Figure 23: Robustness analysis in regression tasks. LimiX consistently exhibits the best performance and superior robustness under perturbations.

7.5 Embedding

Pretrained models (Devlin et al., 2019; Oquab et al., 2024) are generally expected to provide effective and transferable feature representations that can be leveraged for downstream tasks. To assess the quality of embeddings extracted by LimiX-16M, we conduct experiments on six datasets of various sample sizes and numbers of categories. We compare embeddings of LimiX-16M with those of MLP (Goodfellow et al., 2016; Gorishniy et al., 2021), ResNet (He et al., 2016; Gorishniy et al., 2021), ModernNCA (Ye et al., 2025), TabPFN-v2 (Hollmann et al., 2025), and TabICL (Qu et al., 2025). For each model, we treat representations prior to the classification head as embeddings. Figure 24 shows t-SNE visualization of embeddings. We can see that embeddings of different categories extracted by LimiX-16M are more separated than those extracted by other models.

To further evaluate the quality of embeddings extracted by LimiX-16M, we additionally conduct linear probing experiments, which is widely adopted in the analysis of feature representations (Kumar et al., 2022). The experiments are conducted on BCCO-CLS. Table 24 shows that embeddings of LimiX-16M achieve the highest average AUC and ranks among three ICL-based models. Overall, LimiX-16M consistently outperforms baselines in both qualitative and quantitative experiments.

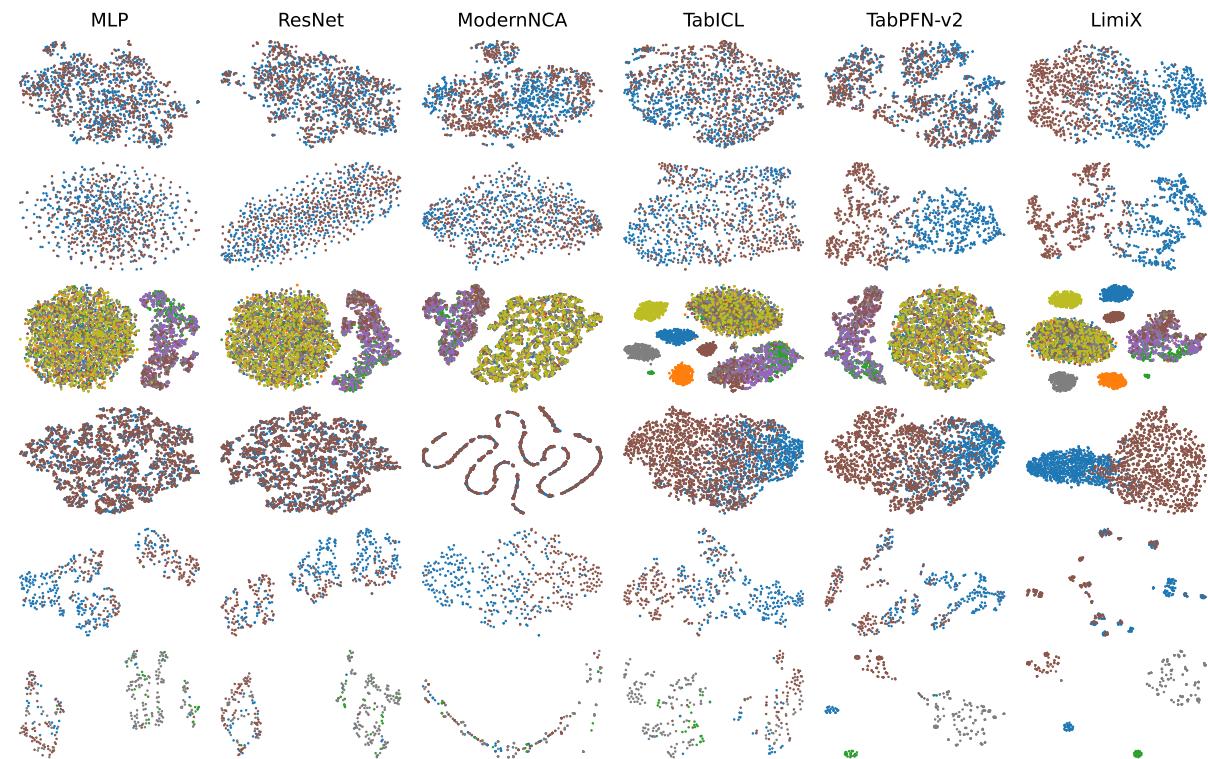


Figure 24: t-SNE visualization of embeddings. Different colors represent different categories. Embeddings extracted by LimiX-16M are better separated between categories than other models.

Table 24: Results of linear probing on BCCO-CLS. LimiX-16M outperforms other two ICL-based models, indicating the strongest embedding capability.

| Model | AUC(\uparrow) | Rank(\downarrow) |
|-----------|-------------------|----------------------|
| TabPFN-v2 | 0.832 | 2.189 |
| TabICL | 0.838 | 1.981 |
| LimiX-16M | 0.850 | 1.792 |

7.6 Fine-tuning

Long contexts increase memory cost and can degrade optimization during fine-tuning. Therefore, it is common practice to subsample the training corpus to a manageable budget, usually via random or KNN selection (Thomas et al., 2024; Xu et al., 2025). Based on our attention-based retrieval strategy, we adopt a retrieval-guided downsampling scheme that concentrates on locally most relevant patterns within each dataset, thus improving sample efficiency and prediction performance.

For each dataset, we split the original training set into a retrieval pool and a query set. For each instance in the query set, we select samples from the retrieval pool via the strategy proposed in Section 5 as in-context samples and construct an episode of in-context learning. Fine-tuning then proceeds over these episodes rather than over full, unfiltered contexts. This strategy substantially reduces the number of epochs required to achieve good performance. Note that the retrieved in-context sets may contain duplicated samples, which leads to a trade-off between relevance and diversity. We empirically find that the retrieved context size controls the balance between computation efficiency and prediction performance.

We compare LimiX-16M with some representative baselines that are trained or fine-tuned on real datasets, including TabDPT (Ma et al., 2024). For the fine-tuning of TabPFN-v2, we adopt the strategy and hyperparameter configurations from its official repository². As shown in Tables 25 and 26, LimiX-16M significantly outperforms the others across various metrics in most benchmarks after fine-tuning. Meanwhile, we illustrate the change of AUC for LimiX-16M on some datasets of BCCO-CLS before and after fine-tuning. From Figure 25, a significant improvement can be observed in most cases.

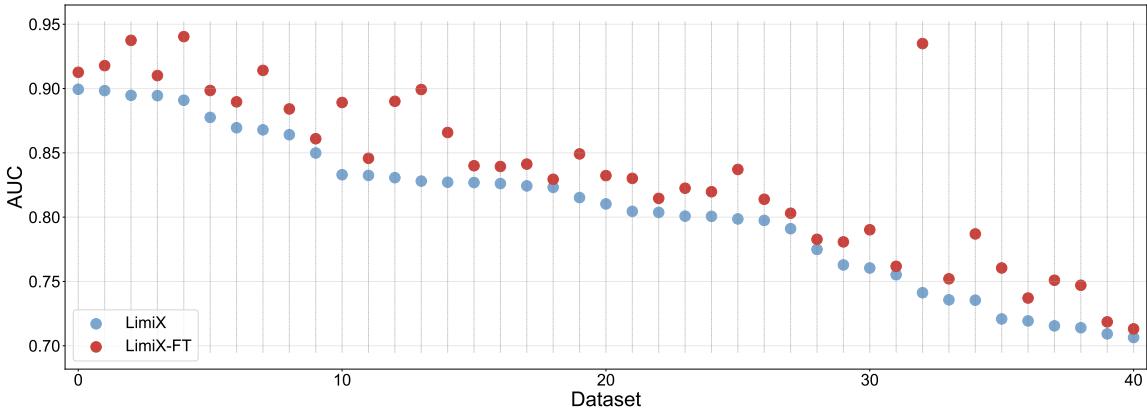


Figure 25: The improvement of fine-tuning in AUC on the BCCO-CLS benchmark.

²<https://github.com/PriorLabs/TabPFN>

Table 25: Performance comparison of representative methods on benchmarks of classification. "FT" indicates the model performance after fine-tuning. TabDPT-NE represents the performance of TabDPT without ensemble while its default version has an ensemble size of 8. The best scores are shown in bold.

| Benchmark | Method | AUC (\uparrow) | Acc. (\uparrow) | F1 (\uparrow) | AUC-Rank (\downarrow) | Acc.-Rank (\downarrow) | F1-Rank (\downarrow) |
|-------------|--------------|--------------------|---------------------|-------------------|---------------------------|----------------------------|--------------------------|
| BCCO-CLS | XGBoost | 0.834 | 0.762 | 0.674 | 5.811 | 5.660 | 5.509 |
| | CatBoost | 0.829 | 0.757 | 0.664 | 6.264 | 6.000 | 5.802 |
| | TabDPT-NE | 0.841 | 0.774 | 0.685 | 5.085 | 4.679 | 4.783 |
| | TabDPT | 0.846 | 0.777 | 0.687 | 4.453 | 4.358 | 4.604 |
| | TabPFN-v2 | 0.843 | 0.772 | 0.679 | 5.170 | 5.208 | 5.274 |
| | TabPFN-v2-FT | 0.842 | 0.773 | 0.678 | 5.198 | 4.925 | 5.330 |
| | Limix-16M | 0.871 | 0.804 | 0.731 | 2.538 | 2.575 | 2.651 |
| | Limix-16M-FT | 0.873 | 0.806 | 0.733 | 1.396 | 1.434 | 1.481 |
| TALENT-CLS | XGBoost | 0.881 | 0.837 | 0.713 | 5.603 | 5.564 | 5.402 |
| | CatBoost | 0.876 | 0.828 | 0.704 | 6.318 | 6.229 | 6.000 |
| | TabDPT-NE | 0.891 | 0.846 | 0.719 | 4.955 | 4.754 | 4.933 |
| | TabDPT | 0.893 | 0.849 | 0.723 | 4.179 | 4.207 | 4.413 |
| | TabPFN-v2 | 0.895 | 0.850 | 0.727 | 4.436 | 4.464 | 4.570 |
| | TabPFN-v2-FT | 0.889 | 0.842 | 0.718 | 5.212 | 5.156 | 5.106 |
| | Limix-16M | 0.903 | 0.861 | 0.752 | 3.061 | 2.860 | 2.933 |
| | Limix-16M-FT | 0.904 | 0.863 | 0.755 | 1.726 | 1.598 | 1.777 |
| OpenML-CC18 | XGBoost | 0.929 | 0.879 | 0.775 | 5.177 | 5.194 | 5.500 |
| | CatBoost | 0.926 | 0.870 | 0.770 | 6.548 | 6.032 | 6.274 |
| | TabDPT-NE | 0.927 | 0.881 | 0.786 | 4.677 | 4.758 | 4.984 |
| | TabDPT | 0.930 | 0.885 | 0.799 | 3.919 | 3.919 | 4.161 |
| | TabPFN-v2 | 0.929 | 0.886 | 0.790 | 5.065 | 4.468 | 4.613 |
| | TabPFN-v2-FT | 0.933 | 0.889 | 0.883 | 4.161 | 3.823 | 3.032 |
| | Limix-16M | 0.939 | 0.893 | 0.811 | 2.952 | 3.677 | 3.613 |
| | Limix-16M-FT | 0.941 | 0.894 | 0.813 | 1.500 | 2.306 | 2.435 |
| TabArena | XGBoost | 0.838 | 0.867 | 0.567 | 5.424 | 6.030 | 5.788 |
| | CatBoost | 0.835 | 0.867 | 0.574 | 6.242 | 5.545 | 5.121 |
| | TabDPT-NE | 0.891 | 0.846 | 0.719 | 4.950 | 4.749 | 4.927 |
| | TabDPT | 0.839 | 0.870 | 0.580 | 4.970 | 4.636 | 4.667 |
| | TabPFN-v2 | 0.838 | 0.872 | 0.589 | 4.939 | 4.303 | 4.818 |
| | TabPFN-v2-FT | 0.850 | 0.874 | 0.713 | 3.697 | 4.273 | 2.682 |
| | Limix-16M | 0.849 | 0.877 | 0.597 | 3.242 | 3.030 | 4.212 |
| | Limix-16M-FT | 0.851 | 0.878 | 0.600 | 2.030 | 1.697 | 2.939 |
| TabZilla | XGBoost | 0.929 | 0.863 | 0.789 | 5.185 | 5.148 | 5.481 |
| | CatBoost | 0.922 | 0.848 | 0.780 | 6.556 | 6.037 | 6.074 |
| | TabDPT-NE | 0.932 | 0.880 | 0.824 | 4.333 | 3.963 | 4.037 |
| | TabDPT | 0.934 | 0.882 | 0.824 | 3.815 | 3.815 | 4.148 |
| | TabPFN-v2 | 0.929 | 0.863 | 0.797 | 5.111 | 4.815 | 5.185 |
| | TabPFN-v2-FT | 0.932 | 0.876 | 0.847 | 4.185 | 4.556 | 3.318 |
| | Limix-16M | 0.943 | 0.885 | 0.836 | 3.556 | 3.444 | 3.889 |
| | Limix-16M-FT | 0.944 | 0.887 | 0.838 | 2.074 | 2.407 | 3.136 |
| PFN-CLS | XGBoost | 0.898 | 0.831 | 0.733 | 6.000 | 5.138 | 5.103 |
| | CatBoost | 0.895 | 0.819 | 0.720 | 6.310 | 5.828 | 6.138 |
| | TabDPT-NE | 0.891 | 0.822 | 0.727 | 5.034 | 5.483 | 5.828 |
| | TabDPT | 0.896 | 0.833 | 0.742 | 4.655 | 4.103 | 4.448 |
| | TabPFN-v2 | 0.910 | 0.845 | 0.756 | 4.138 | 4.655 | 4.552 |
| | TabPFN-v2-FT | 0.900 | 0.837 | 0.759 | 5.379 | 5.310 | 4.828 |
| | Limix-16M | 0.923 | 0.862 | 0.786 | 2.241 | 2.690 | 2.759 |
| | Limix-16M-FT | 0.924 | 0.864 | 0.788 | 1.276 | 1.517 | 1.724 |

Table 26: Performance comparison of representative methods on benchmarks of regression. "FT" indicates the model performance after fine-tuning. TabDPT-NE represents the performance of TabDPT without ensemble while its default version has an ensemble size of 8. The best scores are shown in bold.

| Benchmark | Method | R ² (\uparrow) | RMSE (\downarrow) | R ² -Rank (\downarrow) | RMSE-Rank (\downarrow) |
|------------|--------------|-------------------------------|-----------------------|---------------------------------------|----------------------------|
| BCCO-REG | XGBoost | 0.764 | 0.415 | 5.820 | 5.740 |
| | CatBoost | 0.741 | 0.427 | 6.600 | 6.460 |
| | TabDPT-NE | 0.769 | 0.410 | 5.220 | 5.060 |
| | TabDPT | 0.772 | 0.406 | 4.460 | 4.260 |
| | TabPFN-v2 | 0.772 | 0.404 | 4.600 | 4.280 |
| | TabPFN-v2-FT | 0.777 | 0.399 | 4.020 | 3.720 |
| | Limix-16M | 0.794 | 0.386 | 3.320 | 2.860 |
| | Limix-16M-FT | 0.796 | 0.385 | 1.960 | 2.880 |
| TALENT-REG | XGBoost | 0.710 | 0.462 | 5.545 | 5.404 |
| | CatBoost | 0.700 | 0.471 | 6.485 | 6.434 |
| | TabDPT-NE | 0.709 | 0.461 | 5.061 | 4.929 |
| | TabDPT | 0.711 | 0.458 | 4.263 | 4.081 |
| | TabPFN-v2 | 0.695 | 0.465 | 4.980 | 4.747 |
| | TabPFN-v2-FT | 0.702 | 0.459 | 4.879 | 4.657 |
| | Limix-16M | 0.735 | 0.433 | 2.970 | 2.566 |
| | Limix-16M-FT | 0.737 | 0.417 | 1.818 | 2.177 |
| CTR23 | XGBoost | 0.712 | 0.511 | 5.636 | 5.606 |
| | CatBoost | 0.700 | 0.528 | 6.273 | 6.212 |
| | TabDPT-NE | 0.728 | 0.500 | 4.909 | 4.818 |
| | TabDPT | 0.731 | 0.498 | 4.273 | 4.121 |
| | TabPFN-v2 | 0.716 | 0.503 | 4.545 | 4.394 |
| | TabPFN-v2-FT | 0.722 | 0.498 | 4.636 | 4.545 |
| | Limix-16M | 0.745 | 0.477 | 3.576 | 2.818 |
| | Limix-16M-FT | 0.748 | 0.473 | 2.152 | 2.437 |
| PFN-REG | XGBoost | 0.671 | 0.487 | 4.929 | 4.714 |
| | CatBoost | 0.661 | 0.498 | 6.214 | 6.071 |
| | TabDPT-NE | 0.672 | 0.491 | 5.393 | 5.071 |
| | TabDPT | 0.675 | 0.484 | 4.607 | 4.500 |
| | TabPFN-v2 | 0.676 | 0.475 | 4.393 | 4.143 |
| | TabPFN-v2-FT | 0.687 | 0.466 | 4.179 | 3.857 |
| | Limix-16M | 0.692 | 0.461 | 3.857 | 3.571 |
| | Limix-16M-FT | 0.695 | 0.466 | 2.429 | 4.071 |

7.7 Data Generation

Data generation is a challenging and meaningful task across multiple modalities since it needs to capture the joint distribution of $P(X, Y)$ compared with prediction tasks that focus on modeling $P(Y|X)$. While there is abundant literature on the generation of images (Croitoru et al., 2023), videos (Xing et al., 2024), and text (Li et al., 2024b), relatively less attention has been paid to tabular data generation. However, data generation is even more critical for tabular data due to its scarcity and possible privacy issues (Hernandez et al., 2022). As a foundation model, LimiX models have the ability of tabular data generation given an unseen real dataset. A significant advantage is that it does not require additional training of generative models. Following TabPFN-v2 (Hollmann et al., 2025), firstly we conduct data generation in an iterative style. For the first column, we sample from the empirical categorical distribution if it is a categorical feature, or we use the original first column with added random noise if it is a continuous feature. For other columns, we generate j^{th} column based on the $j - 1$ columns of real data (treated as x^{ct}) and generated data (treated as x^{te}). We iterate this process until the last column is generated. Then for LimiX models, we randomly mask a proportion of cell values of generated data and conduct missing value imputation for multiple times so that we can leverage LimiX’s capability of modeling the joint distribution. We conduct experiments on the following datasets: Early Stage Diabetes 2020 (Islam et al., 2019; 2020), Vertebral Column (Barreto & Neto, 2005), Seeds (Gomes Mantovani, 2015), Wine (Hardik, 2021), and Grub Damage (Vanschoren, 2014b). For evaluation, we use Trend and Shape to evaluate fidelity, which are proposed by SDMetrics³. We also calculate the AUC of XGBoost on a hold-out test dataset using generated data or real data. From Table 27, we find that LimiX-16M outperforms TabPFN-v2 in terms of all metrics on the five classification tasks. On the dataset of Grub Damage, the prediction performance using data generated by LimiX-16M is even higher than using real data. The results show that LimiX models are capable of capturing dependencies between features of X , which is brought by the pretraining strategy of mask prediction. In contrast, TabPFN-v2 is only capable of modeling $P(Y|X)$.

Table 27: Evaluation of data generation on five classification tasks. “Trend” and “Shape” are two fidelity metrics. “AUC” measures the classification performance of XGBoost using generated (or real) data. LimiX-16M consistently outperforms TabPFN-v2 in all three metrics.

| Metric | Method | Early Diabetes | Vertebra | Seeds | Wine | Grub Damage |
|----------------------|-----------|----------------|----------|-------|-------|-------------|
| Trend (\uparrow) | TabPFN-v2 | 0.797 | 0.580 | 0.696 | 0.686 | 0.486 |
| | LimiX-16M | 0.804 | 0.591 | 0.699 | 0.688 | 0.673 |
| Shape (\uparrow) | TabPFN-v2 | 0.889 | 0.754 | 0.739 | 0.622 | 0.635 |
| | LimiX-16M | 0.902 | 0.763 | 0.768 | 0.646 | 0.762 |
| AUC (\uparrow) | TabPFN-v2 | 0.839 | 0.652 | 0.861 | 0.670 | 0.695 |
| | LimiX-16M | 0.879 | 0.783 | 0.932 | 0.912 | 0.727 |
| | Real | 0.969 | 0.915 | 0.982 | 0.996 | 0.710 |

³<https://docs.sdv.dev/sdmetrics>

7.8 Out-of-Distribution Generalization

In real-world applications, tabular data is often subject to distribution shifts between training data and the test data encountered during deployment. For instance, a machine learning model may be trained on patient records collected from one hospital but deployed on data from a different hospital. Such distribution shifts arising from domain variation can lead to substantial degradation in model performance. This challenge is commonly referred to as the Out-of-Distribution (OOD) generalization problem, and has been the focus of extensive research in the machine learning community (Liu et al., 2021; Yu et al., 2024).

We evaluate the OOD generalization performance of various tabular models on 10 public datasets drawn from the TableShift (Gardner et al., 2023), which is a benchmark for distribution shifts in tabular data. It contains 15 binary classification tasks in total, covering a wide range of data sources, including finance, medical diagnosis, policy, etc. More details can be found in Section A.1. To ensure fair comparisons, for each dataset, if the number of training or test samples exceeds 10,000, we randomly subsample 10,000 instances. Each experiment is repeated five times and we report the average.

From Table 28, we can see that LimiX achieves state-of-the-art performance, securing top ranks in both ID (In-Distribution) and OOD evaluations. This could be attributed to LimiX’s integration of causal data with a Context-Conditional Masked Modeling framework. This strategy enables the model to capture robust causal relationships rather than superficial correlations, leading to significantly enhanced generalization on OOD data. As for baselines, models that are also based on in-context learning (ICL), notably TabICL and TabPFN-v2, also deliver competitive results. This indicates that the ICL mechanism could better capture latent invariant patterns in data, thereby endowing models with strong generalization potential.

Table 28: Average AUC and ranks on TableShift. We can see that LimiX consistently outperforms all baselines in terms of ID or OOD performance.

| Model | ID_AUC (\uparrow) | OOD_AUC (\uparrow) | ID_Rank (\downarrow) | OOD_Rank (\downarrow) |
|----------------|-----------------------|------------------------|--------------------------|---------------------------|
| LimiX-16M | 0.848 | 0.806 | 2.5 | 1.3 |
| TabICL | 0.847 | 0.799 | 4.1 | 3.9 |
| AutoGluon | 0.842 | 0.797 | 3.5 | 4.0 |
| TabPFN-v2 | 0.841 | 0.797 | 6.4 | 5.2 |
| CatBoost | 0.840 | 0.793 | 4.1 | 5.9 |
| LightGBM | 0.836 | 0.790 | 6.1 | 7.3 |
| MLP | 0.839 | 0.792 | 7.5 | 7.6 |
| FT-Transformer | 0.840 | 0.789 | 7.8 | 8.3 |
| ResNet | 0.837 | 0.789 | 8.6 | 8.3 |
| NODE | 0.836 | 0.789 | 9.2 | 9.2 |
| XGBoost | 0.830 | 0.783 | 9.9 | 9.5 |
| TabDPT | 0.822 | 0.763 | 12.6 | 12.7 |
| TabR | 0.820 | 0.767 | 13.2 | 13.0 |
| TANGOS | 0.801 | 0.752 | 13.8 | 13.6 |
| ModernNCA | 0.801 | 0.757 | 15.4 | 14.6 |

8 Scaling law

Recent advances in large language models (LLMs) reveal a class of empirical regularities known as *scaling laws*, which describe predictable relationships between model performance and key resource dimensions such as parameter count, dataset size, and compute budget (Kaplan et al., 2020; Hoffmann et al., 2022). These scaling laws enable researchers to forecast model behavior and guide efficient resource allocation when scaling up architectures.

However, most prior studies focus on large language models (LLMs), analyzing how loss scales with model parameters and compute under fixed or approximately fixed data distributions. Despite the growing importance of large structured-data models (LDMs), their scaling behavior remains largely unexplored.

Understanding the quantitative interaction between loss, data, and model parameters informs optimal trade-offs between data acquisition and model expansion. Motivated by this, this work systematically investigates the scaling laws for LDMs, focusing on the empirical relationships among model size, dataset size, and various performance metrics, including Log-Loss, AUC, F1, and accuracy for classification, as

well as RMSE-Loss and R^2 for regression tasks. Our primary focus is on evaluating these metrics on downstream tasks, as they directly reflect the model’s practical effectiveness and its ability to generalize to real-world applications. By analyzing performance on tasks like classification and regression, we aim to identify the optimal balance between model size, data, and task-specific performance.

For the experimental setup, we construct a series of models with parameter counts ranging from 1.05M to 16.53M (and up to 1.6G in extrapolated fits), trained on datasets spanning 1×10^{11} to 1.7×10^{12} tokens. Each configuration maintains consistent architecture and optimization settings, isolating the effects of data and parameter scaling. The downstream benchmark we used is BCCO-CLS and BCCO-REG. For this experiment, the evaluation was performed without the retrieval procedure, which may lead to minor deviations in performance compared to the results reported in the experimental section.

8.1 Scaling with Dataset Size and Model Parameters

Following previous findings in large language models (LLMs), we assume that both the loss and performance metrics of our model follow a power-law relationship with respect to computational resources, including dataset size and model parameter count. This assumption reflects the empirical regularity that as data volume or model capacity increases, the model’s loss tends to decrease while its predictive performance improves in a predictable manner. Formally, we hypothesize that each evaluation metric M (e.g., Log-Loss, RMSE-Loss, AUC, ACC, or R^2) can be expressed as a power-law function of the resource variable N :

$$M = a \cdot N^c + b, \quad (5)$$

where a , b , and c are constants to be determined. To estimate these parameters, we employ an ordinary least squares (OLS) regression over measurements collected across multiple scales of N . For each metric and task configuration, the fitted parameters and the coefficient of determination (R^2) are reported to assess the strength and consistency of the scaling behavior. This procedure enables systematic quantification of how both loss reduction and performance improvement scale with resource growth, facilitating direct comparisons between different model sizes and data regimes.

Figure 26 illustrates the empirical loss scaling with dataset size across different model configurations. All models achieve highly consistent fits with $R^2 > 0.99$, confirming the robustness of the scaling law. As dataset size increases, the loss decreases rapidly at smaller scales and then transitions to a more gradual improvement trend. Larger models achieve consistently lower loss values and exhibit smoother, more stable scaling behavior as additional data are introduced.

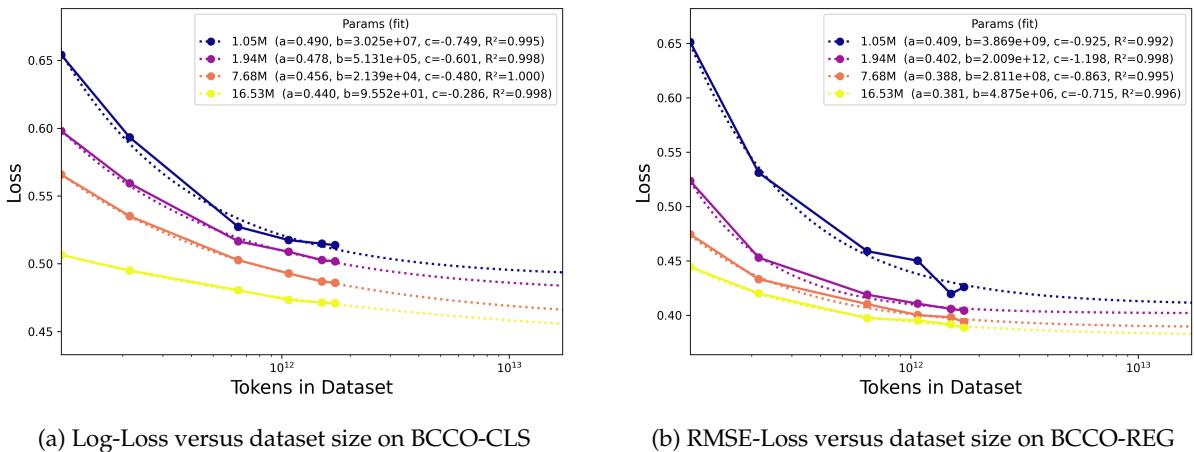


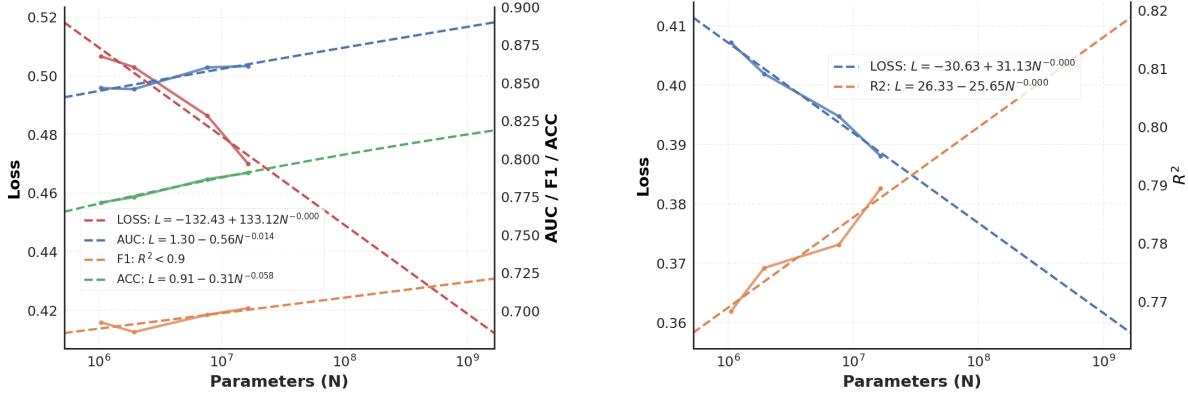
Figure 26: Loss versus dataset size for models of different parameter counts. Dotted lines indicate fitted power-law relations.

Furthermore, we extend our analysis to examine how downstream task loss and performance scale with model parameters when unconstrained by dataset size or computational budget. As shown in Figure 27, multiple performance metrics exhibit clear dependencies on the model parameter count N . For the classification task, the empirical fits achieve $R^2 > 0.9$ for Log-Loss, AUC, and ACC, confirming the robustness of the scaling law, while the fit for F1 yields a slightly lower R^2 of 0.68. For the regression task, both RMSE-Loss and the R^2 score exhibit strong correlations with model size, with empirical fits achieving $R^2 > 0.9$.

The empirical fits are reported below, with only those achieving $R^2 > 0.9$ shown:

$$\begin{aligned}
 \text{Log-Loss} &= -132.43 + 133.12N^{-0.000}, \quad (R^2 = 0.9646) \\
 \text{AUC} &= 1.30 - 0.56N^{-0.014}, \quad (R^2 = 0.9044) \\
 \text{ACC} &= 0.91 - 0.31N^{-0.058}, \quad (R^2 = 0.9933) \\
 \text{RMSE-loss} &= -30.63 + 31.13N^{-0.000}, \quad (R^2 = 0.9890) \\
 R^2 &= 26.33 - 25.65N^{-0.000}, \quad (R^2 = 0.9890)
 \end{aligned} \tag{6}$$

Loss decays sharply with increasing N , while AUC, ACC and F1 improve gradually, indicating that while error convergence is achieved relatively early, fine-grained discriminative improvements require substantially larger models.



(a) Log-Loss and performance versus parameter count on BCCO-CLS (b) RMSE-Loss and performance versus parameter count on BCCO-REG

Figure 27: Joint scaling of Loss, AUC, F1, ACC, and R^2 with respect to parameter count. Real and fitted trends are shown as solid and dashed lines, respectively.

8.2 Conclusions and Insights

This study provides the first systematic characterization of scaling laws in **LDMs**. Our key findings are summarized as follows:

Loss versus Dataset Size: both the classification (BCCO-CLS) and regression (BCCO-REG) tasks exhibit a consistent power-law relationship between loss and dataset size. When the token count increases from approximately 10^{11} to 10^{13} , the loss decreases systematically across all model configurations. Smaller models (e.g., 1.05M parameters) display a sharper initial decline in loss, while larger models (e.g., 16.53M parameters) achieve overall lower loss levels and maintain more stable improvement trends as data volume grows. The fitting results demonstrate exceptionally high explanatory power ($R^2 > 0.99$), confirming the robustness of the scaling behavior. These observations suggest that increasing the dataset size consistently enhances model performance, with improvement rates varying according to model capacity.

Loss and Performance versus Model Parameters As the number of model parameters increases from 1.05M to approximately one billion, the loss decreases markedly while overall predictive performance improves. In classification tasks, loss declines rapidly at smaller scales and stabilizes at larger ones, while AUC and ACC values continue to increase consistently. Similarly, in regression tasks, RMSE decreases and the coefficient of determination (R^2) rises with model size, both following strong power-law trends ($R^2 > 0.98$). These results suggest that scaling up the model enhances predictive stability and generalization. However, the diminishing rate of improvement in loss relative to performance metrics implies that additional parameters primarily contribute to refined discriminative capability rather than further error convergence.

Overall, the scaling behavior of the LDM follows a predictable power-law trend analogous to that observed in large language models. Both increased dataset size and model capacity yield systematic and highly correlated reductions in loss ($R^2 > 0.99$). Nevertheless, the returns on scaling diminish progressively, as larger models require substantially greater data and computational resources to achieve

marginal gains in performance. These results highlight the importance of balancing dataset expansion and model scaling to optimize the trade-off between predictive accuracy and computational efficiency.

9 Conclusion

In this work, we present LimiX, a unified family of our LDM series, and release its first two variants, LimiX-16M and LimiX-2M. LimiX models treat structured-data inputs as a joint distribution over variables and missingness, so that classification, regression, missing value imputation, data generation, and sample selection for interpretability, can all be expressed as conditional queries to a single model. Methodologically, LimiX models adopt a lightweight, scalable architecture that models causal relations among variables while jointly capturing dependencies across features and samples. Meanwhile, LimiX models combine masked joint-distribution pretraining with an episodic, context-conditional objective of per-dataset adaptation for the versatility in downstream tasks. The pretraining data for LimiX models is generated with hierarchical SCMs via graph-aware and solvability-aware sampling. Attention-guided retrieval of LimiX models further supports efficient inference-time ensemble and fine-tuning if desired. Empirically, experiments across 11 large structured-data benchmarks, spanning wide ranges of sample sizes, feature dimensions, number of classes, categorical-to-numerical feature ratios, missingness, and sample-to-feature ratios, confirm the effectiveness of LimiX models . As a single model, LimiX-16M consistently surpasses competitive baselines on various downstream tasks. We present the first systematic investigation of scaling laws in LDMs, revealing a predictable power-law relationship between scale and performance, analogous to that observed in large language models.

10 Contribution

Project Design and Lead

Xingxuan Zhang, Peng Cui

Core Contributors

Gang Ren, Han Yu, Hao Yuan, Hui Wang, Jiansheng Li, Jiayun Wu, Lang Mo, Li Mao, Mingchao Hao, Ningbo Dai, Renzhe Xu, Shuyang Li, Tianyang Zhang, Yue He, Yuanrui Wang, Yunjia Zhang, Zijing Xu

Contributors

Dongzhe Li, Fang Gao, Hao Zou, Jiandong Liu, Jiashuo Liu, Jiawei Xu, Kaijie Cheng, Kehan Li, Linjun Zhou, Qing Li, Shaohua Fan, Xiaoyu Lin, Xinyan Han, Xuanyue Li, Yan Lu, Yuan Xue, Yuanyuan Jiang, Zimu Wang, Zhenlei Wang

References

- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *Ieee Access*, 6:14410–14430, 2018.
- Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, 2019.
- Roohallah Alizadehsani, Mohamad Roshanzamir, and Zahra Sani. Z-alizadeh sani. UCI Machine Learning Repository, 2013. DOI: <https://doi.org/10.24432/C5Q31T>.
- Sercan Ö Arik and Tomas Pfister. Tabnet: Attentive interpretable tabular learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 6679–6687, 2021.
- Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.
- Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.
- 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.
- Dara Bahri, Heinrich Jiang, Yi Tay, and Donald Metzler. Scarf: Self-supervised contrastive learning using random feature corruption. In *International Conference on Learning Representations*, 2022.
- Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.
- Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Transactions on Machine Learning Research*, 2024.
- Guilherme Barreto and Ajalmar Neto. Vertebral column. UCI Machine Learning Repository, 2005. DOI: <https://doi.org/10.24432/C5K89B>.
- Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Pieter Gijsbers, Frank Hutter, Michel Lang, Rafael Gomes Mantovani, Jan N van Rijn, and Joaquin Vanschoren. Openml benchmarking suites. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2017.
- Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
- Pietro Caputo and Daniel Parisi. Block factorization of the relative entropy via spatial mixing. *Communications in Mathematical Physics*, 388(2):793–818, 2021.
- Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. Ieee, 2017.
- Jintai Chen, Kuanlun Liao, Yao Wan, Danny Z Chen, and Jian Wu. Danets: Deep abstract networks for tabular data classification and regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 3930–3938, 2022.
- Jintai Chen, KuanLun Liao, Yanwen Fang, Danny Chen, and Jian Wu. Tabcaps: A capsule neural network for tabular data classification with bow routing. In *The Eleventh International Conference on Learning Representations*, 2023a.
- Jintai Chen, Jiahuan Yan, Qiyuan Chen, Danny Ziyi Chen, Jian Wu, and Jimeng Sun. Excelformer: A neural network surpassing gbdts on tabular data. *arXiv preprint arXiv:2301.02819*, 2023b.
- Kuan-Yu Chen, Ping-Han Chiang, Hsin-Rung Chou, Ting-Wei Chen, and Darby Tien-Hao Chang. Trompt: towards a better deep neural network for tabular data. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 4392–4434, 2023c.

-
- 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*, pp. 785–794, 2016.
- Florinel-Alin Croitoru, Vlad Hondu, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(9):10850–10869, 2023.
- Bruno Rodrigues de Oliveira, Alan Mario Zuffo, Francisco Charles dos Santos Silva, Ricardo Mezzomo, Leandra Matos Barrozo, Tatiane Scilewski da Costa Zanatta, Joel Cabral dos Santos, Carlos Henrique Conceição Sousa, and Yago Pinto Coelho. Dataset: Forty soybean cultivars from subsequent harvests. *Trends in Agricultural and Environmental Sciences*, pp. e230005–e230005, 2023.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.
- Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. Catboost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*, 2018.
- Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. Autogluon-tabular: Robust and accurate automl for structured data. *arXiv preprint arXiv:2003.06505*, 2020.
- Nick Erickson, Lennart Purucker, Andrej Tschalzev, David Holzmüller, Prateek Mutalik Desai, David Salinas, and Frank Hutter. Tabarena: A living benchmark for machine learning on tabular data. *arXiv preprint arXiv:2506.16791*, 2025.
- Xi Fang, Weijie Xu, Fiona Anting Tan, Ziqing Hu, Jian Zhang, Yanjun Qi, Srinivasan H Sengamedu, and Christos Faloutsos. Large language models (llms) on tabular data: Prediction, generation, and understanding-a survey. *Transactions on Machine Learning Research*, 2024.
- Sebastian Felix Fischer, Matthias Feurer, and Bernd Bischl. Openml-ctr23—a curated tabular regression benchmarking suite. In *AutoML Conference 2023 (Workshop)*, 2023.
- Andreas Fuster, Paul Goldsmith-Pinkham, Tarun Ramadorai, and Ansgar Walther. Predictably unequal? the effects of machine learning on credit markets. *The Journal of Finance*, 77(1):5–47, 2022.
- Josh Gardner, Zoran Popovic, and Ludwig Schmidt. Benchmarking distribution shift in tabular data with tableshift. *Advances in Neural Information Processing Systems*, 36:53385–53432, 2023.
- Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.
- Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.
- Rafael Gomes Mantovani. Seeds. OpenML, 2015. URL <https://api.openml.org/d/1499>.
- Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.
- 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.
- Yury Gorishniy, Ivan Rubachev, and Artem Babenko. On embeddings for numerical features in tabular deep learning. *Advances in Neural Information Processing Systems*, 35:24991–25004, 2022.
- Yury Gorishniy, Ivan Rubachev, Nikolay Kartashev, Daniil Shlenskii, Akim Kotelnikov, and Artem Babenko. Tabr: Tabular deep learning meets nearest neighbors. In *The Twelfth International Conference on Learning Representations*, 2024.
- Hardik. Wine. Kaggle, 2021. URL <https://www.kaggle.com/datasets/hrdkcodes/wine-data>.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.
- 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*, pp. 770–778, 2016.

-
- Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.
- Miguel A Hernán and James M Robins. Causal inference, 2010.
- Mikel Hernandez, Gorka Epelde, Ane Alberdi, Rodrigo Cilla, and Debbie Rankin. Synthetic data generation for tabular health records: A systematic review. *Neurocomputing*, 493:28–45, 2022.
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.
- Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations*, 2022.
- Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025.
- David Holzmüller, Léo Grinsztajn, and Ingo Steinwart. Better by default: Strong pre-tuned mlps and boosted trees on tabular data. *Advances in Neural Information Processing Systems*, 37:26577–26658, 2024.
- Xin Huang, Ashish Khetan, Milan Cvitkovic, and Zohar Karnin. Tabtransformer: Tabular data modeling using contextual embeddings. *arXiv preprint arXiv:2012.06678*, 2020.
- MM Faniqul Islam, Rahatara Ferdousi, Sadikur Rahman, and Humayra Yasmin Bushra. Likelihood prediction of diabetes at early stage using data mining techniques. In *Computer Vision and Machine Intelligence in Medical Image Analysis: International Symposium, ISCMIA 2019*, pp. 113–125. Springer, 2019.
- MM Faniqul Islam, Rahatara Ferdousi, Sadikur Rahman, and Humayra Yasmin Bushra. Early stage diabetes risk prediction. UCI Machine Learning Repository, 2020. DOI: <https://doi.org/10.24432/C5VG8H>.
- Daniel Jarrett, Bogdan C Cebere, Tennison Liu, Alicia Curth, and Mihaela van der Schaar. Hyperimpute: Generalized iterative imputation with automatic model selection. In *International Conference on Machine Learning*, pp. 9916–9937. PMLR, 2022.
- Alan Jeffares, Tennison Liu, Jonathan Crabbé, Fergus Imrie, and Mihaela van der Schaar. Tangos: Regularizing tabular neural networks through gradient orthogonalization and specialization. In *The Eleventh International Conference on Learning Representations*, 2023.
- Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.
- 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.
- Tsung-Wei Ke, Nikolaos Gkanatsios, and Katerina Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. In *Conference on Robot Learning*, pp. 1949–1974. PMLR, 2025.
- Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. *Advances in neural information processing systems*, 30, 2017.
- Peter M Krafft, Meg Young, Michael Katell, Karen Huang, and Ghislain Buggingo. Defining ai in policy versus practice. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pp. 72–78, 2020.
- Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations*, 2022.

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Wensi Ai, Benjamin Martinez, et al. Behavior-1k: A human-centered, embodied ai benchmark with 1,000 everyday activities and realistic simulation. *arXiv preprint arXiv:2403.09227*, 2024a.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. Pre-trained language models for text generation: A survey. *ACM Computing Surveys*, 56(9):1–39, 2024b.

Yuchen Li, Alexandre Kirchmeyer, Aashay Mehta, Yilong Qin, Boris Dadachev, Kishore Papineni, Sanjiv Kumar, and Andrej Risteski. Promises and pitfalls of generative masked language modeling: Theoretical framework and practical guidelines. In *International Conference on Machine Learning*, pp. 27969–28017. PMLR, 2024c.

Wei-Chao Lin and Chih-Fong Tsai. Missing value imputation: a review and analysis of the literature (2006–2017). *Artificial Intelligence Review*, 53(2):1487–1509, 2020.

Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2019.

Jiashuo Liu, Zheyuan Shen, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. Towards out-of-distribution generalization: A survey. *arXiv preprint arXiv:2108.13624*, 2021.

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.

Junwei Ma, Valentin Thomas, Rasa Hosseinzadeh, Hamidreza Kamkari, Alex Labach, Jesse C Cresswell, Keyvan Golestan, Guangwei Yu, Maksims Volkovs, and Anthony L Caterini. Tabdpt: Scaling tabular foundation models. *arXiv preprint arXiv:2410.18164*, 2024.

Pierre-Alexandre Mattei and Jes Frellsen. Miwae: Deep generative modelling and imputation of incomplete data sets. In *International conference on machine learning*, pp. 4413–4423. PMLR, 2019.

Duncan McElfresh, Sujay Khandagale, Jonathan Valverde, Vishak Prasad C, Ganesh Ramakrishnan, Micah Goldblum, and Colin White. When do neural nets outperform boosted trees on tabular data? *Advances in Neural Information Processing Systems*, 36:76336–76369, 2023.

Edward Metz. Assistments: From research to practice at scale in education. <https://ies.ed.gov/logs/research/post/assistments-from-research-to-practice-at-scale-in-education>, 2020. Accessed: 2023-06-01.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, pp. 405–421. Springer, 2020.

Pete Mowforth and Barry Shepherd. Statlog (vehicle silhouettes). UCI Machine Learning Repository, 1987. DOI: <https://doi.org/10.24432/C5HG6N>.

Youssef Nader, Leon Sixt, and Tim Landgraf. Dnnr: Differential nearest neighbors regression. In *International Conference on Machine Learning*, pp. 16296–16317. PMLR, 2022.

Maxime Oquab, Timothée Darzet, Théo Moutakanni, Huy V Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

Judea Pearl. *Causality*. Cambridge university press, 2009.

Sergei Popov, Stanislav Morozov, and Artem Babenko. Neural oblivious decision ensembles for deep learning on tabular data. In *International Conference on Learning Representations*, 2020.

Yilong Qin and Andrej Risteski. Fit like you sample: sample-efficient generalized score matching from fast mixing diffusions. In *The Thirty Seventh Annual Conference on Learning Theory*, pp. 4413–4457. PMLR, 2024.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. Tabicl: A tabular foundation model for in-context learning on large data. In *International Conference on Machine Learning*. PMLR, 2025.

Raghu Ramakrishnan, Johannes Gehrke, and Johannes Gehrke. *Database management systems*, volume 3. McGraw-Hill New York, 2003.

Matthew A Reyna, Christopher S Josef, Russell Jeter, Supreeth P Shashikumar, M Brandon Westover, Shamim Nemati, Gari D Clifford, and Ashish Sharma. Early prediction of sepsis from clinical data: the physionet/computing in cardiology challenge 2019. *Critical care medicine*, 48(2):210–217, 2020.

Bruno Rodrigues de Oliveira and Alan Mario Zuffo. Forty soybean cultivars from subsequent harvests. UCI Machine Learning Repository, 2023. DOI: <https://doi.org/10.46420/TAES.e230005>.

Miriam Seoane Santos, Pedro Henriques Abreu, Pedro J García-Laencina, Adélia Simão, and Armando Carvalho. Hcc survival. UCI Machine Learning Repository, 2015a. DOI: <https://doi.org/10.24432/C5TS4S>.

Miriam Seoane Santos, Pedro Henriques Abreu, Pedro J García-Laencina, Adélia Simão, and Armando Carvalho. A new cluster-based oversampling method for improving survival prediction of hepatocellular carcinoma patients. *Journal of biomedical informatics*, 58:49–59, 2015b.

J Paul Siebert. Vehicle recognition using rule based methods. 1987.

Abraham Silberschatz, Henry F Korth, and Shashank Sudarshan. Database system concepts. 2011.

Jeffrey S Simonoff. *Analyzing categorical data*, volume 496. Springer, 2003.

Gowthami Somepalli, Avi Schwarzschild, Micah Goldblum, C Bayan Bruss, and Tom Goldstein. Saint: Improved neural networks for tabular data via row attention and contrastive pre-training. In *NeurIPS 2022 First Table Representation Workshop*, 2022.

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*, pp. 1161–1170, 2019.

Daniel J Stekhoven and Peter Bühlmann. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012.

Michael Stonebraker and Uğur Çetintemel. “one size fits all” an idea whose time has come and gone. In *Making databases work: the pragmatic wisdom of Michael Stonebraker*, pp. 441–462. 2018.

Beata Strack, Jonathan P DeShazo, Chris Gennings, Juan L Olmo, Sebastian Ventura, Krzysztof J Cios, and John N Clore. Impact of hba1c measurement on hospital readmission rates: analysis of 70,000 clinical database patient records. *BioMed research international*, 2014(1):781670, 2014.

Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. Table meets llm: Can large language models understand structured table data? a benchmark and empirical study. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pp. 645–654, 2024.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soriciut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Valentin Thomas, Junwei Ma, Rasa Hosseinzadeh, Keyvan Golestan, Guangwei Yu, Maks Volkovs, and Anthony L Caterini. Retrieval & fine-tuning for in-context tabular models. *Advances in Neural Information Processing Systems*, 37:108439–108467, 2024.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Boris Van Breugel and Mihaela Van Der Schaar. Position: Why tabular foundation models should be a research priority. In *International Conference on Machine Learning*, pp. 48976–48993. PMLR, 2024.

Stef Van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, 45:1–67, 2011.

Aad W Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge university press, 2000.

Jan van Rijn. Eucalyptus. OpenML, 2014. URL <https://api.openml.org/d/188>.

Joaquin Vanschoren. Analcatdata broadwaymult. OpenML, 2014a. URL <https://api.openml.org/d/961>.

Joaquin Vanschoren. Grub damage. OpenML, 2014b. URL <https://api.openml.org/d/338>.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

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*, pp. 1785–1797, 2021.

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, pp. 15924–15933, 2024.

Tiange Xiang, Kai Li, Chengjiang Long, Christian Häne, Peihong Guo, Scott Delp, Ehsan Adeli, and Li Fei-Fei. Repurposing 2d diffusion models with gaussian atlas for 3d generation. *arXiv preprint arXiv:2503.15877*, 2025.

Zhen Xing, Qijun Feng, Haoran Chen, Qi Dai, Han Hu, Hang Xu, Zuxuan Wu, and Yu-Gang Jiang. A survey on video diffusion models. *ACM Computing Surveys*, 57(2):1–42, 2024.

Derek Qiang Xu, F Olcay Cirit, Reza Asadi, Yizhou Sun, and Wei Wang. Mixture of in-context prompters for tabular pfns. In *The Thirteenth International Conference on Learning Representations*, 2025.

Jiahuan Yan, Jintai Chen, Yixuan Wu, Danny Z Chen, and Jian Wu. T2g-former: organizing tabular features into relation graphs promotes heterogeneous feature interaction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 10720–10728, 2023.

Han-Jia Ye, Huai-Hong Yin, De-Chuan Zhan, and Wei-Lun Chao. Revisiting nearest neighbor for tabular data: A deep tabular baseline two decades later. In *The Thirteenth International Conference on Learning Representations*, 2025.

Taoran Yi, Jiemin Fang, Junjie Wang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. Gaussiandreamer: Fast generation from text to 3d gaussians by bridging 2d and 3d diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6796–6807, 2024.

Jinsung Yoon, James Jordon, and Mihaela Schaar. Gain: Missing data imputation using generative adversarial nets. In *International conference on machine learning*, pp. 5689–5698. PMLR, 2018.

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.

Han Yu, Jiashuo Liu, Xingxuan Zhang, Jiayun Wu, and Peng Cui. A survey on evaluation of out-of-distribution generalization. *arXiv preprint arXiv:2403.01874*, 2024.

Wantao Yu, Chee Yew Wong, Roberto Chavez, and Mark A Jacobs. Integrating big data analytics into supply chain finance: The roles of information processing and data-driven culture. *International journal of production economics*, 236:108135, 2021.

Hengrui Zhang, Liancheng Fang, Qitian Wu, and Philip S Yu. Diffputer: Empowering diffusion models for missing data imputation. In *The Thirteenth International Conference on Learning Representations*, 2025.

Xiyuan Zhang and Maddix Robinson Danielle. Mitra: Mixed synthetic priors for enhancing tabular foundation models, 2025. URL <https://www.amazon.science/blog/mitra-mixed-synthetic-priors-for-enhancing-tabular-foundation-models>.

A Experimental Details

A.1 Details of Datasets with Distribution Shifts

The 10 public datasets we adopt from TableShift include Voting⁴, Unemployment⁵, Sepsis (Reyna et al., 2020), Public Health Insurance⁵, Income⁵, Hospital Readmission (Strack et al., 2014)⁶, Food Stamps⁵, Diabetes⁷, College Scorecard⁸, and ASSISTments (Metz, 2020)⁹. Detailed information of the benchmark TableShift we adopt in Section 7.8 can be found in Table 29.

Table 29: Statistics of datasets in TableShift. "#Features" indicates the number of features. "#Train" indicates the number of training samples. "#ID_test" indicates the number of ID (In-Distribution) test samples. "#OOD_test" indicates the number of OOD (Out-of-Distribution) test samples. "Shift Type" indicates how the datasets are split into multiple domains to create distribution shifts. "#Domains" indicates the numbers of training and test domains.

| Dataset | #Features | #Train | #ID_test | #OOD_test | Shift type | #Domains |
|----------------------|-----------|-----------|----------|-----------|-------------------|----------|
| Voting | 365 | 34,796 | 4,350 | 21,231 | Geographic Region | 4/1 |
| ASSISTments | 26 | 2,132,526 | 266,566 | 1,906 | School | 386/10 |
| Diabetes | 142 | 969,229 | 121,154 | 209,375 | Race | 1/5 |
| Food Stamps | 239 | 629,018 | 78,628 | 48,878 | Geographic Region | 9/1 |
| Hospital Readmission | 183 | 34,288 | 4,287 | 50,968 | Admission Source | 15/1 |
| Income | 232 | 1,264,123 | 158,016 | 75,911 | Geographic Region | 9/1 |
| Health Insurance | 135 | 4,006,249 | 500,782 | 817,877 | Disability Status | 1/1 |
| Sepsis | 41 | 1,122,299 | 140,288 | 134,402 | Length of Stay | 47/2 |
| Unemployment | 223 | 1,290,914 | 161,365 | 163,611 | Education Level | 9/15 |
| College ScoreCard | 118 | 98,556 | 12,320 | 1,352 | Institution Type | 26/8 |

A.2 Hyperparameter Search Space for Baselines

The hyperparameter search space for the baseline models is summarized in Table 30.

⁴<https://electionstudies.org/>

⁵<https://www.census.gov/programs-surveys/acs>

⁶<https://archive.ics.uci.edu/ml/datasets/Diabetes+130-US+hospitals+for+years+1999-2008>

⁷<https://www.cdc.gov/brfss/index.html>

⁸<https://collegescorecard.ed.gov/>

⁹<https://new.assistments.org>

Table 30: The hyperparameter search space for tree-based models is utilized by Optuna, which provides automatic suggestions via Bayesian optimization. The only difference between classification and regression configurations lies in the optimization target. All other hyperparameter settings remain identical.

| Baseline | Hyperparameter | Data Type | Log | Search Space |
|----------|-------------------|-------------|-----|---------------------------------|
| RF | n_estimators | int | no | [100, 500] |
| | max_depth | int | no | [3, 20] |
| | min_samples_split | int | no | [2, 20] |
| | min_samples_leaf | int | no | [1, 20] |
| | max_features | categorical | no | {Sqrt, Log ₂ , None} |
| | bootstrap | categorical | no | {True, False} |
| ET | n_estimators | int | no | [100, 500] |
| | max_depth | int | no | [3, 20] |
| | min_samples_split | int | no | [2, 20] |
| | min_samples_leaf | int | no | [1, 20] |
| XGBoost | n_estimators | int | no | [100, 300] |
| | max_depth | int | no | [3, 9] |
| | learning_rate | float | yes | [0.01, 0.3] |
| | subsample | float | no | [0.5, 1.0] |
| | colsample_bytree | float | no | [0.5, 1.0] |
| LightGBM | n_estimators | int | no | [100, 300] |
| | learning_rate | float | yes | [0.01, 0.3] |
| | max_depth | int | no | [-1, 20] |
| | subsample | float | no | [0.5, 1.0] |
| | colsample_bytree | float | no | [0.5, 1.0] |
| CatBoost | iterations | int | no | [100, 300] |
| | depth | int | no | [4, 10] |
| | learning_rate | float | yes | [0.01, 0.3] |

B Omitted Details in Section 6

B.1 Omitted Details in Section 6.1

Proposition B.1 (Formal Version of Theorem 6.1). *Suppose that the distribution $p(\mathbf{X}^{te}|\mathbf{X}^{ct})$ has a strictly positive probability density function. Then there is a one-to-one correspondence between the distribution $p(\mathbf{X}^{te}|\mathbf{X}^{ct})$ and the family of conditionals $\{p(\mathbf{X}_\pi^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) : \forall \pi \in \Pi_k\}$.*

Proof. It is clear that $p(\mathbf{X}^{te}|\mathbf{X}^{ct})$ directly yields all conditionals $\{p(\mathbf{X}_\pi^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) : \pi \in \Pi_k\}$. We now show the converse: $p(\mathbf{X}^{te}|\mathbf{X}^{ct})$ can be recovered from this family of conditionals.

As the first step, we derive $\{p(X_j^{te}|\mathbf{X}_{-j}^{te}, \mathbf{X}^{ct}) : j \in [d]\}$. For any $j \in [d]$, select a mask $\pi \in \Pi_k$ with $j \in \pi$. Then

$$p(X_j^{te}|\mathbf{X}_{-j}^{te}, \mathbf{X}^{ct}) = \frac{p(\mathbf{X}_\pi^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})}{p(\mathbf{X}_{\pi \setminus \{j\}}^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})} = \frac{p(\mathbf{X}_\pi^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})}{\int_{\Omega} p(\mathbf{X}_{\pi \setminus \{j\}}^{te}, X_j^{te'}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) dX_j^{te'}}.$$

Each term on the right-hand side belongs to $\{p(\mathbf{X}_\pi^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) : \pi \in \Pi_k\}$, hence $p(X_j^{te}|\mathbf{X}_{-j}^{te}, \mathbf{X}^{ct})$ can indeed be recovered.

We now use induction to show that the knowledge of $\{p(X_j^{te}|\mathbf{X}_{-j}^{te}, \mathbf{X}^{ct}) : j \in [d]\}$ suffices to recover all conditionals in

$$\mathcal{P}_k = \{p(\mathbf{X}_\pi^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) : |\pi| = k\}.$$

Clearly, \mathcal{P}_1 coincides with $\{p(X_j^{te}|\mathbf{X}_{-j}^{te}, \mathbf{X}^{ct}) : j \in [d]\}$. Suppose $\mathcal{P}_{k'}$ is obtainable for all $k' \leq k$. Consider $p(\mathbf{X}_\pi^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) \in \mathcal{P}_{k+1}$ with $|\pi| = k+1$. Pick $j \in \pi$ and set $\pi' = \pi \setminus \{j\}$. Then

$$\int_{\Omega} \frac{p(X_j^{te}|\mathbf{X}_{\pi'}^{te}, \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})}{p(\mathbf{X}_{\pi'}^{te}|\mathbf{X}_j^{te}, \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})} dX_j^{te} = \int_{\Omega} \frac{p(X_j^{te}|\mathbf{X}_{\pi'}^{te}, \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})}{p(\mathbf{X}_{\pi'}^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})} dX_j^{te} = \frac{1}{p(\mathbf{X}_{\pi'}^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})}.$$

Each term on the left-hand side belongs to \mathcal{P}_1 or \mathcal{P}_k , so $p(\mathbf{X}_{\pi'}^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})$ is obtainable. Finally,

$$p(\mathbf{X}_\pi^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) = p(\mathbf{X}_{\pi'}^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) \cdot p(X_j^{te}|\mathbf{X}_{\pi'}^{te}, \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}),$$

showing that $p(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})$ can be obtained as well. Hence, any element in \mathcal{P}_{k+1} can be obtained. By induction, the claim follows. \square

Now we show that the knowledge of a single conditional distribution $p(X_j^{\text{te}} | \mathbf{X}_{-j}^{\text{te}}, \mathbf{X}^{\text{ct}})$ for some $j \in [d]$ (i.e., when $\Pi = \{j\}$) is insufficient to recover the full conditional distribution $p(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$.

Example B.1. Consider the case $d = 2$ and $m = 0$, i.e., a setting with no in-context samples and a feature dimension of 2. In this case, the target distribution $p(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$ reduces to $p(\mathbf{X}^{\text{te}})$. Suppose further that $\mathbf{X}^{\text{te}} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, where $\boldsymbol{\mu} = (\mu_1, \mu_2) \in \mathbb{R}^2$ and $\Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} \succeq 0$. Then the conditional distribution of X_1^{te} given X_2^{te} is

$$X_1^{\text{te}} | X_2^{\text{te}} \sim \mathcal{N}\left(\Sigma_{12}\Sigma_{22}^{-1}X_2^{\text{te}} + \mu_1 - \Sigma_{12}\Sigma_{22}^{-1}\mu_2, \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}\right).$$

Consequently, knowledge of $p(X_1^{\text{te}} | X_2^{\text{te}})$ alone provides access only to the quantities $\Sigma_{12}\Sigma_{22}^{-1}$, $\mu_1 - \Sigma_{12}\Sigma_{22}^{-1}\mu_2$, and $\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$. These are insufficient to uniquely determine the full parameter set $(\boldsymbol{\mu}, \Sigma)$, and therefore $p(\mathbf{X}^{\text{te}})$ cannot be fully recovered from $p(X_1^{\text{te}} | X_2^{\text{te}})$ alone. By symmetry, $p(\mathbf{X}^{\text{te}})$ also cannot be recovered solely from $p(X_2^{\text{te}} | X_1^{\text{te}})$.

B.2 Omitted Details in Section 6.2

We denote by $q_\theta(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}}), \theta \in \Theta$ the distribution induced by the learned family of conditional probabilities $\{q_\theta(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}}) : \pi \in \Pi_k\}$. We overload the notation by writing $q_\theta(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) := q_\theta(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})p(\mathbf{X}^{\text{ct}})$.

B.2.1 Sample Efficiency

Theorem B.2 (Formal Version of Theorem 6.2). *Suppose there exists $\theta^* \in \Theta$ such that $q_{\theta^*}(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}}) = p(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})$ for all $\mathbf{X}^{\text{ct}} \in \Omega^{m \times d}$, $\mathbf{X}^{\text{te}} \in \Omega^d$, and $\pi \subseteq [d]$, and that the minimizer of $L_k(\theta)$ is unique for every k . Assume that for all $\theta \in \Theta$, $\mathbf{X}^{\text{ct}} \in \Omega^{m \times d}$, $\mathbf{X}^{\text{te}} \in \Omega^d$, and $\pi \subseteq [d]$, the gradient norm $\|\nabla_\theta q_\theta(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})\|_2$ and the Hessian norm $\|\nabla_\theta^2 q_\theta(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})\|_F$ exist and are finite. Assume $\nabla_\theta^2 L_k(\theta^*) \succ 0$ for all $k \in [d]$. Then, for every sufficiently small neighborhood \mathcal{B} of θ^* , there exists a sufficiently large n such that $\hat{\theta}_{k,n}$ is the unique minimizer of $\hat{L}_k(\theta)$ in \mathcal{B} . Moreover,*

$$\sqrt{n}(\hat{\theta}_{k,n} - \theta^*) \xrightarrow{d} \mathcal{N}(0, \Gamma_k),$$

where Γ_k does not depend on n and satisfies $\Gamma_{k+1} \preceq \Gamma_k$.

The proof of Theorem B.2 is based on the following lemma.

Lemma B.3 (Van der Vaart (2000), Theorem 5.23; statement adapted from Qin & Risteski (2024); Li et al. (2024c)). *Consider a loss $L : \Theta \rightarrow \mathbb{R}$, such that $L(\theta) = \mathbb{E}_{\mathbf{X} \sim p}[\ell_\theta(\mathbf{X})]$ for $\ell_\theta : \mathcal{X} \rightarrow \mathbb{R}$. Let Θ^* be the set of global minima of L , i.e.,*

$$\Theta^* = \{\theta^* : L(\theta^*) = \min_{\theta \in \Theta} L(\theta)\}.$$

Suppose the following conditions are met:

- (Gradient bounds on ℓ_θ) The map $\theta \mapsto \ell_\theta$ is measurable and differentiable at every $\theta^* \in \Theta^*$ for p -almost every \mathbf{X} . Furthermore, there exists a function $B(\mathbf{X})$, s.t. $\mathbb{E}[B(\mathbf{X})^2] < \infty$ and for every θ_1, θ_2 near θ^* , we have

$$|\ell_{\theta_1}(\mathbf{X}) - \ell_{\theta_2}(\mathbf{X})| < B(\mathbf{X}) \|\theta_1 - \theta_2\|_2$$

- (Twice-differentiability of L) $L(\theta)$ is twice-differentiable at every $\theta^* \in \Theta^*$ with Hessian $\nabla_\theta^2 L(\theta^*)$, and furthermore $\nabla_\theta^2 L(\theta^*) \succ 0$.
- (Uniform law of large numbers) The loss L satisfies a uniform law of large numbers, that is

$$\sup_{\theta \in \Theta} |\hat{\mathbb{E}}[\ell_\theta(\mathbf{X})] - L(\theta)| \xrightarrow{p} 0.$$

- (Realizability) The data distribution p satisfies: $\exists \theta^* \in \Theta$ such that $p_{\theta^*} = p$.

Then for every $\theta^* \in \Theta^*$, and every sufficiently small neighborhood S of θ^* , there exists a sufficiently large n , such that there is a unique minimizer $\hat{\theta}_n$ of $\hat{\mathbb{E}}[\ell_\theta(\mathbf{X})]$ in S . Furthermore, $\hat{\theta}_n$ satisfies:

$$\sqrt{n}(\hat{\theta}_n - \theta^*) \xrightarrow{d} \mathcal{N}\left(0, \left(\nabla_\theta^2 L(\theta^*)\right)^{-1} \text{Cov}(\nabla_\theta \ell_{\theta^*}(\mathbf{X})) \left(\nabla_\theta^2 L(\theta^*)\right)^{-1}\right).$$

Similar to Lemma 2 in Li et al. (2024c), we have the following lemma.

Lemma B.4. *Under the same assumptions as in Theorem B.2, we have*

$$\nabla_\theta^2 L_k(\theta^*) = \text{Cov}_{(\mathbf{X}^{ct}, \mathbf{X}^{te}) \sim p, \pi \sim \text{Unif}(\Pi_k)} (-\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})). \quad (7)$$

Proof. It is easy to verify that θ^* is the minimizer of $L_k(\theta)$ for any $k \in [d]$ and hence $\theta_k^* = \theta^*$ by assumption.

Rewrite $\nabla_\theta^2 L_k(\theta^*)$ and we get that

$$\begin{aligned} & \nabla_\theta^2 L_k(\theta^*) \\ &= \nabla_\theta^2 \mathbb{E}_{\mathbf{X}^{ct}, \mathbf{X}^{te}, \pi} [-\log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})] \\ &= \mathbb{E}_{\mathbf{X}^{ct}, \mathbf{X}^{te}, \pi} [-\nabla_\theta^2 \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})] \\ &= \mathbb{E}_{\mathbf{X}^{ct}, \mathbf{X}^{te}, \pi} \left[(\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})) (\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}))^\top - \frac{\nabla_\theta^2 q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})}{q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})} \right]. \end{aligned} \quad (8)$$

In addition, we have

$$\begin{aligned} & \mathbb{E}_{\mathbf{X}^{ct}, \mathbf{X}^{te}, \pi} \left[\frac{\nabla_\theta^2 q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})}{q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})} \right] = \mathbb{E}_{\mathbf{X}^{ct}, \pi} \mathbb{E}_{\mathbf{X}_{-\pi}^{te} | \mathbf{X}^{ct}} \mathbb{E}_{\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}} \left[\frac{\nabla_\theta^2 q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})}{q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})} \right] \\ &= \mathbb{E}_{\mathbf{X}^{ct}, \pi} \mathbb{E}_{\mathbf{X}_{-\pi}^{te} | \mathbf{X}^{ct}} \left[\int_{\Omega^k} \frac{\nabla_\theta^2 q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})}{q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})} \cdot p(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) d\mathbf{X}_\pi^{te} \right] \\ &= \mathbb{E}_{\mathbf{X}^{ct}, \pi} \mathbb{E}_{\mathbf{X}_{-\pi}^{te} | \mathbf{X}^{ct}} \left[\int_{\Omega^k} \nabla_\theta^2 q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) d\mathbf{X}_\pi^{te} \right] \quad (\text{Due to the assumption } q_{\theta^*} = p) \\ &= \mathbb{E}_{\mathbf{X}^{ct}, \pi} \mathbb{E}_{\mathbf{X}_{-\pi}^{te} | \mathbf{X}^{ct}} \left[\nabla_\theta^2 \int_{\Omega^k} q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) d\mathbf{X}_\pi^{te} \right] = \mathbb{E}_{\mathbf{X}^{ct}, \pi} \mathbb{E}_{\mathbf{X}_{-\pi}^{te} | \mathbf{X}^{ct}} [\nabla_\theta^2 1] = 0. \end{aligned}$$

Combined with Equation (8), we can get that

$$\nabla_\theta^2 L_k(\theta^*) = \mathbb{E}_{\mathbf{X}^{ct}, \mathbf{X}^{te}, \pi} \left[(\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})) (\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}))^\top \right]. \quad (9)$$

Now rewrite the right-hand side of Equation (7) and we get

$$\begin{aligned} & \text{Cov}(-\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})) \\ &= \mathbb{E}_{\mathbf{X}^{ct}, \mathbf{X}^{te}, \pi} \left[(\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})) (\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}))^\top \right] \\ &\quad - \mathbb{E}_{\mathbf{X}^{ct}, \mathbf{X}^{te}, \pi} [\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})] \mathbb{E}_{\mathbf{X}^{ct}, \mathbf{X}^{te}, \pi} [\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})]^\top. \end{aligned} \quad (10)$$

Note that

$$\begin{aligned} & \mathbb{E}_{\mathbf{X}^{ct}, \mathbf{X}^{te}, \pi} [\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})] = \mathbb{E}_{\mathbf{X}^{ct}, \mathbf{X}^{te}, \pi} \left[\frac{\nabla_\theta q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})}{q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})} \right] \\ &= \mathbb{E}_{\mathbf{X}^{ct}, \pi} \mathbb{E}_{\mathbf{X}_{-\pi}^{te} | \mathbf{X}^{ct}} \mathbb{E}_{\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}} \left[\frac{\nabla_\theta q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})}{q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})} \right] \\ &= \mathbb{E}_{\mathbf{X}^{ct}, \pi} \mathbb{E}_{\mathbf{X}_{-\pi}^{te} | \mathbf{X}^{ct}} \left[\int_{\Omega^k} \frac{\nabla_\theta q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})}{q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})} \cdot p(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) d\mathbf{X}_\pi^{te} \right] \\ &= \mathbb{E}_{\mathbf{X}^{ct}, \pi} \mathbb{E}_{\mathbf{X}_{-\pi}^{te} | \mathbf{X}^{ct}} \left[\int_{\Omega^k} \nabla_\theta q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) d\mathbf{X}_\pi^{te} \right] \quad (\text{Due to the assumption } q_{\theta^*} = p) \\ &= \mathbb{E}_{\mathbf{X}^{ct}, \pi} \mathbb{E}_{\mathbf{X}_{-\pi}^{te} | \mathbf{X}^{ct}} \left[\nabla_\theta \int_{\Omega^k} q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) d\mathbf{X}_\pi^{te} \right] \\ &= \mathbb{E}_{\mathbf{X}^{ct}, \pi} \mathbb{E}_{\mathbf{X}_{-\pi}^{te} | \mathbf{X}^{ct}} [\nabla_\theta 1] = 0. \end{aligned}$$

Combined with Equation (10), we can get that

$$\text{Cov}(-\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})) = \mathbb{E} \left[(\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})) (\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{te} | \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}))^\top \right]. \quad (11)$$

Now the claim follows from Equations (9) and (11). \square

Based on the above lemmas, we can now prove [Theorem B.2](#). The proof follows a similar idea to that of [Theorem 1](#) in [Li et al. \(2024c\)](#).

Proof of Theorem B.2. It is easy to verify that θ^* is the minimizer of $L_k(\theta)$ for any $k \in [d]$ and hence $\theta_k^* = \theta^*$ by assumption. According to [Theorem B.3](#), it holds that for every sufficiently small neighborhood S of θ^* , there exists a sufficiently large n , such that there is a unique minimizer $\hat{\theta}_{k,n}$ in S . Furthermore, $\hat{\theta}_{k,n}$ satisfies:

$$\sqrt{n} (\hat{\theta}_{k,n} - \theta^*) \xrightarrow{d} \mathcal{N}(0, \Gamma_k).$$

Here due to [Theorem B.4](#), it holds that

$$\begin{aligned} \Gamma_k &= \left(\nabla_{\theta}^2 L_k(\theta^*) \right)^{-1} \text{Cov}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_k)} (-\nabla_{\theta} \log q_{\theta^*}(\mathbf{X}_{\pi}^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})) \left(\nabla_{\theta}^2 L_k(\theta^*) \right)^{-1} \\ &= \left(\nabla_{\theta}^2 L_k(\theta^*) \right)^{-1}. \end{aligned} \quad (12)$$

Fix a $k \in [d-1]$. Then for every $\pi \in \Pi_{k+1}$ and $j \in \pi$, let $\gamma = \pi \setminus \{j\}$ and we have

$$\begin{aligned} \log q_{\theta}(\mathbf{X}_{\pi}^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}}) &= \log q_{\theta}(\mathbf{X}_{\gamma}^{\text{te}}, X_j^{\text{te}} | \mathbf{X}_{-(\gamma \cup \{j\})}^{\text{te}}, \mathbf{X}^{\text{ct}}) \\ &= \log q_{\theta}(X_j^{\text{te}} | \mathbf{X}_{-(\gamma \cup \{j\})}^{\text{te}}, \mathbf{X}^{\text{ct}}) + \log q_{\theta}(\mathbf{X}_{\gamma}^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}). \end{aligned} \quad (13)$$

As a result, we have

$$\begin{aligned} &\nabla_{\theta}^2 L_{k+1}(\theta^*) \\ &= \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_{k+1})} \left[(\nabla_{\theta} \log q_{\theta^*}(\mathbf{X}_{\pi}^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})) (\nabla_{\theta} \log q_{\theta^*}(\mathbf{X}_{\pi}^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}}))^{\top} \right] \\ &\quad \text{(Due to Equation (9))} \\ &= \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \gamma \sim \text{Unif}(\Pi_k), j \in \text{Unif}([d] \setminus \gamma)} \left[(\nabla_{\theta} \log q_{\theta^*}(\mathbf{X}_{\pi}^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})) (\nabla_{\theta} \log q_{\theta^*}(\mathbf{X}_{\pi}^{\text{te}} | \mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}}))^{\top} \right] \\ &\quad \text{(Letting } \pi = \gamma \cup \{j\}\text{)} \\ &= \mathbb{E}_{\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}, \gamma, j} \left[\left(\nabla_{\theta} \log q_{\theta^*}(X_j^{\text{te}} | \mathbf{X}_{-(\gamma \cup \{j\})}^{\text{te}}, \mathbf{X}^{\text{ct}}) + \nabla_{\theta} \log q_{\theta^*}(\mathbf{X}_{\gamma}^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right) \times \right. \\ &\quad \left. \left(\nabla_{\theta} \log q_{\theta^*}(X_j^{\text{te}} | \mathbf{X}_{-(\gamma \cup \{j\})}^{\text{te}}, \mathbf{X}^{\text{ct}}) + \nabla_{\theta} \log q_{\theta^*}(\mathbf{X}_{\gamma}^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right)^{\top} \right]. \quad \text{(By Equation (13))} \end{aligned}$$

Define

$$\begin{aligned} A &= \mathbb{E}_{\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}, \gamma, j} \left[\left(\nabla_{\theta} \log q_{\theta^*}(X_j^{\text{te}} | \mathbf{X}_{-(\gamma \cup \{j\})}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right) \left(\nabla_{\theta} \log q_{\theta^*}(X_j^{\text{te}} | \mathbf{X}_{-(\gamma \cup \{j\})}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right)^{\top} \right]. \\ B &= \mathbb{E}_{\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}, \gamma, j} \left[\left(\nabla_{\theta} \log q_{\theta^*}(X_j^{\text{te}} | \mathbf{X}_{-(\gamma \cup \{j\})}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right) \left(\nabla_{\theta} \log q_{\theta^*}(\mathbf{X}_{\gamma}^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right)^{\top} \right]. \\ C &= \mathbb{E}_{\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}, \gamma, j} \left[\left(\nabla_{\theta} \log q_{\theta^*}(\mathbf{X}_{\gamma}^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right) \left(\nabla_{\theta} \log q_{\theta^*}(\mathbf{X}_{\gamma}^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right)^{\top} \right]. \end{aligned}$$

Then we have

$$\nabla_{\theta}^2 L_{k+1}(\theta^*) = A + B + B^{\top} + C.$$

It is easy to verify that $C = \nabla_{\theta}^2 L_k(\theta^*)$. Now consider B . It holds that

$$\begin{aligned} B &= \mathbb{E}_{\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}, \gamma, j} \left[\left(\nabla_{\theta} \log q_{\theta^*}(X_j^{\text{te}} | \mathbf{X}_{-(\gamma \cup \{j\})}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right) \left(\nabla_{\theta} \log q_{\theta^*}(\mathbf{X}_{\gamma}^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right)^{\top} \right] \\ &= \mathbb{E}_{\mathbf{X}^{\text{ct}}, \gamma, j, \mathbf{X}_{-\gamma}^{\text{te}}} \left[\left(\nabla_{\theta} \log q_{\theta^*}(X_j^{\text{te}} | \mathbf{X}_{-(\gamma \cup \{j\})}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right) \cdot \mathbb{E}_{\mathbf{X}_{\gamma}^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}} \left[\left(\nabla_{\theta} \log q_{\theta^*}(\mathbf{X}_{\gamma}^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right)^{\top} \right] \right] \\ &= \mathbb{E}_{\mathbf{X}^{\text{ct}}, \gamma, j, \mathbf{X}_{-\gamma}^{\text{te}}} \left[\left(\nabla_{\theta} \log q_{\theta^*}(X_j^{\text{te}} | \mathbf{X}_{-(\gamma \cup \{j\})}^{\text{te}}, \mathbf{X}^{\text{ct}}) \right) \cdot 0 \right] \quad \text{(See Equation (14) below)} \\ &= 0. \end{aligned}$$

Here the third equation is due to:

$$\begin{aligned}
\mathbb{E}_{\mathbf{X}_\gamma^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}} [\nabla_\theta \log q_{\theta^*} (\mathbf{X}_\gamma^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}})] &= \int_{\Omega^k} \frac{\nabla_\theta q_{\theta^*} (\mathbf{X}_\gamma^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}})}{q_{\theta^*} (\mathbf{X}_\gamma^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}})} \cdot p (\mathbf{X}_\gamma^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}) d\mathbf{X}_\gamma^{\text{te}} \\
&= \int_{\Omega^k} \nabla_\theta q_{\theta^*} (\mathbf{X}_\gamma^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}) d\mathbf{X}_\gamma^{\text{te}} \\
&= \nabla_\theta \int_{\Omega^k} q_{\theta^*} (\mathbf{X}_\gamma^{\text{te}} | \mathbf{X}_{-\gamma}^{\text{te}}, \mathbf{X}^{\text{ct}}) d\mathbf{X}_\gamma^{\text{te}} = \nabla_\theta 1 = 0.
\end{aligned} \tag{14}$$

In addition, since $A \succeq 0$, we have

$$\nabla_\theta^2 L_{k+1}(\theta^*) = A + B + B^\top + C = A + \nabla_\theta^2 L_k(\theta^*) \succeq \nabla_\theta^2 L_k(\theta^*).$$

Noting that $\Gamma_k = (\nabla_\theta^2 L_k(\theta^*))^{-1}$ by Equation (12), it follows that $\Gamma_{k+1} \preceq \Gamma_k$. Now the claim follows. \square

B.2.2 Generalization for Joint Distribution Learning

The generalization error for joint distribution learning is characterized by a key concept termed *approximate tensorization of entropy*. It measures the “complexity” of the distribution over $(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$ by evaluating how easily an algorithm can generate samples from the joint distribution $q(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$ with access to local conditional distributions $q(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}^{\text{te}})$. Technically, approximate tensorization of entropy is associated with the mixing time of Gibbs sampling dynamics, which is the sample generation algorithm to be considered.

Definition B.1 (Approximate Tensorization of Entropy (Caputo & Parisi, 2021)). For a distribution $q(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$ and the set of k -cell masks Π_k , if there exists a constant $C_k(q)$ depending on q and k , such that for any distribution r over $(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$,

$$D_{\text{KL}}(r \| q) \leq C_k(q) \cdot \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_k)} [D_{\text{KL}}(r(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}^{\text{te}}) \| q(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}^{\text{te}}))],$$

then q satisfies approximate tensorization of entropy with respect to the constant C_k and the mask set Π_k . Let $\underline{C}_k(q)$ be the minimum of all possible constants $C_k(q)$ such that q satisfies approximate tensorization of entropy.

Before presenting the main result, we introduce a few regularity conditions in the parametric class $q_\theta(\mathbf{X}^{\text{te}} | \mathbf{X}^{\text{ct}})$ that defines the model.

Assumption B.1 (Regularity Conditions in the Parametric Class (Li et al., 2024c)).

1. There exists $\beta \in (0, 1)$ such that $\forall 1 \leq k \leq d, \forall \pi \in \Pi_k$ and $\forall \theta \in \Theta$, $p(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}^{\text{te}}) > 0$ implies $q_\theta(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}^{\text{te}}) > \beta$.
2. For any $\epsilon > 0$, there exists a partition $\text{Par}(\Theta) = \{\Theta_1, \dots, \Theta_{|\text{Par}(\Theta)|}\}$ of Θ , such that $\forall 1 \leq k \leq d, \forall \pi \in \Pi_k, \forall \Theta_i \in \text{Par}(\Theta), \forall \theta_1, \theta_2 \in \Theta_i$, and any $(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$,

$$|\log q_{\theta_1}(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}^{\text{te}}) - \log q_{\theta_2}(\mathbf{X}_\pi^{\text{te}} | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}^{\text{te}})| \leq \frac{\epsilon}{2}.$$

Let $N(\Theta, \epsilon)$ be the cardinality of the smallest partition $\text{Par}(\Theta)$ that satisfies the condition above.

The first assumption implies that the true distribution $p(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}^{\text{te}})$ supports the parametric distribution $q_\theta(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}^{\text{te}})$. The second assumption specifies the covering number of the parameter space Θ and the lipschitz continuity of the log-likelihood loss function.

Theorem B.5 (Formal Version of Theorem 6.3). For $\theta \in \Theta$, assume $q_\theta(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$ satisfies approximate tensorization of entropy with respect to some constant $C_1(q_\theta)$ and the mask set Π_1 . Then for any $1 \leq k \leq d$, $q_\theta(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$ satisfies approximate tensorization of entropy with respect to some constant $C_k(q_\theta)$ and the mask set Π_k . Furthermore, $\underline{C}_{k+1}(q_\theta) \leq \underline{C}_k(q_\theta)$.

Under Assumption B.1, for any $\epsilon > 0$ and any $\delta \in (0, \frac{1}{d})$, with probability at least $1 - d\delta$, for any $1 \leq k \leq d$ and any $\theta \in \Theta$,

$$\mathbb{E}_{\mathbf{X}^{\text{ct}} \sim p} [D_{\text{TV}}(q_\theta(\cdot | \mathbf{X}^{\text{ct}}) \| p(\cdot | \mathbf{X}^{\text{ct}}))] < \sqrt{\frac{1}{2} \underline{C}_k(q_\theta) \left(\hat{L}_k(\theta) + B \log \frac{1}{\beta} + \epsilon \right) + C},$$

where $B = \sqrt{\frac{1}{\delta} \cdot (8|\Omega|)^{d(m+1)} N(\Theta, \epsilon)} + \sqrt{\frac{1}{2n} \cdot \log \frac{8N(\Theta, \epsilon)}{\delta}}$, and $C = \sqrt{\frac{|\Omega|^{3d(m+1)}}{8\delta n}}$.

With a sufficiently large sample size n and a sufficiently small loss value $\hat{L}_k(\theta)$, the upper bound is dominated by the term $\sqrt{\frac{1}{2} \underline{C}_k(q_\theta) (B \log \frac{1}{\beta} + \epsilon)}$, which is scaled by $\underline{C}_k(q_\theta)$, the constant for the approximate tensorization of entropy. The theorem implies a reduced upper bound for the estimation error of the joint distribution with an increasing number of masked cells.

We prove the monotonicity of $\underline{C}_k(q_\theta)$ with respect to k in [Theorem B.6](#), and prove the upper bound for generalization error in [Theorem B.7](#).

Proposition B.6. *For $\theta \in \Theta$, assume $q_\theta(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$ satisfies approximate tensorization of entropy with respect to some constant $C_1(q_\theta)$ and the mask set Π_1 . Then for any $1 \leq k \leq d$, $q_\theta(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$ satisfies approximate tensorization of entropy with respect to some constant $C_k(q_\theta)$ and the mask set Π_k . Furthermore, $\underline{C}_{k+1}(q_\theta) \leq \underline{C}_k(q_\theta)$.*

Proof of Theorem B.6. Since $q_\theta(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$ satisfies approximate tensorization of entropy with respect to some constant $C_1(q_\theta)$ and the mask set Π_1 , for any distribution r over $(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$,

$$D_{\text{KL}}(r \parallel q_\theta) \leq C_1(q_\theta) \cdot \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_1)} [D_{\text{KL}}(r(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}) \parallel q_\theta(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}))].$$

By [Definition B.1](#),

$$D_{\text{KL}}(r \parallel q_\theta) \leq \underline{C}_1(q_\theta) \cdot \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_1)} [D_{\text{KL}}(r(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}) \parallel q_\theta(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}))].$$

We prove the proposition by deduction. Assume for some $1 \leq k < d$, $q_\theta(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$ satisfies approximate tensorization of entropy with respect to some constant $C_k(q_\theta)$ and the mask set Π_k . It follows that

$$D_{\text{KL}}(r \parallel q_\theta) \leq \underline{C}_k(q_\theta) \cdot \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_k)} [D_{\text{KL}}(r(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}) \parallel q_\theta(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}))]. \quad (15)$$

We have

$$\begin{aligned} & \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_{k+1})} [D_{\text{KL}}(r(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}) \parallel q_\theta(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}))] \\ &= \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_k), a \sim \text{Unif}([d] \setminus \pi)} [D_{\text{KL}}(r(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi \cup \{a\}}) \parallel q_\theta(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi \cup \{a\}}))] \\ &\geq \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_k), a \sim \text{Unif}([d] \setminus \pi)} [D_{\text{KL}}(r(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}) \parallel q_\theta(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}))] \\ &= \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_k)} [D_{\text{KL}}(r(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}) \parallel q_\theta(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}))]. \end{aligned} \quad (16)$$

The inequality follows from the data processing inequality:

$$D_{\text{KL}}(p(x, y) \parallel q(x, y)) = \mathbb{E}_x [D_{\text{KL}}(p(y|x) \parallel q(y|x))] + D_{\text{KL}}(p(x) \parallel q(x)) \geq \mathbb{E}_x [D_{\text{KL}}(p(y|x) \parallel q(y|x))].$$

Combining [Equations \(15\)](#) and [\(16\)](#),

$$D_{\text{KL}}(r \parallel q_\theta) \leq \underline{C}_k(q_\theta) \cdot \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_{k+1})} [D_{\text{KL}}(r(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}) \parallel q_\theta(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}_{-\pi}))].$$

Therefore, $q_\theta(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$ satisfies approximate tensorization of entropy with respect to $\underline{C}_k(q_\theta)$ and the mask set Π_{k+1} . It follows that $\underline{C}_{k+1}(q_\theta) \leq \underline{C}_k(q_\theta)$. By deduction, for any $1 \leq k \leq d$, $q_\theta(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$ satisfies approximate tensorization of entropy with respect to some constant $C_k(q_\theta)$ and the mask set Π_k . \square

Proposition B.7. *Under [Assumption B.1](#) and the condition in [Theorem B.6](#), for any $\epsilon > 0$ and any $\delta \in (0, \frac{1}{d})$, with probability at least $1 - d\delta$, for any $1 \leq k \leq d$ and any $\theta \in \Theta$,*

$$\mathbb{E}_{\mathbf{X}^{\text{ct}} \sim p} [D_{\text{TV}}(q_\theta(\cdot | \mathbf{X}^{\text{ct}}) \parallel p(\cdot | \mathbf{X}^{\text{ct}}))] < \sqrt{\frac{1}{2} \underline{C}_k(q_\theta) \left(\hat{L}_k(\theta) + B \log \frac{1}{\beta} + \epsilon \right)} + C,$$

where $B = \sqrt{\frac{1}{\delta} \cdot (8|\Omega|)^{d(m+1)} N(\Theta, \epsilon)} + \sqrt{\frac{1}{2n} \cdot \log \frac{8N(\Theta, \epsilon)}{\delta}}$, and $C = \sqrt{\frac{|\Omega|^{3d(m+1)}}{8\delta n}}$.

The proposition is a corollary from Theorem 4 in [Li et al. \(2024c\)](#), which provides an upper bound for $D_{\text{TV}}(q_\theta \parallel p)$ in our setting. The remaining gap is an extension of the result to total variation between conditional distributions. We present [Li et al. \(2024c\)](#)'s result as a lemma.

Lemma B.8 ([Li et al. \(2024c\)](#), Theorem 4). *Consider random variables $\mathbf{X} \in \Omega^d$ and $\pi \subset [d]$. We are given n i.i.d. samples $\{\mathbf{x}^{(i)}\}_{i=1}^n$ drawn from $p(\mathbf{X})$. For each sample $\mathbf{x}^{(i)}$, we observe l i.i.d. masks $\{\pi_j^{(i)}\}_{j=1}^l$ drawn from $p(\pi | \mathbf{x}^{(i)})$. Consider the empirical loss function:*

$$\hat{L}(\theta) = \frac{1}{nl} \sum_{i=1}^n \sum_{j=1}^l -\log q_\theta \left(\mathbf{x}_{\pi_j^{(i)}}^{(i)} | \mathbf{x}_{-\pi_j^{(i)}}^{(i)}, \pi_j^{(i)} \right).$$

Suppose the following conditions are met:

1. There exists a constant $C(q_\theta)$ depending on q_θ , such that for any distribution r over \mathbf{X} ,

$$D_{\text{KL}}(r \parallel q) \leq C(q_\theta) \cdot \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x}), \pi \sim p(\pi|\mathbf{x})} [D_{\text{KL}}(r(\cdot|\mathbf{x}_{-\pi}, \pi) \parallel q(\cdot|\mathbf{x}_{-\pi}, \pi)].$$

2. There exists $\beta \in (0, 1)$ such that $\forall \pi \subset [d]$ and $\forall \theta \in \Theta$, $p(\mathbf{x}_\pi | \mathbf{x}_{-\pi}, \pi) > 0$ implies $q_\theta(\mathbf{x}_\pi | \mathbf{x}_{-\pi}, \pi) > \beta$.

3. For any $\epsilon > 0$, there exists a partition $\text{Par}(\Theta) = \{\Theta_1, \dots, \Theta_{|\text{Par}(\Theta)|}\}$ of Θ , such that $\forall \pi \subset [d]$, $\forall \Theta_i \in \text{Par}(\Theta)$, $\forall \theta_1, \theta_2 \in \Theta_i$, and any \mathbf{x} ,

$$|\log q_{\theta_1}(\mathbf{x}_\pi | \mathbf{x}_{-\pi}, \pi) - \log q_{\theta_2}(\mathbf{x}_\pi | \mathbf{x}_{-\pi}, \pi)| \leq \frac{\epsilon}{2}.$$

Let $N(\Theta, \epsilon)$ be the cardinality of the smallest partition $\text{Par}(\Theta)$ that satisfies the condition above.

Then for any $\epsilon > 0$ and any $\delta \in (0, 1)$, with probability at least $1 - \delta$, for any $\theta \in \Theta$,

$$D_{\text{TV}}(q_\theta(\mathbf{X}) \parallel p(\mathbf{X})) < \sqrt{\frac{1}{2}C(q_\theta)\left(\hat{L}(\theta) + B \log \frac{1}{\beta} + \epsilon\right)} + C,$$

where $B = \sqrt{\frac{1}{\delta} \cdot (8|\Omega|)^d N(\Theta, \epsilon)} + \sqrt{\frac{1}{2n} \cdot \log \frac{8N(\Theta, \epsilon)}{\delta}}$, and $C = \sqrt{\frac{|\Omega|^{3d}}{8\delta n}}$.

Remark B.1. Theorem 4 in [Li et al. \(2024c\)](#) is specified for $\hat{\theta}$ as the minimizer of the empirical loss function. In fact, the proof applies uniformly to arbitrarily $\theta \in \Theta$.

Proof of Theorem B.7. For each pair of $(\mathbf{x}^{\text{ct},(i)}, \mathbf{x}^{\text{te},(i)})$, exactly one mask π_i is drawn independently from $\text{Unif}(\Pi_k)$. Therefore, $\pi \perp\!\!\!\perp (\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}})$. It follows from [Theorem B.8](#) that for each $1 \leq k \leq d$, for any $\epsilon > 0$ and any $\delta \in (0, 1)$, with probability at least $1 - \delta$, for any $\theta \in \Theta$,

$$D_{\text{TV}}(q_\theta(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}}) \parallel p(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})) < \sqrt{\frac{1}{2}C_k(q_\theta)\left(\hat{L}_k(\theta) + B \log \frac{1}{\beta} + \epsilon\right)} + C, \quad (17)$$

where $B = \sqrt{\frac{1}{\delta} \cdot (8|\Omega|)^{d(m+1)} N(\Theta, \epsilon)} + \sqrt{\frac{1}{2n} \cdot \log \frac{8N(\Theta, \epsilon)}{\delta}}$, and $C = \sqrt{\frac{|\Omega|^{3d(m+1)}}{8\delta n}}$.

By union bound, for any $\epsilon > 0$ and any $\delta \in (0, \frac{1}{d})$, with probability at least $1 - d\delta$, [Equation \(17\)](#) is satisfied for any $1 \leq k \leq d$ and any $\theta \in \Theta$.

Furthermore, we have

$$\begin{aligned} \mathbb{E}_{\mathbf{X}^{\text{ct}} \sim p}[D_{\text{TV}}(q_\theta(\cdot | \mathbf{X}^{\text{ct}}) \parallel p(\cdot | \mathbf{X}^{\text{ct}}))] &= \sum_{\mathbf{x}^{\text{ct}} \in \Omega^{m \times d}} p(\mathbf{x}^{\text{ct}}) D_{\text{TV}}(q_\theta(\cdot | \mathbf{x}^{\text{ct}}) \parallel p(\cdot | \mathbf{x}^{\text{ct}})) \\ &= \sum_{\mathbf{x}^{\text{ct}} \in \Omega^{m \times d}} p(\mathbf{x}^{\text{ct}}) \cdot \frac{1}{2} \sum_{\mathbf{x}^{\text{te}} \in \Omega^d} |q_\theta(\mathbf{x}^{\text{te}} | \mathbf{x}^{\text{ct}}) - p(\mathbf{x}^{\text{te}} | \mathbf{x}^{\text{ct}})| \\ &= \frac{1}{2} \sum_{\mathbf{x}^{\text{ct}} \in \Omega^{m \times d}, \mathbf{x}^{\text{te}} \in \Omega^d} |p(\mathbf{x}^{\text{ct}})q_\theta(\mathbf{x}^{\text{te}} | \mathbf{x}^{\text{ct}}) - p(\mathbf{x}^{\text{ct}})p(\mathbf{x}^{\text{te}} | \mathbf{x}^{\text{ct}})| \quad (18) \\ &= \frac{1}{2} \sum_{\mathbf{x}^{\text{ct}} \in \Omega^{m \times d}, \mathbf{x}^{\text{te}} \in \Omega^d} |q_\theta(\mathbf{x}^{\text{te}}, \mathbf{x}^{\text{ct}}) - p(\mathbf{x}^{\text{te}}, \mathbf{x}^{\text{ct}})| \\ &= D_{\text{TV}}(q_\theta(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}}) \parallel p(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})). \end{aligned}$$

The proof is complete by combining [Equations \(17\)](#) and [\(18\)](#). \square