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, the first installment of our large structured-data models (LDMs). LimiX treats 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. LimiX is pretrained using masked joint-distribution modeling with an episodic, context-conditional objective, where the model predicts for query subsets conditioned on dataset-specific contexts, supporting rapid, training-free adaptation at inference. We evaluate LimiX across 10 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 interface, LimiX consistently surpasses strong baselines including gradient-boosting trees, deep tabular networks, recent tabular foundation models, and automated ensembles, on a wide range of tasks—classification, regression, missing value imputation, and data generation—often by substantial margins, while avoiding task-specific architectures or bespoke training per task. To facilitate reproducibility and community-driven research and development, all LimiX models are publicly accessible under Apache 2.0.

Contents

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 (OpenAI, 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 (Xiang et al., 2025; Li et al., 2024a; Yi et al., 2024; Ke et al., 2024; Mildenhall et al., 2020). 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 (Noriega et al., 2023; Yu et al., 2021; Johnson et al., 2023; Benoit & colleagues, 2024). The structural consistency and inherent order of structured data provide a robust framework for quantitative analysis and reliable operations (Silberschatz et al., 2020; Ramakrishnan & Gehrke, 2003; Stonebraker & Çetintemel, 2005), enabling precise prediction, automated reasoning, and rigorous causal inference (Little & Rubin, 2019; Hernán & Robins, 2020; Pearl, 2009). 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).

Traditionally, practitioners deploy pipelines of specialized models with gradient-boosting trees and automated ensembles such as LightGBM (Ke et al., 2017), CatBoost (Dorogush et al., 2018), XGBoost (Chen & Guestrin, 2016), 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; Gorishniy et al., 2021a; Somepalli et al., 2021; Bahri et al., 2021; Yoon et al., 2020). 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 this progress, these attempts 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 introduce LimiX, the first installment of our LDM series. LimiX aims 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 adopts 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 Semantic Encoding (DSE) that encodes feature identities. Pretraining of LimiX 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 retrieves informative samples and features using its own attention scores, aggregates predictions across a handful of lightweight pipelines, and delivers 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 on 10 large structured-data benchmarks with broad regimes of sample size, feature dimensionality, number of classes, categorical–to-numerical feature ratios, missing values, and sample-to-feature ratios. Results show that LimiX demonstrates the strongest predictive performance on all the benchmarks. With a single model and a unified inference interface, LimiX 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 delivers consistent gains, often by large margins, while avoiding task-specific model architectures, customized ensembles, or per-dataset training. On most benchmarks, such as OpenML-CC18 (Bischl et al., 2017), TabArena (Bahri et al., 2021), TALENT-REG (Liu et al., 2024), LimiX is the only model that consistently outperforms AutoGluon, which is considered an outstanding baseline across various tabular-data tasks.

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 $\mathbf{y}^R = \{y_i^R\}_{i=1}^m$; the superscript R indicates the raw input. Here, $\mathbf{x}_i^R \in \mathbb{R}^d$ and $y_i^R \in \mathbb{R}$ 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 Semantic Encoding

The attention score between feature j and j' for sample i is $\frac{1}{\sqrt{p}}\left\langle \mathbf{x}_{i,j}\mathbf{W}_{\mathrm{Q}}, \mathbf{x}_{i,j'}\mathbf{W}_{\mathrm{K}} \right\rangle$. 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 Semantic Encoding* (DSE) 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.

For implementation, let s denote the rank of DSE, 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 approx-

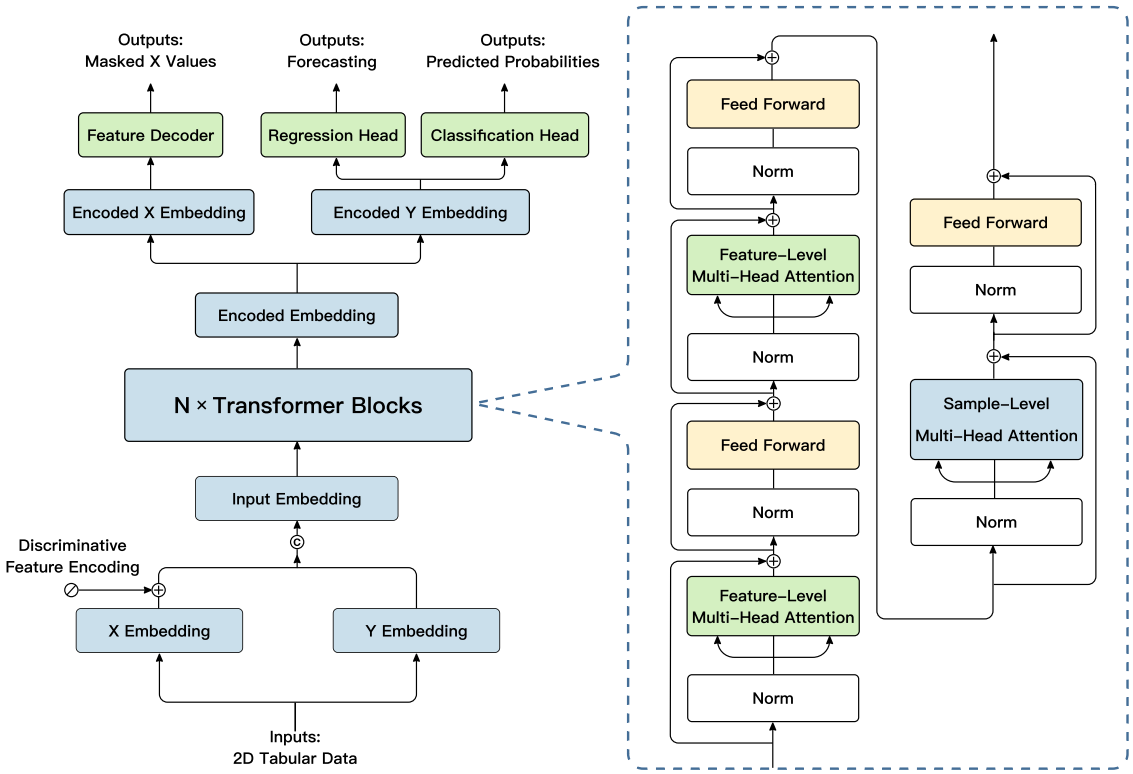


Figure 1: The overall model structure of LimiX.

imately 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 DSE 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 1, LimiX comprises 12 transformer blocks. Each block performs axis-wise self-attention along the feature axis twice and along the sample axis once, and each attention sublayer is followed by a position-wise feed-forward network (FFN). We adopt this asymmetric design with two feature-level passes and one sample-level pass because ablations show that using equal numbers underrepresent feature interactions, whereas doubling attention along the feature axis improves modeling capacity on heterogeneous schemas with minimal overhead. All sublayers use LayerNorm (Ba et al., 2016) in a pre-normalization configuration to stabilize optimization and support scaling in depth.

3 Pretraining

3.1 Context-Conditional Masked Modeling for Joint Distribution Learning

We pretrain LimiX 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.

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 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 to develop causal reasoning capabilities.

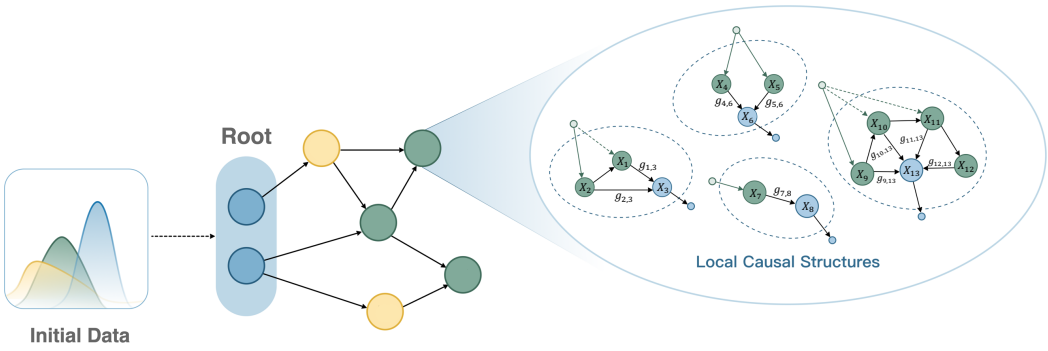


Figure 2: An example of the generated DAG, where $g_{i,j}$ is an edge function defining the relationship between a parent node X_i and its child node X_j in the local causal structures.

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 2, 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 $X_i = f(\{g_{i,k}(X_k)_{k\in \mathrm{PA}(X_i)}, \varepsilon_i)\}$ where $g_{i,k}(\cdot)$ represents the edge function from X_k to X_i, $\mathrm{PA}(X_i)$ 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.

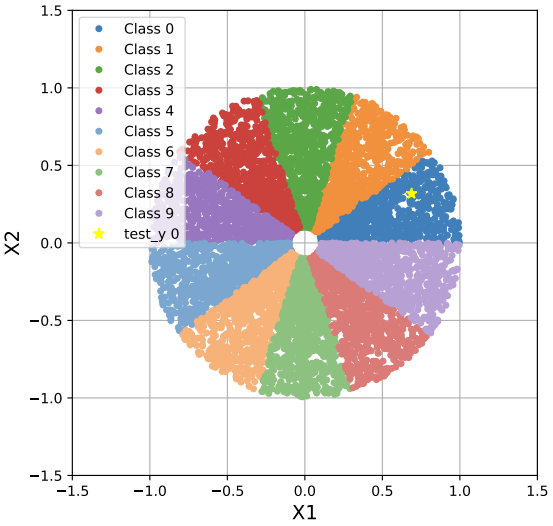
5 Retrieval-based Ensemble

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.

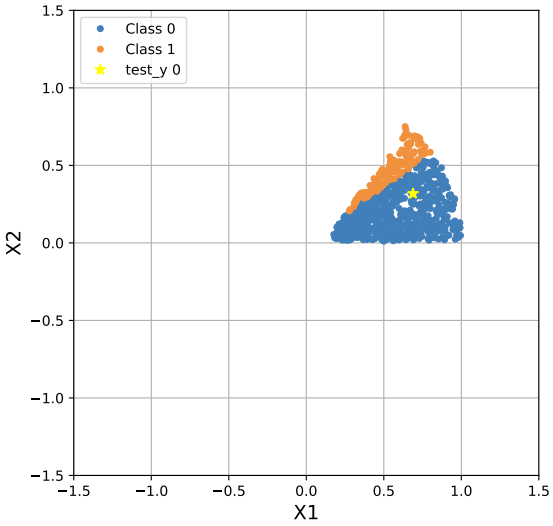
In terms of ensemble, we run multiple inference pipelines per dataset and aggregate the results. In each pipeline, we (i) randomly permute the feature columns and reorder the labels for categorical features or outcomes, and (ii) augment a subset of features with simple, schema-preserving transformations 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 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, the module of 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 $(\mathbf{x}_{ct}, \mathbf{y}_{ct})$. Finally, for a given test sample, we retrieve in-context samples with highest reweighted attention scores as the 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 using $\mathbf{a}_f$ as the feature weights.

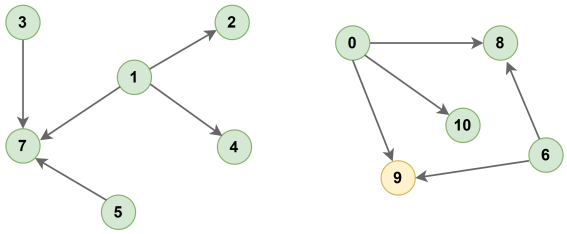


(a) Visualization of synthetic samples. Each sector represents a category of in-context samples. The yellow pentagram represent the test sample.

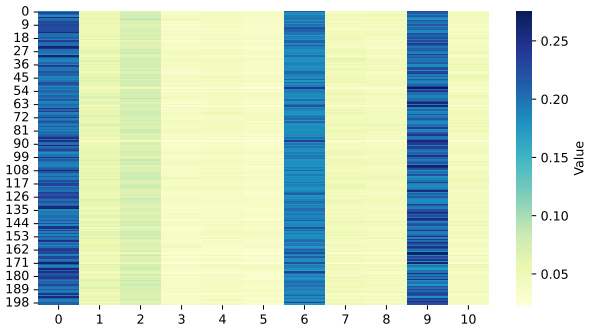


(b) Top 10% of in-context samples sorted by sample-level attention scores.

Figure 3: 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.



(a) Causal DAG of synthetic samples.



(b) Heatmap of feature-level attention scores of Y.

Figure 4: 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.

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 3a. 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 3b, 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 4a, where each blue 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 4b, 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 $\mathbf{X}^{\text{te}} = (X_1^{\text{te}}, \ldots, X_d^{\text{te}}) \in \Omega^d$ to denote the test sample[1]. 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 $\mathbf{X}^{\text{te}}$ 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:

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

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

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

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

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

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

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

At test time, the target variable y may correspond to any feature X_j^{te} of the test sample $\mathbf{X}^{\text{te}}$, 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 Proposition 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 $p(X_\pi^{\text{te}}|\mathbf{X}_{-\pi}^{\text{te}}, \mathbf{X}^{\text{ct}})$ 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 $\mathbf{X}^{\text{ct}} \perp \mathbf{X}^{\text{te}} \mid S$, which yields

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

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.

[1]For 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 Proposition 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 $\{q_\theta(\mathbf{X}^{\text{te}}_\pi|\mathbf{X}^{\text{te}}_{-\pi}, \mathbf{X}^{\text{ct}}) : \pi \in \Pi_k\}$.

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 $q_{\theta^*}(\mathbf{X}^{te}_\pi|\mathbf{X}^{te}_{-\pi}, \mathbf{X}^{ct}) = p(\mathbf{X}^{te}_\pi|\mathbf{X}^{te}_{-\pi}, \mathbf{X}^{ct})$ for all $\mathbf{X}^{ct} \in \Omega^{m\times d}$, $\mathbf{X}^{te} \in \Omega^d$, and $\pi \subseteq [d]$, and that the minimizer of $L_k(\theta)$ is unique for every k. Then, under mild regularity conditions, as $n \to \infty$,*

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

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 $q_\theta(\mathbf{X}^{te}|\mathbf{X}^{ct})$ and the true conditional distribution $p(\mathbf{X}^{te}|\mathbf{X}^{ct})$ is at most:*

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

where constants $C_k(q_\theta)$ depend only on q_θ and k. Furthermore, $C_{k+1}(q_\theta) \le C_k(q_\theta)$ 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 (Bahri et al., 2021).

Furthermore, we introduce **B**alanced **C**omprehensive **C**hallenging **O**mni-domain (BCCO) Benchmark, comprising BCCO-CLS and BCCO-REG, for the evaluation of LimiXand 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), 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-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 with a range of state-of-the-art baseline models, categorized into tree-based models, neural networks, and recent ICL-based approaches.

- **Tree-based approaches.** We include CatBoost (Dorogush et al., 2018), XGBoost (Chen & Guestrin, 2016), LightGBM (Ke et al., 2017), 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 Appendix 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.
- **Neural Network (NN) based approaches.** We evaluate against TabNet (Arik & Pfister, 2021), DCN-V2 (Wang et al., 2021), AutoInt (Song et al., 2019), Node (Popov et al., 2019), ResNet (He et al., 2016), RealMLP (Holzmüller et al., 2024), MLP-PLR (Gorishniy et al., 2022), TabR (Gorishniy et al., 2023), TANGOS (Jeffares et al., 2023), ModernNCA (Ye et al., 2024), T2G-Former (Yan et al., 2023), FT-Transformer (Gorishniy et al., 2021b), ExcelFormer (Chen et al., 2023a), SAINT (Somepalli et al., 2021), TabTransformer (Huang et al., 2020), SwitchTab (Wu et al., 2024), DANets (Chen et al., 2022), GrowNet (Badirli et al., 2020), Trompt (Chen et al., 2023b), and SNN (Chatiras, 2024). 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), Mitra (Zhang & Danielle, 2025), and TabICL (Qu et al., 2025).

Metrics. To evaluate model performance, we employ ROC AUC (area under the receiver operating characteristic curve; One-vs-One), accuracy, and F1 score for classification tasks.

Results. Figure 5 shows that LimiX achieves the best performance among all methods on most datasets in BCCO-CLS. 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 6 to 11, we can see that LimiX 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.

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

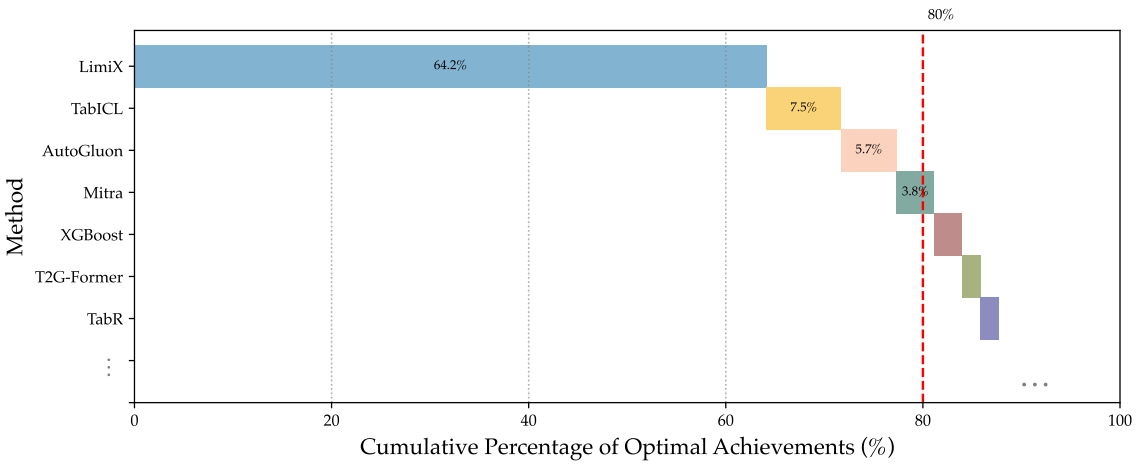


Figure 5: The proportion of models achieving the best AUC. The length of each bar represents the proportion of the 106 datasets in BCCO-CLS where a given method achieves the highest AUC.

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.

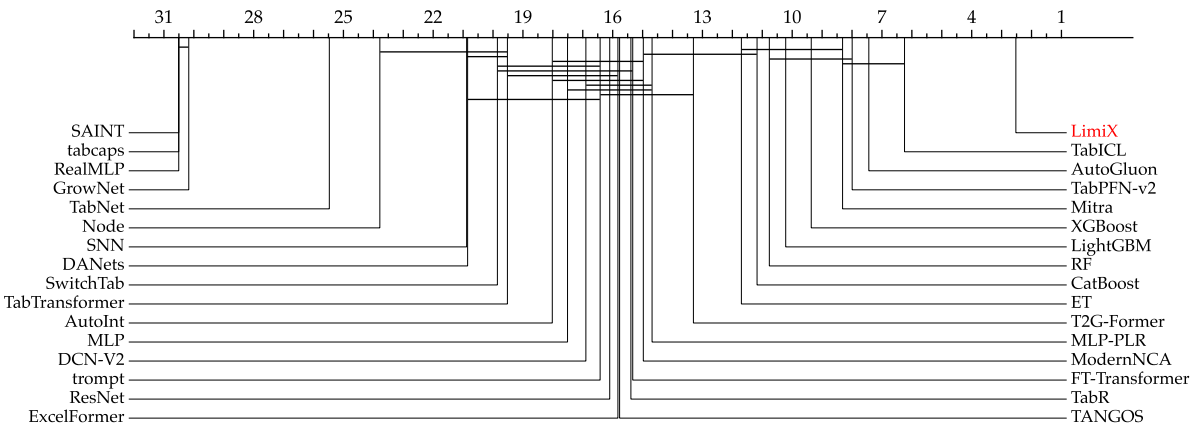
From Figure 12, we observe that LimiX exhibits leading performance across all subgroups compared with other methods. Notably, after stratification, under some subgroups like the third subgroup in Figure 12c that indicates a larger training sample size, AutoGluon proves to be a strong competitor to ICL-based models. In these cases, LimiX is the only ICL-based model that outperforms AutoGluon. Notably, Figure 12f shows that, as the proportion of categorical features increases, performance for most baselines drops sharply, reflecting the sparsity and high-cardinality challenges. In contrast, LimiX 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.

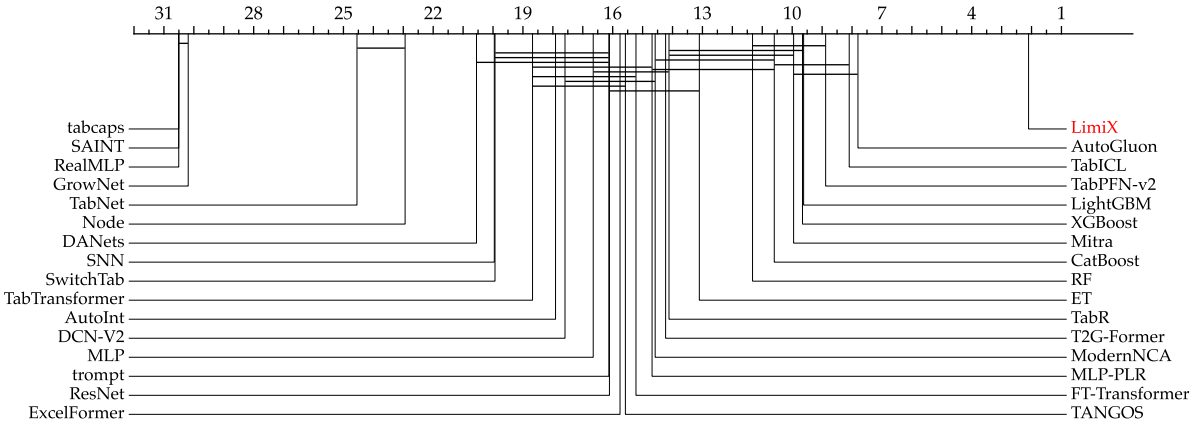
	BCCO-CLS					
	Mean			Rank		
Model	AUC (↑)	Acc. (↑)	F1 (↑)	AUC (↓)	Acc. (↓)	F1 (↓)
LimiX	**0.871**$_{\pm 0.003}$	**0.804**$_{\pm 0.003}$	**0.731**$_{\pm 0.008}$	**2.509**	**2.085**	**2.491**
TabICL	0.847$_{\pm 0.008}$	0.768$_{\pm 0.010}$	0.672$_{\pm 0.016}$	6.236	7.906	9.340
AutoGluon	0.846$_{\pm 0.000}$	0.771$_{\pm 0.001}$	0.677$_{\pm 0.001}$	7.434	7.491	8.528
TabPFN-v2	0.844$_{\pm 0.004}$	0.771$_{\pm 0.005}$	0.679$_{\pm 0.008}$	7.972	8.660	9.745
Mitra	0.841$_{\pm 0.000}$	0.769$_{\pm 0.000}$	0.679$_{\pm 0.000}$	8.283	9.623	10.670
XGBoost	0.834$_{\pm 0.005}$	0.762$_{\pm 0.007}$	0.674$_{\pm 0.012}$	9.349	9.453	9.887
LightGBM	0.832$_{\pm 0.005}$	0.763$_{\pm 0.006}$	0.678$_{\pm 0.011}$	10.208	9.396	9.840
CatBoost	0.829$_{\pm 0.008}$	0.757$_{\pm 0.010}$	0.664$_{\pm 0.014}$	10.755	11.028	11.934
RF	0.829$_{\pm 0.007}$	0.756$_{\pm 0.008}$	0.652$_{\pm 0.014}$	11.170	10.321	10.472
ET	0.825$_{\pm 0.005}$	0.745$_{\pm 0.006}$	0.618$_{\pm 0.014}$	11.689	12.774	15.358
ModernNCA	0.813$_{\pm 0.008}$	0.750$_{\pm 0.007}$	0.657$_{\pm 0.014}$	14.962	14.302	13.528
T2G-Former	0.805$_{\pm 0.012}$	0.735$_{\pm 0.011}$	0.637$_{\pm 0.020}$	13.283	14.028	13.075
TabR	0.804$_{\pm 0.011}$	0.744$_{\pm 0.010}$	0.652$_{\pm 0.016}$	15.377	13.925	13.151
MLP-PLR	0.802$_{\pm 0.007}$	0.734$_{\pm 0.010}$	0.635$_{\pm 0.016}$	14.670	14.481	13.877
ExcelFormer	0.799$_{\pm 0.012}$	0.733$_{\pm 0.013}$	0.635$_{\pm 0.021}$	15.811	15.528	14.557
TANGOS	0.798$_{\pm 0.009}$	0.730$_{\pm 0.011}$	0.632$_{\pm 0.019}$	15.764	15.283	15.245
FT-Transformer	0.797$_{\pm 0.016}$	0.732$_{\pm 0.016}$	0.623$_{\pm 0.028}$	15.330	14.972	15.113
ResNet	0.793$_{\pm 0.014}$	0.724$_{\pm 0.015}$	0.634$_{\pm 0.022}$	16.085	15.925	14.962
AutoInt	0.785$_{\pm 0.023}$	0.719$_{\pm 0.019}$	0.606$_{\pm 0.031}$	18.019	17.708	17.321
DCN-V2	0.783$_{\pm 0.020}$	0.717$_{\pm 0.018}$	0.599$_{\pm 0.034}$	16.887	17.349	16.925
MLP	0.783$_{\pm 0.011}$	0.720$_{\pm 0.009}$	0.612$_{\pm 0.019}$	17.509	16.453	17.038
SNN	0.763$_{\pm 0.024}$	0.701$_{\pm 0.018}$	0.570$_{\pm 0.030}$	20.877	19.755	19.915
SwitchTab	0.762$_{\pm 0.018}$	0.701$_{\pm 0.017}$	0.592$_{\pm 0.037}$	19.821	19.736	18.613
DANets	0.756$_{\pm 0.033}$	0.693$_{\pm 0.024}$	0.570$_{\pm 0.047}$	20.830	20.311	20.726
TabTransformer	0.754$_{\pm 0.018}$	0.700$_{\pm 0.015}$	0.572$_{\pm 0.026}$	19.500	18.453	17.821
Node	0.738$_{\pm 0.018}$	0.677$_{\pm 0.014}$	0.502$_{\pm 0.039}$	23.783	22.689	23.792
Trompt	0.728$_{\pm 0.014}$	0.663$_{\pm 0.011}$	0.560$_{\pm 0.021}$	16.208	15.689	16.123
TabNet	0.696$_{\pm 0.041}$	0.669$_{\pm 0.027}$	0.544$_{\pm 0.043}$	25.443	24.481	23.679

Table 2: Statistical profile of the benchmark BCCO-CLS, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively.

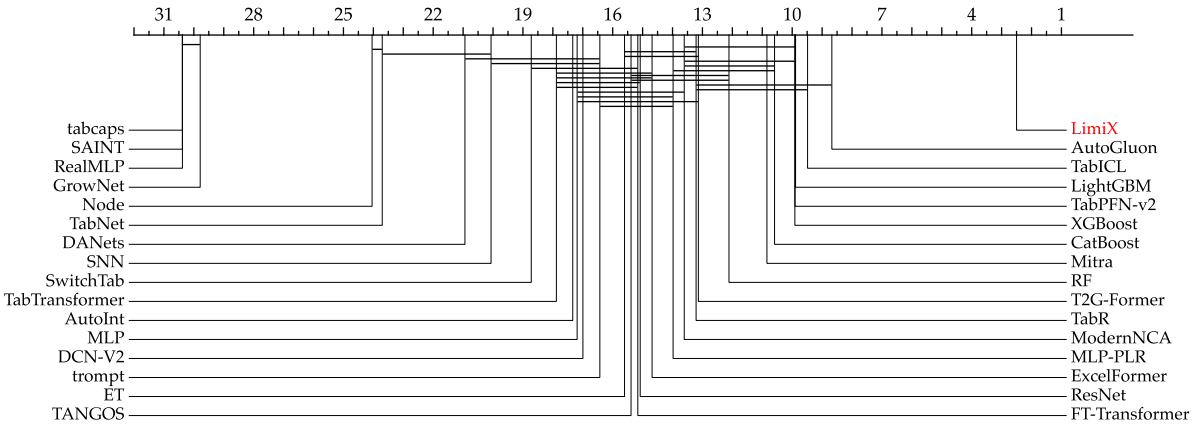
	Statistics						
Metric	Q10	Q50	Q90	Mean	Std	Min	Max
# Features	4	12	48	23	33	1	259
# Classes	2	2	5	3	2	2	10
Missing Values (Ratio)	0	0	0.069	0.024	0.068	0	0.403
Categorical Features (Ratio)	0	0.384	0.929	0.399	0.337	0	1
Features w/ Missing Values (Ratio)	0	0	0.523	0.126	0.241	0	0.909



(a) AUC on the BCCO-CLS benchmark.



(b) Accuracy on the BCCO-CLS benchmark.



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

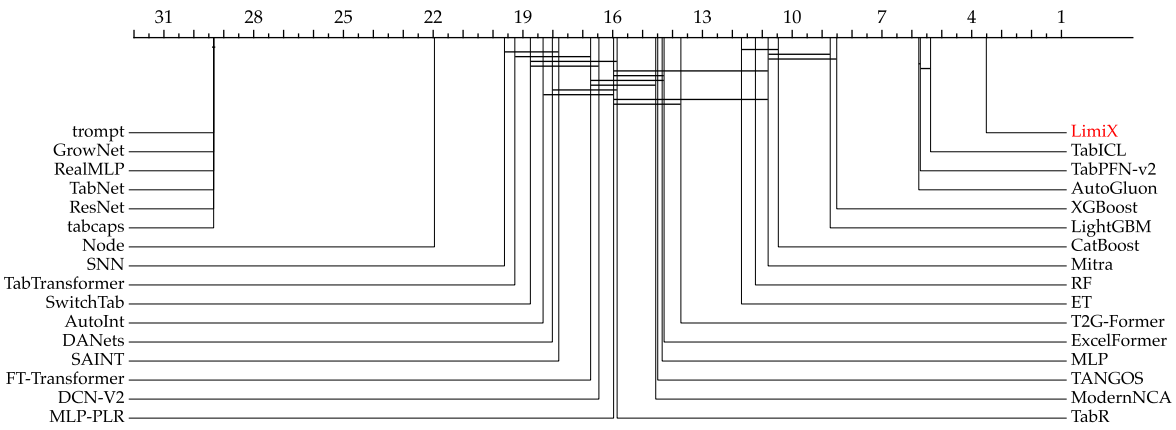
Figure 6: Critical difference diagrams on BCCO-CLS benchmark.

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

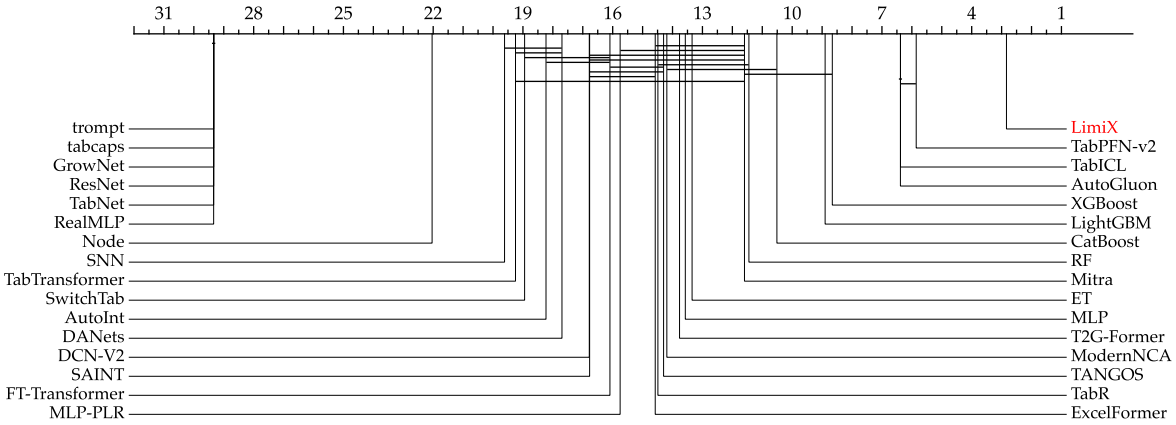
	TALENT-CLS					
	Mean			Rank		
Model	AUC (↑)	Acc. (↑)	F1 (↑)	AUC (↓)	Acc. (↓)	F1 (↓)
LimiX	**0.903**$_{\pm 0.003}$	**0.861**$_{\pm 0.004}$	**0.752**$_{\pm 0.008}$	**3.480**	**2.765**	**3.497**
TabPFNv2	0.895$_{\pm 0.002}$	0.850$_{\pm 0.003}$	0.728$_{\pm 0.005}$	5.564	5.514	6.581
TabICL	0.894$_{\pm 0.003}$	0.845$_{\pm 0.005}$	0.715$_{\pm 0.008}$	5.223	6.028	6.972
AutoGluon	0.891$_{\pm 0.000}$	0.845$_{\pm 0.000}$	0.719$_{\pm 0.001}$	5.615	5.978	6.447
XGBoost	0.881$_{\pm 0.003}$	0.837$_{\pm 0.004}$	0.713$_{\pm 0.009}$	8.413	8.335	8.469
LightGBM	0.880$_{\pm 0.003}$	0.836$_{\pm 0.004}$	0.713$_{\pm 0.009}$	8.609	8.536	8.682
RF	0.877$_{\pm 0.003}$	0.828$_{\pm 0.004}$	0.691$_{\pm 0.010}$	11.112	11.089	11.872
CatBoost	0.876$_{\pm 0.005}$	0.828$_{\pm 0.006}$	0.704$_{\pm 0.014}$	10.346	10.201	10.296
ET	0.875$_{\pm 0.003}$	0.821$_{\pm 0.006}$	0.662$_{\pm 0.017}$	11.570	13.095	14.324
MLP	0.862$_{\pm 0.007}$	0.818$_{\pm 0.006}$	0.677$_{\pm 0.014}$	14.257	13.363	13.274
ModernNCA	0.862$_{\pm 0.004}$	0.826$_{\pm 0.004}$	0.681$_{\pm 0.008}$	14.503	14.000	13.128
TANGOS	0.860$_{\pm 0.005}$	0.816$_{\pm 0.005}$	0.676$_{\pm 0.012}$	14.391	14.128	13.430
T2G-Former	0.858$_{\pm 0.006}$	0.820$_{\pm 0.005}$	0.676$_{\pm 0.010}$	13.592	13.536	13.056
TabR	0.856$_{\pm 0.007}$	0.822$_{\pm 0.006}$	0.679$_{\pm 0.014}$	15.749	14.307	13.229
MLP-PLR	0.851$_{\pm 0.005}$	0.816$_{\pm 0.006}$	0.668$_{\pm 0.011}$	15.866	15.525	14.559
DCN-V2	0.848$_{\pm 0.011}$	0.808$_{\pm 0.008}$	0.649$_{\pm 0.019}$	16.335	16.598	15.994
DANets	0.841$_{\pm 0.015}$	0.794$_{\pm 0.018}$	0.636$_{\pm 0.033}$	17.955	17.458	17.425
AutoInt	0.839$_{\pm 0.008}$	0.802$_{\pm 0.006}$	0.639$_{\pm 0.017}$	18.268	18.056	17.486
SwitchTab	0.836$_{\pm 0.007}$	0.791$_{\pm 0.006}$	0.624$_{\pm 0.014}$	18.721	18.793	18.374
ExcelFormer	0.830$_{\pm 0.011}$	0.787$_{\pm 0.008}$	0.607$_{\pm 0.023}$	14.028	14.229	13.235
SNN	0.830$_{\pm 0.006}$	0.790$_{\pm 0.005}$	0.655$_{\pm 0.014}$	19.559	19.436	19.480
TabTransformer	0.821$_{\pm 0.009}$	0.780$_{\pm 0.007}$	0.605$_{\pm 0.018}$	19.140	19.061	18.687
SAINT	0.811$_{\pm 0.009}$	0.780$_{\pm 0.006}$	0.631$_{\pm 0.016}$	17.648	16.508	15.687
Node	0.800$_{\pm 0.015}$	0.749$_{\pm 0.011}$	0.502$_{\pm 0.030}$	21.899	21.749	22.313
FT-Transformer	0.784$_{\pm 0.009}$	0.752$_{\pm 0.008}$	0.621$_{\pm 0.015}$	16.391	15.648	15.380

Table 4: Statistical profile of the benchmark TALENT-CLS, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively.

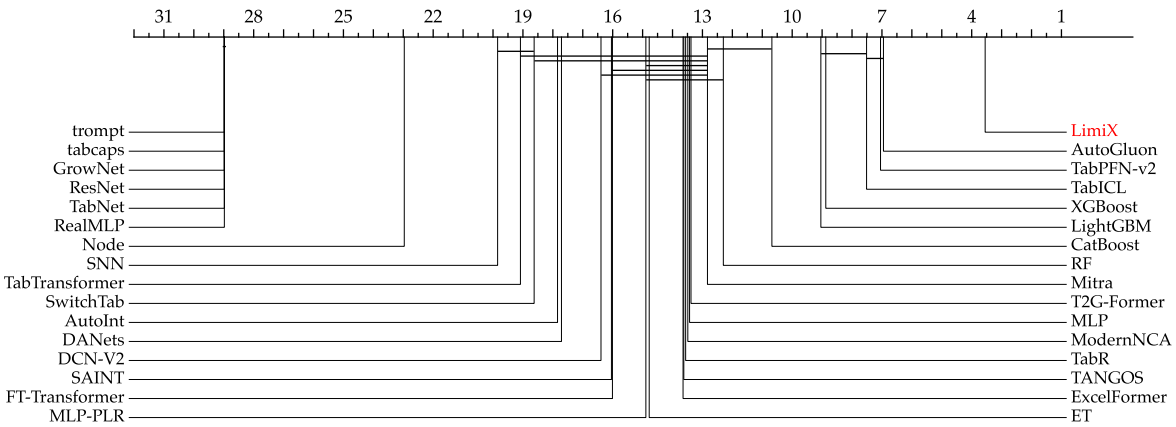
	Statistics						
Metric	Q10	Q50	Q90	Mean	Std	Min	Max
# Features	7	19	70	33	47	3	308
# Classes	2	2	10	6	14	2	100
Missing Values (Ratio)	0	0	0	0.001	0.008	0	0.1
Categorical Features (Ratio)	0	0.121	0.972	0.306	0.365	0	1
Features w/ Missing Values (Ratio)	0	0	0	0.025	0.121	0	0.979



(a) AUC on the TALENT-CLS benchmark.



(b) Accuracy on the TALENT-CLS benchmark.



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

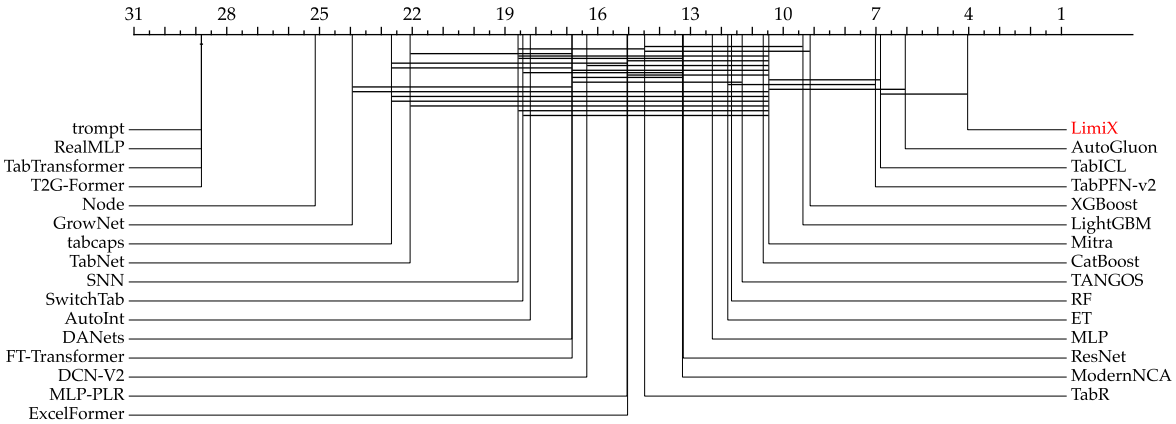
Figure 7: Critical difference diagram on the TALENT-CLS benchmark

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

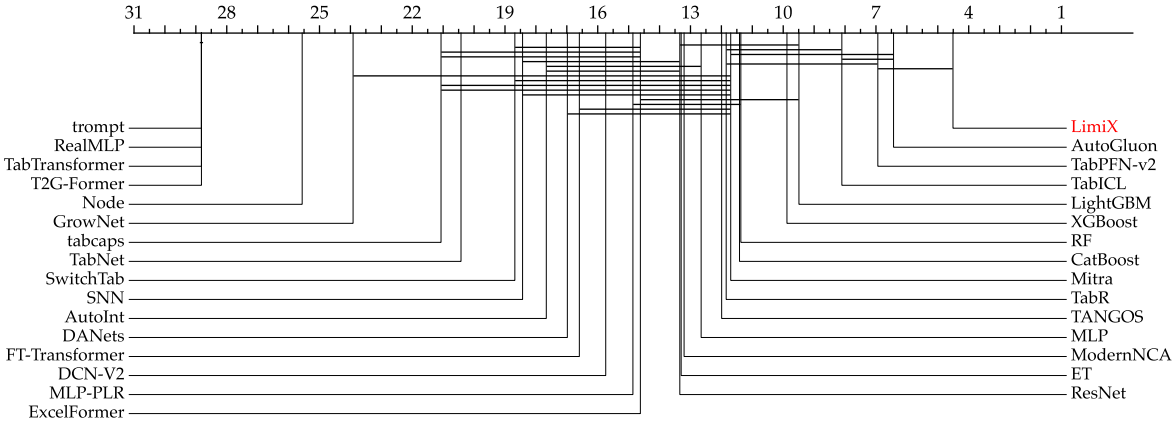
	OpenML-CC18					
	Mean			Rank		
Model	AUC (↑)	Acc. (↑)	F1 (↑)	AUC (↓)	Acc. (↓)	F1 (↓)
LimiX	**0.939**$_{\pm 0.004}$	**0.893**$_{\pm 0.004}$	**0.811**$_{\pm 0.008}$	**4.097**	**3.532**	**3.742**
AutoGluon	0.932$_{\pm 0.000}$	0.885$_{\pm 0.001}$	0.790$_{\pm 0.003}$	5.823	5.306	6.060
TabICL	0.927$_{\pm 0.004}$	0.875$_{\pm 0.008}$	0.782$_{\pm 0.011}$	7.661	6.500	7.468
TabPFN-v2	0.867$_{\pm 0.002}$	0.829$_{\pm 0.006}$	0.735$_{\pm 0.010}$	7.355	7.355	6.968
XGBoost	0.919$_{\pm 0.002}$	0.875$_{\pm 0.005}$	0.771$_{\pm 0.011}$	8.968	8.355	8.952
LightGBM	0.917$_{\pm 0.003}$	0.874$_{\pm 0.005}$	0.771$_{\pm 0.014}$	8.629	8.661	8.629
Mitra	0.759$_{\pm 0.005}$	0.718$_{\pm 0.008}$	0.629$_{\pm 0.016}$	11.016	9.742	10.532
CatBoost	0.915$_{\pm 0.005}$	0.866$_{\pm 0.008}$	0.766$_{\pm 0.016}$	10.903	10.306	10.839
TANGOS	0.911$_{\pm 0.004}$	0.861$_{\pm 0.008}$	0.749$_{\pm 0.015}$	11.629	10.871	11.694
ET	0.911$_{\pm 0.002}$	0.858$_{\pm 0.005}$	0.719$_{\pm 0.016}$	12.726	11.242	14.810
RF	0.915$_{\pm 0.003}$	0.868$_{\pm 0.006}$	0.759$_{\pm 0.016}$	10.758	11.339	11.323
MLP	0.884$_{\pm 0.008}$	0.832$_{\pm 0.011}$	0.724$_{\pm 0.027}$	12.355	11.823	12.677
DCN-V2	0.894$_{\pm 0.011}$	0.840$_{\pm 0.010}$	0.704$_{\pm 0.022}$	13.597	13.048	13.532
ModernNCA	0.882$_{\pm 0.007}$	0.837$_{\pm 0.008}$	0.723$_{\pm 0.017}$	11.032	11.290	10.855
TabR	0.874$_{\pm 0.000}$	0.840$_{\pm 0.000}$	0.722$_{\pm 0.000}$	12.129	9.758	10.097
MLP-PLR	0.869$_{\pm 0.011}$	0.827$_{\pm 0.014}$	0.698$_{\pm 0.024}$	12.339	12.435	12.694
SNN	0.853$_{\pm 0.018}$	0.795$_{\pm 0.017}$	0.639$_{\pm 0.029}$	15.710	15.548	16.210
SwitchTab	0.845$_{\pm 0.021}$	0.782$_{\pm 0.022}$	0.628$_{\pm 0.043}$	15.661	15.903	15.581
TabNet	0.834$_{\pm 0.026}$	0.794$_{\pm 0.020}$	0.641$_{\pm 0.037}$	18.339	17.081	17.500
tabcaps	0.829$_{\pm 0.000}$	0.803$_{\pm 0.000}$	0.622$_{\pm 0.000}$	18.887	17.613	17.742
DANets	0.826$_{\pm 0.018}$	0.769$_{\pm 0.018}$	0.644$_{\pm 0.041}$	14.032	14.161	15.000
ResNet	0.824$_{\pm 0.009}$	0.775$_{\pm 0.013}$	0.697$_{\pm 0.026}$	10.758	11.000	10.210
GrowNet	0.674$_{\pm 0.076}$	0.618$_{\pm 0.066}$	0.426$_{\pm 0.098}$	19.935	19.952	19.081

Table 6: Statistical profile of the benchmark OpenML-CC18, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively.

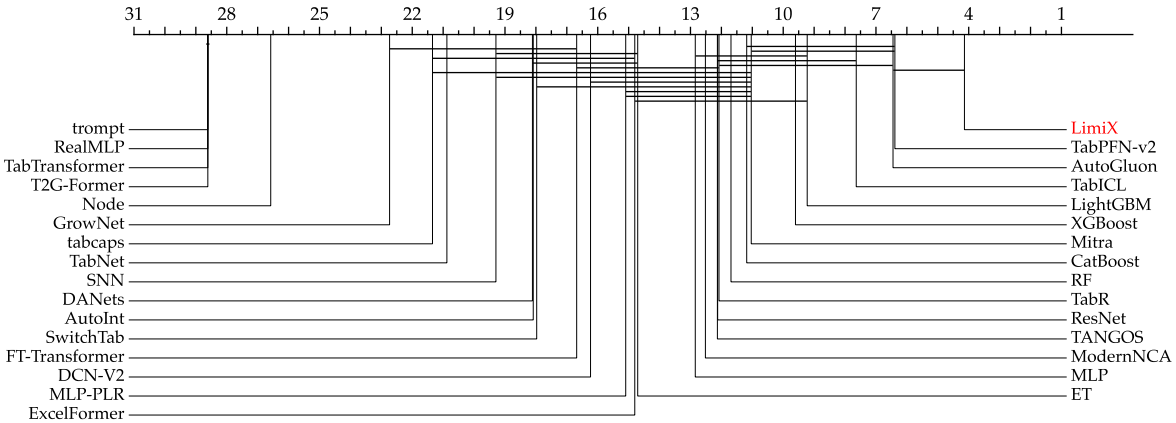
	Statistics						
Metric	Q10	Q50	Q90	Mean	Std	Min	Max
# Features	6	30	611	336	1344	4	10935
# Classes	2	3	10	6	7	2	46
Missing Values (Ratio)	0	0	0.002	0.004	0.019	0	0.139
Categorical Features (Ratio)	0	0.091	1	0.327	0.406	0	1
Features w/ Missing Values (Ratio)	0	0	0.204	0.054	0.165	0	0.757



(a) AUC on the OpenML-CC18 benchmark.



(b) Accuracy on the OpenML-CC18 benchmark.



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

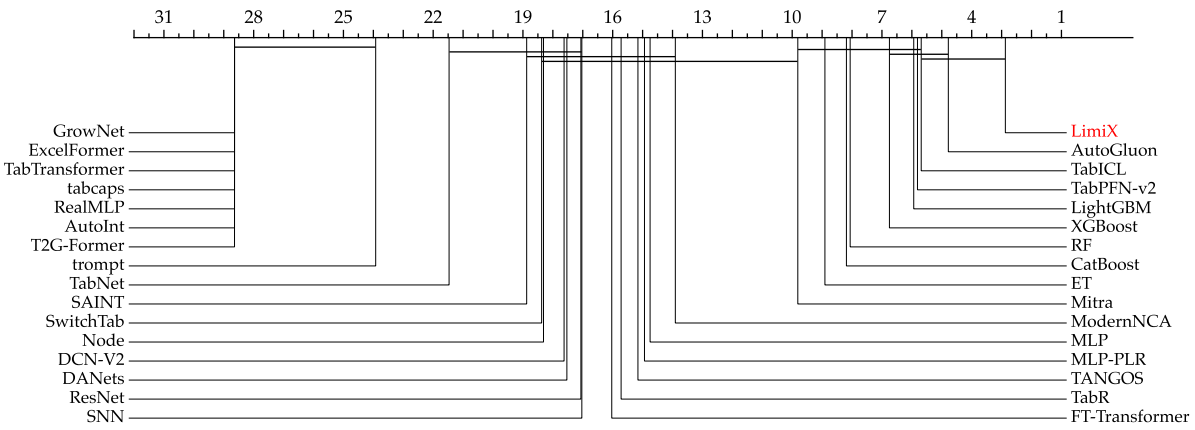
Figure 8: Critical difference diagrams on the OpenML-CC18 benchmark.

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

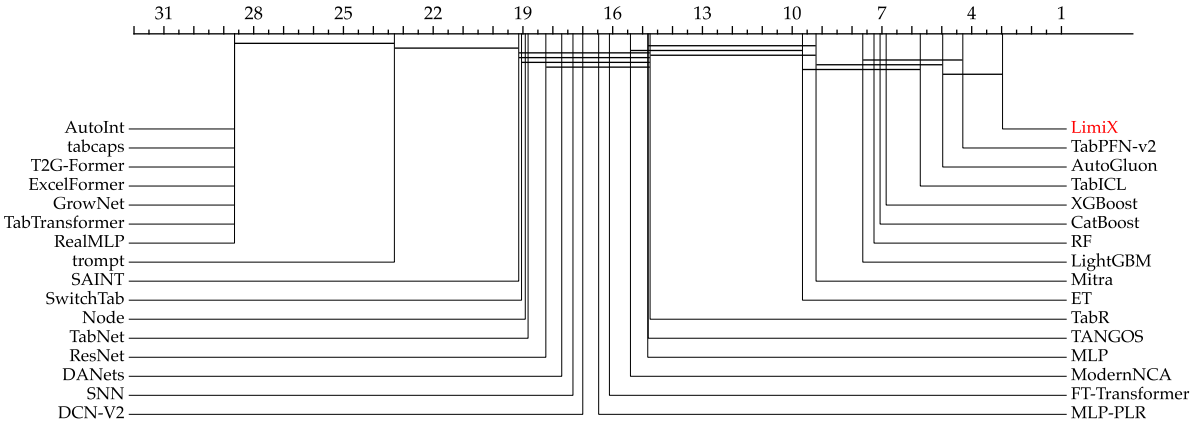
	TabArena					
	Mean			Rank		
Model	AUC (↑)	Acc. (↑)	F1 (↑)	AUC (↓)	Acc. (↓)	F1 (↓)
LimiX	**0.849**$_{\pm 0.004}$	**0.877**$_{\pm 0.001}$	**0.597**$_{\pm 0.006}$	**2.875**	**2.969**	**5.719**
AutoGluon	0.844$_{\pm 0.001}$	0.870$_{\pm 0.000}$	0.574$_{\pm 0.001}$	4.781	4.344	6.812
TabICL	0.840$_{\pm 0.006}$	0.870$_{\pm 0.002}$	0.553$_{\pm 0.007}$	5.688	5.062	8.094
TabPFN-v2	0.841$_{\pm 0.002}$	0.869$_{\pm 0.002}$	0.576$_{\pm 0.006}$	5.812	3.594	6.562
LightGBM	0.841$_{\pm 0.002}$	0.868$_{\pm 0.002}$	0.574$_{\pm 0.011}$	5.938	7.281	8.562
XGBoost	0.838$_{\pm 0.002}$	0.867$_{\pm 0.003}$	0.567$_{\pm 0.018}$	6.750	6.656	8.344
CatBoost	0.835$_{\pm 0.005}$	0.867$_{\pm 0.004}$	0.574$_{\pm 0.015}$	8.152	6.576	7.455
RF	0.837$_{\pm 0.002}$	0.864$_{\pm 0.003}$	0.558$_{\pm 0.011}$	7.758	6.485	8.788
ET	0.833$_{\pm 0.003}$	0.857$_{\pm 0.003}$	0.505$_{\pm 0.019}$	8.788	8.939	12.000
Mitra	0.805$_{\pm 0.000}$	0.862$_{\pm 0.000}$	0.522$_{\pm 0.000}$	10.030	8.485	11.303
ModernNCA	0.794$_{\pm 0.007}$	0.848$_{\pm 0.004}$	0.505$_{\pm 0.014}$	13.667	15.212	12.636
MLP-PLR	0.781$_{\pm 0.008}$	0.834$_{\pm 0.004}$	0.466$_{\pm 0.013}$	14.606	15.970	13.909
TANGOS	0.778$_{\pm 0.013}$	0.842$_{\pm 0.006}$	0.489$_{\pm 0.021}$	15.030	14.636	13.030
TabR	0.769$_{\pm 0.011}$	0.838$_{\pm 0.002}$	0.474$_{\pm 0.008}$	15.455	14.636	12.394
ResNet	0.765$_{\pm 0.022}$	0.803$_{\pm 0.025}$	0.501$_{\pm 0.044}$	16.727	17.879	13.394
MLP	0.760$_{\pm 0.023}$	0.812$_{\pm 0.026}$	0.450$_{\pm 0.031}$	14.697	14.485	14.909
FT-Transformer	0.752$_{\pm 0.001}$	0.767$_{\pm 0.000}$	0.410$_{\pm 0.000}$	15.727	15.818	14.485
SNN	0.738$_{\pm 0.016}$	0.797$_{\pm 0.013}$	0.425$_{\pm 0.023}$	16.606	16.636	14.939
DCN-V2	0.737$_{\pm 0.029}$	0.810$_{\pm 0.019}$	0.413$_{\pm 0.037}$	17.273	16.394	15.455
SwitchTab	0.734$_{\pm 0.032}$	0.788$_{\pm 0.014}$	0.360$_{\pm 0.024}$	18.242	18.091	18.061
DANets	0.725$_{\pm 0.032}$	0.771$_{\pm 0.056}$	0.421$_{\pm 0.054}$	17.364	17.424	14.455
Node	0.699$_{\pm 0.040}$	0.739$_{\pm 0.052}$	0.342$_{\pm 0.050}$	18.091	18.091	17.333
SAINT	0.694$_{\pm 0.018}$	0.732$_{\pm 0.016}$	0.434$_{\pm 0.029}$	18.364	18.576	15.818
TabNet	0.651$_{\pm 0.044}$	0.749$_{\pm 0.044}$	0.383$_{\pm 0.036}$	21.273	18.182	17.030

Table 8: Statistical profile of the benchmark TabArena, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively.

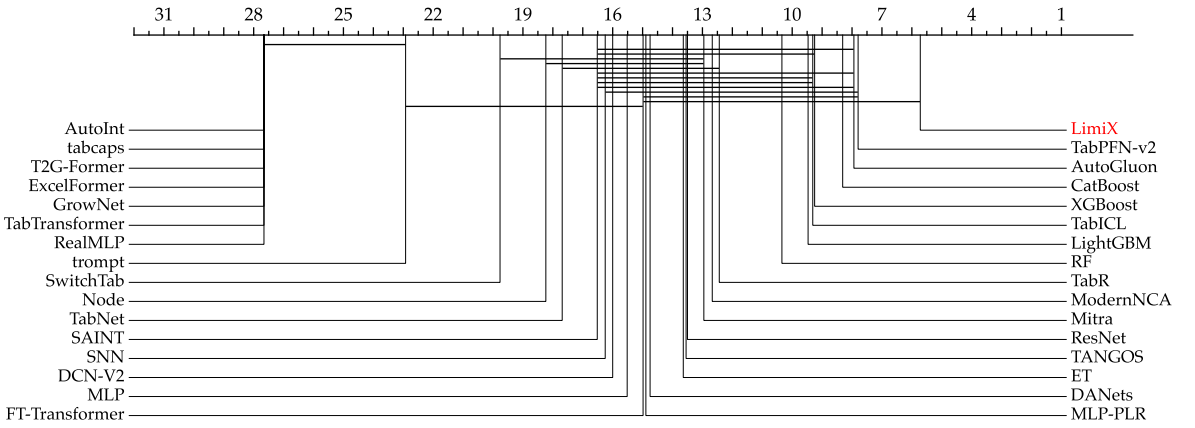
	Statistics						
Metric	Q10	Q50	Q90	Mean	Std	Min	Max
# Features	9	21	129	125	374	4	1776
# Classes	2	2	3	262	1577	2	9856
Missing Values (Ratio)	0	0	0.056	0.027	0.109	0	0.672
Categorical Features (Ratio)	0	0.47	1	0.45	0.366	0	1
Features w/ Missing Values (Ratio)	0	0	0.696	0.139	0.294	0	0.994



(a) AUC on the TabArena benchmark.



(b) Accuracy on the TabArena benchmark.



(c) F1-score on the TabArena benchmark.

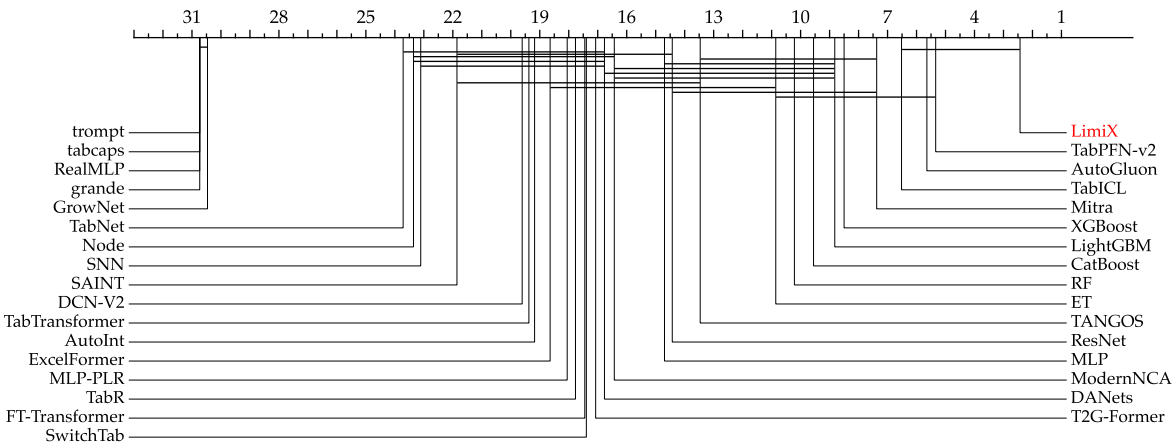
Figure 9: Critical difference diagrams on the TabArena benchmark.

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

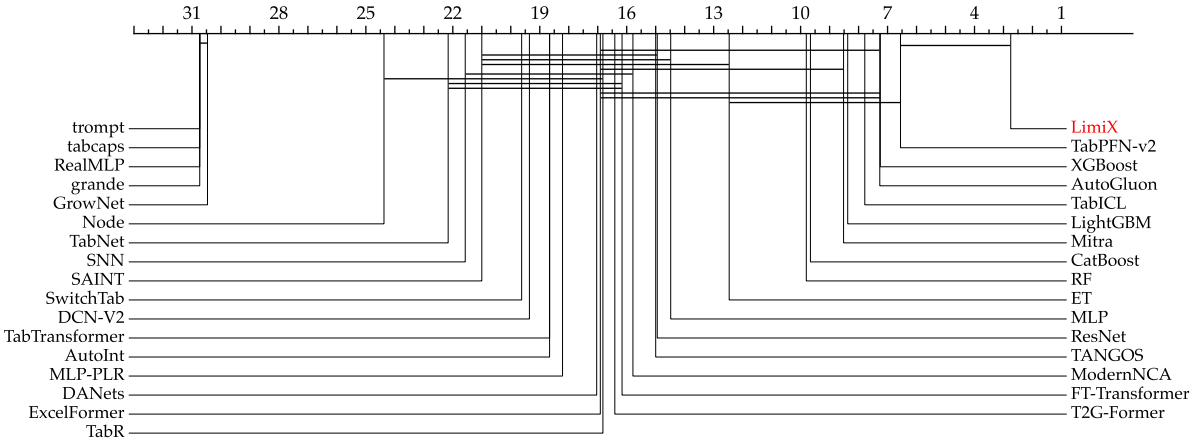
	PFN-CLS					
	Mean			Rank		
Model	AUC (↑)	Acc. (↑)	F1 (↑)	AUC (↓)	Acc. (↓)	F1 (↓)
LimiX	**0.923**$_{\pm 0.003}$	**0.862**$_{\pm 0.002}$	**0.786**$_{\pm 0.006}$	**2.750**	**2.143**	**2.786**
TabPFN-v2	0.912$_{\pm 0.003}$	0.848$_{\pm 0.006}$	0.754$_{\pm 0.008}$	6.214	5.143	6.107
AutoGluon	0.906$_{\pm 0.001}$	0.835$_{\pm 0.001}$	0.738$_{\pm 0.002}$	6.786	5.357	6.464
TabICL	0.903$_{\pm 0.005}$	0.832$_{\pm 0.008}$	0.742$_{\pm 0.014}$	7.786	6.429	8.107
Mitra	0.900$_{\pm 0.000}$	0.834$_{\pm 0.000}$	0.734$_{\pm 0.000}$	7.893	7.179	8.321
XGBoost	0.898$_{\pm 0.003}$	0.831$_{\pm 0.007}$	0.733$_{\pm 0.023}$	6.714	8.250	7.464
LightGBM	0.893$_{\pm 0.004}$	0.826$_{\pm 0.006}$	0.725$_{\pm 0.016}$	7.821	8.500	8.393
RF	0.896$_{\pm 0.006}$	0.822$_{\pm 0.008}$	0.721$_{\pm 0.019}$	9.621	8.897	9.655
CatBoost	0.895$_{\pm 0.005}$	0.819$_{\pm 0.009}$	0.720$_{\pm 0.015}$	9.034	8.414	9.069
ET	0.893$_{\pm 0.003}$	0.809$_{\pm 0.007}$	0.675$_{\pm 0.018}$	10.207	11.276	12.448
TANGOS	0.869$_{\pm 0.008}$	0.797$_{\pm 0.008}$	0.692$_{\pm 0.022}$	12.966	14.034	12.931
MLP	0.864$_{\pm 0.010}$	0.794$_{\pm 0.009}$	0.687$_{\pm 0.019}$	13.793	13.759	12.793
ResNet	0.864$_{\pm 0.009}$	0.793$_{\pm 0.010}$	0.698$_{\pm 0.019}$	14.276	13.483	13.414
ModernNCA	0.859$_{\pm 0.005}$	0.795$_{\pm 0.008}$	0.685$_{\pm 0.014}$	15.552	15.000	15.034
SwitchTab	0.853$_{\pm 0.008}$	0.770$_{\pm 0.011}$	0.607$_{\pm 0.026}$	16.448	18.276	16.828
T2G-Former	0.852$_{\pm 0.012}$	0.786$_{\pm 0.008}$	0.679$_{\pm 0.012}$	16.069	15.276	15.034
FT-Transformer	0.852$_{\pm 0.009}$	0.785$_{\pm 0.009}$	0.668$_{\pm 0.016}$	16.759	15.138	16.034
TabR	0.849$_{\pm 0.009}$	0.782$_{\pm 0.009}$	0.671$_{\pm 0.019}$	16.724	15.448	14.414
MLP-PLR	0.845$_{\pm 0.004}$	0.783$_{\pm 0.006}$	0.650$_{\pm 0.013}$	16.966	17.345	17.310
DCN-V2	0.843$_{\pm 0.016}$	0.770$_{\pm 0.014}$	0.642$_{\pm 0.033}$	18.724	18.000	19.310
DANets	0.839$_{\pm 0.021}$	0.752$_{\pm 0.023}$	0.601$_{\pm 0.035}$	15.759	15.552	16.207
AutoInt	0.834$_{\pm 0.013}$	0.766$_{\pm 0.011}$	0.630$_{\pm 0.021}$	18.345	17.655	17.517
TabTransformer	0.828$_{\pm 0.009}$	0.760$_{\pm 0.008}$	0.605$_{\pm 0.015}$	18.172	17.586	17.103
TabNet	0.809$_{\pm 0.028}$	0.749$_{\pm 0.020}$	0.591$_{\pm 0.049}$	22.586	21.069	21.517
SNN	0.802$_{\pm 0.028}$	0.742$_{\pm 0.015}$	0.553$_{\pm 0.030}$	21.793	20.034	20.759
Node	0.797$_{\pm 0.020}$	0.712$_{\pm 0.010}$	0.431$_{\pm 0.036}$	22.276	23.138	23.897

Table 10: Statistical profile of the benchmark PFN-CLS, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively.

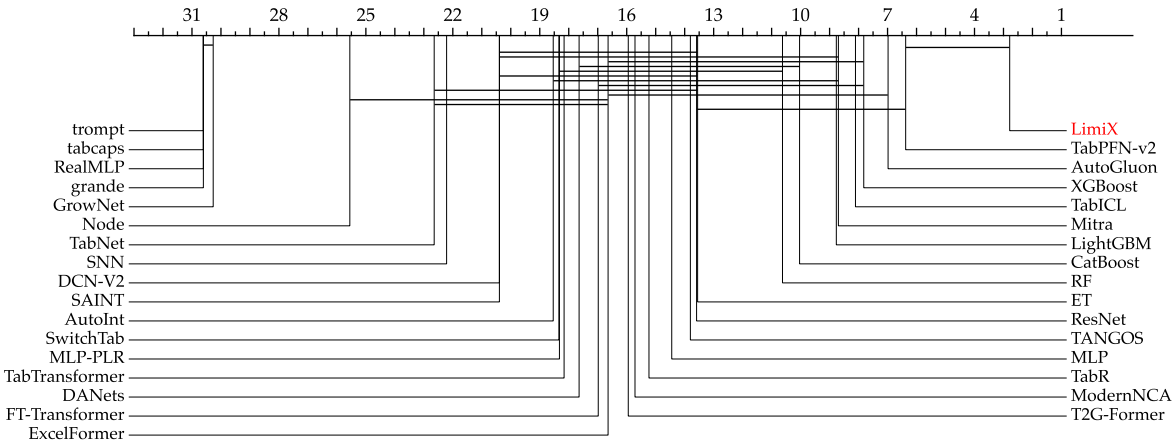
	Statistics						
Metric	Q10	Q50	Q90	Mean	Std	Min	Max
# Features	6	21	187	584	80	4	308
# Classes	2	2	7	4	2	2	10
Missing Values (Ratio)	0	0	0	0.001	0.006	0	0.032
Categorical Features (Ratio)	0	0.081	0.956	0.291	0.381	0	1
Features w/ Missing Values (Ratio)	0	0	0	0.016	0.086	0	0.474



(a) AUC on the PFN-CLS benchmark



(b) Accuracy on the PFN-CLS benchmark



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

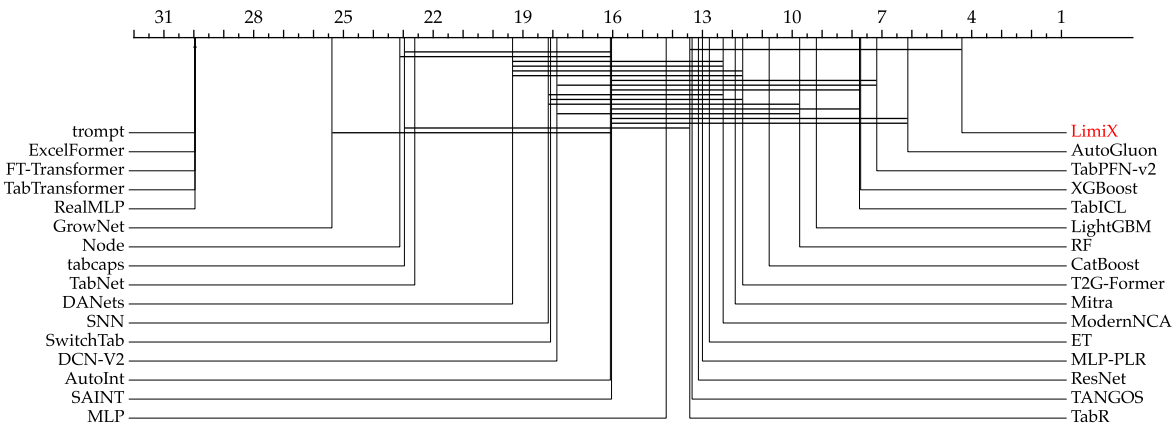
Figure 10: Critical difference diagrams on PFN-CLS benchmark

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

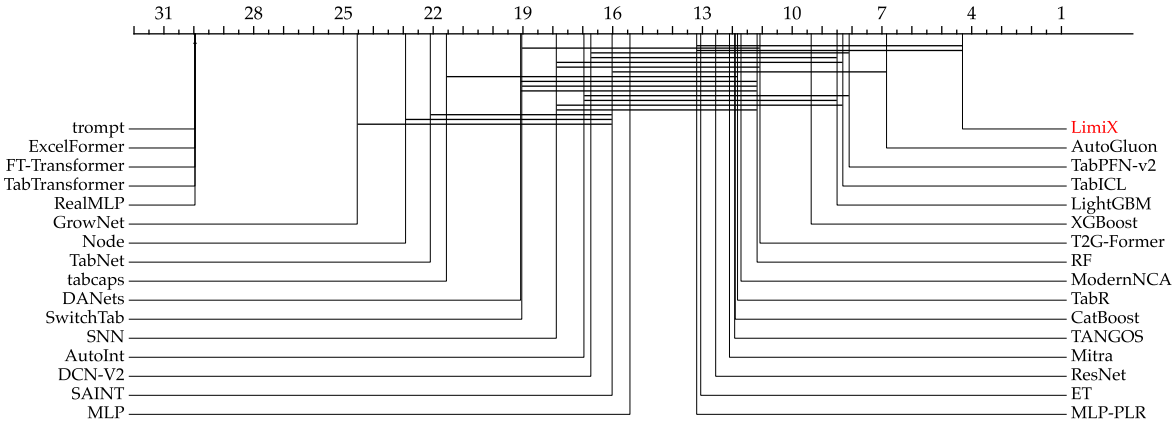
	TabZilla					
	Mean			Rank		
Model	AUC (↑)	Acc. (↑)	F1 (↑)	AUC (↓)	Acc. (↓)	F1 (↓)
LimiX	**0.945**$_{\pm 0.004}$	**0.885**$_{\pm 0.004}$	**0.836**$_{\pm 0.008}$	**3.962**	**4.231**	**4.769**
AutoGluon	0.936$_{\pm 0.001}$	0.871$_{\pm 0.002}$	0.803$_{\pm 0.008}$	6.154	5.249	7.308
TabPFN-v2	0.932 $_{\pm 0.004}$	0.832$_{\pm 0.009}$	0.770$_{\pm 0.011}$	7.538	6.808	7.500
TabICL	0.935$_{\pm 0.006}$	0.864$_{\pm 0.016}$	0.803$_{\pm 0.023}$	7.808	7.385	8.385
XGBoost	0.929$_{\pm 0.003}$	0.863$_{\pm 0.005}$	0.789$_{\pm 0.011}$	9.038	7.692	9.846
LightGBM	0.927$_{\pm 0.003}$	0.863$_{\pm 0.007}$	0.796$_{\pm 0.010}$	7.885	8.923	8.731
RF	0.924$_{\pm 0.004}$	0.852$_{\pm 0.006}$	0.773$_{\pm 0.012}$	9.148	10.37	10.481
CatBoost	0.922$_{\pm 0.006}$	0.848$_{\pm 0.011}$	0.780$_{\pm 0.020}$	10.481	11.259	11.296
Mitra	0.919$_{\pm 0.000}$	0.847$_{\pm 0.000}$	0.774$_{\pm 0.000}$	11.407	11.000	11.333
ET	0.912$_{\pm 0.004}$	0.837$_{\pm 0.008}$	0.745$_{\pm 0.014}$	12.630	12.630	13.778
T2G-Former	0.911$_{\pm 0.003}$	0.854$_{\pm 0.006}$	0.798$_{\pm 0.007}$	11.704	10.593	10.370
ModernNCA	0.908$_{\pm 0.003}$	0.851$_{\pm 0.009}$	0.795$_{\pm 0.009}$	11.630	11.148	10.963
TANGOS	0.908$_{\pm 0.005}$	0.837$_{\pm 0.011}$	0.776$_{\pm 0.021}$	13.074	11.704	11.889
ResNet	0.908$_{\pm 0.006}$	0.835$_{\pm 0.013}$	0.777$_{\pm 0.018}$	13.148	12.556	12.370
MLP	0.905$_{\pm 0.008}$	0.826$_{\pm 0.012}$	0.751$_{\pm 0.025}$	14.222	15.333	15.852
MLP-PLR	0.904$_{\pm 0.009}$	0.851$_{\pm 0.011}$	0.789$_{\pm 0.014}$	12.889	12.926	12.037
TabR	0.904$_{\pm 0.005}$	0.842$_{\pm 0.007}$	0.772$_{\pm 0.016}$	13.037	11.630	10.778
DCN-V2	0.888$_{\pm 0.019}$	0.825$_{\pm 0.020}$	0.740$_{\pm 0.043}$	17.704	16.370	17.333
DANets	0.883$_{\pm 0.028}$	0.800$_{\pm 0.035}$	0.705$_{\pm 0.070}$	19.185	18.704	19.481
AutoInt	0.879$_{\pm 0.012}$	0.812$_{\pm 0.012}$	0.699$_{\pm 0.025}$	15.889	16.593	16.704
SNN	0.870$_{\pm 0.008}$	0.800$_{\pm 0.012}$	0.674$_{\pm 0.020}$	18.185	17.444	17.667
SwitchTab	0.860$_{\pm 0.019}$	0.755$_{\pm 0.019}$	0.645$_{\pm 0.028}$	17.852	19.111	18.852
tabcaps	0.842$_{\pm 0.000}$	0.774$_{\pm 0.000}$	0.657$_{\pm 0.000}$	22.963	21.444	21.370
Node	0.835$_{\pm 0.029}$	0.730$_{\pm 0.038}$	0.539$_{\pm 0.082}$	23.148	22.778	23.889
TabNet	0.832$_{\pm 0.039}$	0.745$_{\pm 0.033}$	0.626$_{\pm 0.051}$	22.704	22.037	21.259
SAINT	0.825$_{\pm 0.006}$	0.761$_{\pm 0.009}$	0.670$_{\pm 0.019}$	15.963	15.815	15.259
GrowNet	0.661$_{\pm 0.042}$	0.577$_{\pm 0.029}$	0.419$_{\pm 0.063}$	25.074	24.000	24.000

Table 12: Statistical profile of the benchmark TabZilla, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively.

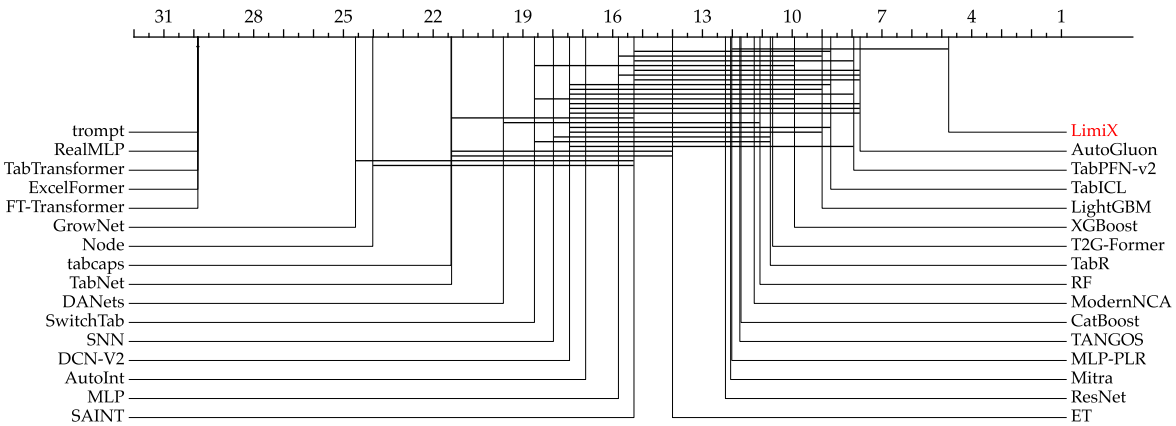
	Statistics						
Metric	Q10	Q50	Q90	Mean	Std	Min	Max
# Features	6	24	132	226	756	4	4296
# Classes	2	2	10	7	16	2	100
Missing Values (Ratio)	0	0	0.078	0.022	0.059	0	0.205
Categorical Features (Ratio)	0	0.528	1	0.474	0.39	0	1
Features w/ Missing Values (Ratio)	0	0	0.483	0.102	0.218	0	0.808



(a) AUC on the TabZilla benchmark.

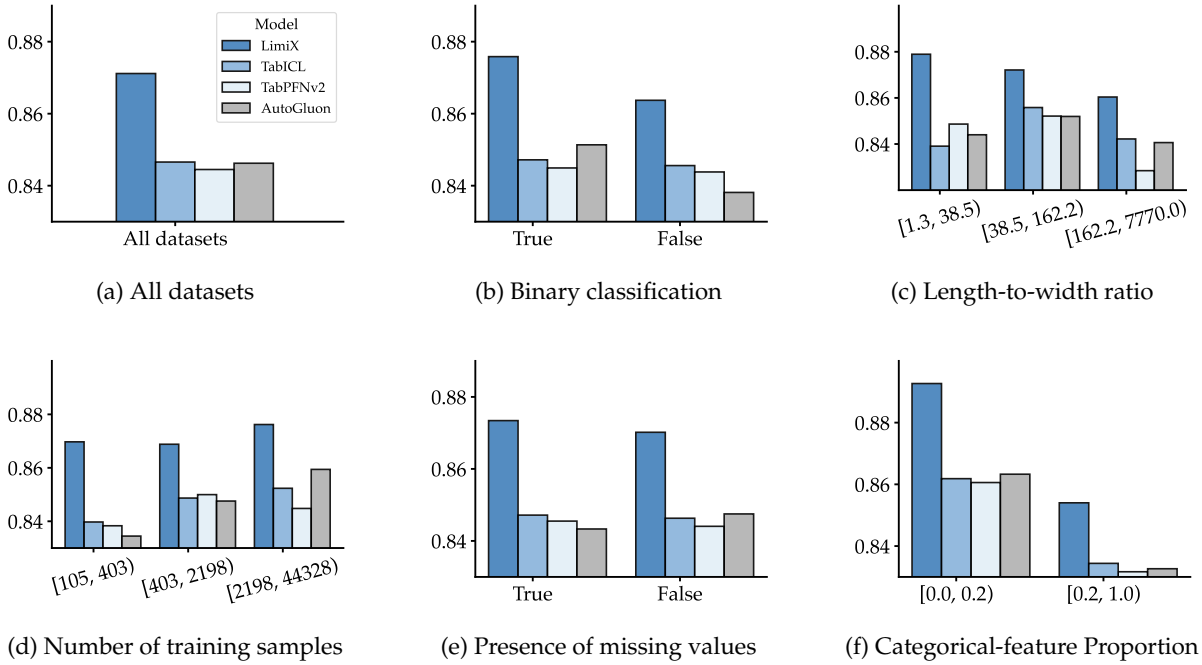


(b) Accuracy on the TabZilla benchmark.



(c) F1-score on the TabZilla benchmark.

Figure 11: Critical difference diagrams on the TabZilla benchmark.



(a) All datasets (b) Binary classification (c) Length-to-width ratio

(d) Number of training samples (e) Presence of missing values (f) Categorical-feature Proportion

Figure 12: AUC on subset with various sample size, feature dimensionality, number of classes, categorical–to-numerical feature ratios, missing values, and sample-to-feature ratios.

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), and CTR23 (Fischer et al., 2023). 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. However, TabICL is excluded due to its incapability of regression. Additionally, 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 ranks of models with respect to both metrics.

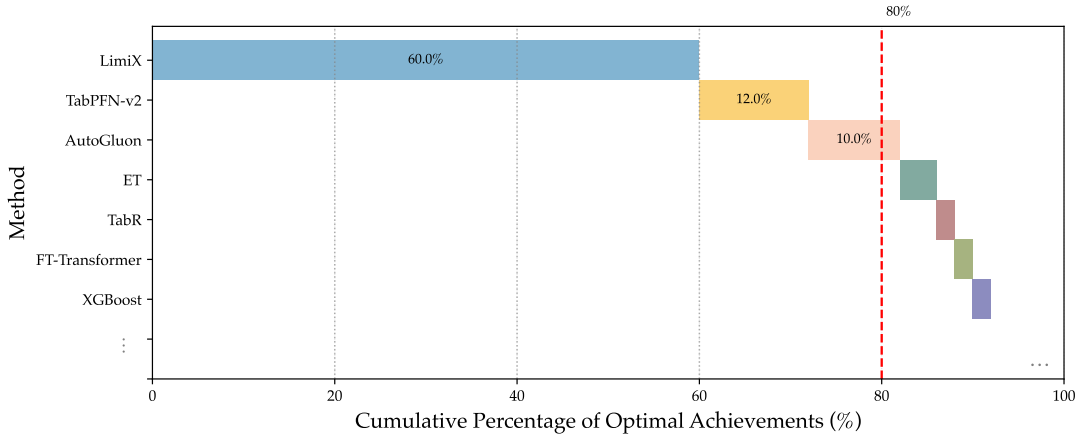
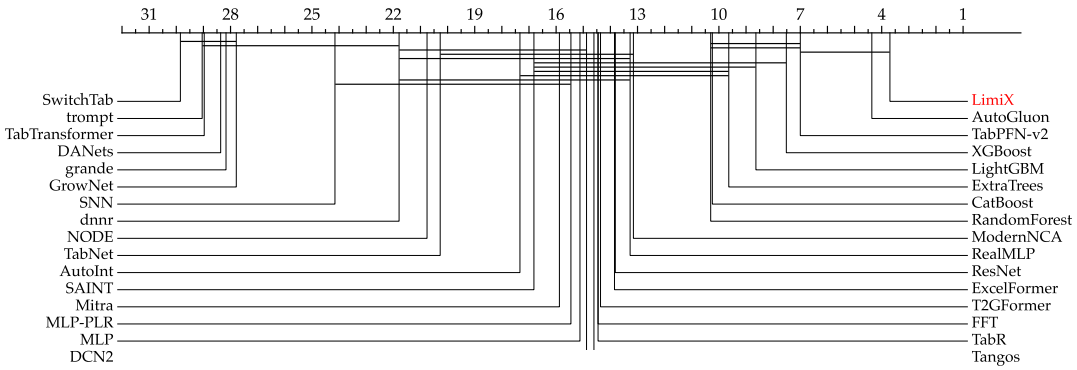
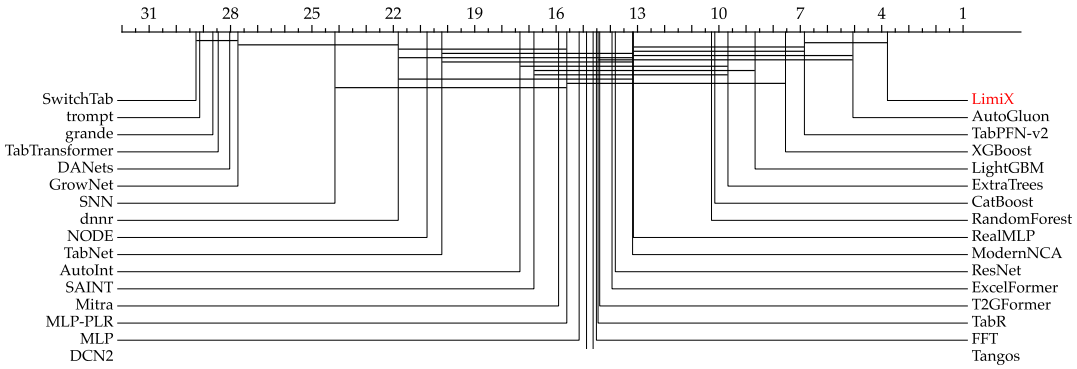


Figure 13: The proportion of models achieving the best R^2. The length of each bar represents the proportion of the 106 datasets in which a given method achieved the highest R^2 on the BCCO-REG benchmark.

Results. As shown by critical difference diagrams, i.e. Figures 14 to 17, LimiX always achieves the best performance, outperforming strong baselines like AutoGluon, XGBoost, and TabPFN-v2. Quantitative results in Tables 13, 15, 17 and 19 also confirms that LimiX achieves state-of-the-art regression performance, leading in both normalized RMSE and R^2.

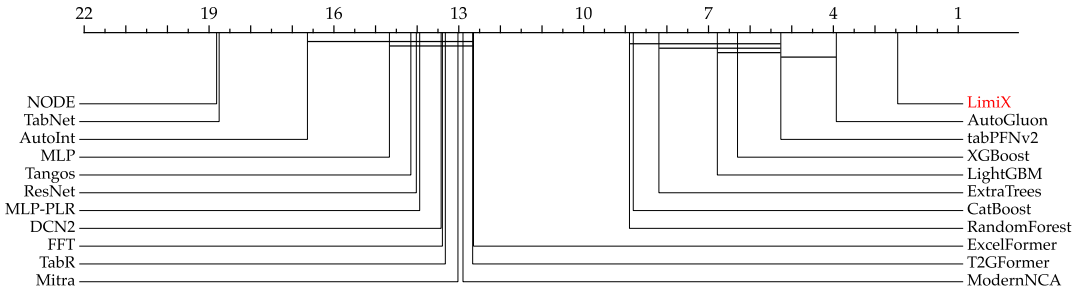


(a) R^2 on the CTR23 benchmark

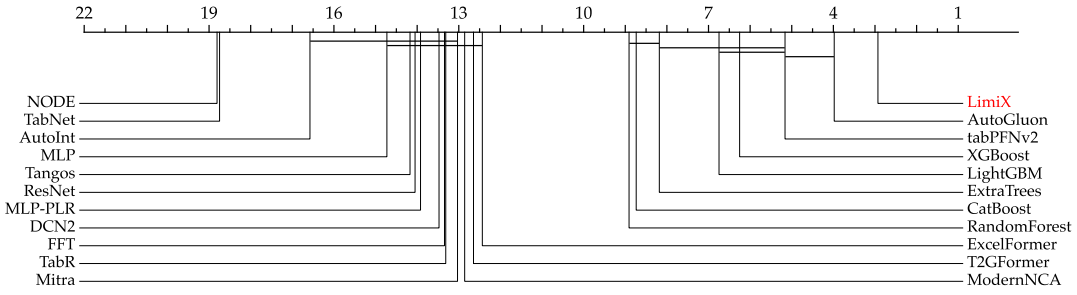


(b) RMSE on the CTR23 benchmark

Figure 14: Critical difference diagram on the CTR23 benchmark

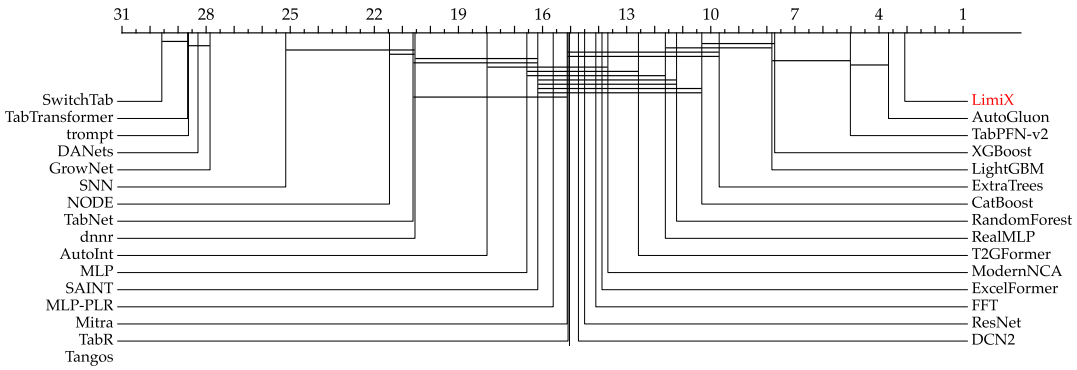


(a) R^2 on the TALENT-REG benchmark

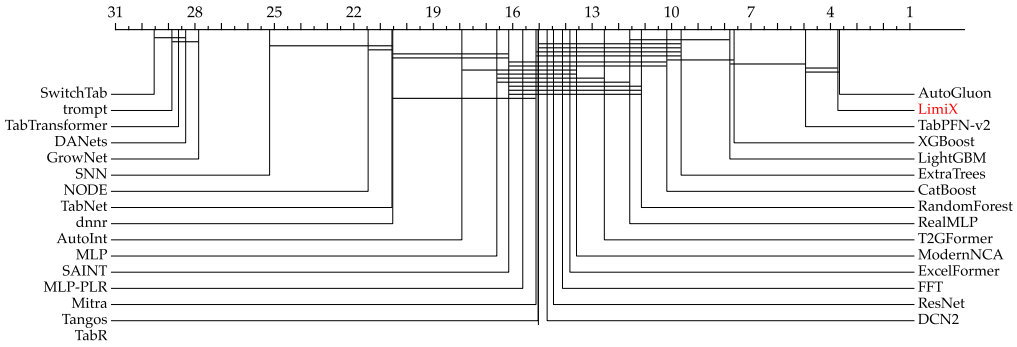


(b) RMSE on TALENT-REG benchmark

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

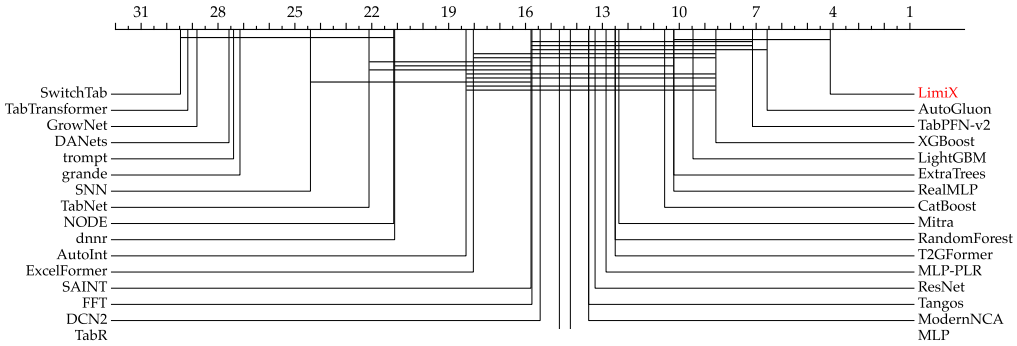


(a) R^2 on the BCCO-REG benchmark

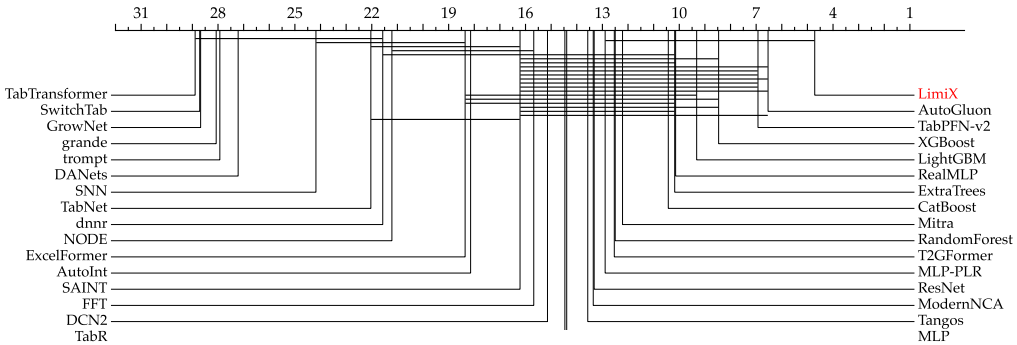


(b) RMSE on the BCCO-REG benchmark

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



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



(b) RMSE on PFN-REG benchmark

Figure 17: Critical difference diagram on the PFN-REG benchmark

Table 13: Regression results on the CTR23 benchmark. The best scores are shown in bold.

	CTR23			
	Mean		Rank	
Model	R^2 (↑)	RMSE (↓)	R^2 (↓)	RMSE (↓)
LimiX	**0.745**$_{\pm 0.002}$	**0.477**$_{\pm 0.002}$	**3.606**	**3.697**
AutoGluon	0.726$_{\pm 0.001}$	0.512$_{\pm 0.001}$	4.212	3.939
TabPFN-v2	0.716$_{\pm 0.002}$	0.503$_{\pm 0.002}$	6.788	6.636
XGBoost	0.712$_{\pm 0.008}$	0.511$_{\pm 0.007}$	7.333	7.364
LightGBM	0.706$_{\pm 0.004}$	0.516$_{\pm 0.005}$	8.455	8.485
CatBoost	0.700$_{\pm 0.022}$	0.528$_{\pm 0.030}$	10.000	9.909
ET	0.697$_{\pm 0.007}$	0.535$_{\pm 0.008}$	9.424	9.455
RF	0.694$_{\pm 0.010}$	0.539$_{\pm 0.012}$	10.091	10.061
ModernNCA	0.672$_{\pm 0.009}$	0.543$_{\pm 0.008}$	12.667	12.697
T2G-Former	0.667$_{\pm 0.000}$	0.551$_{\pm 0.000}$	13.788	13.818
TabR	0.665$_{\pm 0.000}$	0.549$_{\pm 0.000}$	13.939	13.939
DCN2	0.663$_{\pm 0.004}$	0.553$_{\pm 0.003}$	14.333	14.333
FT-Transformer	0.663$_{\pm 0.000}$	0.554$_{\pm 0.000}$	13.939	14.000
RealMLP	0.661$_{\pm 0.018}$	0.546$_{\pm 0.010}$	12.848	12.758
AutoInt	0.649$_{\pm 0.007}$	0.581$_{\pm 0.009}$	16.636	16.636
ResNet	0.649$_{\pm 0.007}$	0.583$_{\pm 0.007}$	13.333	13.364
TANGOS	0.648$_{\pm 0.008}$	0.580$_{\pm 0.007}$	14.121	14.152
ExcelFormer	0.647$_{\pm 0.000}$	0.574$_{\pm 0.000}$	13.273	13.333
SAINT	0.644$_{\pm 0.000}$	0.570$_{\pm 0.000}$	16.091	16.121
Mitra	0.631$_{\pm 0.018}$	0.581$_{\pm 0.009}$	15.273	15.364
MLP	0.607$_{\pm 0.004}$	0.625$_{\pm 0.004}$	14.636	14.667
TabNet	0.551$_{\pm 0.061}$	0.642$_{\pm 0.032}$	19.545	19.515
Node	0.494$_{\pm 0.000}$	0.734$_{\pm 0.000}$	20.030	20.030
dnnr	-0.454$_{\pm 0.417}$	0.853$_{\pm 0.072}$	20.848	20.939
SNN	0.356$_{\pm 0.012}$	0.840$_{\pm 0.008}$	23.273	23.273
GrowNet	0.061$_{\pm 0.017}$	1.021$_{\pm 0.008}$	26.515	26.576
SwitchTab	-0.018$_{\pm 0.018}$	1.062$_{\pm 0.010}$	28.091	27.970
TabTransformer	-0.014$_{\pm 0.000}$	1.061$_{\pm 0.000}$	27.364	27.273
Trompt	-0.010$_{\pm 0.000}$	1.059$_{\pm 0.000}$	27.606	27.788
DANets	-0.006$_{\pm 0.003}$	1.056$_{\pm 0.001}$	26.909	26.879

Table 14: Statistical profile of the benchmark CTR23, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively.

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

	TALENT-REG			
	Mean		Rank	
Model	R^2 (↑)	RMSE (↓)	R^2 (↓)	RMSE (↓)
LimiX	**0.784**$_{\pm 0.002}$	**0.394**$_{\pm 0.002}$	**2.714**	**2.833**
AutoGluon	0.770$_{\pm 0.001}$	0.410$_{\pm 0.001}$	3.643	3.690
TabPFN-v2	0.759$_{\pm 0.001}$	0.417$_{\pm 0.002}$	5.619	5.548
XGBoost	0.750$_{\pm 0.007}$	0.428$_{\pm 0.005}$	7.476	7.405
LightGBM	0.691$_{\pm 0.022}$	0.438$_{\pm 0.005}$	7.571	7.619
CatBoost	0.723$_{\pm 0.030}$	0.443$_{\pm 0.023}$	10.405	10.286
ET	0.742$_{\pm 0.006}$	0.446$_{\pm 0.007}$	10.190	10.143
RF	0.717$_{\pm 0.015}$	0.454$_{\pm 0.008}$	11.262	11.214
RealMLP	0.737$_{\pm 0.009}$	0.454$_{\pm 0.011}$	11.810	11.786
T2G-Former	0.725$_{\pm 0.007}$	0.459$_{\pm 0.006}$	12.571	12.548
DCN-V2	0.722$_{\pm 0.006}$	0.465$_{\pm 0.005}$	14.262	14.262
FT-Transformer	0.720$_{\pm 0.009}$	0.465$_{\pm 0.007}$	14.048	14.071
TabR	0.720$_{\pm 0.013}$	0.465$_{\pm 0.010}$	14.833	14.786
ModernNCA	0.697$_{\pm 0.041}$	0.466$_{\pm 0.010}$	13.952	13.833
MLP-PLR	0.718$_{\pm 0.005}$	0.469$_{\pm 0.005}$	15.286	15.286
ExcelFormer	0.703$_{\pm 0.021}$	0.481$_{\pm 0.020}$	13.952	13.952
AutoInt	0.705$_{\pm 0.010}$	0.485$_{\pm 0.008}$	17.429	17.381
SAINT	0.687$_{\pm 0.000}$	0.489$_{\pm 0.000}$	14.929	14.857
ResNet	0.700$_{\pm 0.007}$	0.489$_{\pm 0.006}$	14.571	14.548
Tangos	0.696$_{\pm 0.006}$	0.492$_{\pm 0.007}$	14.690	14.714
Mitra	0.620$_{\pm 0.000}$	0.499$_{\pm 0.000}$	15.286	15.286
MLP	0.683$_{\pm 0.008}$	0.507$_{\pm 0.007}$	16.548	16.619
TabNet	0.630$_{\pm 0.034}$	0.550$_{\pm 0.026}$	21.119	21.071
Node	0.543$_{\pm 0.003}$	0.640$_{\pm 0.002}$	22.786	22.833
SNN	0.377$_{\pm 0.022}$	0.755$_{\pm 0.014}$	25.048	25.024
dnnr	-0.429$_{\pm 0.195}$	0.831$_{\pm 0.038}$	21.000	21.000
GrowNet	0.053$_{\pm 0.026}$	0.951$_{\pm 0.012}$	27.738	27.786
DANets	-0.002$_{\pm 0.002}$	0.979$_{\pm 0.001}$	28.310	28.405
Trompt	-0.005$_{\pm 0.000}$	0.981$_{\pm 0.000}$	28.548	28.833
TabTransformer	-0.009$_{\pm 0.009}$	0.982$_{\pm 0.005}$	28.571	28.571
SwitchTab	-0.028$_{\pm 0.035}$	0.990$_{\pm 0.017}$	29.810	29.786

Table 16: Statistical profile of the benchmark TALENT-REG, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively.

	Statistics						
Metric	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 BCCO-REG benchmark. The best scores are shown in bold.

	BCCO-REG			
	Mean		Rank	
Model	R^2 (↑)	RMSE (↓)	R^2 (↓)	RMSE (↓)
LimiX	**0.784**$_{\pm 0.002}$	**0.394**$_{\pm 0.001}$	**2.714**	**2.833**
AutoGluon	0.770$_{\pm 0.001}$	0.410$_{\pm 0.001}$	3.643	3.690
TabPFN-v2	0.759$_{\pm 0.001}$	0.417$_{\pm 0.002}$	5.619	5.548
XGBoost	0.750$_{\pm 0.007}$	0.428$_{\pm 0.005}$	7.476	7.405
LightGBM	0.691$_{\pm 0.022}$	0.438$_{\pm 0.005}$	7.571	7.619
CatBoost	0.723$_{\pm 0.030}$	0.443$_{\pm 0.023}$	10.405	10.286
ExtraTrees	0.742$_{\pm 0.006}$	0.446$_{\pm 0.007}$	10.190	10.143
RandomForest	0.717$_{\pm 0.015}$	0.454$_{\pm 0.008}$	11.262	11.214
RealMLP	0.737$_{\pm 0.009}$	0.454$_{\pm 0.011}$	11.810	11.786
T2GFormer	0.725$_{\pm 0.007}$	0.459$_{\pm 0.006}$	12.571	12.548
DCN2	0.722$_{\pm 0.006}$	0.465$_{\pm 0.005}$	14.262	14.262
FFT	0.720$_{\pm 0.009}$	0.465$_{\pm 0.007}$	14.048	14.071
TabR	0.720$_{\pm 0.013}$	0.465$_{\pm 0.010}$	14.833	14.786
ModernNCA	0.697$_{\pm 0.041}$	0.466$_{\pm 0.010}$	13.952	13.833
MLP-PLR	0.718$_{\pm 0.005}$	0.469$_{\pm 0.005}$	15.286	15.286
ExcelFormer	0.703$_{\pm 0.021}$	0.481$_{\pm 0.020}$	13.952	13.952
AutoInt	0.705$_{\pm 0.010}$	0.485$_{\pm 0.008}$	17.429	17.381
SAINT	0.687$_{\pm 0.000}$	0.489$_{\pm 0.000}$	14.929	14.857
ResNet	0.700$_{\pm 0.007}$	0.489$_{\pm 0.006}$	14.571	14.548
Tangos	0.696$_{\pm 0.006}$	0.492$_{\pm 0.007}$	14.690	14.714
Mitra	0.620$_{\pm 0.000}$	0.499$_{\pm 0.000}$	15.286	15.286
MLP	0.683$_{\pm 0.008}$	0.507$_{\pm 0.007}$	16.548	16.619
TabNet	0.630$_{\pm 0.034}$	0.550$_{\pm 0.026}$	21.119	21.071
NODE	0.543$_{\pm 0.003}$	0.640$_{\pm 0.002}$	22.786	22.833
SNN	0.377$_{\pm 0.022}$	0.755$_{\pm 0.014}$	25.048	25.024
dnnr	-0.429$_{\pm 0.195}$	0.831$_{\pm 0.038}$	21.000	21.000
GrowNet	0.053$_{\pm 0.026}$	0.951$_{\pm 0.012}$	27.738	27.786
DANets	-0.002$_{\pm 0.002}$	0.979$_{\pm 0.001}$	28.310	28.405
Trompt	-0.005$_{\pm 0.000}$	0.981$_{\pm 0.000}$	28.548	28.833
TabTransformer	-0.009$_{\pm 0.009}$	0.982$_{\pm 0.005}$	28.571	28.571
SwitchTab	-0.028$_{\pm 0.035}$	0.990$_{\pm 0.017}$	29.810	29.786

Table 18: Statistical profile of the benchmark BCCO-REG, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively.

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

	PFN-REG			
	Mean		Rank	
Model	R^2 (↑)	RMSE (↓)	R^2 (↓)	RMSE (↓)
LimiX	**0.692**$_{\pm 0.001}$	**0.461**$_{\pm 0.001}$	**3.714**	**4.143**
AutoGluon	0.677$_{\pm 0.000}$	0.482$_{\pm 0.001}$	5.786	5.750
TabPFNv2	0.676$_{\pm 0.004}$	0.475$_{\pm 0.002}$	6.286	6.071
XGBoost	0.671$_{\pm 0.007}$	0.487$_{\pm 0.007}$	7.536	7.429
LightGBM	0.664$_{\pm 0.008}$	0.496$_{\pm 0.008}$	8.393	8.250
ET	0.652$_{\pm 0.009}$	0.515$_{\pm 0.009}$	9.036	9.036
CatBoost	0.661$_{\pm 0.014}$	0.498$_{\pm 0.014}$	9.464	9.357
Mitra	0.630$_{\pm 0.000}$	0.531$_{\pm 0.000}$	10.857	10.857
RF	0.643$_{\pm 0.008}$	0.524$_{\pm 0.010}$	11.214	11.250
T2G-Former	0.642$_{\pm 0.017}$	0.504$_{\pm 0.012}$	11.286	11.321
MLP-PLR	0.640$_{\pm 0.007}$	0.507$_{\pm 0.006}$	11.536	11.571
ModernNCA	0.646$_{\pm 0.012}$	0.509$_{\pm 0.010}$	11.929	11.893
ResNet	0.596$_{\pm 0.011}$	0.558$_{\pm 0.011}$	12.036	12.071
TANGOS	0.594$_{\pm 0.010}$	0.558$_{\pm 0.011}$	12.143	12.179
TabR	0.633$_{\pm 0.017}$	0.515$_{\pm 0.013}$	12.786	12.714
MLP	0.586$_{\pm 0.013}$	0.565$_{\pm 0.010}$	12.821	12.964
DCN-V2	0.628$_{\pm 0.012}$	0.522$_{\pm 0.009}$	13.500	13.393
FT-Transformer	0.610$_{\pm 0.011}$	0.527$_{\pm 0.009}$	13.679	13.679
ExcelFormer	0.558$_{\pm 0.017}$	0.578$_{\pm 0.011}$	15.286	15.357
AutoInt	0.599$_{\pm 0.021}$	0.552$_{\pm 0.017}$	15.929	15.893
TabNet	0.501$_{\pm 0.049}$	0.645$_{\pm 0.040}$	19.143	19.179

Table 20: Statistical profile of the benchmark PFN-REG, where Q10, Q50, and Q90 correspond to the 10%, 50%, and 90% quantiles, respectively.

	Statistics						
Metric	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

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 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 is the first model to be capable of 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), Early Stage Diabetes 2020 (Islam et al., 2019), Forty Soybean Cultivars from Subsequent Harvests (de Oliveira et al., 2023), HCC survival (Santos et al., 2015), Vehicle (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), KNN, MICE (Van Buuren & Groothuis-Oudshoorn, 2011), MissForest (Stekhoven & Bühlmann, 2012), GAIN (Yoon et al., 2018), MIWAE (Stekhoven & Bühlmann, 2012), 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 21 shows that LimiX LimiX consistently outperforms previous baselines, all of which require additional training or fitting.

Table 21: 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 (↓)					Classification Error (↓)	
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	**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 18, 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 19, 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 sharply increases when the outlier factor grows from 100 to 10000, while LimiX do not. This demonstrates the superior robustness of LimiX compared with TabPFN-v2 in regression tasks. Overall, LimiX exhibits superior robustness relative to the baselines.

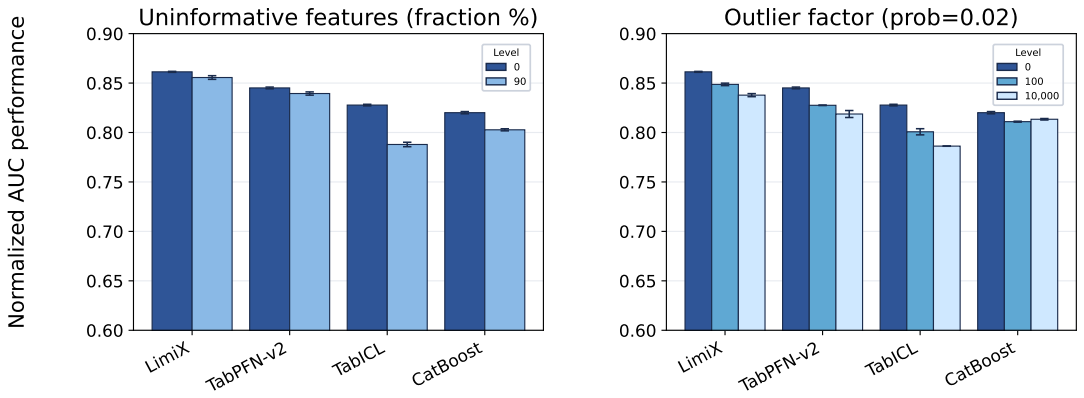


Figure 18: Robustness analysis in classification Tasks. LimiX consistently exhibits the best performance and superior robustness under perturbations.

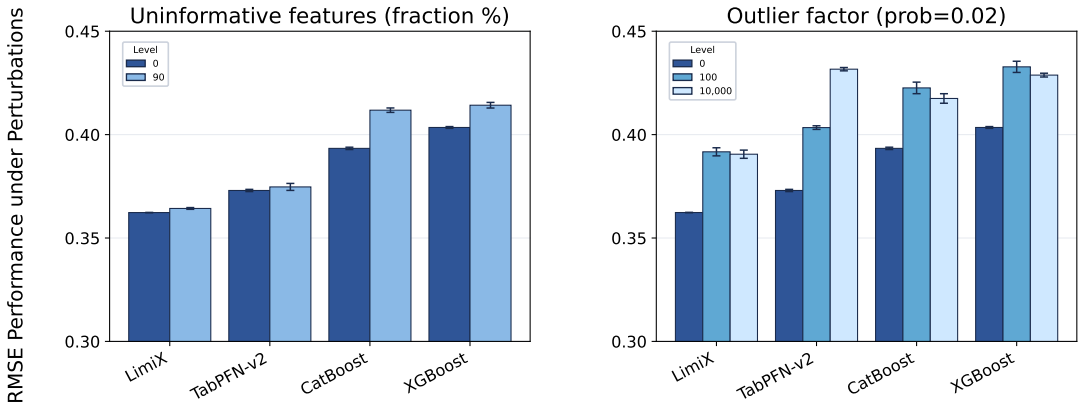


Figure 19: Robustness analysis in regression tasks. LimiX consistently exhibits the best performance and superior robustness under perturbations.

7.5 Embedding

Pretrained models (Devlin et al., 2019; Oquab et al., 2023) 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, we conduct experiments on six datasets of various sample sizes and numbers of categories. We compare embeddings of LimiX with those of MLP, ResNet (He et al., 2016), ModernNCA (Ye et al., 2024), 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 20 shows t-SNE visualization of embeddings. We can see that embeddings of different categories extracted by LimiX are more separated than those extracted by other models.

To further evaluate the quality of embeddings extracted by LimiX, 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 22 shows that embeddings of LimiX achieve the highest average AUC and ranks among three ICL-based models. Overall, LimiX consistently outperforms baselines in both qualitative and quantitative experiments.

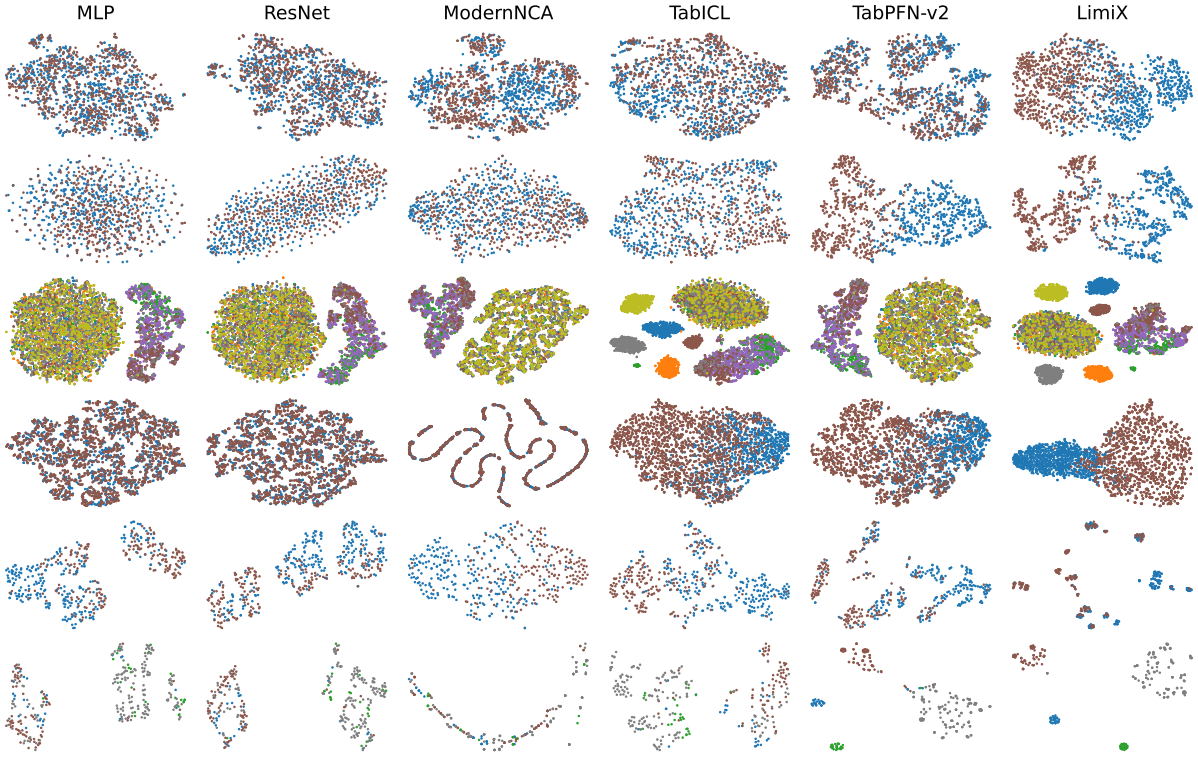


Figure 20: t-SNE visualization of embeddings. Different colors represent different categories. Embeddings extracted by LimiX are better separated between categories than other models.

Table 22: Results of linear probing on BCCO-CLS. LimiX outperforms other two ICL-based models, indicating the strongest embedding capability.

Model	AUC(↑)	Rank(↓)
TabPFN-v2	0.832	2.189
TabICL	0.838	1.981
LimiX	**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 k-nearest-neighbor selection (Xu et al., 2024; Thomas et al., 2024). 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 with other proprietary and general models that are trained or fine-tuned on real datasets. As shown in Tables 23 and 24, LimiX significantly outperforms the others across various metrics in both classification and regression tasks after fine-tuning.

Figure 21 shows the points where ensemble and finetuning achieved the most significant improvement in AUC metric on the BCCO-CLS benchmark for LimiX.

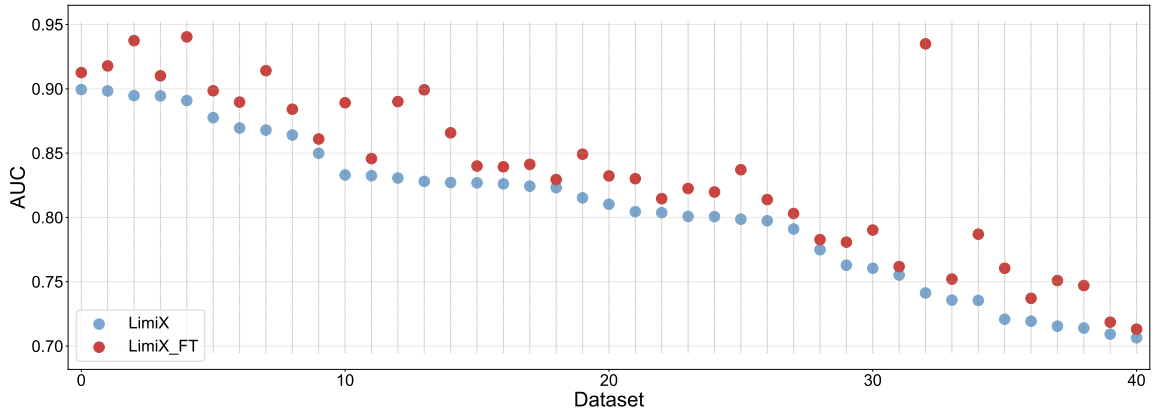


Figure 21: The improvement of fine-tuning in AUC on the BCCO-CLS benchmark.

Table 23: Performance comparison of fine-tuned models and models supervised trained with real datasets on classification benchmarks.

Benchmark	Method	AUC	Acc.	F1	AUC-rank	Acc.-rank	f1-rank
BCCO	LimiX	0.871	0.804	0.731	2.689	2.575	3.047
	LimiX-FT	**0.873**	**0.806**	0.733	**1.528**	**1.415**	**1.877**
	TabPFN-v2	0.844	0.771	0.679	5.896	5.925	6.160
	TabPFN-v2-FT[2]	0.850	0.775	**0.760**	4.792	5.283	4.349
	TabDPT-e1	0.841	0.774	0.685	5.330	5.019	5.340
	TabDPT-e8	0.846	0.777	0.687	4.651	4.632	5.113
	XGboost	0.834	0.762	0.674	6.085	6.028	6.000
	Catboost	0.829	0.757	0.664	6.519	6.387	6.321
	TabR	0.804	0.744	0.652	7.425	6.915	6.519
TALENT-CLS	LimiX	0.903	0.861	0.752	3.274	2.955	3.469
	LimiX-FT	**0.904**	**0.863**	0.755	**1.894**	**1.715**	**2.263**
	TabPFN-v2	0.892	0.844	0.720	5.330	5.514	5.659
	TabPFN-v2-FT	0.898	0.853	**0.844**	4.140	4.402	3.050
	TabDPT-e1	0.891	0.846	0.719	5.145	5.056	5.397
	TabDPT-e8	0.893	0.849	0.723	4.402	4.380	4.844
	XGBoost	0.881	0.837	0.713	5.944	5.933	5.894
	CatBoost	0.876	0.828	0.704	6.682	6.665	6.570
	TabR	0.856	0.822	0.679	7.654	7.425	7.056
OpenML-CC18	LimiX	0.939	0.893	0.811	3.048	3.758	3.645
	LimiX-FT	**0.941**	**0.894**	**0.893**	**1.516**	**2.387**	**2.500**
	TabPFN-v2	0.927	0.879	0.788	5.968	5.387	5.532
	TabPFN-v2-FT	0.933	0.889	0.883	4.226	4.000	3.145
	TabDPT-e1	0.927	0.881	0.786	4.677	4.871	5.032
	TabDPT-e8	0.930	0.885	0.799	3.952	3.984	4.306
	XGBoost	0.902	0.859	0.755	5.371	5.468	5.742
	CatBoost	0.899	0.850	0.750	6.823	6.323	6.581
	TabR	0.842	0.808	0.689	6.919	6.323	6.645
TabArena	LimiX	0.849	0.877	0.597	3.273	3.030	4.394
	LimiX-FT	**0.851**	**0.878**	0.800	**2.091**	**1.697**	3.212
	TabPFN-v2	0.836	0.870	0.582	5.606	5.455	5.545
	TabPFN-v2-FT	0.852	0.874	**0.860**	3.667	4.030	**2.273**
	TabDPT-e1	0.839	0.869	0.576	5.485	5.394	5.364
	TabDPT-e8	0.839	0.870	0.580	5.030	4.636	4.818
	XGBoost	0.838	0.867	0.567	5.333	6.030	5.970
	CatBoost	0.835	0.867	0.574	6.152	5.455	5.273
	TabR	0.769	0.838	0.474	8.333	8.182	7.061
TabZilla	LimiX	0.943	0.885	0.836	3.704	3.519	4.111
	LimiX-FT	**0.944**	**0.887**	0.838	**2.111**	**2.481**	**2.556**
	TabPFN-v2	0.925	0.855	0.790	5.889	5.889	5.963
	TabPFN-v2-FT	0.932	0.873	**0.868**	4.296	4.704	3.889
	TabDPT-e1	0.932	0.880	0.824	4.556	4.185	4.259
	TabDPT-e8	0.934	0.882	0.824	4.111	4.000	4.407
	XGBoost	0.929	0.863	0.789	5.222	5.333	5.889
	CatBoost	0.922	0.848	0.780	6.815	6.370	6.556
	TabR	0.904	0.851	0.789	6.667	6.222	5.926
PFN-CLS	LimiX	0.923	0.862	0.786	2.483	2.793	3.103
	LimiX-FT	**0.924**	**0.864**	0.818	**1.448**	**1.621**	**2.034**
	TabPFN-v2	0.910	0.841	0.750	5.000	5.379	5.448
	TabPFN-v2-FT	0.912	0.847	**0.840**	3.897	4.276	3.414
	TabDPT-e1	0.891	0.822	0.727	5.310	6.000	6.310
	TabDPT-e8	0.896	0.833	0.742	4.862	4.448	4.793
	XGBoost	0.898	0.831	0.733	6.276	5.379	5.414
	CatBoost	0.895	0.819	0.720	6.552	6.103	6.379
	TabR	0.849	0.782	0.671	7.966	7.759	7.414

Table 24: Performance comparison of fine-tuned models and models supervised trained with real datasets on regression benchmarks.

Benchmark	Method	R^2	RMSE	R^2-rank	RMSE-rank
BCCO-REG	LimiX	0.784	**0.394**	3.476	**2.952**
	LimiX-FT	**0.787**	**0.394**	**1.976**	3.667
	TabPFN-v2	0.713	0.432	5.548	5.381
	TabPFN-v2-FT	0.764	0.411	4.762	4.500
	TabDPT-e1	0.758	0.418	5.571	5.429
	TabDPT-e8	0.761	0.414	4.738	4.548
	XGBoost	0.750	0.428	6.762	6.714
	CatBoost	0.723	0.443	7.929	7.857
	TabR	0.720	0.465	8.548	8.548
TALENT-REG	LimiX	0.735	**0.433**	3.162	**2.848**
	LimiX-FT	**0.737**	**0.433**	**1.980**	3.465
	TabPFN-v2	0.700	0.459	5.899	5.697
	TabPFN-v2-FT	0.702	0.459	5.354	5.141
	TabDPT-e1	0.709	0.461	5.697	5.576
	TabDPT-e8	0.711	0.458	4.808	4.626
	XGBoost	0.710	0.462	6.323	6.182
	CatBoost	0.700	0.471	7.475	7.424
	TabR	0.642	0.526	8.758	8.717
PFN-REG	LimiX	0.692	**0.461**	4.393	**4.143**
	LimiX-FT	**0.695**	**0.461**	**2.786**	4.643
	TabPFN-v2	0.686	0.465	4.964	4.714
	TabPFN-v2-FT	0.687	0.466	4.750	4.429
	TabDPT-e1	0.672	0.491	6.286	5.964
	TabDPT-e8	0.675	0.484	5.429	5.321
	XGBoost	0.671	0.487	5.821	5.607
	CatBoost	0.661	0.498	7.393	7.250
	TabR	0.633	0.515	8.143	8.143
CTR23	LimiX	0.745	**0.477**	3.939	**3.242**
	LimiX-FT	**0.748**	**0.477**	**2.455**	3.909
	TabDPT-e1	0.728	0.500	5.606	5.515
	TabDPT-e8	0.731	0.498	4.970	4.818
	TabPFN-v2	0.719	0.501	5.576	5.394
	TabPFN-v2-FT	0.722	0.498	5.182	5.091
	XGBoost	0.712	0.511	6.606	6.576
	CatBoost	0.700	0.528	7.394	7.333
	TabR	0.665	0.549	8.091	8.091

[2]The fine-tuning method of TabPFN-v2 is based on the work https://github.com/PriorLabs/TabPFN.git

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 has 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 $\mathbf{x}^{ct}$) and generated data (treated as $\mathbf{x}^{\text{te}}$). We iterate this process until the last column is generated. Then for LimiX, 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), Vertebral Column (Barreto & Neto, 2005), Seeds (Gomes Mantovani, 2015), Wine (Hardik, 2021), and Grub Damage (Vanschoren, 2014). For evaluation, we use Trend and Shape to evaluate fidelity, which are proposed by SDMetrics[3]. We also calculate the AUC of XGBoost on a hold-out test dataset using generated data or real data. From Table 25, we find that LimiX 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 is even higher than using real data. The results show that LimiX is 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 25: 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 consistently outperforms TabPFN-v2 in all three metrics.

Metric	Method	Early Diabetes	Vertebra	Seeds	Wine	Grub Damage
Trend (↑)	TabPFN-v2	0.797	0.580	0.696	0.686	0.486
	LimiX	0.804	0.591	0.699	0.688	0.673
Shape (↑)	TabPFN-v2	0.889	0.754	0.739	0.622	0.635
	LimiX	0.902	0.763	0.768	0.646	0.762
AUC (↑)	TabPFN-v2	0.839	0.652	0.861	0.670	0.695
	LimiX	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 Appendix 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 26, 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 26: 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 (↑)	OOD_AUC (↑)	ID_rank (↓)	OOD_rank (↓)
LimiX	**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 Conclusion

In this report, we introduce LimiX, an LDM that treats 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 adopts a lightweight, scalable architecture that models causal relations among variables while jointly capturing dependencies across features and samples. Meanwhile, LimiX combines 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 is generated with hierarchical SCMs via graph-aware and solvability-aware sampling. Attention-guided retrieval of LimiX further supports efficient inference-time ensemble and fine-tuning if desired. Empirically, experiments across 10 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. As a single model, LimiX consistently surpasses competitive baselines on various downstream tasks.

9 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

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.

Mahmoud Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025. URL https://arxiv.org/abs/2506.09985.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. URL https://arxiv.org/abs/1607.06450.

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. *arXiv preprint arXiv:2106.15147*, 2021. URL https://arxiv.org/abs/2106.15147.

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, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024. URL https://arxiv.org/abs/2404.08471.

Guilherme Barreto and Ajalmar Neto. Vertebral column. UCI Machine Learning Repository, 2005. DOI: https://doi.org/10.24432/C5K89B.

Kenneth Benoit and colleagues. Ai and data science for public policy. *LSE Public Policy Review*, 4(2):1–6, 2024. doi: 10.31389/lseppr.115.

Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Pieter Gijsbers, Frank Hutter, Michel Lang, Rafael G Mantovani, Jan N van Rijn, and Joaquin Vanschoren. Openml benchmarking suites. *arXiv preprint arXiv:1708.03731*, 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.

Marios Chatiras. Spiking neural networks on tabular data for time series classification. Master's thesis, Aalto University, 2024.

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, 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*, 2023a.

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, 2023b.

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. ACM, 2016. doi: 10.1145/2939672.2939785.

Florinel-Alin Croitoru, Vlad Hondru, 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.

Xiang Fang, Yichen Wang, Jiayi Liu, Lihong Li, Wei Liu, and Ce Zhang. Large language models on tabular data: A survey. *arXiv preprint arXiv:2402.17944*, 2024. URL https://arxiv.org/abs/2402.17944.

Sebastian Felix Fischer, Matthias Feurer, and Bernd Bischl. Openml-ctr23–a curated tabular regression benchmarking suite. In *AutoML Conference 2023 (Workshop)*, 2023.

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. Version 1.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. In *Advances in Neural Information Processing Systems*, volume 34, 2021a. URL https://arxiv.org/abs/2106.11959.

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, 2021b.

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 2023. *arXiv preprint arXiv:2307.14338*, 2023.

Hardik. Wine. Kaggle, 2021. URL https://www.kaggle.com/datasets/hrdkcodes/wine-data. Version 1.

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: What If*. Chapman and Hall/CRC, 2020. URL https://miguelhernan.org/whatifbook.

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.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. *arXiv preprint arXiv:2207.01848*, 2022. URL https://arxiv.org/abs/2207.01848.

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. URL https://arxiv.org/abs/2012.06678.

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, ISCMM 2019*, pp. 113–125. Springer, 2019.

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. *arXiv preprint arXiv:2303.05506*, 2023.

Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Steven Horng, Leo Anthony Celi, and Roger Mark. Mimic-iv (version 2.2). PhysioNet, 2023. URL https://physionet.org/content/mimiciv/2.2/.

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. *arXiv preprint arXiv:2402.10885*, 2024.

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 J. A. Little and Donald B. Rubin. *Statistical Analysis with Missing Data*. Wiley, Hoboken, NJ, 3 edition, 2019. doi: 10.1002/9781119482260.

Jiashuo Liu, Zheyan 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, 2024. URL https://arxiv.org/abs/2407.04057.

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. URL https://arxiv.org/abs/2410.18164.

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.

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 (ECCV)*, 2020. URL https://arxiv.org/abs/2003.08934.

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

Juan P. Noriega, Alfredo García, and Joaquín Tinguaro Rodríguez. Machine learning for credit risk prediction: A systematic review. *Data*, 8(11):169, 2023. doi: 10.3390/data8110169.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. URL https://arxiv.org/abs/2303.08774.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Zachary A Pardos, Ryan SJD Baker, Maria OCZ San Pedro, Sujith M Gowda, and Supreeth M Gowda. Affective states and state tests: Investigating how affect throughout the school year predicts end of year learning outcomes. In *Proceedings of the third international conference on learning analytics and knowledge*, pp. 117–124, 2013.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, UK, 2 edition, 2009.

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

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 *Proceedings of the 42nd International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2502.05564. Also available as arXiv:2502.05564.

Raghu Ramakrishnan and Johannes Gehrke. *Database Management Systems*. McGraw–Hill, 3 edition, 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.

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, 2015.

Avi Silberschatz, Henry F. Korth, and S. Sudarshan. *Database System Concepts*. McGraw-Hill, 7 edition, 2020. ISBN 9780078022159.

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

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

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 *Conference on Innovative Data Systems Research (CIDR)*, 2005.

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.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, 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 et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. URL https://arxiv.org/abs/2307.09288.

Boris Van Breugel and Mihaela Van Der Schaar. Position: Why tabular foundation models should be a research priority. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 48976–48993. PMLR, 2024. URL https://proceedings.mlr.press/v235/van-breugel24a.html.

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. Version 1.

Joaquin Vanschoren. Grub damage. OpenML, 2014. URL https://api.openml.org/d/338. Version 1.

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 Xu, Olcay Cirit, Reza Asadi, Yizhou Sun, and Wei Wang. Mixture of in-context prompters for tabular pfns. *arXiv preprint arXiv:2405.16156*, 2024.

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. *arXiv preprint arXiv:2407.03257*, 2024.

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. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/7d97667a3e056acab9aaf653807b4a03-Abstract.html.

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.

W. Yu, G. Zhao, Q. Liu, and H. Song. Integrating big data analytics into supply chain finance: A system dynamics and empirical study. *International Journal of Production Economics*, 236:108122, 2021. doi: 10.1016/j.ijpe.2021.108122.

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 Voting[4], Unemployment[5], Sepsis (Reyna et al., 2020), Public Health Insurance[5], Income[5], Hospital Readmission (Strack et al., 2014), Food Stamps[5], Diabetes[6], College Scorecard[7], and ASSISTments (Pardos et al., 2013). Detailed information of the benchmark TableShift we adopt in Section 7.8 can be found in Table 27.

Table 27: Dataset Statistics of datasets in TableShift

Dataset	Num features	Train size	IID test size	OOD test size	Domain split	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
Diabets	142	969,229	121,154	209,375	Race	1/5
Food Stamps	239	629,018	78,628	48,878	Geographic Region	9/1
Hospital Readm.	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 28.

Table 28: 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]

[4] https://electionstudies.org/
[5] https://www.census.gov/programs-surveys/acs
[6] https://www.cdc.gov/brfss/index.html
[7] https://collegescorecard.ed.gov/

B Omitted Details in Section 6

B.1 Omitted Details in Section 6.1

Proposition B.1 (Formal Version of Proposition 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 $\{p(\mathbf{X}^{te}_{\pi}|\mathbf{X}^{te}_{-\pi}, \mathbf{X}^{ct}) : \forall \pi \in \Pi_k\}$.*

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

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

$$p(X^{\text{te}}_j|\mathbf{X}^{\text{te}}_{-j}, \mathbf{X}^{\text{ct}}) = \frac{p(\mathbf{X}^{\text{te}}_{\pi}|\mathbf{X}^{\text{te}}_{-\pi}, \mathbf{X}^{\text{ct}})}{p(\mathbf{X}^{\text{te}}_{\pi\setminus\{j\}}|\mathbf{X}^{\text{te}}_{-\pi}, \mathbf{X}^{\text{ct}})} = \frac{p(\mathbf{X}^{\text{te}}_{\pi}|\mathbf{X}^{\text{te}}_{-\pi}, \mathbf{X}^{\text{ct}})}{\int_{\Omega} p\left(\mathbf{X}^{\text{te}}_{\pi\setminus\{j\}}, X^{\text{te}\prime}_j|\mathbf{X}^{\text{te}}_{-\pi}, \mathbf{X}^{\text{ct}}\right) \mathrm{d}X^{\text{te}\prime}_j}.$$

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

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

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

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

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

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

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

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

Now we show that the knowledge of a single conditional distribution $p(X^{\text{te}}_j|\mathbf{X}^{\text{te}}_{-j}, \mathbf{X}^{\text{ct}})$ 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^{te}_1 given X^{te}_2 is

$$X^{\text{te}}_1 \mid X^{\text{te}}_2 \sim \mathcal{N}\left(\Sigma_{12}\Sigma_{22}^{-1}X^{\text{te}}_2 + \mu_1 - \Sigma_{12}\Sigma_{22}^{-1}\mu_2, \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}\right).$$

Consequently, knowledge of $p(X^{\text{te}}_1|X^{\text{te}}_2)$ 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 $p(X^{\text{te}}_1|X^{\text{te}}_2)$ alone. By symmetry, $p(\mathbf{X}^{\text{te}})$ also cannot be recovered solely from $p(X^{\text{te}}_2|X^{\text{te}}_1)$.

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 $\{q_\theta(\mathbf{X}^{\text{te}}_{\pi}|\mathbf{X}^{\text{te}}_{-\pi}, \mathbf{X}^{\text{ct}}) : \pi \in \Pi_k\}$. 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 $q_{\theta^*}(\mathbf{X}_\pi^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}) = p(\mathbf{X}_\pi^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})$ for all $\mathbf{X}^{ct} \in \Omega^{m\times d}$, $\mathbf{X}^{te} \in \Omega^d$, and $\pi \subseteq [d]$, and that the minimizer of $L_k(\theta)$ is unique for every k. Assume that for all $\theta \in \Theta$, $\mathbf{X}^{ct} \in \Omega^{m\times d}$, $\mathbf{X}^{te} \in \Omega^d$, and $\pi \subseteq [d]$, the gradient norm $\|\nabla_\theta q_\theta(\mathbf{X}_\pi^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})\|_2$ and the Hessian norm $\|\nabla_\theta^2 q_\theta(\mathbf{X}_\pi^{te}|\mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct})\|_F$ 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 $L : \Theta \to \mathbb{R}$, such that $L(\theta) = \mathbb{E}_{\mathbf{X}\sim p}[\ell_\theta(\mathbf{X})]$ for $\ell_\theta : \mathcal{X} \to \mathbb{R}$. Let Θ^* be the 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 and for every θ_1, θ_2 near θ^*, we have*
$$\left|\ell_{\theta_1}(\mathbf{X}) - \ell_{\theta_2}(\mathbf{X})\right| < B(\mathbf{X}) \left\|\theta_1 - \theta_2\right\|_2$$
- *(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*
$$\sup_{\theta\in\Theta} \left|\hat{\mathbb{E}}[\ell_\theta(\mathbf{X})] - L(\theta)\right| \xrightarrow{p} 0.$$
- *(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:

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

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*

$$\nabla_\theta^2 L_k(\theta^*) = \mathrm{Cov}_{(\mathbf{X}^{ct}, \mathbf{X}^{te})\sim p,\, \pi\sim\mathrm{Unif}(\Pi_k)}\left(-\nabla_\theta \log q_{\theta^*}\left(\mathbf{X}_\pi^{te} \mid \mathbf{X}_{-\pi}^{te}, \mathbf{X}^{ct}\right)\right). \tag{5}$$

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

$$\begin{aligned}
&\nabla_\theta^2 L_k(\theta^*) \\
&= \nabla_\theta^2 \mathbb{E}_{\mathbf{X}^{\mathrm{ct}}, \mathbf{X}^{\mathrm{te}}, \pi}\left[-\log q_{\theta^*}(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}}, \mathbf{X}^{\mathrm{ct}})\right] \\
&= \mathbb{E}_{\mathbf{X}^{\mathrm{ct}}, \mathbf{X}^{\mathrm{te}}, \pi}\left[-\nabla_\theta^2 \log q_{\theta^*}(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}}, \mathbf{X}^{\mathrm{ct}})\right] \\
&= \mathbb{E}_{\mathbf{X}^{\mathrm{ct}}, \mathbf{X}^{\mathrm{te}}, \pi}\left[\left(\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}}, \mathbf{X}^{\mathrm{ct}})\right)\left(\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}}, \mathbf{X}^{\mathrm{ct}})\right)^\top - \frac{\nabla_\theta^2 q_{\theta^*}(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}}, \mathbf{X}^{\mathrm{ct}})}{q_{\theta^*}(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}}, \mathbf{X}^{\mathrm{ct}})}\right].
\end{aligned} \tag{6}$$

In addition, we have

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

Combined with Equation (6), we can get that

$$\nabla_\theta^2 L_k(\theta^*) = \mathbb{E}_{\mathbf{X}^{\mathrm{ct}},\mathbf{X}^{\mathrm{te}},\pi}\left[\left(\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}})\right)\left(\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}})\right)^\top\right]. \tag{7}$$

Now rewrite the right-hand side of Equation (5) and we get

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

Note that

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

Combined with Equation (8), we can get that

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

Now the claim follows from Equations (7) and (9). □

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 Lemma 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:

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

Here due to Lemma B.4, it holds that

$$\begin{aligned}
\Gamma_k &= \left(\nabla_\theta^2 L_k\left(\theta^*\right)\right)^{-1}\mathrm{Cov}_{(\mathbf{X}^{\mathrm{ct}},\mathbf{X}^{\mathrm{te}})\sim p,\,\pi\sim\mathrm{Unif}(\Pi_k)}\left(-\nabla_\theta \log q_{\theta^*}\left(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\right)\left(\nabla_\theta^2 L_k\left(\theta^*\right)\right)^{-1}\\
&= \left(\nabla_\theta^2 L_k\left(\theta^*\right)\right)^{-1}.
\end{aligned} \tag{10}$$

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

$$\begin{aligned}
\log q_\theta(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}) &= \log q_\theta\left(\mathbf{X}_\gamma^{\mathrm{te}}, X_j^{\mathrm{te}}|\mathbf{X}_{-(\gamma\cup\{j\})}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\\
&= \log q_\theta\left(X_j^{\mathrm{te}}|\mathbf{X}_{-(\gamma\cup\{j\})}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right) + \log q_\theta\left(\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right).
\end{aligned} \tag{11}$$

As a result, we have

$$
\begin{aligned}
&\nabla_\theta^2 L_{k+1}(\theta^*) \\
&= \mathbb{E}_{(\mathbf{X}^{\mathrm{ct}},\mathbf{X}^{\mathrm{te}})\sim p,\pi\sim\mathrm{Unif}(\Pi_{k+1})} \left[\left(\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}})\right)\left(\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}})\right)^\top \right] && \text{(Due to Equation (7))} \\
&= \mathbb{E}_{(\mathbf{X}^{\mathrm{ct}},\mathbf{X}^{\mathrm{te}})\sim p,\gamma\sim\mathrm{Unif}(\Pi_{k}),j\in\mathrm{Unif}([d]\setminus\gamma)} \left[\left(\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}})\right)\left(\nabla_\theta \log q_{\theta^*}(\mathbf{X}_\pi^{\mathrm{te}}|\mathbf{X}_{-\pi}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}})\right)^\top \right] && \text{(Letting } \pi=\gamma\cup\{j\}) \\
&= \mathbb{E}_{\mathbf{X}^{\mathrm{ct}},\mathbf{X}^{\mathrm{te}},\gamma,j}\Bigg[\left(\nabla_\theta \log q_{\theta^*}\left(X_j^{\mathrm{te}}|\mathbf{X}_{-(\gamma\cup\{j\})}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right) + \nabla_\theta \log q_{\theta^*}\left(\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\right)\times \\
&\qquad \left(\nabla_\theta \log q_{\theta^*}\left(X_j^{\mathrm{te}}|\mathbf{X}_{-(\gamma\cup\{j\})}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right) + \nabla_\theta \log q_{\theta^*}\left(\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\right)^\top\Bigg]. && \text{(By Equation (11))}
\end{aligned}
$$

Define

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

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

$$
\begin{aligned}
B &= \mathbb{E}_{\mathbf{X}^{\mathrm{ct}},\mathbf{X}^{\mathrm{te}},\gamma,j}\left[\left(\nabla \log q_{\theta^*}\left(X_j^{\mathrm{te}}|\mathbf{X}_{-(\gamma\cup\{j\})}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\right)\left(\nabla \log q_{\theta^*}\left(\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\right)^\top\right] \\
&= \mathbb{E}_{\mathbf{X}^{\mathrm{ct}},\gamma,j,\mathbf{X}_{-\gamma}^{\mathrm{te}}}\left[\left(\nabla \log q_{\theta^*}\left(X_j^{\mathrm{te}}|\mathbf{X}_{-(\gamma\cup\{j\})}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\right)\cdot \mathbb{E}_{\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}}\left[\left(\nabla \log q_{\theta^*}\left(\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\right)^\top\right]\right] \\
&= \mathbb{E}_{\mathbf{X}^{\mathrm{ct}},\gamma,j,\mathbf{X}_{-\gamma}^{\mathrm{te}}}\left[\left(\nabla \log q_{\theta^*}\left(X_j^{\mathrm{te}}|\mathbf{X}_{-(\gamma\cup\{j\})}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\right)\cdot 0\right] && \text{(See Equation (12) below)} \\
&= 0.
\end{aligned}
$$

Here the third equation is due to:

$$
\begin{aligned}
\mathbb{E}_{\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}}\left[\nabla_\theta \log q_{\theta^*}\left(\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\right] &= \int_{\Omega^k} \frac{\nabla_\theta q_{\theta^*}\left(\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)}{q_{\theta^*}\left(\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)}\cdot p\left(\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\mathrm{d}\mathbf{X}_\gamma^{\mathrm{te}} \\
&= \int_{\Omega^k} \nabla_\theta q_{\theta^*}\left(\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\mathrm{d}\mathbf{X}_\gamma^{\mathrm{te}} \\
&= \nabla_\theta \int_{\Omega^k} q_{\theta^*}\left(\mathbf{X}_\gamma^{\mathrm{te}}|\mathbf{X}_{-\gamma}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}}\right)\mathrm{d}\mathbf{X}_\gamma^{\mathrm{te}} = \nabla_\theta 1 = 0.
\end{aligned} \tag{12}
$$

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 $\Gamma_k = \left(\nabla_\theta^2 L_k(\theta^*)\right)^{-1}$ by Equation (10), it follows that $\Gamma_{k+1} \preceq \Gamma_k$. Now the claim follows. □

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 $(\mathbf{X}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}})$ by evaluating how easily an algorithm can generate samples from the joint distribution $q(\mathbf{X}^{\mathrm{te}},\mathbf{X}^{\mathrm{ct}})$ with access to local conditional distributions $q(\cdot|\mathbf{X}^{\mathrm{ct}},\mathbf{X}_{-\pi}^{\mathrm{te}})$. 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 $(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$,

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

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 $\forall\ \theta \in \Theta$, $p(\mathbf{X}^{\text{te}}_\pi | \mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi}) > 0$ implies $q_\theta(\mathbf{X}^{\text{te}}_\pi | \mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi}) > \beta$.
2. For any $\epsilon > 0$, there exists a partition $\text{Par}(\Theta) = \{\Theta_1, \ldots, \Theta_{|\text{Par}(\Theta)|}\}$ 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 $(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$,

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

   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 $p(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi})$ supports the parametric distribution $q_\theta(\cdot | \mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi})$. 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 $q_\theta(\mathbf{X}^{te}, \mathbf{X}^{ct})$ satisfies approximate tensorization of entropy with respect to some constant $C_1(q_\theta)$ and the mask set Π_1. Then for any $1 \leq k \leq d$, $q_\theta(\mathbf{X}^{te}, \mathbf{X}^{ct})$ 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 $\theta \in \Theta$,

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

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 Proposition B.6, and prove the upper bound for generalization error in Proposition B.7.

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

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

By Definition B.1,

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

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

$$D_{\text{KL}}(r \parallel q_\theta) \le \underline{C_k}(q_\theta) \cdot \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_k)} \left[D_{\text{KL}}(r(\cdot|\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi}) \parallel q_\theta(\cdot|\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi})) \right]. \tag{13}$$

We have

$$\begin{aligned}
&\mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_{k+1})} \left[D_{\text{KL}}(r(\cdot|\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi}) \parallel q_\theta(\cdot|\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi})) \right] \\
&= \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_k), a \sim \text{Unif}([d] \setminus \pi)} \left[D_{\text{KL}}(r(\cdot|\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi \cup \{a\}}) \parallel q_\theta(\cdot|\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi \cup \{a\}})) \right] \\
&\ge \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_k), a \sim \text{Unif}([d] \setminus \pi)} \left[D_{\text{KL}}(r(\cdot|\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi}) \parallel q_\theta(\cdot|\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi})) \right] \\
&= \mathbb{E}_{(\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}) \sim p, \pi \sim \text{Unif}(\Pi_k)} \left[D_{\text{KL}}(r(\cdot|\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi}) \parallel q_\theta(\cdot|\mathbf{X}^{\text{ct}}, \mathbf{X}^{\text{te}}_{-\pi})) \right].
\end{aligned} \tag{14}$$

The inequality follows from the data processing inequality:

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

Combining Equations (13) and (14),

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

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 $\underline{C_{k+1}}(q_\theta) \le \underline{C_k}(q_\theta)$. By deduction, for any $1 \le k \le d$, $q_\theta(\mathbf{X}^{\text{te}}, \mathbf{X}^{\text{ct}})$ satisfies approximate tensorization of entropy with respect to some constant $C_k(q_\theta)$ and the mask set Π_k. □

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

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

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:*

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

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 $\mathbf{X}$,*
$$D_{\text{KL}}(r \parallel q) \le C(q_\theta) \cdot \mathbb{E}_{\mathbf{X} \sim p(\mathbf{X}), \pi \sim p(\pi|\mathbf{X})} \left[D_{\text{KL}}(r(\cdot|\mathbf{X}_{-\pi}, \pi) \parallel q(\cdot|\mathbf{X}_{-\pi}, \pi)) \right].$$
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 $Par(\Theta) = \{\Theta_1, \ldots, \Theta_{|Par(\Theta)|}\}$ of Θ, such that $\forall\ \pi \subset [d]$, $\forall\ \Theta_i \in Par(\Theta)$, $\forall\ \theta_1, \theta_2 \in \Theta_i$, and any $\mathbf{x}$,*
$$\left| \log q_{\theta_1}(\mathbf{x}_\pi|\mathbf{x}_{-\pi}, \pi) - \log q_{\theta_2}(\mathbf{x}_\pi|\mathbf{x}_{-\pi}, \pi) \right| \le \frac{\epsilon}{2}.$$
Let $N(\Theta, \epsilon)$ be the cardinality of the smallest partition $Par(\Theta)$ 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 $B = \sqrt{\frac{1}{l\delta} \cdot (8|\Omega|)^d N(\Theta, \epsilon)} + \sqrt{\frac{1}{2n} \cdot \log \frac{8N(\Theta,\epsilon)}{\delta}}$, 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 Proposition B.7. For each pair of $(\mathbf{x}^{\mathrm{ct},(i)}, \mathbf{x}^{\mathrm{te},(i)})$, exactly one mask π_i is drawn independently from $\mathrm{Unif}(\Pi_k)$. Therefore, $\pi \perp\!\!\!\perp (\mathbf{X}^{\mathrm{ct}}, \mathbf{X}^{\mathrm{te}})$. It follows from Lemma B.8 that for each $1 \le k \le d$, for any $\epsilon > 0$ and any $\delta \in (0,1)$, with probability at least $1-\delta$, for any $\theta \in \Theta$,

$$D_{\mathrm{TV}}(q_\theta(\mathbf{X}^{\mathrm{te}}, \mathbf{X}^{\mathrm{ct}}) \parallel p(\mathbf{X}^{\mathrm{te}}, \mathbf{X}^{\mathrm{ct}})) < \sqrt{\frac{1}{2}\underline{C_k}(q_\theta)\left(\hat{L}_k(\theta) + B\log\frac{1}{\beta} + \epsilon\right)} + C, \tag{15}$$

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 (15) is satisfied for any $1 \le k \le d$ and any $\theta \in \Theta$.

Furthermore, we have

$$\begin{aligned}
\mathbb{E}_{\mathbf{X}^{\mathrm{ct}}\sim p}\left[D_{\mathrm{TV}}(q_\theta(\cdot|\mathbf{X}^{\mathrm{ct}}) \parallel p(\cdot|\mathbf{X}^{\mathrm{ct}}))\right] &= \sum_{\mathbf{x}^{\mathrm{ct}}\in\Omega^{m\times d}} p(\mathbf{x}^{\mathrm{ct}}) D_{\mathrm{TV}}(q_\theta(\cdot|\mathbf{X}^{\mathrm{ct}} = \mathbf{x}^{\mathrm{ct}}) \parallel p(\cdot|\mathbf{X}^{\mathrm{ct}} = \mathbf{x}^{\mathrm{ct}})) \\
&= \sum_{\mathbf{x}^{\mathrm{ct}}\in\Omega^{m\times d}} p(\mathbf{x}^{\mathrm{ct}}) \cdot \frac{1}{2} \sum_{\mathbf{x}^{\mathrm{te}}\in\Omega^{d}} \left|q_\theta(\mathbf{x}^{\mathrm{te}}|\mathbf{x}^{\mathrm{ct}}) - p(\mathbf{x}^{\mathrm{te}}|\mathbf{x}^{\mathrm{ct}})\right| \\
&= \frac{1}{2} \sum_{\mathbf{x}^{\mathrm{ct}}\in\Omega^{m\times d}, \mathbf{x}^{\mathrm{te}}\in\Omega^{d}} \left|p(\mathbf{x}^{\mathrm{ct}})q_\theta(\mathbf{x}^{\mathrm{te}}|\mathbf{x}^{\mathrm{ct}}) - p(\mathbf{x}^{\mathrm{ct}})p(\mathbf{x}^{\mathrm{te}}|\mathbf{x}^{\mathrm{ct}})\right| \\
&= \frac{1}{2} \sum_{\mathbf{x}^{\mathrm{ct}}\in\Omega^{m\times d}, \mathbf{x}^{\mathrm{te}}\in\Omega^{d}} \left|q_\theta(\mathbf{x}^{\mathrm{te}}, \mathbf{x}^{\mathrm{ct}}) - p(\mathbf{x}^{\mathrm{te}}, \mathbf{x}^{\mathrm{ct}})\right| \\
&= D_{\mathrm{TV}}(q_\theta(\mathbf{X}^{\mathrm{te}}, \mathbf{X}^{\mathrm{ct}}) \parallel p(\mathbf{X}^{\mathrm{te}}, \mathbf{X}^{\mathrm{ct}})).
\end{aligned} \tag{16}$$

The proof is complete by combining Equations (15) and (16). □