TabK: Amortized Bayesian Estimation of the Number of Clusters in Tabular Data

Mohammadreza Bakhtyari[1]* **Bogdan Mazoure**[2] **Renato Cordeiro de Amorim**[3]
Guillaume Rabusseau[2,4] **Vladimir Makarenkov**[1,2]

[1]Université du Québec à Montréal, Canada [2]Mila, Canada
[3]University of Essex, UK [4]DIRO, Université de Montréal, Canada

bakhtyari.mohammadreza@courrier.uqam.ca, bogdan.mazoure@gmail.com,
r.amorim@essex.ac.uk, guillaume.rabusseau@umontreal.ca, makarenkov.vladimir@uqam.ca

Abstract

Estimating the number of clusters, k, is a fundamental prerequisite for most clustering algorithms. Traditionally, this estimation has relied on computationally expensive grid searches over internal cluster validity indices (CVIs) or adaptive clustering algorithms, both constrained by specific statistical assumptions. We introduce TabK, a zero-shot, permutation-invariant transformer architecture that frames the estimation of k as an amortized Bayesian inference problem for tabular data. TabK leverages a novel Quantile Feature Encoder (QFE) to extract position-free, content-derived representations from empirical marginal distributions. By learning a generalized structural bias over a diverse synthetic prior, TabK estimates the number of clusters in a single forward pass per ensemble member. Evaluated across 50 diverse real-world benchmarks, TabK outperforms all competing methods, achieving a mean absolute error (MAE) value of 1.04 compared to 2.14 for the closest competitor. TabK also achieves the lowest average inference time, producing estimates in less than one second on average and running $2\times$ faster than TabClustPFN and $30\times$ faster than traditional CVIs such as Silhouette and Calinski–Harabasz. Code is available at `github.com/mrbakhtyari/TabK`.

1 Introduction

The performance of most clustering algorithms hinges on a critical prerequisite: the number of clusters, k, which governs partition quality. This dependency spans partition-based k-means [21], graph-based spectral clustering [26], and probabilistic Gaussian Mixture Models (GMMs) [25], and extends to modern deep clustering methods, including Deep Embedded Clustering (DEC) [38], Deep Clustering Network (DCN) [39], and Contrastive Clustering (CC) [19], all of which enforce an inductive bias based on a predefined k. In practice, however, k is rarely known in advance. In domains such as customer segmentation [28], patient stratification [37], or cell-type discovery [16], the number of clusters is itself a quantity of interest. An incorrect choice of k can merge distinct clusters or fragment coherent ones, thereby distorting downstream analyses.

Conventionally, determining the number of clusters in a given tabular dataset involves a validation over a discrete search space via geometric heuristics [33], internal cluster validity indices (CVIs) [5, 31], statistically grounded null-reference comparisons [34], or adaptive clustering algorithms [27]. However, these approaches incur substantial computational overhead, rely on specific statistical assumptions, and introduce an inductive bias toward a predefined clustering objective. While recent deep learning approaches attempt to modernize this pipeline by integrating the estimation of k directly

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

into representation learning via joint gradient optimization [30], they typically require dataset-specific optimization. This paradigm is particularly challenging for tabular data, which lacks shared spatial or sequential patterns common in image and text domains. The necessity for per-dataset retraining to process heterogeneous tabular data incurs high computational costs, hindering practical deployment of deep learning models in this domain [12].

Despite these challenges, the emergence of Prior-Data Fitted Networks (PFNs) [13] has recently redefined tabular deep learning. By employing the in-context learning capabilities of transformers, PFNs amortize Bayesian inference into a single forward pass, shifting the learning paradigm from dataset-specific gradient optimization to optimization over a diverse generative prior. While models like ZEUS [23] provide zero-shot clustering on tabular datasets, they still require a predefined k. Subsequent architectures, such as TabClustPFN [42], attempt to bridge this gap by jointly solving zero-shot clustering and cardinality inference problems. However, constraining a single model to infer global dataset structure and perform local cluster assignments tends to limit its capacity and tightly couples the resulting k-estimation task to a model-specific embedding space. This limitation motivates a fundamentally different design principle: treating the estimation of k as a structural inference problem, which is solved independently of the downstream clustering algorithm.

This separation offers two practical advantages. First, it avoids grid search over candidate values of k, as is often done within clustering pipelines. Second, it allows the same predicted k to be reused with different clustering algorithms. To this end, we introduce TabK, a transformer-based double-invariant architecture—ensuring permutation invariance to both objects and features—designed for zero-shot cluster cardinality estimation. Instead of relying on isolated cell-value embeddings, TabK employs a Quantile Feature Encoder (QFE) to extract learned quantile representations for each feature, explicitly encoding its global distributional statistics. Following this feature-level encoding, the model contextualizes embeddings via cross-column attention before compressing the resulting feature matrix into a dataset-level representation via inter-row attention and dynamic pooling.

Finally, acknowledging that the number of clusters follows an ordinal structure, TabK relies on Deep Label Distribution Learning (DLDL) [11] to model a soft probability distribution over candidate k values, respecting their ordinal relationships. By isolating the estimation of k, TabK operates as a lightweight, plug-and-play precursor that integrates seamlessly into existing clustering pipelines. Evaluating such an estimator requires a reference value of k. Following standard practice in the clustering literature [1, 30, 23, 42, 3], throughout the paper, we use the annotated class cardinality k^* of labeled benchmarks as a proxy for the ground-truth number of clusters. In summary, our primary contributions are as follows:

- We propose TabK, a transformer-based architecture that reframes the estimation of k in tabular datasets from a manual, iterative, algorithm-dependent heuristic to a search-free zero-shot inference task.
- We introduce a quantile-based feature encoder that summarizes column-wise marginal distributions in a permutation-invariant manner.
- We frame estimation of the number of clusters as an ordinal-aware distribution learning task using DLDL, leveraging soft Gaussian targets and a Kullback–Leibler (KL) divergence loss to encode the ordinal relationships among candidate values of k.
- Across 50 real-world benchmarks, TabK outperforms all 17 competing methods for estimating k, achieving a mean absolute error (MAE) of 1.04, compared to 2.14 for the closest competitor and to 2.24 for TabClustPFN. TabK also yields the highest average Adjusted Rand Index (ARI) when its estimates of k are supplied to k-means. Moreover, it runs more than $2\times$ faster than TabClustPFN and $30\times$ faster than classical CVIs.

2 Related work

Classical estimation of the number of clusters. Estimating the number of clusters, k, has historically relied on geometric heuristics and internal CVIs, both typically evaluated over a grid search. Prominent examples include the Elbow method [33], which tracks the marginal reduction of within-cluster variance, alongside established CVIs such as the Silhouette [31], Davies–Bouldin [6], Dunn [8], and Calinski–Harabasz [5] indices. To provide a more rigorous statistical grounding, the Gap Statistic [34] formalizes structural inference by comparing observed data dispersion against a suitable

null reference distribution. However, these estimation techniques rely on computationally expensive iterative execution and geometric assumptions that fail to capture heterogeneous feature spaces typical of real-world tabular data. Furthermore, because heuristic evaluation requires instantiating a specific clustering algorithm at each step of the grid search, the estimated k is biased by the geometric assumptions of the chosen algorithm rather than reflecting the intrinsic structure of the data.

Joint estimation of the number of clusters. To bypass external grid searches, several algorithms integrate k-estimation directly into the clustering objective. For instance, X-Means [27] recursively splits k-means clusters based on the Bayesian Information Criterion (BIC), while subsequent approaches such as Intelligent Minkowski k-means (IMWk) [7] and Cluster Number Assisted k-means (CNAK) [32] incorporate adaptive feature weighting and random data sampling. In the deep learning domain, dynamic clustering architectures have emerged to estimate the number of clusters internally. DeepDPM [30] frames the problem as a Dirichlet Process Mixture, allowing the network to split or merge clusters during training. Similarly, UNSEEN [18] estimates k by progressively removing clusters during training, supported by a nearest-neighbor regularization loss. While these deep clustering methods eliminate the need for a grid search, they are constrained by iterative, dataset-specific gradient optimization, which is computationally expensive for heterogeneous tabular data that lacks shared inductive biases of image or text modalities.

Prior-data fitted networks and tabular data. Although models like TabPFN [13] have achieved state-of-the-art performance in supervised tabular classification, extending this paradigm to unsupervised domains remains an open challenge. In the clustering landscape, ZEUS [23] successfully generates zero-shot tabular embeddings but requires an externally provided k for clustering. Most closely related to our work is TabClustPFN [42], which jointly performs clustering and k-estimation. However, its k-estimation is architecturally subordinate to clustering: it must compute a soft-assignment Gram matrix ($G_k = P_k^\top P_k$) for every candidate $k \in \{2, \ldots, k_{\max}\}$. This design incurs $k_{\max}-1$ redundant decoder passes whose outputs are subsequently discarded during inference and couples k-estimation to the quality of the assignment logits. Rather than deriving k as an artifact of cluster assignment, our work treats structural inference as a decoupled modular precursor that estimates a distribution over k in a single forward pass. This decoupling frees the architecture from the need to balance local assignment accuracy against global structural inference and yields a plug-and-play module compatible with any k-dependent clustering algorithm.

3 Methodology

The number of clusters, k, can be viewed primarily as a global property of the data distribution, rather than an instance-level quantity derived from individual object assignments to clusters. While joint-estimation models treat k as an artifact of local object-to-cluster mappings and classical heuristics rely on exhaustive evaluations across varying k, TabK reframes the objective. Rather than predicting per-object cluster assignments, TabK operates directly on the full dataset matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$ to learn a global mapping that approximates $p(k \mid \mathbf{X})$, where n is the number of objects and m is the number of features. To faithfully model this posterior distribution, the network must respect the permutation symmetries inherent to tabular data while bypassing the computational bottleneck of iterative local assignments. We introduce a double-invariant transformer architecture trained over a diverse generative prior with an ordinal-aware predictive head tailored to predict k in tabular data. TabK first summarizes each feature using a quantile feature encoder, then models interactions among feature representations through cross-column self-attention, and finally aggregates row embeddings into a dataset-level representation via inter-row attention and dynamic pooling. This design preserves permutation invariance over both rows and columns, avoiding the introduction of arbitrary positional information. An overview of the proposed architecture is presented in Figure 1.

3.1 Double-invariant architecture

A core challenge in permutation-invariant tabular processing is distinguishing features without relying on explicit positional indices. While networks like TabPFN [13] process rows as permutation-invariant sets, they map features using fixed linear projections. Consequently, they lack theoretical invariance to column permutations, and only achieve approximate column-permutation invariance empirically. Existing methods often treat individual cells as independent tokens [14, 22], incurring

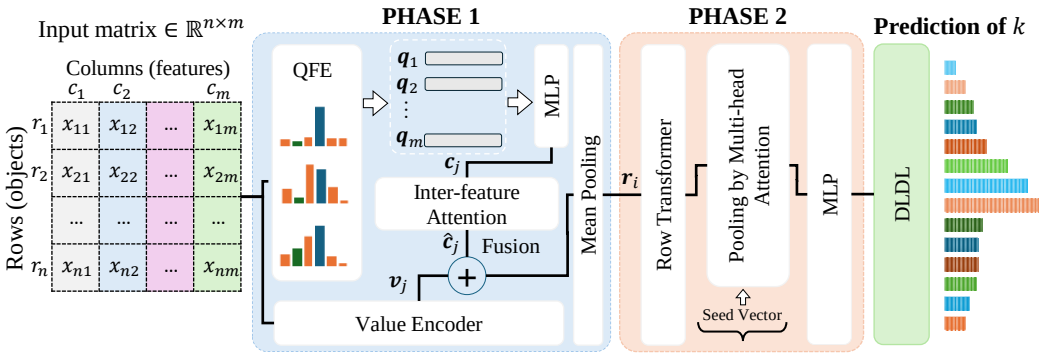


Figure 1: Overview of the TabK architecture. The model processes the input matrix $\mathbf{X}$ via a double-invariant transformer and a Quantile Feature Encoder (QFE) for zero-shot k estimation.

an $\mathcal{O}(nm^2)$ attention time complexity for feature-wise interactions. Alternatively, projecting the inputs into a fixed-dimensional latent space before attention reduces this complexity to $\mathcal{O}(m^2)$, but may prematurely compress the space, discarding the fine-grained marginal distributions essential for structural separability. To resolve this, TabK employs the empirical quantile function of each feature as a content-derived representation. This statistical signature provides an initial feature representation that, when combined with subsequent cross-column attention, enables the model to differentiate features based entirely on their global properties rather than arbitrary spatial coordinates. Furthermore, by isolating feature-wise interactions from row-wise dimensionality, we establish a *set-of-sets* architecture. This architecture decomposes the model into two sequential phases: feature identity and row compression (Section 3.1.1), followed by dataset-level structural aggregation (Section 3.1.2).

3.1.1 Phase 1: feature identity and row compression

The main objective of this phase is to transform the input matrix $\mathbf{X}$ into a sequence of row embeddings $\mathbf{r}_i \in \mathbb{R}^d$ through a process that is invariant to the input column order, capturing global feature context without combining the row and column dimensions.

Component 1: quantile feature encoder. To provide an initial structural signature for features without positional indices, we encode the statistics of each column j via a Quantile Feature Encoder (QFE). For a valid column vector $\mathbf{x}_{\cdot j}$, we extract b evenly spaced quantiles from its empirical cumulative distribution function (CDF):

$$\mathbf{q}_j = \left[Q_0(\mathbf{x}_{\cdot j}),\, Q_{\frac{1}{b-1}}(\mathbf{x}_{\cdot j}),\, \ldots,\, Q_1(\mathbf{x}_{\cdot j}) \right] \in \mathbb{R}^b. \tag{1}$$

Unlike basic moments, quantiles are order statistics that robustly capture distribution shape while remaining invariant to row permutations, with a fixed dimension regardless of the total number of objects n. The QFE natively captures distributional characteristics, allowing the network to leverage data artifacts, such as median imputation and categorical sparsity in one-hot encoding, as structural signals (for more details, see Appendix E). Furthermore, evaluating quantiles at fixed probability levels provides a common representation of marginal distributions across different dataset sizes, which facilitates extrapolation beyond the size range of the synthetic prior (see Appendix F.6). To integrate this robust statistical profile into the network's latent space, the representation $\mathbf{q}_j$ is projected into the model dimension d via a two-layer Multi-Layer Perceptron (MLP), producing the initial column identity vector $\mathbf{c}_j$.

Component 2: inter-feature attention. While marginal distributions establish an initial feature identity, each identity is computed from a single column. To contextualize each identity within the set of features, we apply a transformer encoder exclusively over the set of column identities. By running self-attention over the matrix in $\mathbb{R}^{m\times d}$ rather than the full cell representation in $\mathbb{R}^{n\times m\times d}$, we compute column identities $\hat{\mathbf{c}}_j$ at $\mathcal{O}(m^2)$, allowing features to contextualize their structural roles.

Component 3: value-identity fusion and row aggregation. An isolated scalar cell-value embedding is inherently context-agnostic; it cannot ascertain whether a specific value represents an extreme outlier or a median baseline. TabK supplies this context through additive contextual fusion with the refined column identity. We project the raw scalar cell value into the latent space using learned parameters $\mathbf{w}_v$ and $\mathbf{b}_v \in \mathbb{R}^d$, and fuse it with its context-aware feature identity:

$$\mathbf{e}_{ij} = \underbrace{\mathbf{w}_v x_{ij} + \mathbf{b}_v}_{\text{value embedding}} + \underbrace{\hat{\mathbf{c}}_j}_{\text{refined column identity}} . \tag{2}$$

Because $\hat{\mathbf{c}}_j$ already encodes the marginal distribution shape, the resulting cell representation $\mathbf{e}_{ij}$ is natively contextualized within its global statistical profile. To produce a row representation $\mathbf{r}_i$ that is invariant to column permutation, we aggregate the set of cell embeddings of row i, $\mathcal{F}_i = \{\mathbf{e}_{ij}\}_{j=1}^m$, using masked mean pooling, followed by a final projection via an MLP. This yields a vector $\mathbf{r}_i$ that combines pooled cell-value information with global column context.

3.1.2 Phase 2: dataset-level structural representation

With each row now represented as a context-aware vector $\mathbf{r}_i \in \mathbb{R}^d$, the model transitions to capturing the global distributional structure of the dataset through inter-object interaction and dynamic pooling.

Component 4: row set interaction. We process a collection of row embeddings $\{\mathbf{r}_1, \ldots, \mathbf{r}_n\}$ through a standard transformer encoder. By omitting positional encodings, this sequence modeling remains permutation-equivariant with respect to the rows. This inter-object attention is used for assessing dataset-level properties, enabling TabK to model relationships among row embeddings.

Component 5: pooling by multihead attention. The final architectural stage aggregates the variable-size set of interacting rows into a global distributional summary $\mathbf{z} \in \mathbb{R}^d$. While standard mean pooling often attenuates the sparse signals critical for identifying rare clusters, we utilize Pooling by Multihead Attention (PMA) [17]. PMA employs a learned, dataset-agnostic seed vector as a query to attend over the sequence of row embeddings. By selectively extracting representative row embeddings rather than uniformly averaging all instances, this dynamic pooling mechanism naturally adapts to varying dataset sizes and cluster imbalances.

3.1.3 Formal symmetry guarantees

Theorem 1 (Double-permutation invariance)**.** *Let $\mathcal{P}_t$ denote the set of $t \times t$ permutation matrices. For any tabular matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$, row permutation $\mathbf{\Pi}_r \in \mathcal{P}_n$, and column permutation $\mathbf{\Pi}_c \in \mathcal{P}_m$, the TabK model f_θ satisfies:*

$$f_\theta(\mathbf{\Pi}_r \mathbf{X} \mathbf{\Pi}_c) = f_\theta(\mathbf{X}) \tag{3}$$

A comprehensive mathematical proof detailing the specific equivariance and invariance guarantees of each component, alongside our empirical evaluation confirming these symmetries, is provided in Appendix C.1. Furthermore, all architectural hyperparameters and complete optimization protocols are detailed in Appendix C.

3.2 Amortized Bayesian inference via generative priors

TabK learns to generalize across structural patterns by amortizing Bayesian inference over a diverse synthetic prior. We define this prior, $p_{\text{prior}}(\mathbf{X}, k)$, as a joint distribution over tabular matrices $\mathbf{X}$ and their ground-truth numbers of clusters $k \in \{k_{\min}, \ldots, k_{\max}\}$. During training, TabK seeks the optimal parameters θ^* that minimize the expected loss over datasets drawn from this distribution:

$$\theta^* = \arg\min_\theta \mathbb{E}_{(\mathbf{X},k)\sim p_{\text{prior}}} \left[\mathcal{L}(f_\theta(\mathbf{X}), k)\right]. \tag{4}$$

Instead of a hard one-hot target, we employ DLDL [11], whose target for a dataset with k^* clusters is the discretized Gaussian:

$$q_{k^*}(k) \propto \exp\left(-\frac{(k-k^*)^2}{2\sigma^2}\right), \quad \text{for } k \in \{k_{\min}, \ldots, k_{\max}\}, \tag{5}$$

and the loss is $\mathcal{L} = \mathrm{KL}(q_{k^*} \,\|\, \hat{p})$, where $\hat{p}$ is the predicted distribution and $\sigma = 0.5$ controls the degree of label smoothing.

We train the model over a diverse mixture of synthetic topologies, including controlled Gaussians [29], overlapping elliptical distributions [41], elongated line-supported clusters [9], high-dimensional shells, and complex networked manifolds [15] (as detailed in Appendix B.2). Through this training procedure, TabK learns an approximation of $p(k \mid \mathbf{X})$ that reflects the ordinal distance between candidate values of k. For an unrestricted predictor, the expected loss is minimized by $\sum_{k'} q_{k'}(k)\, p_{\text{prior}}(k' \mid \mathbf{X})$, where the posterior under the synthetic prior is smoothed by the ordinal target kernel. In total, we generated 200,000 synthetic datasets (40,000 configurations $\times$ 5 strategies), 180,000 of which were used for training and 20,000 were held out as a test set (as detailed in Appendix B.1). We train TabK using 5-fold cross-validation on the training split. At inference, we average the log-probability outputs of the models from these folds and normalize the resulting scores to obtain the ensemble distribution, whose *argmax* gives the predicted k. All reported results, including latency, use this five-model ensemble (Appendix C.3).

4 Experiments

4.1 Experimental setup

Datasets and baselines. We evaluate TabK on a diverse representative suite of 50 real-world benchmarks sourced from OpenML [4]. Because the vast majority of practical tabular datasets contain fewer than 10,000 objects [14], we scope our primary evaluation to datasets with up to 10,000 objects, 200 features, and ground-truth cluster counts $k^* \in [2, 15]$. Notably, this requires TabK to extrapolate to scales up to 3.25 times larger than those present in its synthetic training prior. Crucially, these benchmarks include missing values and categorical features, reflecting the heterogeneity of real-world tabular data (full dataset profiles are provided in Appendix D). The benchmark was assembled to cover heterogeneous application domains and structural regimes: it spans 57 to 8,124 objects, 3 to 90 features, and $k^* \in [2, 15]$, and includes 16 datasets with categorical features and 9 with missing values; duplicate entries and datasets used to estimate the synthetic prior were excluded. All methods received the same preprocessed inputs: categorical features were one-hot encoded, missing values were median-imputed, and features were standardized.

We compare TabK against 17 established competitors, spanning the clustering literature: internal validity indices including Silhouette (Sil), Calinski–Harabasz (CH), Davies–Bouldin (DB), and Dunn; statistical frameworks such as Gap Statistic (Gap) and GMM-BIC [10]; and adaptive methods including X-Means, IMWk, and CNAK. We further compare TabK against deep learning architectures, specifically UNSEEN, and the state-of-the-art PFN-based method, TabClustPFN (TCP). We pair three self-supervised tabular encoders, VIME [40], SCARF [2], and SubTab [35], with the Silhouette and CH indices computed on their learned embeddings; in figures, each encoder is shown with its best-performing index (SCARF+CH, SubTab+Sil, and VIME+Sil) and labeled by the encoder name.

Evaluation protocol. To ensure a rigorous comparison, all classical baselines are evaluated across the same range of k using official implementations and author-recommended hyperparameters. Latency benchmarks were standardized on an Intel Core i9-13900 CPU. Additionally, as demonstrated in Section 4.3, we evaluate how TabK's latency scales when executed under GPU acceleration (NVIDIA RTX 3060 Ti). Comprehensive implementation details for all baselines are provided in Appendix G.

Metrics. We assess the model's performance across four dimensions: structural error, prediction accuracy, downstream utility, and computational efficiency. Structural error is quantified via the MAE between the predicted $\bar{k}$ and ground truth k^*. Accuracy is reported as both the Exact Match (EM) rate, the proportion of datasets for which $\bar{k} = k^*$, and the ± 1 Tolerance rate ($\tau = |\bar{k} - k^*| \leq 1$). To evaluate practical downstream utility, we report the ARI computed when supplying each model's predicted $\bar{k}$ to k-means and GMM. We additionally define the *oracle* ARI as the clustering quality achieved when the ground-truth k^* is supplied directly to k-means and GMM, providing a reference point. Finally, operational efficiency is measured via average wall-clock inference latency.

4.2 Performance on real-world benchmarks

Table 1 and Figure 2 present the aggregate performance across the 50 OpenML benchmarks. TabK exhibits the lowest median error with a low variance (see Figure 2a), in contrast to the high vari-

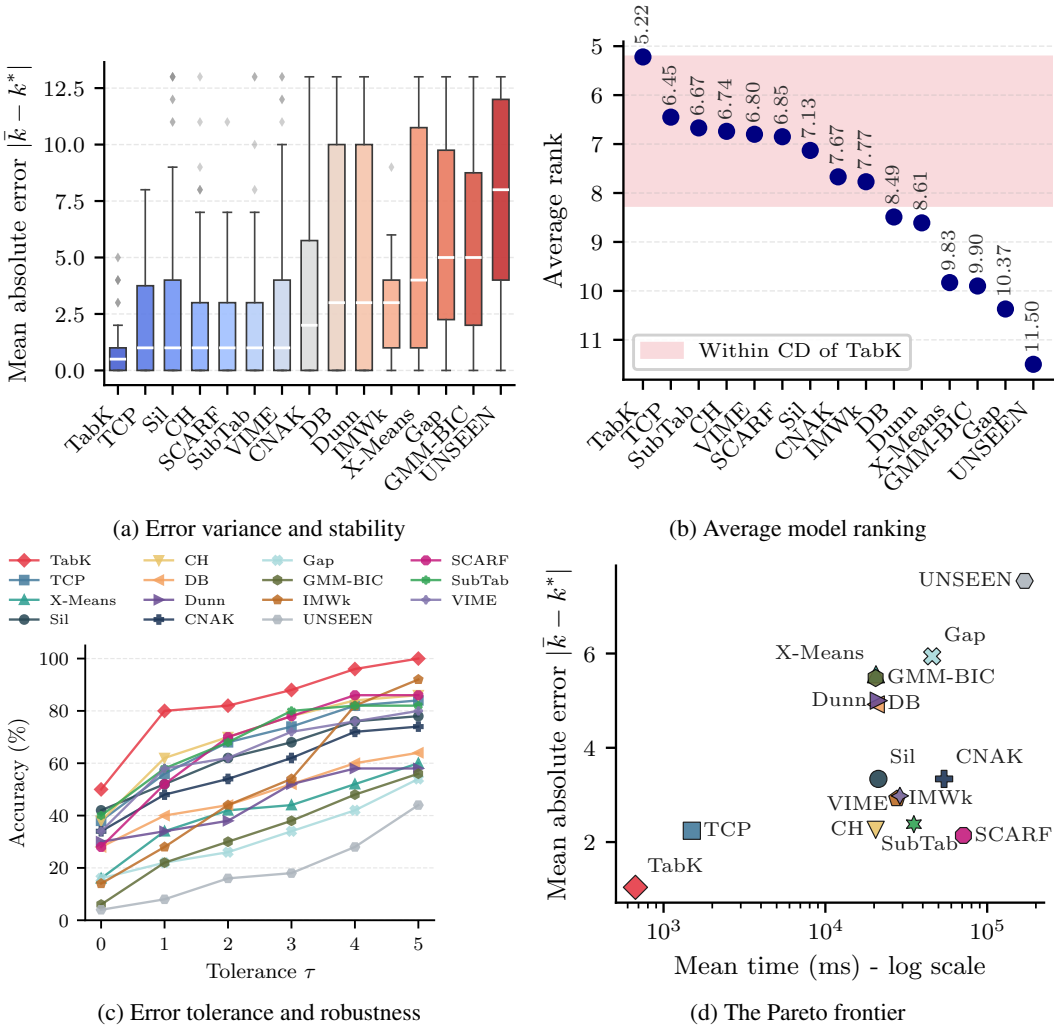
Table 1: Aggregate performance on 50 OpenML benchmarks. TabK achieves the lowest MAE, and the highest Exact Match (EM) and ± 1 Tolerance rates. ARI_1 and ARI_2 correspond to the ARIs obtained with k-means and GMM, respectively. Best results are **bolded**, second-best underlined.

	Performance metrics						**Wilcoxon test**
Method	MAE ↓	ARI_1 ↑	ARI_2 ↑	EM (%) ↑	$\tau \leq 1$ (%) ↑	Latency (s) ↓	p-value
TabK	**1.04**	**0.232**	0.221	**50.0**	**80.0**	**0.67**	–
SCARF+CH	2.14	0.218	0.207	28.0	52.0	71.3	1.5×10^{-3}
TCP	2.24	0.210	0.213	38.0	56.0	1.50	6.5×10^{-3}
CH	2.26	0.207	0.200	38.0	62.0	20.5	1.7×10^{-2}
SubTab+Sil	2.38	0.214	0.213	40.0	58.0	35.2	1.3×10^{-2}
SCARF+Sil	2.50	0.200	0.166	36.0	54.0	72.9	3.6×10^{-3}
IMWk	2.92	0.221	0.210	14.0	28.0	27.7	6.0×10^{-5}
VIME+Sil	2.98	0.230	**0.243**	34.0	58.0	28.9	3.9×10^{-3}
VIME+CH	3.08	0.190	0.201	42.0	56.0	27.4	7.1×10^{-3}
Sil	3.34	0.202	0.199	42.0	52.0	21.3	2.0×10^{-3}
CNAK	3.34	0.203	0.199	34.0	48.0	54.2	1.8×10^{-4}
SubTab+CH	3.72	0.211	0.205	32.0	52.0	33.9	8.0×10^{-4}
DB	4.92	0.205	0.210	28.0	40.0	20.5	1.7×10^{-5}
Dunn	5.00	0.183	0.194	30.0	34.0	21.4	2.7×10^{-6}
GMM-BIC	5.48	0.188	0.198	6.00	22.0	20.4	1.6×10^{-7}
X-Means	5.56	0.228	0.220	16.0	34.0	20.5	4.9×10^{-7}
Gap	5.94	0.191	0.195	16.0	22.0	45.7	4.9×10^{-8}
UNSEEN	7.54	0.171	0.166	4.00	8.00	170	3.7×10^{-9}

ance observed in both classical heuristics and modern frameworks. Moreover, TabK achieves the top average Friedman rank of 5.22 across all datasets (Figure 2b). The substantial performance margin between TabK (MAE $= 1.04$; 50.0% Exact Match) and the closest PFN-based competitor, TabClustPFN (MAE $= 2.24$; 38.0% Exact Match), provides empirical support for the proposed decoupled architectural design. To ensure this gap is not an artifact of different prior scale, we evaluated TabK within TabClustPFN's training-scale upper bounds ($n \leq 1000$, $m \leq 64$, $k \leq 10$). As detailed in Appendix F.3, TabK outperforms TabClustPFN even within these constraints. Furthermore, TabK's MAE improvements are statistically significant against each competitor (Wilcoxon test p-value < 0.05). This advantage does not stem from ensembling alone: a single ensemble member already attains an average MAE of 2.06, which is better than the results of all competing methods. Moreover, TabK's estimates are robust under irrelevant features and outliers: its MAE rises to at most 1.94, which is still lower than the MAE of every competing method on the original data (Appendix F.5).

Beyond Exact Match, TabK exhibits strong error boundedness. TabK outperforms all evaluated methods at every tolerance threshold τ (see Figure 2c), achieving a ± 1 Tolerance rate ($\tau \leq 1$) of 80.0%. In real-world tabular domains—where the ground-truth k is frequently ambiguous and dependent on human annotation—this bounded error is highly practical. Thus, when TabK diverges from the ground truth, it degrades gracefully, confining its predictions to values close to k^*. Crucially, this precise estimation of k translates directly to downstream clustering. When supplying the predicted $\bar{k}$ to standard k-means, TabK yields the highest ARI (0.232), recovering 92.18% of the *oracle* ARI (0.251), while maintaining a bounded MAE across difficulty strata (see Figure 3). It is worth noting that this advantage is not driven by datasets whose labels are poorly aligned with geometric structure: restricted to the 20 datasets with the *oracle* ARI ≥ 0.30, TabK attains an MAE of 1.25 vs 2.05 for the closest competitor (VIME+Sil), and it ranks first at every threshold between 0.10 and 0.30. When $\bar{k}$ is supplied to a GMM, TabK yields an average ARI of 0.221, recovering 85.20% of the *oracle* ARI (0.259), and outperforms all raw-feature baselines, including algorithms explicitly optimized for Gaussian geometries such as X-Means (0.220) and GMM-BIC (0.198); only VIME+Sil attains a higher average GMM ARI (0.243), with a substantially higher MAE (2.98).

Figure 2d illustrates the Pareto frontier, revealing that TabK achieves a highly favorable efficiency-performance trade-off among all evaluated methods. By amortizing the search into one forward pass per ensemble member, TabK executes in an average of 0.67 seconds per dataset. This represents



(a) Error variance and stability

(b) Average model ranking

(c) Error tolerance and robustness

(d) The Pareto frontier

Figure 2: Empirical evaluation on 50 OpenML real-world datasets. **(a)** MAE distributions. **(b)** Average Friedman ranks (Nemenyi test, $\alpha = 0.05$). **(c)** Cumulative accuracy at varying error tolerances (τ). **(d)** Efficiency-performance Pareto frontier. TabK demonstrates the highest accuracy and average ranking, the tightest error distribution, and the lowest latency among evaluated methods.

a more than 2× speedup over TabClustPFN (1.5s) and far surpasses the exhaustive iterative grid searches used by classical baselines (requiring ≥20s).

4.3 Predictive uncertainty and scalability analysis

While aggregate metrics establish TabK's performance advantage, a deeper analysis against the closest PFN-based competitor, TCP, reveals improvements in predictive uncertainty and empirical scaling. Since both models output probability distributions and support GPU acceleration, this comparison highlights differences in predictive uncertainty and computational latency.

Risk-coverage. Because TabK frames the estimation of k as a distribution-aware learning task via DLDL, the network outputs a probability mass function over the discrete candidate space of k. Thus, we measure confidence as the geometric mean (across the five ensemble members) of the probability assigned to the predicted value of k; this confidence decreases when the members disagree. To evaluate whether this confidence reflects prediction error, we perform a risk-coverage analysis (see Figure 4a). TabK demonstrates reliable confidence ranking: as the confidence threshold increases, the MAE generally decreases, approaching zero error on its most confident predictions.

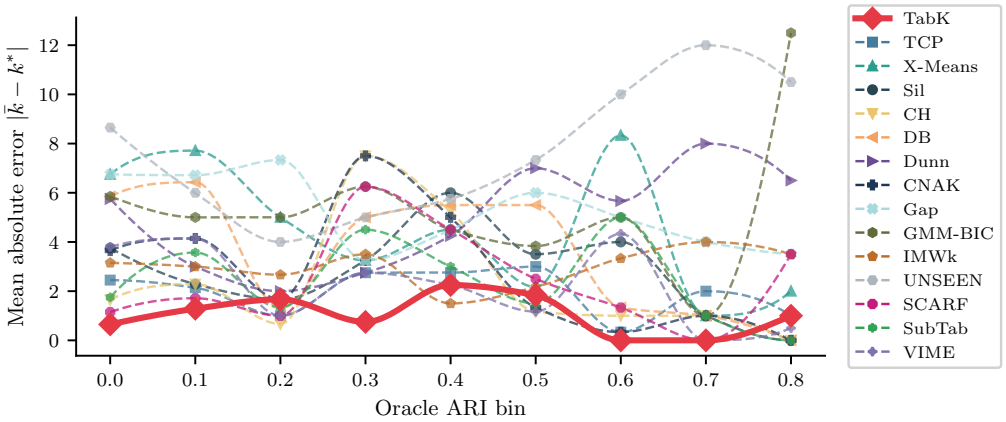
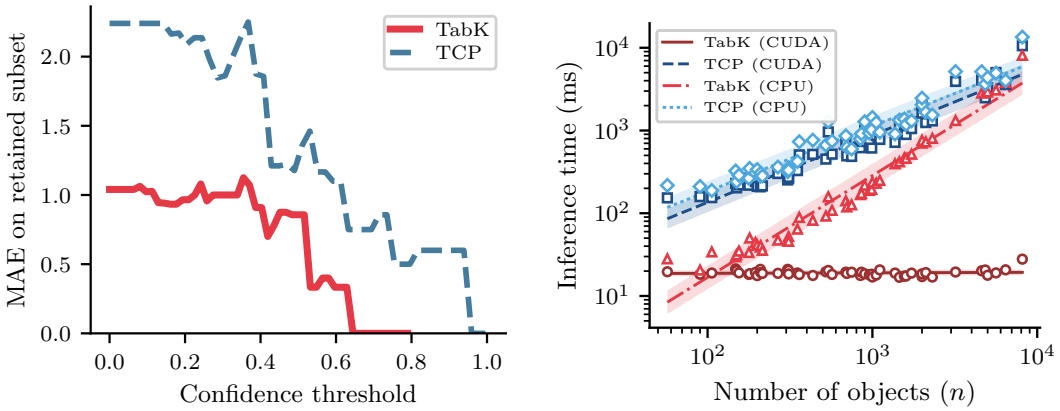


Figure 3: TabK performance across difficulty strata. MAE remains bounded across all *oracle* ARIs of k-means, whereas evaluated competitors exhibit high variance and marked failures. Datasets are grouped by *oracle* ARI into bins of width 0.1, and markers show the mean MAE per bin.



(a) Risk-coverage analysis

(b) Empirical scaling (latency vs. size)

Figure 4: Head-to-head analysis of TabK vs. TabClustPFN (TCP). **(a)** Risk-coverage analysis showing TabK's and TCP's confidence ranking. **(b)** Log-log scalability plot demonstrating TabK's order-of-magnitude speedup on GPU, with CPU execution frequently outpacing TCP on GPU.

Conversely, TabClustPFN exhibits a less monotonic risk-coverage curve, with MAE fluctuating even at higher confidence thresholds. TabK also achieves a lower negative log-likelihood (NLL) of 2.08, compared to 3.07 for TabClustPFN, confirming improved alignment between predicted distributions and observed outcomes.

Computational scalability. Figure 4b compares the empirical inference latency of TabK and TabClustPFN across varying dataset sizes. On GPU, TabK achieves an order-of-magnitude speedup over its main competitor. Notably, in 96% of cases, TabK, executed on a standard CPU, remains faster than TabClustPFN, executed on a dedicated GPU, highlighting the practical efficiency of the proposed architecture. Beyond the primary benchmark, TabK also outperforms TCP on 25 OpenML-CC18 datasets, including 10 that exceed the size of the largest synthetic training table by up to 13× (Appendix F.6).

4.4 Ablation study

To validate the proposed TabK architecture, we systematically ablated its core architectural components, predictive heads, and synthetic prior configurations. Table 2 reports the MAE results obtained for the set of 50 OpenML benchmarks. The reference TabK model, trained on the full mixture prior with a DLDL head and the complete architecture, achieves an MAE of 1.04.

Table 2: Ablation study results in terms of MAE evaluated across architectural components, predictive heads, and synthetic prior configurations. The Δ column indicates performance degradation relative to the full TabK model with an MAE of 1.04. Leave-one-generator-out removes a single generator from the prior mixture while keeping the total number of synthetic datasets fixed.

Model variation	**MAE** $\downarrow$	Δ	**Prior variation**	**MAE** $\downarrow$	Δ
Architectural components			*Prior volume*		
w/o QFE	1.82	+0.78	25% of synthetic prior	1.72	+0.68
w/o column attention	1.48	+0.44	50% of synthetic prior	1.46	+0.42
w/o PMA	1.38	+0.34	75% of synthetic prior	1.42	+0.38
Head formulations			*Leave-one-generator-out*		
Ordinal	2.34	+1.30	w/o networked [15]	1.70	+0.66
Regression	2.10	+1.06	w/o concentric shells	1.84	+0.80
Focal Loss	1.44	+0.40	w/o elliptical [41]	1.84	+0.80
Classification	1.40	+0.36	w/o Gaussians [29]	1.86	+0.82
			w/o elongated [9]	3.56	+2.52

Architectural components. The largest architectural degradation occurs when the QFE is replaced with standard parametric moments, increasing the MAE to 1.82. While higher-order parametric moments may partially recover this performance, quantiles provide a non-parametric alternative that requires no distributional assumptions, making them more appropriate for heterogeneous tabular data. Similarly, removing cross-column attention (MAE $= 1.48$) restricts the model's ability to capture interactions among column representations. Finally, replacing PMA with standard mean pooling (MAE $= 1.38$) impairs the model's capacity to selectively attend to representative rows.

Predictive head formulations. TabK uses DLDL to capture the ordinal relationships between candidate values of k. Reverting to standard categorical classification (MAE $= 1.40$) or Focal Loss (MAE $= 1.44$) degrades performance, as these objectives treat all misclassifications as equally incorrect, failing to impose a penalty proportional to the distance from k^*. Conversely, scalar regression (MAE $= 2.10$) treats k as a continuous variable, ignoring its discrete nature. The ordinal head suffers the most severe degradation (MAE $= 2.34$). DLDL provides the most reliable objective, as its soft Gaussian targets natively bridge these gaps, respecting the discrete, ordinal hierarchy of k.

Prior volume and diversity. Amortized Bayesian inference benefits from the scale and diversity of the generative prior. Training on only 25% of the synthetic datasets yields an MAE of 1.72, with performance improving monotonically with prior volume and the most substantial gain observed at the full prior. Crucially, every generator contributes to generalization: removing any single generator from the mixture, while keeping the prior size fixed, increases the MAE from 1.04 to between 1.70 (without networked manifolds) and 3.56 (without elongated line-supported clusters). Thus, training on the full mixture prior yields the best performance overall, allowing TabK to generalize across different cluster geometries and adapt to variable tabular distributions.

5 Conclusion

We introduced TabK, a double-invariant architecture designed to transition the estimation of the number of clusters, k, from a computationally expensive iterative heuristic to a rapid, zero-shot inference task. By treating k-estimation as an independent, global distributional objective rather than a localized object-assignment problem, TabK provides a modular, accurate, and efficient precursor for downstream clustering. Equipped with a Quantile Feature Encoder to process feature distributions and an ordinal-aware Deep Label Distribution Learning objective, TabK outperforms all evaluated methods in estimating k, achieving state-of-the-art performance on real-world benchmarks within the evaluated operating window ($n \leq 10{,}000$, $m \leq 200$, $k \in [2, 15]$). Furthermore, by amortizing inference over a diverse synthetic prior, TabK estimates k in less than one second on average. A thorough discussion of the limitations of this work, including the generative prior constraints, computational complexity, and ground-truth ambiguity, is presented in Appendix A.

Acknowledgments and Disclosure of Funding

This work was supported by grant 371537 through the partnership between the Ministère de l'Enseignement supérieur du Québec and the Fonds de recherche du Québec, and by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant 249644. The authors declare no competing interests.

References

[1] Olatz Arbelaitz, Ibai Gurrutxaga, Javier Muguerza, Jesús M. Pérez, and Iñigo Perona. An extensive comparative study of cluster validity indices. *Pattern Recognition*, 46(1):243–256, 2013. ISSN 0031-3203. doi: 10.1016/j.patcog.2012.07.021.

[2] Dara Bahri, Heinrich Jiang, Yi Tay, and Donald Metzler. SCARF: Self-supervised contrastive learning using random feature corruption. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=CuV_qYkmKb3.

[3] Mohammadreza Bakhtyari, Bogdan Mazoure, Renato Cordeiro de Amorim, Guillaume Rabusseau, and Vladimir Makarenkov. ClustRecNet: A novel end-to-end deep learning framework for clustering algorithm recommendation. *IEEE Access*, 14:81352–81365, 2026. doi: 10.1109/ACCESS.2026.3697689.

[4] Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Pieter Gijsbers, Frank Hutter, Michel Lang, Rafael Gomes Mantovani, Jan N. van Rijn, and Joaquin Vanschoren. OpenML benchmarking suites. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=OCrD8ycKjG.

[5] T. Caliński and J Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3(1):1–27, 1974. doi: 10.1080/03610927408827101.

[6] David L. Davies and Donald W. Bouldin. A cluster separation measure. *IEEE Trans. Pattern Anal. Mach. Intell.*, 1(2):224–227, February 1979. ISSN 0162-8828. doi: 10.1109/TPAMI.1979.4766909.

[7] Renato Cordeiro de Amorim and Christian Hennig. Recovering the number of clusters in data sets with noise features using feature rescaling factors. *Information Sciences*, 324:126–145, 2015. ISSN 0020-0255. doi: 10.1016/j.ins.2015.06.039. URL https://www.sciencedirect.com/science/article/pii/S0020025515004715.

[8] J. C. Dunn. Well-separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4(1):95–104, 1974. doi: 10.1080/01969727408546059.

[9] Nuno Fachada and Diogo de Andrade. Generating multidimensional clusters with support lines. *Knowledge-Based Systems*, 277:110836, 2023. ISSN 0950-7051. doi: 10.1016/j.knosys.2023.110836.

[10] Chris Fraley and Adrian E Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97(458):611–631, 2002. doi: 10.1198/016214502760047131.

[11] Bin-Bin Gao, Chao Xing, Chen-Wei Xie, Jianxin Wu, and Xin Geng. Deep label distribution learning with label ambiguity. *IEEE Transactions on Image Processing*, 26(6):2825–2838, 2017. doi: 10.1109/TIP.2017.2689998.

[12] Leo Grinsztajn, Edouard Oyallon, and Gael Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 507–520. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/0378c7692da36807bdec87ab043cdadc-Paper-Datasets_and_Benchmarks.pdf.

[13] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=cp5PvcI6w8_.

[14] 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, Jan 2025. ISSN 1476-4687. doi: 10.1038/s41586-024-08328-6.

[15] Philipp Jahn, Christian M. M. Frey, Anna Beer, Collin Leiber, and Thomas Seidl. Data with density-based clusters: A generator for systematic evaluation of clustering algorithms. In Albert Bifet, Jesse Davis, Tomas Krilavičius, Meelis Kull, Eirini Ntoutsi, and Indrė Žliobaitė, editors, *Machine Learning and Knowledge Discovery in Databases. Research Track*, pages 3–21, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-70368-3. doi: 10.1007/978-3-031-70368-3_1.

[16] Vladimir Yu Kiselev, Kristina Kirschner, Michael T Schaub, Tallulah Andrews, Andrew Yiu, Tamir Chandra, Kedar N Natarajan, Wolf Reik, Mauricio Barahona, Anthony R Green, et al. SC3: consensus clustering of single-cell RNA-seq data. *Nature methods*, 14(5):483–486, 2017. doi: 10.1038/nmeth.4236.

[17] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3744–3753. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/lee19d.html.

[18] Collin Leiber, Niklas Strauß, Matthias Schubert, and Thomas Seidl. Dying clusters is all you need - deep clustering with an unknown number of clusters. In *2024 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 726–733, 2024. doi: 10.1109/ICDMW65004.2024.00100.

[19] Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. Contrastive clustering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 8547–8555, 2021. doi: 10.1609/aaai.v35i10.17037.

[20] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, Oct 2017.

[21] Stuart Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982. doi: 10.1109/TIT.1982.1056489.

[22] Lars Lorch, Scott Sussex, Jonas Rothfuss, Andreas Krause, and Bernhard Schölkopf. Amortized inference for causal structure learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=eV4JI-MMeX.

[23] Patryk Marszałek, Tomasz Kuśmierczyk, Witold Wydmański, Jacek Tabor, and Marek Śmieja. ZEUS: Zero-shot embeddings for unsupervised separation of tabular data. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=fjBsE0kKk4.

[24] M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979. doi: 10.1080/00401706.1979.10489755.

[25] Geoffrey J. McLachlan and Kaye E. Basford. *Mixture Models: Inference and Applications to Clustering*. Statistics: Textbooks and Monographs. Marcel Dekker, New York, 1988.

[26] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: analysis and an algorithm. In *Proceedings of the 15th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, pages 849–856, Cambridge, MA, USA, 2001. MIT Press. URL https://proceedings.neurips.cc/paper_files/paper/2001/file/801272ee79cfde7fa5960571fee36b9b-Paper.pdf.

[27] Dan Pelleg and Andrew W. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 727–734, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1558607072.

[28] Girish Punj and David W. Stewart. Cluster analysis in marketing research: Review and suggestions for application. *Journal of Marketing Research*, 20(2):134–148, 1983. doi: 10.1177/002224378302000204.

[29] Mayra Z. Rodriguez, Cesar H. Comin, Dalcimar Casanova, Odemir M. Bruno, Diego R. Amancio, Luciano da F. Costa, and Francisco A. Rodrigues. Clustering algorithms: A comparative approach. *PLOS ONE*, 14(1):1–34, 2019. doi: 10.1371/journal.pone.0210236.

[30] Meitar Ronen, Shahaf E. Finder, and Oren Freifeld. DeepDPM: Deep clustering with an unknown number of clusters. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9851–9860, 2022. doi: 10.1109/CVPR52688.2022.00963.

[31] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. ISSN 0377-0427. doi: 10.1016/0377-0427(87)90125-7.

[32] Jayasree Saha and Jayanta Mukherjee. CNAK: Cluster number assisted k-means. *Pattern Recognition*, 110:107625, 2021. ISSN 0031-3203. doi: 10.1016/j.patcog.2020.107625.

[33] Robert L. Thorndike. Who belongs in the family? *Psychometrika*, 18(4):267–276, 1953. doi: 10.1007/BF02289263.

[34] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 63(2):411–423, 2001. ISSN 1369-7412. doi: 10.1111/1467-9868.00293.

[35] Talip Ucar, Ehsan Hajiramezanali, and Lindsay Edwards. SubTab: Subsetting features of tabular data for self-supervised representation learning. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=vrhNQ7aYSdr.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[37] Bo Wang, Aziz M Mezlini, Feyyaz Demir, Marc Fiume, Zhuowen Tu, Michael Brudno, Benjamin Haibe-Kains, and Anna Goldenberg. Similarity network fusion for aggregating data types on a genomic scale. *Nature methods*, 11(3):333–337, 2014. doi: 10.1038/nmeth.2810.

[38] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 478–487, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/xieb16.html.

[39] Bo Yang, Xiao Fu, Nicholas D. Sidiropoulos, and Mingyi Hong. Towards k-means-friendly spaces: simultaneous deep learning and clustering. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 3861–3870. JMLR.org, 2017.

[40] 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 H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 11033–11043. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/7d97667a3e056acab9aaf653807b4a03-Paper.pdf.

[41] Michael J Zellinger and Peter Bühlmann. Natural language-based synthetic data generation for cluster analysis. *Journal of Classification*, pages 1–27, 2025. doi: 10.1007/s00357-025-09501-w.

[42] Tianqi Zhao, Guanyang Wang, Yan Shuo Tan, and Qiong Zhang. TabClustPFN: A prior-data fitted network for tabular data clustering, 2026. URL https://arxiv.org/abs/2601.21656.

A Limitations

Generative prior scale and generalization. Although amortized inference over a synthetic prior eliminates iterative grid searches, its synthetic training prior covers a finite range: up to 2,500 objects, 200 features, and 15 clusters. While this represents a significant expansion over previous PFNs (e.g., TabClustPFN limits its domain to 1,000 objects, 64 features, and 10 clusters), the prediction head remains restricted to $k \in \{2, \ldots, 15\}$. Nevertheless, the evaluation in Appendix F.6 provides empirical evidence of extrapolation in both object and feature counts, with datasets reaching 96,320 objects and 1,776 features, processed without subsampling or truncation. On the 10 datasets with more cells than the largest training table (up to $13\times$ as many), TabK achieves an MAE of 1.00. We attribute this extrapolation in object count to the QFE, whose quantile-based encoding produces fixed-size representations by evaluating empirical quantiles at fixed probability levels.

Computational complexity of attention. A further limitation stems from the quadratic time and memory complexity of the transformer attention mechanism across both the row and column dimensions. In practice, TabK mitigates the practical computational cost on GPUs through automatic mixed-precision (AMP) (detailed in Appendix C.3), which contributes to the inference speedup reported in Figure 4b. In addition, on the primary benchmarks exceeding the training object-count ceiling, we show in Appendix F.4 that row subsampling largely preserves prediction consistency. Investigating and integrating linear attention mechanisms adapted to the tabular domain is a promising direction for extending zero-shot inference to million-row datasets.

Ground-truth ambiguity in evaluation. A fundamental challenge in evaluating unsupervised algorithms is the reliance on human-annotated class labels to define the ground-truth number of clusters, k^*. In reality, these labels frequently encode semantic or arbitrary boundaries that do not necessarily correspond to distinct, geometrically separable clusters. Consequently, the annotations themselves often fail to provide a well-defined ground truth for the clustering task. On hard datasets—where k-means used with k^* yields a low *oracle* ARI—a low MAE does not unambiguously certify that a model has recovered the correct geometric structure. It may merely indicate that the model has learned to predict the human-annotated k^*, regardless of intrinsic spatial separability. As demonstrated in Figure 3, TabK maintains bounded prediction error relative to the annotated k^* across the evaluated real-world benchmarks. Nevertheless, practitioners should interpret standard evaluation metrics cautiously; a model predicting a geometrically coherent number of clusters that differs from k^* may be more useful than one that matches the subjective annotation. We therefore recommend treating TabK's probabilistic output as a data-driven estimation of k to be validated against domain knowledge.

B Synthetic prior generation

TabK's amortized framework requires a synthetic prior $p_{\text{prior}}(\mathbf{X}, k)$ whose geometric diversity is designed to approximate the heterogeneous real-world tabular data. We therefore construct a structured prior by decoupling two complementary design decisions. First, we sample configuration triples (k, n, m) from distributions grounded in real tabular benchmarks. Second, we instantiate each configuration through five generative strategies, each targeting a distinct cluster geometry. Together, these two components ensure that TabK is exposed during training to a configuration space and a geometric diversity that jointly approximate the heterogeneous structure of real-world tabular data.

B.1 Configuration sampling

Each synthetic dataset is parameterized by a configuration triple (k, n, m), representing the number of clusters, objects, and features, respectively. To ground this prior in real-world statistics, we collected classification datasets from OpenML, filtered to those whose (k, n, m) fell within our target ranges $k \in [2, 15]$, $n \in [100, 2500]$, and $m \in [2, 200]$, and removed duplicates. We used this curated corpus to choose parametric marginals for k, n, and m that reflect the empirical distributional landscape of real tabular benchmarks.

We generated $N_{\text{cfg}} = 40{,}000$ configurations using Latin Hypercube Sampling (LHS) [24] over $[0, 1)^3$. Each uniform marginal (u_k, u_n, u_m) was mapped through the corresponding inverse CDF: a truncated Zipf distribution ($\alpha = 1.6$) for k, and log-uniform distributions for n and m, providing stratified coverage of the specified marginal distributions.

Table 3: Hyperparameter specifications and sampling distributions used within our synthetic data generation strategies.

Strategy	Parameter	Value ranges / distributions
Rodriguez et al. [29]	α	Log-Uniform $[0.05, 5.0]$
Fachada and de Andrade [9]	`direction`	Random Unit Vector
	`angle_disp`	Uniform $[0.1, 0.6]$
	`llength`	Log-Uniform $[10.0, 60.0]$
	`cluster_sep`	$(\texttt{llength}/\sqrt{m}) \times [0.1, 0.6]$ per coordinate
	`llength_disp`	`llength` $\times$ $[0.6, 1.1]$
	`lateral_disp`	Uniform $[0.5, 3.0]$
	`proj_dist_fn`	Norm (0.6), Unif (0.4)
	`point_dist_fn`	`n` (0.8), `n-1` (0.2)
Jahn et al. [15]	`radius`	Log-Uniform $[0.6, 1.8]$
	`step`	`radius` $\times$ $[0.7, 1.6]$
	`min_dist`	Uniform $[0.92, 1.06]$
	`ratio_noise`	Uniform $[0.05, 0.35]$
	`connections`	$\{0\} \cup [1, \min(6, \text{max_edges})]$
	`ratio_con`	$\{0\} \cup$ Unif $[0.03, 0.25]$
	`con_min_dist`	Uniform $[0.9, 1.05]$
	`con_step`	`step` $\times$ Unif $[0.6, 1.1]$
	`dens_factors`	Boolean ($P = 0.8$)
	`momentum`	Beta(1.5, 1.5) $\rightarrow [0.25, 0.9]$
	`distribution`	`studentt` (0.5), `gaussian` (0.3), `paper` (0.2)
Zellinger and Bühlmann [41]	`min_max_overlap`	(0.001, 0.002), (0.002, 0.25), (0.25, 0.5)
	`transform_type`	none (0.6), distort (0.2), wrap (0.2)
	`aspect_ref`	$\{1, 3\}$
	`aspect_maxmin`	$\{1, 3\}$
	`radius_maxmin`	$\{1, 3, 10\}$
	`imbalance_ratio`	$\{1, 3, 10\}$
Concentric Hyperspheres	`noise`	Uniform $[0.005, 0.1]$
	`factor`	Uniform $[0.4, 0.8]$



(a) (b) (c) (d) (e)

Figure 5: Visualization of synthetic geometries used for the generative prior ($n = 1000, m = 2, k = 5$): (a) Gaussian clusters [29], (b) anisotropic ellipsoids [41], (c) non-convex concentric shells, (d) networked manifolds [15], and (e) elongated line-supported clusters [9].

B.2 Generative strategies

To ensure TabK learns generalized structural representations, we constructed a diverse synthetic prior using a mixture of five distinct geometric generation strategies, illustrated in Figure 5. Strategy-specific hyperparameters were sampled independently per instance, ensuring that identical (k, n, m) triples generate geometrically distinct datasets. Table 3 summarizes all hyperparameter distributions.

The first strategy follows the method by Rodriguez et al. [29] that generates Gaussian clusters via cluster-specific factor matrices with random centroid offsets, where a scaling coefficient α controls the degree of separation, ranging from heavily overlapping to well-separated configurations. The

second strategy uses the Repliclust framework [41] to produce ellipsoidal clusters with anisotropic covariance, heterogeneous scales, and calibrated overlap, optionally passed through a nonlinear manifold transform. The third strategy samples objects uniformly on the m-dimensional unit hypersphere and arranges them into concentric shells of increasing radius, producing non-convex nested geometries that linear and centroid-based methods struggle to cluster. The fourth strategy employs the Densired generator [15] to construct cluster cores as random-walk graphs with optional inter-cluster bridging filaments, per-cluster density scaling, and background noise, capturing some of the complex irregular geometries common in real-world data. The fifth strategy uses PyClugen [9] to distribute objects along randomly oriented line segments in $\mathbb{R}^m$, yielding elongated cigar-shaped clusters with variable orientation and lateral scatter.

C Architecture

C.1 Proof of double-permutation invariance

In this section, we provide the formal proof of Theorem 1. We demonstrate that the TabK architecture preserves symmetry by verifying how the permutations transform during the two sequential phases of the model.

Lemma 2 (Equivariance of transformer layers)**.** *Let $T(\cdot)$ be a transformer encoder layer without positional encodings. For any permutation matrix $\mathbf{\Pi} \in \mathcal{P}_t$ acting on a sequence $\mathbf{Z} \in \mathbb{R}^{t\times d}$, the operation is permutation-equivariant: $T(\mathbf{\Pi Z}) = \mathbf{\Pi} T(\mathbf{Z})$.*

Proof. The self-attention mechanism is defined as $\text{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}})\mathbf{V}$. Since $\mathbf{Q}, \mathbf{K}$, and $\mathbf{V}$ are row-wise linear projections of $\mathbf{Z}$, permuting the rows of $\mathbf{Z}$ results in $\mathbf{\Pi Q}, \mathbf{\Pi K}$, and $\mathbf{\Pi V}$, respectively. The resulting attention map $\mathbf{\Pi A \Pi}^\top$ is permuted both row-wise and column-wise. Multiplying this by the permuted value matrix yields $\mathbf{\Pi}(\mathbf{AV})$, preserving the permutation order [36, 17]. □

Theorem 1. Let $\mathcal{P}_t$ denote the set of $t \times t$ permutation matrices. For any tabular matrix $\mathbf{X} \in \mathbb{R}^{n\times m}$, row permutation $\mathbf{\Pi}_r \in \mathcal{P}_n$, and column permutation $\mathbf{\Pi}_c \in \mathcal{P}_m$, the TabK model f_θ satisfies:

$$f_\theta(\mathbf{\Pi}_r \mathbf{X} \mathbf{\Pi}_c) = f_\theta(\mathbf{X}) \tag{6}$$

Proof. Let $\mathbf{X}$ be the input matrix transformed by row permutation $\mathbf{\Pi}_r$ and column permutation $\mathbf{\Pi}_c$. We denote the transformed input as $\mathbf{X}' = \mathbf{\Pi}_r \mathbf{X} \mathbf{\Pi}_c$.

Phase 1: Feature identity and row compression. First, we consider **Component 1 (QFE)**. The empirical quantiles $\mathbf{q}_j$ are calculated over the set of values in column j. Since the set $\{x_{1j}, \ldots, x_{nj}\}$ is invariant to the row-ordering i, the column identities $\mathbf{c}_j$ are invariant to $\mathbf{\Pi}_r$. However, permuting the columns by $\mathbf{\Pi}_c$ reorders the resulting set of identities. Thus, the matrix of identities $\mathbf{C} \in \mathbb{R}^{m\times d}$ is transformed as follows: $\mathbf{C}' = \mathbf{\Pi}_c^\top \mathbf{C}$.

For **Component 2 (inter-feature attention)**, by Lemma 2, the refined identities $\hat{\mathbf{C}}$ remain column-equivariant: $\hat{\mathbf{C}}' = \mathbf{\Pi}_c^\top \hat{\mathbf{C}}$.

For **Component 3 (value-identity fusion and row aggregation)**, the cell embeddings are defined as $\mathbf{e}_{ij} = (\mathbf{w}_v x_{ij} + \mathbf{b}_v) + \hat{\mathbf{c}}_j$. Under both permutations, the transformed cell embedding becomes:

$$\mathbf{e}'_{ij} = \mathbf{e}_{\sigma_r(i),\sigma_c(j)}, \tag{7}$$

where σ_r and σ_c are the permutations associated with $\mathbf{\Pi}_r$ and $\mathbf{\Pi}_c$. The row embedding $\mathbf{r}_i$ is then generated by mean pooling, which is a symmetric function over the set $\mathcal{F}_i$; thus, it is invariant to the column permutation σ_c. Consequently, the row embedding vector $\mathbf{r}_i$ is invariant to $\mathbf{\Pi}_c$. The resulting sequence of row embeddings $\mathbf{R} \in \mathbb{R}^{n\times d}$ satisfies:

$$\mathbf{R}(\mathbf{\Pi}_r \mathbf{X} \mathbf{\Pi}_c) = \mathbf{\Pi}_r \mathbf{R}(\mathbf{X}). \tag{8}$$

Phase 2: Dataset-level structural representation. In **Component 4 (row set interaction)**, the row transformer processes $\mathbf{R}$. By Lemma 2, the interaction is row-equivariant, yielding: $\hat{\mathbf{R}}' = \mathbf{\Pi}_r \hat{\mathbf{R}}$.

Finally, in **Component 5 (Pooling by Multihead Attention)**, the global summary $\mathbf{z}$ is computed using a learnable seed vector $\mathbf{s}$. As the attention mechanism computes a weighted sum over the input tokens (rows) based on their similarity to the query $\mathbf{s}$, the output is invariant to the ordering of the input rows [17]. Therefore, $\text{PMA}(\mathbf{\Pi}_r \hat{\mathbf{R}}) = \text{PMA}(\hat{\mathbf{R}})$.

Combining these steps, we have $f_\theta(\mathbf{\Pi}_r \mathbf{X} \mathbf{\Pi}_c) = f_\theta(\mathbf{X})$, which completes the proof. □

To empirically validate these theoretical guarantees, we evaluated the model across 500 randomized permutations of rows and columns. Across all trials, the predicted number of clusters remained identical, with maximum absolute differences in output logits below 10^{-6}.

C.2 Model configuration

The selected TabK architecture uses an embedding dimension of $d = 256$ with $h = 8$ attention heads. The QFE encodes 50 evenly spaced quantiles per column, processed by a two-layer MLP to produce column identity vectors $\mathbf{c}_j \in \mathbb{R}^d$. A transformer encoder with a feed-forward dimension of $4d$ refines these identities via inter-column self-attention. The row-level transformer encoder uses 4 layers with identical hyperparameters. Dropout is set to 0.3 throughout the backbone. PMA pooling uses $h = 8$ heads, followed by LayerNorm. The prediction head applies a LayerNorm to $\mathbf{z}$, followed by a two-layer MLP with a GELU activation and an intermediate dropout of 0.2, mapping to a distribution over the 14 candidate values of $k \in \{2, \ldots, 15\}$. The logits are passed through log-softmax, and the training loss is the DLDL objective $\mathcal{L} = \text{KL}(q_{k^*} \,\|\, \hat{p})$ of Section 3.2, with $\sigma = 0.5$. The full TabK model contains 4.4M trainable parameters.

We additionally evaluated four alternative prediction heads in the ablation study (see Table 2). The classification head applies standard cross-entropy loss over one-hot targets. The Focal Loss head uses cross-entropy with $(1 - p_y)^\gamma$ modulation [20]. The ordinal head employs a cumulative threshold model, where each of the 13 binary logits predicts $P(k > t)$. Finally, the regression head produces a single scalar output trained with Huber loss ($\delta = 1.0$). For the classification and focal heads, predictions are decoded via argmax over raw logits. For the regression head, the scalar output is rounded and clamped to the valid range. For the ordinal head, predictions are obtained by counting the number of cumulative probabilities exceeding 0.5 and adding $k_{\min} = 2$. All heads share the same MLP backbone, differing only in output dimensionality and loss function.

C.3 Training and inference protocol

Training uses 5-fold cross-validation on the training split of the synthetic prior. Each fold trains a fresh model for up to 20 epochs with early stopping (patience of 5 epochs, monitored on validation MAE). The best checkpoint per fold is restored after training completes. We use AdamW with a learning rate of 2×10^{-4} and a weight decay of 10^{-3}. Weight decay is applied exclusively to parameters with dimensionality greater than 1 (i.e., weight matrices); biases and LayerNorm scales are exempt. Gradient accumulation over 8 micro-batches of 16 samples yields an effective batch size of 128. A two-phase schedule is employed for the learning rate. A linear warmup from 1% of the base learning rate over $\max(100, 0.05 \times T)$ steps, where T is the total number of optimizer steps, is followed by cosine annealing to 10% of the base rate over the remaining steps. These two phases are composed via a sequential scheduler. To mitigate the quadratic time and memory complexity of the attention mechanism, training uses automatic mixed-precision (AMP) with BFloat16 on Ampere or newer GPUs, while GPU inference uses AMP with Float16. At inference time, the checkpoints are loaded as an ensemble. Each loaded model independently produces a log-probability vector in a single forward pass. The 5 output vectors are averaged in log-space and renormalized, and the final prediction $\bar{k}$ is the argmax of the averaged output. No explicit calibration or temperature scaling is applied.

C.4 Computational resources

Training was conducted on a single NVIDIA A100-SXM4-40GB GPU using Python 3.13 and PyTorch 2.9.1+cu126. The final 5-fold training run completed in 135,262 seconds (≈37.6 hours).

Table 4: Hyperparameter search space. Bold values denote the selected configuration for the final TabK model.

Hyperparameter	Symbol	Values
DLDL bandwidth	σ	0.1, 0.25, $\mathbf{0.5}$, 1.0
Learning rate	η	10^{-4}, $\mathbf{2 \times 10^{-4}}$, 5×10^{-4}
Embedding dimension	d_{model}	128, $\mathbf{256}$, 512
QFE quantile bins	b	25, $\mathbf{50}$, 100
Dropout rate	p_{drop}	0.1, $\mathbf{0.3}$, 0.5
Attention heads	n_{head}	4, $\mathbf{8}$, 16
Transformer layers	n_{layers}	2, $\mathbf{4}$, 6
Weight decay	λ	$\mathbf{10^{-3}}$, 5×10^{-3}

Following the methodology of ML CO2 Impact, we estimate the carbon footprint as follows. The A100-SXM4-40GB has a thermal design power of 400W. Assuming a power usage effectiveness (PUE) of 1.1 and the average carbon intensity of 0.39 kg CO_2/kWh, the final model consumed approximately 16.5 kWh ($\approx$6.4 kg CO_2).

C.5 Hyperparameter selection

Hyperparameter optimization was conducted via grid search (see Table 4), with model selection determined by validation MAE across a 5-fold cross-validation scheme. The quantile resolution of the QFE directly affects the model's structural representation capacity. Empirically, 50 quantile bins achieve the best performance-efficiency trade-off, suggesting that 25 bins may be insufficient to capture fine-grained distributional structure, and 100 bins introduce computational overhead without improving accuracy. Similarly, the DLDL bandwidth $\sigma = 0.5$ consistently outperforms both narrower ($\sigma = 0.1$) and wider ($\sigma = 1.0$) targets. This suggests that a moderate degree of ordinal smoothing provides the ideal bias-variance trade-off, proportionally penalizing predictions without excessively flattening the target distribution.

D Benchmark dataset specifications and profiles

To rigorously evaluate TabK's generalization capacity across heterogeneous tabular geometries, we constructed a comprehensive benchmark suite through a systematic selection process from the OpenML database [4]. To align the evaluation with our target operational window while explicitly testing the model's extrapolation capacity beyond the training object-count ceiling, datasets were restricted to classification tasks with a maximum of 10,000 objects, 200 features, and 15 classes (which serve as the ground-truth number of clusters, k^*). We subsequently removed duplicate entries and excluded all datasets used in the structural parameter analysis (see Appendix B.1) to prevent distributional leakage into the configuration prior, yielding a final evaluation suite of 50 datasets. The complete specifications for the benchmark suite are summarized in Table 5.

E Preprocessing pipeline

To ensure a fair comparison across all evaluated baselines, many of which lack native mechanisms for handling incomplete or non-numeric data, we applied a standard preprocessing pipeline to all datasets. Categorical variables were transformed via one-hot encoding (OHE), missing values were replaced using median imputation, and features were standardized to zero mean and unit variance. While deep tabular networks frequently require specialized embedding tables or explicit missingness indicators to process these artifacts without degradation, TabK captures the statistical signatures introduced by standard preprocessing through its Quantile Feature Encoder (QFE). For an OHE column, the sorted quantiles manifest as a distinct two-level step function, explicitly encoding the empirical category frequency as the column's structural identity. Similarly, median imputation introduces an artificial point mass at the median value. In the QFE, this manifests as a flat plateau within the sampled quantiles. By passing this distributional anomaly through the statistical encoder, TabK enables the subsequent attention layers to identify distributional signatures of heavily imputed features. This reduces reliance on explicit missingness indicators while preserving the structural signal introduced

Table 5: Summary of the 50 real-world benchmarks used in our evaluation. OpenML dataset IDs are provided for reproducibility. n: number of objects, m: number of features, k: ground-truth number of clusters.

ID	Name	Objects (n)	Features (m)	k	#Missing	#Categorical
2	anneal	898	38	5	22175	32
3	kr-vs-kp	3196	36	2	0	36
4	labor	57	16	2	326	8
10	lymph	148	18	4	0	15
14	mfeat-fourier	2000	76	10	0	0
15	breast-w	699	9	2	16	0
22	mfeat-zernike	2000	47	10	0	0
28	optdigits	5620	64	10	0	0
29	credit-a	690	15	2	67	9
31	credit-g	1000	20	2	0	13
36	segment	2310	19	7	0	0
43	haberman	306	3	2	0	1
44	spambase	4601	57	2	0	0
50	tic-tac-toe	958	9	2	0	9
54	vehicle	846	18	4	0	0
55	hepatitis	155	19	2	167	13
56	vote	435	16	2	392	16
59	ionosphere	351	34	2	0	0
61	iris	150	4	3	0	0
182	satimage	6430	36	6	0	0
187	wine	178	13	3	0	0
299	libras_move	360	90	15	0	0
694	diggle_table_a2	310	8	9	0	0
721	pwLinear	200	10	2	0	0
733	machine_cpu	209	6	2	0	0
960	postoperative-patient-data	90	8	2	3	8
1049	pc4	1458	37	2	0	0
1050	pc3	1563	37	2	0	0
1063	kc2	522	21	2	0	0
1067	kc1	2109	21	2	0	0
1068	pc1	1109	21	2	0	0
1462	banknote-authentication	1372	4	2	0	0
1464	blood-transfusion	748	4	2	0	0
1465	breast-tissue	106	9	6	0	0
1488	parkinsons	195	22	2	0	0
1494	qsar-biodeg	1055	41	2	0	0
1499	seeds	210	7	3	0	0
1510	wdbc	569	30	2	0	0
1523	vertebra-column	310	6	3	0	0
1600	SPECTF	267	44	2	0	0
4153	Smartphone	180	67	6	0	1
6332	cylinder-bands	540	39	2	999	21
40682	thyroid-new	215	5	3	0	0
40691	wine-quality-red	1599	11	6	0	0
40701	churn	5000	20	2	0	4
40975	car	1728	6	4	0	6
40983	wilt	4839	5	2	0	0
43922	mushroom	8124	22	2	0	22
46592	cervical-cancer	858	32	5	3622	0
47154	vowel	990	10	11	0	0

Table 6: TabK performance on the 10% synthetic holdout set, stratified by generative strategy. The model demonstrates competitive exact recovery and ± 1 Tolerance ($\tau \leq 1$) across diverse geometric domains.

Generative strategy	**MAE** $\downarrow$	**Exact Match (%)** $\uparrow$	$\tau \leq 1$ **(%)** $\uparrow$
Concentric Hyperspheres	1.17	47.3	73.6
Rodriguez et al. [29]	1.27	52.3	73.6
Fachada and de Andrade [9]	1.56	41.5	66.3
Jahn et al. [15]	1.76	43.8	65.3
Zellinger and Bühlmann [41]	2.05	42.3	61.8
Overall synthetic holdout	**1.56**	**45.5**	**68.2**

by imputation. We evaluated the model's performance in our stratified performance analysis as shown in Figure 7, where TabK maintains a consistently low error profile even on datasets with missing values and categorical features.

F Extended experimental results

F.1 Performance on synthetic holdout sets

While our real-world benchmarks establish that TabK generalizes beyond its training distribution, we evaluated its performance on a synthetic holdout set of 20,000 datasets (a 10% split of the total generated prior) to isolate its performance across specific geometric topologies. As shown in Table 6, TabK achieves an overall MAE of 1.56 and an Exact Match rate of 45.5%. These metrics are consistent with, though slightly below, the real-world OpenML results (MAE 1.04, 50.0% Exact Match). We attribute this difference in part to the distribution shift of k: the synthetic holdout samples k from a Zipf prior over [2, 15], whereas the real-world benchmarks are skewed toward smaller k^*, producing a less challenging marginal distribution for inference. Across synthetic datasets used in our experiments, performance degrades most on the Zellinger and Bühlmann [41] data (MAE 2.05) that encompasses non-linear manifold transformations warping standard anisotropic overlaps into highly complex, non-convex boundaries. Crucially, the ± 1 Tolerance ($\tau \leq 1$) column shows that predictions fall within ± 1 of the true k in 61.8–73.6% of cases across all geometric domains.

F.2 Stratified performance analysis

Figure 6 presents the results of structural sensitivity analysis in which MAE is stratified by the number of clusters, objects, and features. TabK maintains a consistent error profile across all three dimensions, achieving the MAE values of 1.05 and 1.00 for low and high numbers of clusters, 0.94 and 1.22 for small and large object counts, and 1.00 and 1.67 for low and high feature counts. The modest degradation in high-feature regimes is anticipated, as high-dimensional spaces attenuate pairwise separability signals; however, TabK's QFE mitigates this by operating on marginal quantile representations rather than pairwise distances, retaining the best high-feature MAE among all evaluated methods. TabClustPFN is most sensitive to object count, degrading to an MAE of 2.56 on small datasets, likely because its assignment-based cardinality predictor requires a sufficient object density to form stable cluster assignments.

In Figure 7, MAE is stratified by dataset feature composition across four distinct regimes: complete numerical data, numerical data with missing values, data containing categorical features (without missing values), and the highly challenging mixed regime containing both. TabK maintains a consistently low error profile across all four settings. On categorical inputs, one-hot encoding produces characteristic sparsity patterns that the QFE detects via point masses in the empirical distribution. Consequently, even in the hardest mixed regime, where both sparsity patterns and imputation plateaus co-occur, TabK successfully encodes these co-occurring structural signatures and maintains a low error level.

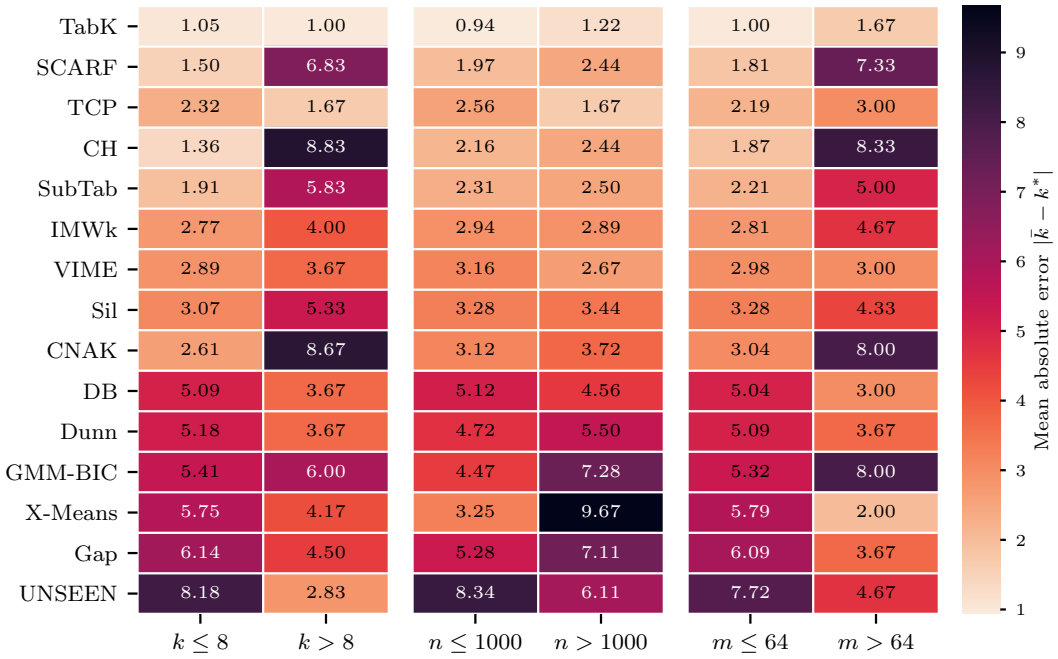


Figure 6: Structural sensitivity analysis. MAE stratified by ground-truth number of clusters (k), objects (n), and features (m). TabK maintains consistent performance across all structural regimes.

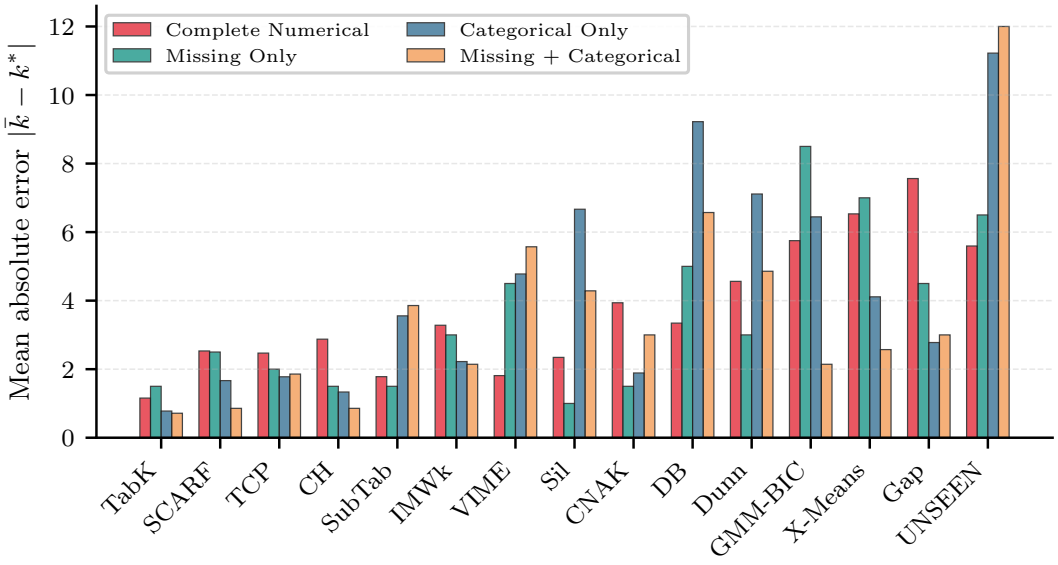


Figure 7: MAE stratified by the missing values and categorical features. TabK exhibits a bounded and consistent error profile across all evaluated domains compared to baseline methods and state-of-the-art TabClustPFN.

F.3 Prior-constrained domain evaluation

A potential confounding factor when comparing PFNs is the scale of their respective synthetic training distributions. TabClustPFN is trained on a restricted prior bounded by $n \leq 1000$, $m \leq 64$, and $k \leq 10$. To verify that TabK's advantage reported in Table 1 persists within TabClustPFN's training-scale upper bounds, we isolated the 27 OpenML datasets in our benchmark that satisfy these upper bounds for a direct head-to-head comparison. As shown in Table 7, TabK outperforms TabClustPFN across all metrics even within this restricted sub-domain. For instance, TabK reduces

Table 7: Head-to-head evaluation restricted to TabClustPFN's training-scale upper bounds ($n \leq 1000$, $m \leq 64$, $k \leq 10$), comprising 27 OpenML benchmarks.

Method	MAE ↓	ARI (k-means) ↑	EM (%) ↑	$\tau \leq 1$ (%) ↑	Latency (s) ↓
TabK	**0.78**	**0.269**	**55.6**	**85.2**	**0.08**
TabClustPFN	2.56	0.235	29.6	59.3	0.50

Table 8: Evaluation of TabK's structural inference stability under uniform row subsampling. Results are aggregated over 15 independent trials for each sample size. The metrics demonstrate that TabK structural inference remains stable across a wide range of sample sizes.

Sample size	Exact Match rate vs. Full	Mean $\lvert\Delta k\rvert$ vs. Full	Latency (ms)
$N = 2500$	94.3%	0.16	33.91
$N = 2000$	89.5%	0.18	25.83
$N = 1000$	90.5%	0.34	20.44
$N = 500$	86.7%	0.34	20.96
$N = 250$	75.2%	0.62	19.27
$N = 100$	61.9%	0.73	19.20

MAE from 2.56 to 0.78 (a 70% reduction) and achieves an Exact Match rate of 55.6%, compared to 29.6% for TabClustPFN. These results indicate that the performance gap is not explained by evaluation beyond TabClustPFN's training-scale upper bounds.

F.4 Stability under uniform subsampling

In Appendix A, we noted that TabK's reliance on full-matrix self-attention imposes an $\mathcal{O}(n^2)$ time and memory complexity. However, because the number of clusters, k, is inherently a global property rather than an instance-level artifact, the model does not require observing every object to infer the latent structure. To validate this, we evaluated TabK's prediction consistency under uniform row subsampling on the subset of benchmark datasets exceeding the training prior ceiling ($n > 2,500$), across 15 independent draws per sample size. As detailed in Table 8, TabK exhibits high stability under moderate-to-aggressive subsampling rates. When uniformly subsampling the input matrix to $N = 2500$ rows, TabK recovers the same k as the full dataset in 94.3% of trials. This empirical robustness demonstrates that practitioners can apply uniform row subsampling to mitigate the $\mathcal{O}(n^2)$ attention bottleneck with only modest reductions in prediction consistency, provided a sufficient sample size is retained.

F.5 Robustness to noisy features and outliers

We evaluated TabK on all 50 real-world benchmarks under two perturbations that leave the annotated k^* unchanged, averaging each nonzero level over five seeds. For noisy features, we append independent standard Gaussian columns equal to 25%, 50%, or 100% of the original feature count. For diffuse outliers, we append points equal to 1%, 5%, or 15% of the sample size, drawn independently per feature from the observed range extended by 25% on either side. Each perturbed dataset is then standardized. Table 9 reports the MAE and the agreement with TabK's prediction on the clean data. Even when the feature count is doubled with uninformative columns, TabK's MAE (1.94) remains below the clean-data MAE of every competing method in Table 1.

F.6 Evaluation beyond the training scale

To evaluate TabK beyond the primary benchmark, we assembled 25 datasets from the OpenML-CC18 suite [4], after removing every dataset in the primary benchmark and every dataset used to estimate the prior's (k, n, m) marginals. The suite spans 500 to 96,320 objects and 5 to 1,776 features; the 10 datasets above 500,000 cells exceed the largest synthetic training table ($2{,}500 \times 200$) by up to $13\times$. Each dataset is processed at its full size, without subsampling or truncation. Table 10 summarizes the results and Table 11 lists the per-dataset predictions. On both subsets, TabK improves on TabClustPFN in every reported metric.

Table 9: TabK under noisy features and diffuse outliers on the 50 benchmarks (mean over five seeds). Agreement is the percentage of datasets on which the prediction matches the prediction on clean data.

Perturbation	Level	MAE ↓	Agreement (%) ↑
None (clean)	–	1.04	100.0
Noisy features	+25% of m	1.43	86.8
	+50% of m	1.56	85.6
	+100% of m	1.94	76.8
Diffuse outliers	1% of n	1.30	80.4
	5% of n	1.43	68.0
	15% of n	1.61	67.6

Table 10: Evaluation beyond the primary benchmark. OpenML-CC18: 25 datasets disjoint from the primary benchmark and from the datasets used to estimate the prior's (k, n, m) marginals; the >500k subset exceeds the largest synthetic training table.

Subset	Method	MAE ↓	EM (%) ↑	$\tau \leq 1$ (%) ↑	Brier ↓	NLL ↓	Latency (s) ↓
CC18, all (25)	**TabK**	**1.56**	**44.0**	**60.0**	**0.718**	**2.106**	**0.70**
	TCP	2.60	40.0	56.0	0.897	5.274	4.31
CC18, >500k cells (10)	**TabK**	**1.00**	**60.0**	**80.0**	**0.542**	**1.238**	**1.66**
	TCP	2.00	50.0	70.0	0.904	6.180	9.88

F.7 Failure analysis

While TabK consistently improves structural recovery compared to evaluated methods, an analysis of its failure modes provides valuable insight into the inherent ambiguity of tabular clustering. By examining the datasets where TabK registers its highest absolute errors, we observe that these failures in part stem from a misalignment between human-annotated class labels and the intrinsic spatial geometry of the data (see Appendix A). We categorize these failure instances into two distinct phenomena.

Hierarchical ambiguity. In datasets with highly granular human annotations, the most mathematically robust geometric partitions often do not align with the fine-grained label structure. For example, on the *Smartphone* dataset (`ID=4153`), the annotated number of clusters is $k^* = 6$ (representing specific movements like walking, sitting, and standing). TabK assigns its highest confidence (51.9%) to $k = 2$. While evaluated as an error of 4, an inspection of the baseline algorithms reveals that the Calinski–Harabasz index, the Davies–Bouldin index, the Silhouette width, and the deep baseline TabClustPFN model all unanimously predict $k = 2$. This indicates that the global geometry is dominated by two highly separable macro-clusters (e.g., active versus passive movements). Consistent with these baselines, TabK identifies the dominant macro-level partition rather than fine-grained sub-class boundaries.

Unresolvable geometries and flat posteriors. The second category of error occurs in datasets that appear to lack clear spatial separability. For instance, on the *vehicle* (`ID=54`, $k^* = 4$, predicted $\bar{k} = 8$) and *wine-quality-red* (`ID=40691`, $k^* = 6$, predicted $\bar{k} = 2$) datasets, TabK fails to recover the ground truth. However, the DLDL predictive head provides a crucial diagnostic advantage over classical heuristics: it outputs a probability mass function. For the *vehicle* dataset, TabK's maximum predicted confidence is merely 10.7%, whereas it is 12.1% for the *wine-quality-red* dataset, with the probability mass distributed almost uniformly across k. Rather than generating an overconfident exact prediction, TabK's output distribution reflects the uncertainty of these estimates.

F.8 Full per-dataset results

To ensure complete empirical transparency, Table 12 details the exact numbers of clusters, $\bar{k}$, predicted by TabK and all other evaluated methods across the 50 OpenML benchmarks considered.

Table 11: Per-dataset predictions $\bar{k}$ on the 25 OpenML-CC18 datasets used for the scale evaluation, ordered by table size. Bold entries indicate exact recovery; underlined entries indicate $|\bar{k} - k^*| = 1$.

ID	Dataset	n	m	Cells	k^*	TabK	TCP
1480	ilpd	583	12	6,996	2	**2**	**2**
469	analcatdata_dmft	797	9	7,173	6	2	10
40994	climate-model-simulation-crashes	540	18	9,720	2	**2**	**2**
23	cmc	1,473	9	13,257	3	7	<u>2</u>
1489	phoneme	5,404	5	27,020	2	4	9
40982	steel-plates-fault	1,941	27	52,407	7	2	10
188	eucalyptus	736	96	70,656	5	2	2
23381	dresses-sales	500	167	83,500	2	**2**	**2**
1497	wall-robot-navigation	5,456	24	130,944	4	8	10
1487	ozone-level-8hr	2,534	72	182,448	2	5	8
40499	texture	5,500	40	220,000	11	9	2
1053	jm1	10,885	21	228,585	2	**2**	<u>3</u>
38	sick	3,772	74	279,128	2	**2**	**2**
4534	PhishingWebsites	11,055	30	331,650	2	<u>3</u>	**2**
151	electricity	45,312	8	362,496	2	<u>3</u>	7
40670	dna	3,186	180	573,480	3	**3**	<u>2</u>
1468	cnae-9	1,080	856	924,480	9	14	2
46	splice	3,190	347	1,106,930	3	<u>2</u>	<u>2</u>
23517	numerai28.6	96,320	21	2,022,720	2	**2**	10
1461	bank-marketing	45,211	60	2,712,660	2	**2**	**2**
1486	nomao	34,465	118	4,066,870	2	<u>3</u>	**2**
40978	Internet-Advertisements	3,279	1,558	5,108,682	2	**2**	**2**
1590	adult	48,842	113	5,519,146	2	**2**	**2**
1478	har	10,299	561	5,777,739	6	3	3
4134	Bioresponse	3,751	1,776	6,661,776	2	**2**	**2**

G Baseline implementation details

To ensure a rigorous and reproducible comparison, we provide comprehensive implementation details for the 17 competing methods evaluated in our study (see Table 1). All baselines were evaluated over the same candidate range $k \in \{2, \ldots, 15\}$, except TabClustPFN, whose pretrained output range is limited to $k \leq 10$ (see Appendix F.3 for a comparison within its domain), using a fixed random seed.

No method was tuned per dataset or against the real-world annotations: baselines used the configurations recommended by their authors where available, with identical preprocessing. For the four internal cluster validity indices, first, k-means (`scikit-learn`, `n_init=100`, `k-means++` initialization) is executed for every candidate k, producing a set of label vectors $\{\hat{\mathbf{y}}_k\}$. Each CVI is then computed on the data–label pairs $(\mathbf{X}, \hat{\mathbf{y}}_k)$. We used the `scikit-learn` implementations for the Silhouette, Calinski–Harabasz, and Davies–Bouldin indices. We used our own implementation of the Dunn index. Additionally, we implemented the Gap statistic [34] and GMM-BIC [10] according to their original specifications, and integrated the author-provided MATLAB implementation of IMWk [7] into our benchmarking pipeline. All implementations and scripts are available in our repository at `github.com/mrbakhtyari/TabK`.

Specific details for the remaining methods are as follows:

- **X-Means** [27]: We used the `PyClustering` implementation with the k`-means++` initialization, and the `BIC` splitting criterion.
- **CNAK** [32]: We used the official source code at: Jayasree-Saha/CNAK.
- **UNSEEN** [18]: We used the official source code at: collinleiber/UNSEEN.
- **TabClustPFN** [42]: We used the official source code at: Tianqi-Zhao/TabClustPFN.
- **VIME, SCARF, and SubTab** [40, 2, 35]: We trained each encoder on the preprocessed dataset and applied the same Silhouette and Calinski–Harabasz selection routine and k-means setting (`n_init=100`) to the learned embeddings as for the raw-feature CVIs.

Table 12: Predicted numbers of clusters, $\bar{k}$, reported for the 50 OpenML benchmarks. OpenML dataset IDs are indicated. Methods are ordered by the MAE performance (best first). Bold entries indicate exact recovery ($\bar{k} = k^*$). Underlined entries indicate a prediction with $\tau \leq 1$ ($|\bar{k} - k^*| = 1$).

ID	k^*	TabK	SCARF+CH	TCP	CH	SubTab+Sil	SCARF+Sil	IMWk	VIME+Sil	VIME+CH	Sil	CNAK	SubTab+CH	DB	Dunn	GMM-BIC	X-Means	Gap	UNSEEN
2	5	2	**5**	2	3	15	3	4	15	2	8	2	14	12	4	**5**	15	2	13
3	2	**2**	4	**2**	**2**	9	**2**	4	**2**	**2**	7	**2**	3	15	13	5	15	**2**	15
4	2	3	**2**	10	**2**	**2**	3	4	**2**	**2**	15	**2**	**2**	15	14	4	**2**	3	15
10	4	3	3	2	2	3	**4**	**4**	2	2	2	2	5	14	7	3	2	2	15
14	10	**10**	3	**10**	2	2	2	8	15	2	2	3	2	15	3	3	15	15	15
15	2	**2**	5	3	**2**	**2**	**2**	7	**2**	3	**2**	**2**	**2**	**2**	5	15	6	8	11
22	10	9	4	**10**	2	3	2	9	7	2	7	2	2	14	8	5	15	15	14
28	10	13	6	**10**	2	13	**10**	6	9	2	14	2	13	14	13	11	15	2	12
29	2	**2**	3	3	3	**2**	**2**	5	5	**2**	**2**	3	3	**2**	**2**	6	4	8	15
31	2	**2**	4	**2**	**2**	**2**	6	7	**2**	**2**	15	**2**	**2**	15	12	7	3	**2**	15
36	7	2	13	4	2	6	4	**7**	4	2	**7**	2	6	6	14	14	15	2	10
43	2	**2**	4	**2**	5	**2**	4	7	15	15	4	5	15	14	12	8	**2**	5	7
44	2	**2**	**2**	**2**	**2**	**2**	**2**	6	3	**2**	**2**	4	**2**	**2**	**2**	12	15	15	10
50	2	3	**2**	10	**2**	8	**2**	3	13	**2**	14	9	**2**	14	14	14	**2**	12	15
54	4	8	**4**	8	2	2	**4**	6	2	2	2	2	**4**	2	**4**	7	15	15	10
55	2	**2**	4	**2**	3	**2**	7	6	14	**2**	**2**	12	3	**2**	**2**	**2**	4	3	15
56	2	**2**	3	3	**2**	3	9	**2**	**2**	**2**	3	**2**	3	3	13	8	3	8	14
59	2	3	**2**	10	**2**	4	**2**	4	3	**2**	4	**2**	**2**	6	**2**	10	3	10	11
61	3	**3**	2	4	2	5	2	5	2	2	2	2	15	2	2	2	7	5	15
182	6	11	3	2	3	3	2	**6**	3	4	3	2	4	3	15	5	15	15	11
187	3	5	7	4	**3**	**3**	7	5	2	2	**3**	**3**	**3**	**3**	13	15	**3**	4	15
299	15	14	4	10	2	12	2	6	**15**	2	14	2	5	**15**	**15**	2	**15**	**15**	**15**
694	9	**9**	2	5	2	2	2	5	3	15	2	2	15	3	3	15	15	15	14
721	2	3	3	10	**2**	**2**	3	**2**	**2**	**2**	14	**2**	**2**	14	**2**	5	**2**	**2**	15
733	2	**2**	8	**2**	**2**	**2**	6	6	**2**	15	**2**	**2**	14	12	6	11	4	11	3
960	2	3	**2**	**2**	3	15	9	3	15	**2**	15	8	3	15	12	3	**2**	**2**	14
1049	2	3	4	**2**	**2**	**2**	**2**	5	**2**	**2**	**2**	**2**	**2**	10	15	10	15	15	7
1050	2	3	4	4	3	**2**	4	7	7	**2**	**2**	12	9	**2**	**2**	12	15	15	7
1063	2	**2**	3	3	6	4	3	4	3	7	**2**	15	15	**2**	**2**	6	15	15	7
1067	2	**2**	**2**	4	**2**	**2**	**2**	6	**2**	3	**2**	**2**	15	**2**	**2**	12	15	15	**2**
1068	2	**2**	**2**	**2**	3	**2**	**2**	8	3	**2**	**2**	15	8	3	**2**	11	15	15	4
1462	2	3	3	9	5	**2**	**2**	8	3	15	15	6	15	13	14	15	15	7	10
1464	2	**2**	3	10	13	3	**2**	7	3	15	**2**	8	5	14	8	10	13	13	4
1465	6	2	3	**6**	7	3	3	5	3	4	2	2	8	3	3	11	5	10	14
1488	2	**2**	**2**	8	3	3	**2**	6	**2**	4	**2**	5	3	**2**	14	3	6	15	10
1494	2	**2**	**2**	10	**2**	**2**	**2**	6	9	**2**	**2**	**2**	**2**	15	15	12	15	15	9
1499	3	**3**	**3**	5	2	2	4	7	**3**	**3**	2	2	**3**	2	11	4	4	7	15
1510	2	**2**	3	**2**	**2**	**2**	3	6	**2**	**2**	**2**	**2**	**2**	**2**	15	4	10	15	7
1523	3	2	**3**	2	2	2	2	6	**3**	2	2	4	2	4	6	**3**	2	8	7
1600	2	**2**	**2**	**2**	**2**	**2**	**2**	5	**2**	**2**	**2**	**2**	**2**	**2**	**2**	4	3	12	14
4153	6	2	2	2	2	2	2	3	2	2	2	2	2	2	2	2	7	12	15
6332	2	**2**	4	**2**	3	5	4	6	3	**2**	**2**	3	**2**	14	**2**	4	5	6	15
40682	3	2	2	2	**3**	2	2	7	**3**	15	**3**	**3**	6	5	5	4	**3**	8	7
40691	6	2	2	4	2	4	2	7	2	2	2	2	2	**6**	14	9	11	15	10
40701	2	**2**	3	**2**	**2**	**2**	6	**2**	**2**	**2**	**2**	**2**	**2**	7	**2**	7	8	6	15
40975	4	3	5	2	3	3	15	**4**	3	3	15	3	2	15	15	14	3	**4**	15
40983	2	**2**	5	**2**	**2**	**2**	**2**	8	3	3	**2**	3	3	**2**	**2**	9	15	**2**	4
43922	2	**2**	4	**2**	4	15	3	6	14	4	13	**2**	15	5	5	14	15	**2**	15
46592	5	2	3	2	2	2	3	6	14	**5**	3	2	2	15	2	9	15	2	9
47154	11	10	5	10	2	4	5	7	4	4	2	2	4	14	15	15	15	14	12
MAE		**1.04**	2.14	2.24	2.26	2.38	2.50	2.92	2.98	3.08	3.34	3.34	3.72	4.92	5.00	5.48	5.56	5.94	7.54

We used the official implementations of VIME (jsyoon0823/VIME) and SubTab (AstraZeneca/SubTab), and a public implementation of SCARF (clabrugere/pytorch-scarf).

H Broader impacts

TabK improves the efficiency of unsupervised learning for tabular data, reducing the computational cost and potentially the energy consumption of standard clustering pipelines. However, automating cluster discovery carries potential societal risks if applied to sensitive human-centric data without oversight. We urge practitioners to manually audit the resulting clusters for fairness and privacy implications before deploying them in sensitive domains.

I Asset licenses

All datasets and codebases used in this work are publicly available and properly credited; datasets and code are used under the terms of their sources. Identified software licenses include BSD-3-Clause (scikit-learn, PyClustering, ClustPy, and repliclust), MIT (PyClugen, DENSIRED, and the SCARF implementation), Apache-2.0 (SubTab), and CC BY-NC 4.0 (TabClustPFN).

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Section 1 clearly state the primary contributions, including the TabK architecture, QFE, and empirical results (MAE, ARI, latency) on real-world datasets.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We comprehensively discuss the limitations of our work in Appendices A and F.7, including constraints on the generative prior scale, the computational complexity of attention, and ground-truth ambiguity.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Theorem 1 (Double-Permutation Invariance) is stated in Section 3.1.3, and the formal, complete mathematical proof is provided in Appendix C.1.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Implementation details for all baselines are provided in Appendix G. Model configuration, training protocol, and hyperparameter selection are detailed in Appendix C, synthetic data generation in Appendix B, and benchmark datasets and preprocessing in Appendices D and E. Code and pretrained model weights are publicly available (see the abstract).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is publicly available at `github.com/mrbakhtyari/TabK`, including the model, synthetic data generation, training, implementations of the classical baselines, and the benchmark results together with the analysis code that regenerates the main experimental tables and figures. Pretrained weights are hosted on the Hugging Face Hub, and the OpenML dataset IDs are listed in Table 5; the remaining baselines use the official implementations linked in Appendix G.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Training details are in Appendix C. Dataset specifications and preprocessing pipelines are detailed in Appendices D and E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report Wilcoxon signed-rank test p-values in Table 1 to establish the statistical significance of TabK's performance gains over all evaluated baselines.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute resources are detailed in Appendix C.4 and Section 4.1, including the exact hardware used, inference time, total training time, and an estimate of the carbon footprint.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This research conforms to the NeurIPS Code of Ethics in its methodology, dataset usage, and transparency.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: We discuss potential positive and negative societal impacts in Appendix H.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: Our model predicts a single integer k describing the structural properties of tabular data and does not inherently generate high-risk content.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All assets are cited and credited; their sources and terms of use are given in Appendices G and I.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code and pretrained TabK model weights introduced in this paper are publicly released at `github.com/mrbakhtyari/TabK` and on the Hugging Face Hub under the MIT license, with documentation covering installation, inference, and training. The repository also includes the benchmark results and the analysis code that regenerates the main experimental tables and figures.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This research relies entirely on existing, publicly available benchmark datasets and synthetic data generation. No human subjects were involved.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: No human subjects research was conducted.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: Large Language Models were not used as an important, original, or non-standard component of the core methodology in this research.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.