Low-Rank Density Matrices as Concept Bottlenecks for Tabular Classification

Anonymous Authors[1]

Abstract

Tabular prediction is largely dominated by tree-based methods and, more recently, foundation models, leaving alternative modeling paradigms underexplored. Concept Bottleneck Models (CBMs), which route predictions through intermediate concept representations, are one such direction. Prior work on tabular data (TabCBM) showed competitive performance, but has seen limited follow-up. This work revisits CBMs for tabular learning. We introduce a modern implementation of TabCBM and show that it stays competitive with strong baselines. We then propose **QuantumBind** (QB), a novel concept layer that represents concepts as low-rank density matrices interacting through learned positive semi-definite observables. Sequential evaluation of these interactions induces structured nonlinearities beyond standard concept scoring mechanisms.

On OpenML-CC18, QB achieves 88.7% mean accuracy, outperforming TabCBM and strong non-foundation baselines including TabM (88.3%), CatBoost (87.5%), and XGBoost (87.3%). We find that multiple small matrices outperform a single large one, and that the benefits of sequential evaluation are dataset-dependent. A unique property of QB is that its receptor observables are feature-count invariant: trained on one set of datasets, they transfer to new tasks with different numbers of features: a form of cross-dataset concept transfer not available to other tabular methods. These results position CBMs as a competitive alternative for modern tabular learning, with QB providing a strong extension.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

1. Introduction

Concept bottleneck models (Koh et al., 2020) have become a standard approach for interpretable classification in vision and language, where predictions are routed through human-understandable intermediate concepts. For tabular data, however, concept-based methods have received surprisingly little attention despite the domain's natural affinity for interpretable features where tabular columns often correspond directly to meaningful quantities.

TabCBM (Zarlenga et al., 2023) demonstrated that unsupervised concept discovery is feasible for tabular classification, achieving competitive accuracy while providing concept-based explanations. Yet TabCBM has been largely overlooked in the tabular ML literature, which has focused on tree-based methods (Chen & Guestrin, 2016; Prokhorenkova et al., 2018), attention-based architectures (Gorishniy et al., 2021), and more recently, in-context learning approaches (Qu et al., 2025). Concept bottleneck architectures for tabular data remain underexplored despite promising early results. We show that, when implemented within modern training and architectural frameworks, TabCBM remains competitive with state-of-the-art tabular models.

We further demonstrate that relaxing strict interpretability constraints, by allowing concepts to be learned latent features rather than human-specified, transforms the bottleneck into a structured inductive bias. This preserves partial interpretability while improving predictive performance.

Building on these observations, we introduce **QuantumBind** (QB), a physics-inspired concept layer that replaces TabCBM's ad-hoc concept generators with *low-rank density matrices* measured by learned positive semi-definite (PSD) observables. In QB, low-rank structure is enforced by design: concepts are parameterized as rank-r density matrices (typically r=2–4), enabling controlled expressivity and structured interactions.

Our contributions:

1. We revisit concept bottleneck models for tabular data and show they are competitive when interpretability constraints are relaxed.

2. We introduce QB, where concepts are trace measurements $\mathrm{Tr}(\rho \cdot O)$ of low-rank density matrices by PSD observables, with an optional sequential state collapse mechanism for nonlinear interactions.
3. We show QB outperforms all non-foundation baselines on OpenML-CC18 (0.887 vs. TabM 0.883, CatBoost 0.875, XGBoost 0.873) while improving over TabCBM (0.872) by +1.5%.
4. We identify practical findings: sigmoid removal (+4–6%), multi-binder architectures, and dataset-dependent benefits of sequential collapse.

2. Background

2.1. Concept Bottleneck Models

A concept bottleneck model (Koh et al., 2020) factorizes a classifier $f : \mathcal{X} \to \mathcal{Y}$ through an intermediate concept space $\mathcal{C}$:

$$f(x) = g(c(x)), \quad c : \mathcal{X} \to \mathcal{C}, \quad g : \mathcal{C} \to \mathcal{Y}. \tag{1}$$

The concept encoder c maps inputs to concept activations, and the label predictor g classifies from concepts alone. When concepts are interpretable, this enables concept interventions ; an expert can then correct mispredicted concepts to improve predictions.

2.2. TabCBM

TabCBM (Zarlenga et al., 2023) extends CBMs to tabular data without requiring predefined concepts. It uses a shared encoder followed by N concept generators, each producing a concept embedding that is scored via cosine similarity with a learned prototype. Training proceeds in three phases: (1) pretrain the encoder for classification, (2) self-supervised denoising pretraining for concept generators, and (3) joint end-to-end training through the bottleneck.

While TabCBM demonstrated competitive accuracy and the ability to discover meaningful concepts, it has not been widely adopted in the tabular ML community. We hypothesize that this is partly due to instability in the learned concepts, which can hinder reproducibility and interpretability in practice. In addition, its concept scoring mechanism, based on cosine similarity with prototypes, does not explicitly model feature interactions, limiting each concept to primarily first-order feature patterns and constraining overall expressiveness.

2.3. Density Matrices and Trace Measurement

A density matrix $\rho \in \mathbb{R}^{d\times d}$ is a positive semidefinite (PSD) matrix with unit trace: $\rho \succeq 0$, $\mathrm{Tr}(\rho) = 1$. In quantum mechanics, density matrices represent mixed states and encode second-order statistics of the underlying system.

A measurement is defined by an observable $O \succeq 0$, and the expected measurement outcome is $\mathrm{Tr}(\rho \cdot O)$. This trace inner product is a natural similarity measure between PSD matrices, generalizing the dot product to the cone of positive semidefinite matrices.

A rank-r density matrix can be parameterized via a low-rank factor $F \in \mathbb{R}^{d\times r}$:

$$\rho = \frac{FF^\top}{\mathrm{Tr}(FF^\top)}, \tag{2}$$

which automatically satisfies $\rho \succeq 0$ and $\mathrm{Tr}(\rho) = 1$. The rank r is an explicit architectural choice controlling model capacity, and $r \ll d$ imposes a low-rank bottleneck on the feature interaction matrix.

State collapse (optional) can provide nonlinear feature interactions. After measuring ρ with observable O_1, the post-measurement state is:

$$\rho' = \frac{O_1 \rho O_1}{\mathrm{Tr}(O_1 \rho O_1)}, \tag{3}$$

and a subsequent measurement $\mathrm{Tr}(\rho' \cdot O_2)$ depends nonlinearly on ρ through the collapse.

3. Related Work

Tabular deep learning. Recent work has sought neural alternatives to gradient-boosted trees for tabular data. TabNet (Arik & Pfister, 2021) uses sequential attention, FT-Transformer (Gorishniy et al., 2021) applies transformers to feature embeddings, and TabM (Gorishniy et al., 2025) uses parameter-efficient ensembling of MLPs. TabICLv2 (Qu et al., 2025) applies in-context learning as a foundation model for tabular tasks. QB is complementary: it provides a structured bottleneck that could be integrated into any of these architectures.

Concept-based models. Beyond CBMs (Koh et al., 2020) and TabCBM (Zarlenga et al., 2023), post-hoc concept extraction (Kim et al., 2018) and concept embedding models (Espinosa Zarlenga et al., 2022) offer different interpretability trade-offs. QB's concepts (receptor measurements) are inherently interpretable as projections onto PSD bases, similar to spectral analysis.

Low-rank and tensor methods. Low-rank matrix factorization appears in recommender systems (Koren et al., 2009), compressed sensing (Recht et al., 2010), and neural network compression (Hu et al., 2022). QB differs in that low-rank structure is imposed on the *feature interaction matrix* (density matrix) rather than on model weights. Polynomial networks (Chrysos et al., 2021) also capture higher-order interactions but without PSD constraints or the measurement interpretation.

4. Method: QuantumBind

QuantumBind (QB) is not a direct extension of TabCBM but a redesign of its concept layer and its surrounding architecture. By replacing prototype-based concept scoring with a density-based formulation, QB enables structured interactions between features and removes key limitations of similarity-based concepts. This change unlocks a different set of architectural choices, such as low-rank multi-binder representations and sequential measurement, that together yield improved expressivity and performance while preserving a bottleneck structure.

The resulting architecture (Figure 1) consists of four stages: (i) per-feature encoding, (ii) construction of multiple low-rank density matrices, (iii) measurement through learned observables (optionally with sequential state collapse), and (iv) linear classification over the resulting concept representation.

Stage 1: Per-feature encoding. Each input feature x_i is independently embedded using a NAM-style (Agarwal et al., 2021) per-feature MLP:

$$h_i = \mathrm{MLP}_i(x_i) \in \mathbb{R}^H. \tag{4}$$

The resulting embeddings are concatenated and projected to a shared representation:

$$h = \mathrm{ReLU}\left(W\left[h_1 \| \cdots \| h_d\right] + b\right) \in \mathbb{R}^D. \tag{5}$$

In our experiments, H=32 and D=64.

Stage 2: Multi-binder density matrices. Rather than forming a single large density matrix, we use K binders, each projecting h to a small $d_\rho \times d_\rho$ factor:

$$F_k = \mathrm{reshape}(W_k h,\ [d_\rho, d_\rho]), \quad \rho_k = \frac{F_k F_k^\top}{\mathrm{Tr}(F_k F_k^\top)}. \tag{6}$$

With K=9 and d_ρ=4, each ρ_k is a rank-$\leq$ 4 matrix capturing a different view of feature interactions. This multi-binder design outperforms a single large ρ (Section 6).

Stage 3: Receptor measurement (parallel or sequential). Each binder k is associated with S receptor observables

$$O_{k,s} = L_{k,s} L_{k,s}^\top, \quad s = 1, \ldots, S, \tag{7}$$

where $L_{k,s}$ are learnable lower-triangular matrices, ensuring $O_{k,s} \succeq 0$. Measurement proceeds sequentially, starting from the initial state

$$\rho_k^{(0)} = \rho_k. \tag{8}$$

At each step s, we compute a fingerprint and update the state:

$$\mathrm{fp}_{k,s} = \mathrm{Tr}\left(\rho_k^{(s-1)} O_{k,s}\right), \tag{9}$$

$$\rho_k^{(s)} = \frac{O_{k,s}\, \rho_k^{(s-1)}\, O_{k,s}}{\mathrm{Tr}\left(O_{k,s}\, \rho_k^{(s-1)}\, O_{k,s}\right)}. \tag{10}$$

In *parallel* mode, fingerprints are computed independently as $\mathrm{fp}_{k,s} = \mathrm{Tr}(\rho_k \cdot O_{k,s})$ without collapse, yielding $K \times S$ bilinear features. In *sequential* mode, the state collapse after each measurement makes subsequent fingerprints depend on previous ones, capturing order-dependent, nonlinear interactions that cannot be recovered in parallel mode. On CC18, both modes perform comparably (Section 6), suggesting that the density matrix parameterization is the primary driver of performance; sequential collapse may be more beneficial in settings with strong higher-order feature interactions.

No sigmoid on fingerprints. A critical practical finding: applying sigmoid to $\mathrm{Tr}(\rho \cdot O)$ compresses all values to the $[0.5, 0.65]$ range, destroying class separation. We use raw trace values and let the classifier handle nonlinearity.

Stage 4: Linear concept classifier. The $K \times S$ fingerprints are concatenated into a concept vector

$$z = [\mathrm{fp}_{1,1}, \ldots, \mathrm{fp}_{1,S}, \ldots, \mathrm{fp}_{K,1}, \ldots, \mathrm{fp}_{K,S}] \in \mathbb{R}^{KS}. \tag{11}$$

Predictions are obtained via a linear classifier:

$$\hat{y} = W_{\mathrm{cls}}\, z + b_{\mathrm{cls}}. \tag{12}$$

This layer defines the concept bottleneck: all predictive information must pass through the low-dimensional fingerprint representation z, enforcing a structured intermediate representation while keeping the final decision rule simple.

4.1. Training

We adopt TabCBM's three-phase training:

1. **Pretrain** (Phase 1): Train NAM + dense mixer + direct classification head, bypassing the concept bottleneck.
2. **Self-supervised** (Phase 2): Freeze encoder, train receptors via denoising: randomly mask features, form ρ, measure, and reconstruct unmasked features from fingerprints.
3. **Joint** (Phase 3): End-to-end training through the full bottleneck with classification loss + regularization.

Regularization is applied at the level of the observables and binders. We introduce a receptor repulsion term to

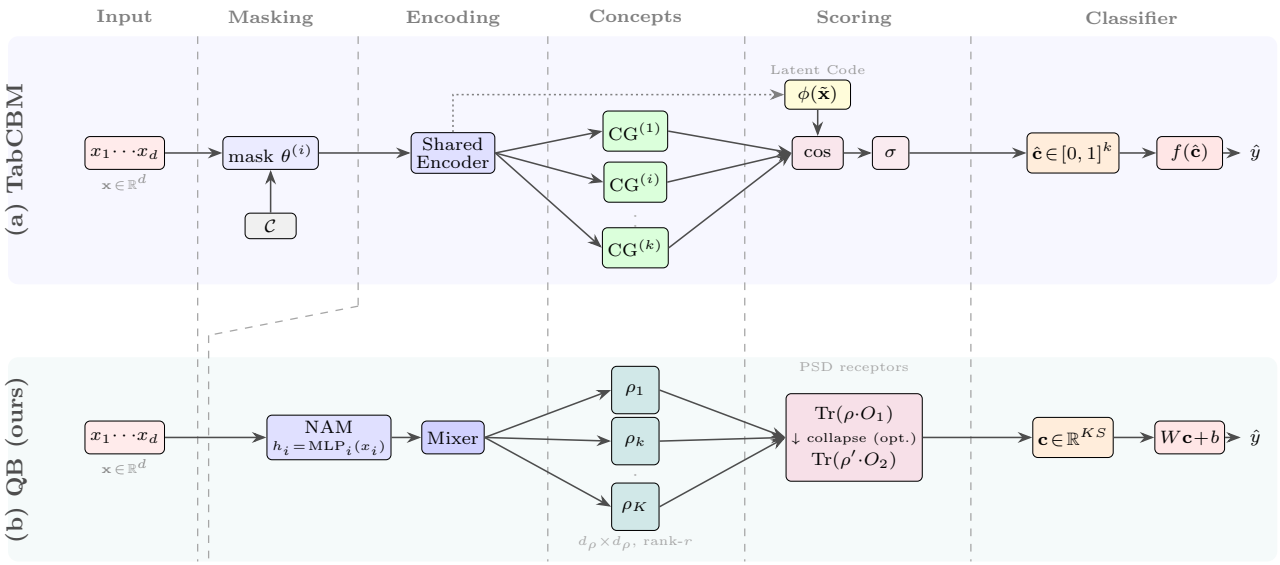

Figure 1. Comparison of (a) TabCBM and (b) QuantumBind architectures. TabCBM uses concept generators scored via cosine similarity with a latent code encoder. QB replaces this with K low-rank density matrices ρ_k measured by PSD receptor observables via $\mathrm{Tr}(\rho \cdot O)$, with optional state collapse between measurements. Both route all information through a concept bottleneck before the final classifier. QB uses a neural additive model (NAM) encoder, raw trace values (no sigmoid), and a linear classifier.

discourage overlap between observables, defined as

$$\mathcal{L}_{\text{rep}} = \sum_{i \neq j} \frac{\mathrm{Tr}(O_i O_j)^2}{\mathrm{Tr}(O_i^2)\,\mathrm{Tr}(O_j^2)}, \tag{13}$$

which penalizes aligned operators independently of their scale. In addition, we regularize the variance of factor norms across binders to promote a balanced allocation of representation capacity.

For prediction, we employ a boosting-style ensemble of $M = 5$ models, combined with post-hoc temperature calibration.

4.2. Connection to Existing Frameworks

QB's trace measurement $\mathrm{Tr}(\rho \cdot O)$ is a bilinear form $\mathrm{vec}(\rho)^\top \mathrm{vec}(O)$, a dot product in the space of PSD matrices. This connects to:

- **Multi-head attention / slot attention** (Locatello et al., 2020): like attention heads, QB's binders produce multiple parallel views of the input. Unlike slot attention, binders do not compete via iterative refinement: each independently projects h to a density matrix. The diversity comes from the receptor repulsion regularizer rather than from a competition mechanism.
- **Polynomial networks** (Chrysos et al., 2021): the trace product is a degree-2 polynomial in the input features. When sequential collapse is used, it composes these into higher-degree interactions.
- **Tensor networks**: density matrices are rank-r tensor decompositions of the feature interaction matrix.

5. Experiments

5.1. Setup

We evaluate on **OpenML-CC18** (Vanschoren et al., 2014), a curated suite of 72 classification datasets. We restrict to the 52 datasets where tabular foundation methods, in particular TabICLv2, can readily scale. All methods use the same 5-fold cross-validation splits. Ablation studies and intermediate model selection were conducted on a separate 14-dataset subset (TabICL14), chosen to be diverse and to include datasets where TabICLv2 has a clear advantage, providing a challenging target for measuring progress. Of these, 11 overlap with CC18; the remaining 3 (Australian, monks-problems-2, soybean) are drawn from the broader OpenML repository. No architectural or hyperparameter decisions were made based on full CC18 performance; the CC18 evaluation is purely held-out.

QB configuration: K=9 binders, d_ρ=4, r=4 (full-rank within d_ρ), S=2 measurement sites (parallel or sequential), no sigmoid. NAM hidden dim H=32, dense mixer dim 64, AdamW (lr=10^{-3}, weight decay 0.01), cosine annealing, early stopping (patience 25), boosted ensemble M=5.

Baselines: TabICLv2 (Qu et al., 2025) (in-context learning foundation model), TabPFNv2 (Hollmann et al., 2025) (prior-data fitted network), and TabCBM (Zarlenga et al., 2023). All models, including baselines, use their de-

Table 1. Results on OpenML-CC18 (52 datasets, 5-fold CV). QB outperforms all non-foundation baselines. $\bar{\sigma}$ = mean per-dataset fold std. Rank = mean rank (lower is better). Wins = datasets with best accuracy.

Method	Type	Acc.	$\bar{\sigma}$	Rank	Wins
TabICLv2	ICL	**.900**	.010	**1.8**	**35**
TabPFNv2[†]	ICL	.890	.011	3.8	9
QB-par (ours)	CBM	.887	.011	4.5	6
QB-seq (ours)	CBM	.886	.011	4.9	6
TabM	NN	.883	.013	5.4	4
CatBoost	Tree	.875	.014	6.4	3
RealMLP	NN	.874	.013	7.6	1
XGBoost	Tree	.873	.013	7.2	3
LightGBM	Tree	.873	.013	6.9	0
TabCBM	CBM	.872	.011	6.6	3

[†]6/52 datasets imputed (GBDT mean).

fault hyperparameters with no per-dataset tuning. We re-implemented TabCBM in PyTorch with modernized training: vectorized concept generators, Kaiming initialization, entity embeddings for categoricals, and the same boosted ensemble (M=5) as QB. These improvements boost TabCBM's accuracy beyond the original TensorFlow implementation, making our TabCBM baseline stronger and the comparison with QB fairer. For the TabICL14 analysis, we additionally compare against XGBoost (Chen & Guestrin, 2016), TabM (Gorishniy et al., 2025), and RealMLP (Holzmüller et al., 2024).

5.2. Main Results: OpenML-CC18

Table 1 presents our main result on CC18. QB achieves 0.887 mean accuracy (parallel variant), outperforming all non-foundation baselines: TabM (0.883), CatBoost (0.875), XGBoost (0.873), and RealMLP (0.874). Only the two foundation models, TabICLv2 (0.900) and TabPFNv2 (0.890), surpass QB. Parallel and sequential QB perform comparably (0.887 vs. 0.886), indicating that the collapse mechanism's benefit is dataset-dependent (Section 6).

Two key observations emerge. First, *QB is the best non-foundation model*: it edges out TabM (0.887 vs. 0.883), CatBoost (0.875), and XGBoost (0.873), though the margin over TabM is small (+0.4%). Second, *TabCBM is comparable to tree-based methods*: at 0.872, our modernized re-implementation performs on par with XGBoost (0.873) and LightGBM (0.873), demonstrating that concept bottleneck architectures are viable for tabular classification at this scale. QB improves on TabCBM by +1.5%, winning on 38 of 52 datasets. The largest gains are on cylinder-bands (+37.8%), diabetes (+6.5%), and eucalyptus (+5.8%), *i.e.* datasets with moderate feature counts (8–20) where pairwise interactions are discriminative, precisely the regime where QB's second-order representations have an advantage over TabCBM's first-order cosine scoring. Conversely, TabCBM wins on datasets with many well-separated classes (vowel: 11 classes, soybean: 19 classes), where individual feature patterns suffice.

Table 2. Results on TabICL14 (14 datasets, 5-fold CV). Same format as Table 1.

Method	Type	Acc.	$\bar{\sigma}$	Rank	Wins
TabICLv2	ICL	**.929**	.009	**1.6**	**12**
QB-seq (ours)	CBM	.919	.011	3.2	2
QB-par (ours)	CBM	.919	.011	3.3	2
TabCBM	CBM	.913	.013	4.6	1
TabM	NN	.911	.013	4.2	1
RealMLP	NN	.905	.014	6.1	0
XGBoost	Tree	.900	.014	4.9	2

5.3. TabICL14 Analysis

For detailed analysis and ablation studies, we use a curated subset of 14 OpenML datasets (TabICL14) selected to span diverse classification tasks (binary and multi-class, 500–3,200 samples, 4–36 features) where all baselines are available (Table 2).

On TabICL14, TabCBM (0.913) outperforms XGBoost (0.900) and RealMLP (0.905). QB improves further to 0.919, surpassing TabM (0.911) and approaching TabICLv2 (0.929). QB also shows the lowest fold variance among non-foundation models ($\bar{\sigma}$=0.011), matching TabCBM and suggesting that concept bottleneck architectures produce stable predictions.

6. Analysis

6.1. What Do QB Concepts Capture?

A key question is what QB's density matrix concepts represent compared to TabCBM's cosine-prototype concepts. In TabCBM, each concept generator produces a vector $\tilde{c}^{(i)} \in \mathbb{R}^m$ scored against a latent prototype via $\sigma(\cos(\tilde{c}^{(i)}, \phi(\tilde{x})))$. This captures *first-order* feature patterns: each concept responds to a weighted subset of features.

In QB, each concept is a trace measurement $\mathrm{Tr}(\rho_k \cdot O_s)$ where ρ_k is a $d_\rho \times d_\rho$ density matrix encoding pairwise feature interactions. This captures *second-order* patterns: the concept score depends on the outer product structure of feature embeddings. When sequential collapse is enabled, it chains two such measurements: the second acts on a state projected by the first observable, yielding a degree-4 rational function of the input features without additional parameters. However, our CC18 results show that this higher-order capacity is not always necessary: the second-order density matrix parameterization alone accounts for most of QB's improvement over TabCBM.

Table 3. Effect of sigmoid on fingerprint scores (vehicle dataset). Sigmoid compresses all values to a narrow range, destroying class separation.

Activation	FP range	Accuracy
Sigmoid	$[0.50, 0.65]$	0.796
None (raw $\mathrm{Tr}(\rho{\cdot}O)$)	$[0.0, 1.8]$	**0.837**

Concretely, expanding $\mathrm{Tr}(\rho \cdot O)$ in terms of the input:

$$\mathrm{Tr}(\rho \cdot O) = \frac{\sum_{ij}(Wh)_i(Wh)_j\, O_{ij}}{\sum_i (Wh)_i^2}, \qquad (14)$$

where h is the mixed feature embedding. This is a *normalized bilinear form* over feature embeddings, a ratio of degree-2 polynomials. TabCBM's cosine similarity is also a normalized bilinear form, but between two *different* vectors (concept and prototype), while QB measures a single vector against a *fixed PSD basis*. Intuitively, each TabCBM concept detects a *direction* in feature space ("is this sample similar to prototype k?"), whereas each QB concept detects a *covariance pattern* ("does this sample exhibit a specific pattern of pairwise feature interactions?"). This distinction explains QB's +1.5% gain over TabCBM: on datasets where feature interactions are discriminative, measuring covariance patterns is more informative than measuring directions alone. A detailed visualization of learned receptor observables and their correspondence to interpretable feature interactions is left to future work.

6.2. Sigmoid Kills Concept Scores

Table 3 shows the most impactful finding. The original TabCBM uses sigmoid on concept scores to bound them in $[0, 1]$, which is natural for cosine-similarity-based scores that can be negative. QB inherits this design choice by default. However, since ρ and O are both PSD, $\mathrm{Tr}(\rho \cdot O) \geq 0$ always holds, so sigmoid maps all values to the saturated region $[0.5, 1)$. In practice, fingerprints concentrate in $[0.5, 0.65]$, destroying class-discriminative variation. Removing sigmoid gives +4–6% across all configurations. The key insight is that PSD-parameterized scores have a fundamentally different range than cosine similarities: they are non-negative by construction, so sigmoid's role as a normalizer becomes counterproductive.

6.3. Ablation Study

Table 4 shows ablation results across the full 14-dataset benchmark. **Measurement mode**: Parallel and sequential QB with $S{=}2$ achieve identical accuracy (0.919) on TabICL14, and perform comparably on CC18 (0.887 vs. 0.886). The collapse mechanism does not provide a consistent advantage, suggesting that QB's improvement over TabCBM stems from the density matrix parameterization itself. Adding more collapse steps ($S{=}3, 4$) consistently reduces accuracy by 6–7%: each collapse projects the state onto a lower-dimensional subspace, progressively losing information. **Binders**: doubling from $K{=}9$ to $K{=}18$ also hurts (-6.5%), suggesting overfitting from too many density matrices on these small-to-medium datasets (500–3,200 samples). **Boosting**: gradient-boosting-style sample reweighting underperforms uniform weighting by 6.8%, indicating that the base model is already well-calibrated and reweighting introduces noise on small datasets.

Table 4. Ablation study on TabICL14 (14 datasets, mean accuracy). $S{=}2$ is optimal; more sites or more binders hurt. Sequential vs. parallel is dataset-dependent (see text).

Variant	K	d_ρ	S	Acc.
Measurement mode and sites:				
Parallel, $S{=}2$	9	4	2	**0.919**
Sequential, $S{=}2$	9	4	2	**0.919**
Sequential, $S{=}3$	9	4	3	0.851
Sequential, $S{=}4$	9	4	4	0.852
Number of binders:				
$K{=}9$ (default)	9	4	2	**0.919**
$K{=}18$	18	4	2	0.854
Boosted ensemble:				
$M{=}5$ (default)	9	4	2	**0.919**
$M{=}5$, boosted reweight	9	4	2	0.851

Table 5. Multi-binder ablation (vehicle). Many small ρ outperform one large ρ.

Config	K	d_ρ	Accuracy
Single ρ	1	6	0.796
3 binders	3	4	0.806
9 binders	**9**	**3–4**	**0.819–0.837**
12 binders	12	3	0.819
18 binders	18	3	0.812

6.4. Multi-Binder vs. Single Density Matrix

A single $6{\times}6$ density matrix compresses the $d{\times}H$-dimensional NAM output (e.g., 64-dimensional after mixing) into 21 unique values: a severe information bottleneck. Multiple small density matrices preserve more information while maintaining PSD structure. Table 5 shows $K{=}9$ binders at $d_\rho{=}3$–4 is the sweet spot.

6.5. Receptors as Transferable Concepts

Since density matrices are $d_\rho{\times}d_\rho$ regardless of input dimension d, receptor observables are feature-count invariant: a property unique among concept bottleneck approaches. We pre-trained receptors on 90 OpenML datasets, entirely disjoint from both CC18 and TabICL14, and transferred them to held-out tasks. On CC18 (52 datasets), QB with pre-trained receptors achieves 0.886, identical to random

initialization (0.886). Pre-training helps in low-data regimes (standalone QB: 0.676 → 0.716, +4%), but the full architecture with boosted ensemble and end-to-end training renders the initialization irrelevant. This suggests that the current training pipeline is sufficient to learn good receptors from scratch, though pre-training may prove valuable for few-shot or cross-domain transfer settings.

7. Discussion and Limitations

When does QB help? QB excels on datasets where second-order feature interactions are informative. On CC18, QB achieves its largest gains over TabCBM on cylinder-bands, diabetes, and eucalyptus: all moderate-dimensional datasets where pairwise feature structure matters. Both concept bottleneck methods outperform tree-based methods on vehicle (+7–9% over XGBoost), suggesting that the bottleneck architecture provides useful regularization for small datasets with many features. QB struggles relative to trees on wall-robot-navigation and tic-tac-toe, where decision boundaries are axis-aligned and trees' native splitting is optimal.

TabCBM vs. QB: when to use which? TabCBM's cosine-prototype scoring is simpler, faster to train, and provides more directly interpretable concepts (each concept has a clear feature mask). QB's density matrix scoring captures richer second-order interactions (winning 38/52 over TabCBM), but at the cost of a less transparent concept space. For applications where concept interventions are critical, TabCBM may be preferred; for maximum accuracy within a concept bottleneck, QB is the better choice.

Sequential vs. parallel measurement. On the full CC18, parallel and sequential QB perform comparably (0.887 vs. 0.886), with sequential winning 19/52 datasets. The collapse mechanism does not consistently help across diverse datasets, suggesting that the primary advantage of QB over TabCBM comes from the density matrix parameterization itself, not from the sequential measurement protocol.

The bottleneck trade-off. QB's PSD constraint restricts the function class. At equal parameter budgets, an unconstrained MLP head outperforms QB (0.843 vs. 0.819 on vehicle). QB's value lies in the structured, transferable, and partially interpretable concept space, not raw predictive power. However, the CC18 results show that this trade-off is favorable: QB's mean rank (4.5) is competitive with TabM (5.3) and ahead of all tree-based methods.

Limitations. *No hyperparameter tuning.* All results use default hyperparameters for both QB and TabCBM (K=9, d_ρ=4, S=2, lr=10^{-3}). Per-dataset hyperparameter optimization would likely improve results for both methods, but we have not conducted such a search. The relative ranking of QB vs. TabCBM under tuned settings remains an open question.

Classification only. QB is currently designed for classification. Extending to regression would require replacing the softmax readout and adapting the loss, but the density matrix construction and measurement stages are task-agnostic in principle.

Scaling. The NAM encoder creates one MLP per input feature, scaling linearly with d. For high-dimensional inputs ($d > 100$), this becomes expensive in both parameters and compute. The density matrix construction (K matrices of size d_ρ^2) and measurement ($K \times S$ trace products) are cheap, but the NAM bottleneck limits applicability to very wide datasets.

Ensemble cost. The ensemble (M=5) multiplies training time and inference cost by $5\times$. A single-model QB achieves lower accuracy; reducing ensemble size without sacrificing performance is an open problem.

Interpretability. While QB concepts (trace measurements) are more structured than MLP features, they are less interpretable than TabCBM's concepts, which have explicit feature masks. The density matrix ρ_k encodes feature interactions implicitly, making it harder to explain *which* features a concept responds to.

Physics analogy. The connection to quantum mechanics is an analogy. We do not claim tabular features have quantum structure, only that the mathematical framework (PSD matrices, trace measurement, state collapse) provides useful inductive biases.

8. Conclusion

We have shown that concept bottleneck models for tabular data are more competitive than commonly believed. A modernized TabCBM already matches tree-based methods (0.872 vs. XGBoost 0.873 on CC18), and replacing its concept scoring with low-rank density matrices (QB) yields the best non-foundation model on this benchmark (0.887), outperforming TabM (0.883), CatBoost (0.875), and all tree-based baselines.

The key practical insight is that the density matrix parameterization itself, not the sequential collapse, drives the improvement over TabCBM. Multiple small density matrices (K=9, d_ρ=4) capture second-order feature interactions that cosine-prototype scoring cannot, while the PSD constraint provides geometric structure that aids optimization. Removing sigmoid from concept scores (+4–6%) is broadly applicable to any PSD-parameterized bottleneck.

Beyond performance, QB's density matrix formulation en-

ables a property unavailable to other tabular methods: receptor observables live in a fixed $d_\rho \times d_\rho$ space regardless of input dimensionality, making them transferable across datasets with different feature counts. While this does not yet improve accuracy in our full pipeline, it opens a path toward pre-trained concept libraries that could benefit few-shot and cross-domain tabular learning.

The low-rank structure at QB's core – density matrices of rank r=2–4 in a $d_\rho \times d_\rho$ space – demonstrates that explicit low-rank priors over feature interactions can match or exceed implicit structure learning for tabular classification, positioning concept bottleneck models as a competitive and principled alternative to both tree-based and neural tabular methods.

Code availability. Code for QB and our TabCBM reimplementation is available at https://github.com/NamelessAuthor/quantumbind.

References

Agarwal, R., Melnick, L., Frosst, N., Zhang, X., Lengerich, B., Caruana, R., and Hinton, G. E. Neural additive models: Interpretable machine learning with neural nets. *Advances in Neural Information Processing Systems*, 34, 2021.

Arik, S. Ö. and Pfister, T. TabNet: Attentive interpretable tabular learning. In *AAAI Conference on Artificial Intelligence*, volume 35, pp. 6679–6687, 2021.

Chen, T. and Guestrin, C. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016.

Chrysos, G. G., Moschoglou, S., Bouritsas, G., Deng, J., Panagakis, Y., and Zafeiriou, S. Deep polynomial neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44:4021–4034, 2021.

Espinosa Zarlenga, M., Barbiero, P., Ciravegna, G., Marra, G., Giannini, F., Diligenti, M., Shams, Z., Precioso, F., Melacci, S., Weller, A., et al. Concept embedding models: Beyond the accuracy-explainability trade-off. In *Adv Neural Inf Process Syst*, volume 35, 2022.

Gorishniy, Y., Rubachev, I., Khrulkov, V., and Babenko, A. Revisiting deep learning models for tabular data. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Gorishniy, Y., Kotelnikov, A., and Babenko, A. TabM: Advancing tabular deep learning with parameter-efficient ensembling. In *International Conference on Learning Representations*, 2025.

Hollmann, N., Müller, S., Purucker, L., Krishnakumar, A., Körfer, M., Hoo, S. B., Schirrmeister, R. T., and Hutter, F. Accurate predictions on small data with a tabular foundation model. *Nature*, 2025.

Holzmüller, D., Grinsztajn, L., and Steinwart, I. Better by default: Strong pre-tuned MLPs and boosted trees on tabular data. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.

Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *ICML*, pp. 2668–2677, 2018.

Koh, P. W., Nguyen, T., Tang, Y. S., Mussmann, S., Pierson, E., Kim, B., and Liang, P. Concept bottleneck models. In *International Conference on Machine Learning*, pp. 5338–5348. PMLR, 2020.

Koren, Y., Bell, R., and Volinsky, C. Matrix factorization techniques for recommender systems. *Computer*, 42(8): 30–37, 2009.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

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

Qu, J., Holzmüller, D., Varoquaux, G., and Morvan, M. L. TabICL: A tabular foundation model for in-context learning on large data. *arXiv preprint arXiv:2502.05564*, 2025.

Recht, B., Fazel, M., and Parrilo, P. A. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

Vanschoren, J., van Rijn, J. N., Bischl, B., and Torgo, L. OpenML: Networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60, 2014.

Zarlenga, M. E., Shams, Z., Nelson, M. E., Kim, B., and Jamnik, M. Tabcbm: Concept-based interpretable neural networks for tabular data. *Transactions on Machine Learning Research*, 2023.

A. Supplementary Results

A.1. Statistical Significance

We conduct a Friedman test with Nemenyi post-hoc analysis (α=0.05, CD=1.67) on the 52 CC18 datasets. TabPFNv2 is excluded due to missing datasets. TabICLv2 is significantly better than all other methods. QB-par and QB-seq form a clique with TabM and CatBoost: their differences are not statistically significant. QB-par is significantly better than RealMLP, XGBoost, and LightGBM. TabCBM is not significantly different from tree-based methods, confirming that concept bottleneck models are statistically on par with standard baselines. The critical difference diagram is shown below.

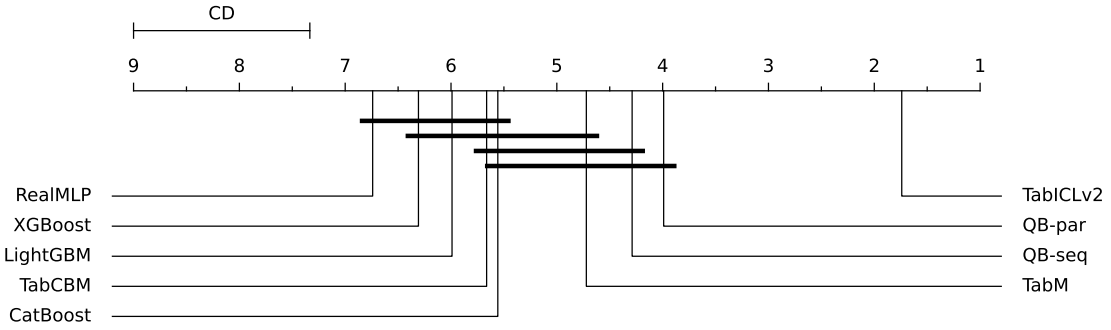

Figure A.1. Critical difference diagram (Nemenyi test, α=0.05) on 52 CC18 datasets. Connected methods are not significantly different.

A.2. Per-Dataset Results on OpenML-CC18

Table A.1 reports per-dataset accuracy for all methods on the 52 common OpenML-CC18 datasets. TabPFNv2 results for 6 datasets are imputed using the mean of GBDT baselines (marked *).

Table A.1. Per-dataset accuracy on 52 OpenML-CC18 datasets (5-fold CV). Bold = best, underline = second best. All models evaluated on all 52 datasets except TabPFNv2 (46/52).

Dataset	XGB	CatB	RMLP	TabM	TCBM	QB-p	QB-s	PFN	ICL
Australian	**.872**	.870	.843	.867	.846	.861	.859	.870*	.872
MiceProtein	.980	.996	.997	.988	.999	.995	.994	**1.00**	1.00
authorship	.988	.990	.993	.996	.996	.995	.995	**1.00**	1.00
dmft	.200	.173	.205	**.223**	.191	.218	.215	.204	.217
balance-scale	.866	.883	.973	.986	.987	.989	**.990**	.974	.990
banknote	.996	.997	**1.00**	.996	1.00	1.00	1.00	1.00	1.00
blood-transf.	.766	.714	.795	.781	.775	**.802**	.795	.789	.793
breast-w	.960	.969	.958	.967	.954	.967	.966	**.970**	.969
car	.996	.977	.986	.997	.997	**.999**	.997	.989	.991
churn	.957	.959	.932	.962	.961	.964	.963	.966	**.970**
climate	.946	.946	.924	.946	.937	.946	.943	**.952**	.950
cmc	.538	.509	.544	.541	.520	.549	**.563**	.563	.559
cnae-9	.907	.930	.928	.935	.954	**.958**	.956	.892*	.954
credit-appr.	.852	.855	.857	.859	.829	.865	.867	.867	**.870**
credit-g	.749	.763	.738	.738	.742	.743	.753	**.764**	.760
cylinder-bands	.815	.822	.752	.757	.422	.796	.800	.802	**.850**
diabetes	.745	.745	.759	.760	.703	.766	**.768**	.766	.762
dna	.963	.958	.938	.962	.954	.957	.959	.965	**.966**
dresses-sales	.560	.546	.564	**.614**	.564	.604	.574	.602	.612
eucalyptus	.668	.685	.687	.693	.637	.708	.696	.715	**.723**
ilpd	.700	.695	.702	**.726**	.700	.714	.714	.708	.724
kc1	.859	.826	.843	.846	.856	.858	.860	.865	**.866**
kc2	.822	.805	.816	.843	.818	.835	.826	.843	**.845**
kr-vs-kp	.993	**.996**	.994	.994	.992	.993	.994	.993	.996
mfeat-factors	.966	.975	.973	.975	.982	.979	.980	.980	**.988**
mfeat-fourier	.834	.843	.823	.856	.831	.841	.833	.885	**.911**
mfeat-karhunen	.958	.970	.954	.973	.965	.972	.969	.980	**.985**
mfeat-morph.	.693	.724	.726	.744	.752	.755	.755	**.758**	.758
mfeat-pixel	.965	.975	.968	.972	.978	.972	.970	.976	**.989**
mfeat-zernike	.784	.796	.823	.821	.826	.855	.848	.872	**.904**
monks-2	.963	**1.00**	.993	1.00	1.00	1.00	1.00	.953*	1.00
optdigits	.981	.984	.982	.986	.988	.985	.983	.990	**.995**
ozone-level	.943	.946	.946	.937	.941	.946	.944	.947	**.949**
pc1	.931	.924	.926	.931	.926	.931	.928	.937	**.941**
pc3	.886	.882	.898	.895	.884	.894	.896	.898	**.903**
pc4	.918	.911	.912	.913	**.927**	.912	.910	.922	.927
phoneme	.885	.894	.887	.896	.891	.894	.890	.919	**.921**
qsar-biodeg	.864	.868	.864	.864	.875	.885	.886	.885	**.896**
segment	.985	.982	.972	.980	.977	.978	.981	.947	**.988**
semeion	.925	.940	.918	.938	.945	.935	.935	.940	**.966**
sick	.988	.987	.970	.987	.984	.982	.981	.985	**.990**
soybean	.933	.941	.927	.927	.941	.933	.925	.937*	**.958**
spambase	.950	.956	.941	.951	.945	.949	.950	.956	**.962**
splice	.958	.958	.945	.961	.948	.965	.966	**.970**	.968
steel-plates	.804	.796	.745	.776	.766	.770	.762	**.847**	.837
texture	.985	.988	.997	.993	.998	.997	.996	.986*	**1.00**
tic-tac-toe	**1.00**	1.00	.974	.993	.985	.991	.990	.985	.987
vehicle	.762	.762	.787	.764	.856	.830	.837	.865	**.878**
vowel	.905	.955	.969	.954	.991	.977	.976	.928*	**.999**
wall-robot-nav	**.998**	.996	.938	.993	.958	.971	.972	.993	.991
wdbc	.961	.961	.972	.972	.972	.974	.975	.977	**.981**
wilt	.986	.986	.987	.985	.988	**.989**	.989	.989	.989
Mean	.873	.875	.874	.883	.872	.887	.886	.890	**.900**

XGB=XGBoost, CatB=CatBoost, RMLP=RealMLP, TCBM=TabCBM, QB-p=QB-par, QB-s=QB-seq, PFN=TabPFNv2, ICL=TabICLv2.
*TabPFNv2 imputed on 6 datasets using GBDT mean (XGBoost, CatBoost, LightGBM).