TABFSA: FEATURE-TO-STRATEGY ADAPTATION FOR TABULAR IN-CONTEXT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Tabular foundation models (TFMs) enable in-context prediction across datasets, but target settings involving temporal transfer, unseen groups, or rare or unseen categorical levels may require targeted adaptation. We study how target data can improve their generalization to new prediction settings without full-model fine-tuning or pretraining a new backbone. We propose Feature-to-Strategy Adaptation (FSA), which uses support–query training tasks representing the target settings to learn a bank of low-rank expert adapters, each specialized to one task family. With the backbone and adapters frozen, controlled comparisons show that favorable adaptation strengths and allocations vary across tasks, and some weighted combinations outperform directly applying any one expert at maximum strength. These observations motivate learning the adaptation strategy from the current task context. FSA constructs a dual-view strategy feature from labeled support examples and unlabeled query inputs, comparing support and query through feature and sample views. We jointly learn this strategy feature and its mapping to adaptation strength and expert allocation by minimizing query prediction loss, enabling task-dependent composition without further parameter optimization on unseen tasks. We instantiate FSA as TabFSA with TabICLv2 and synthetic target training tasks. On BeyondArena, TabFSA raises aggregate Elo from 1178 to 1216 and improves average rank from 4.40 to 3.90. On 200 TALENT classification datasets, it records 136 wins, 15 ties, and 49 losses against the frozen backbone. Retraining FSA on frozen TabPFN also improves prediction, demonstrating applicability across backbones.

1 INTRODUCTION

Tabular foundation models (TFMs) enable in-context prediction on previously unseen datasets by conditioning on labeled support examples. TabPFN and TabPFNv2 (Hollmann et al., 2023; 2025), followed by TabPFN-3 and TabPFN-3.5 (Grinsztajn et al., 2026; Jäger et al., 2026), together with TabICL and TabICLv2 (Qu et al., 2025; 2026), demonstrate the effectiveness of this paradigm across tabular datasets.

However, target prediction settings may differ from the training distribution in ways that challenge generalization. For example, temporal prediction transfers evidence from earlier to later observations; unseen-group prediction requires generalization to groups absent from support; and high-cardinality categorical inputs provide limited evidence for rare or unseen levels. Model behavior varies across these target prediction settings (Purucker et al., 2026). We study how to leverage target data in these settings to enhance an existing tabular foundation model while retaining its pretrained backbone.

We propose *Feature-to-Strategy Adaptation (FSA)*. FSA organizes target data into training tasks and uses them to learn a bank of specialized low-rank expert adapters while keeping the pretrained backbone frozen. Our main implementation, TabFSA, instantiates FSA with TabICLv2 (Qu et al., 2026), a state-of-the-art open-source TFM. We construct synthetic target tasks covering four mechanism families: Extended-IID varies response and observation mechanisms under common-population sampling; temporal tasks transfer from earlier to later observations; grouped tasks introduce group-dependent structure and transfer across groups; and high-cardinality tasks vary the evidence available for rare or unseen categorical levels.

The learned expert updates show mechanism-specific specialization, but their predictive effects also depend on how they are applied. FSA controls expert composition through an adaptation strength s

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

and expert allocation $\boldsymbol{\pi}$. Controlled comparisons show that favorable strengths and allocations vary across tasks, and that some weighted combinations outperform directly applying any one expert at maximum strength (Figure 2). These observations motivate learning the adaptation strategy from the current task context.

To predict a task-dependent strategy, FSA constructs a dual-view strategy feature from labeled support examples and unlabeled query inputs. The feature view compares support and query within feature groups, while the sample view compares row-representation sets after cross-feature interaction. With the backbone and expert-adapter bank frozen, we jointly learn the strategy feature and its mapping to s and $\boldsymbol{\pi}$ by minimizing query prediction loss. On an unseen task, the learned mapping produces the adaptation strategy directly from the support set and unlabeled query inputs. We evaluate TabFSA on BeyondArena (Purucker et al., 2026) and TALENT (Liu et al., 2025), together with controlled analyses of expert specialization, ablation studies, and cross-backbone transfer to TabPFN.

Our contributions are:

1. We introduce FSA, a framework that learns specialized expert adapters from target training tasks and adapts their composition to each prediction task while retaining the pretrained backbone. In TabFSA, these tasks extend TabICLv2's synthetic task construction to extended-IID, temporal, grouped, and high-cardinality settings.
2. We show that applying a frozen expert-adapter bank with different strengths and allocations produces task-dependent prediction changes, with some evaluated combinations improving on directly applying any one expert at maximum strength. We then learn adaptation strength and expert allocation from a dual-view strategy feature combining label-aware support information with label-free support–query comparisons.
3. We instantiate FSA as TabFSA on TabICLv2, improving performance on BeyondArena and TALENT, and validate its applicability to another backbone through retraining on frozen TabPFN.

2 Related Work

Tabular foundation models and adaptation. Prior-data fitted networks learn prediction procedures from sampled tasks and apply them to new datasets through in-context learning (Müller et al., 2022; Hollmann et al., 2023). TabPFNv2 broadens this approach for tabular prediction (Hollmann et al., 2025), while TabICL and TabICLv2 use hierarchical table representations and scalable attention (Qu et al., 2025; 2026). Recent TFMs have expanded their training distributions through mixed synthetic priors (Zhang et al., 2025), real datasets (Ma et al., 2026), and temporal mechanisms (Helli et al., 2024). Adaptation methods further modify model parameters (Rubachev et al., 2025; Tanna et al., 2026), select local context (Thomas et al., 2024; Xu et al., 2025), modify model inputs (Nguyen et al., 2026), or perform test-time adaptation and posterior adjustment (Lee, 2026; He & Shi, 2026). FSA learns specialized expert adapters from target training tasks and learns how to reuse their updates across new task contexts.

Parameter-efficient adaptation and adapter composition. Low-rank adaptation (LoRA) represents parameter updates with trainable low-rank factors while retaining pretrained weights (Hu et al., 2022). AdapterFusion combines representations from previously learned task adapters (Pfeiffer et al., 2021). LoraHub reuses independently trained LoRA modules and optimizes their combination coefficients from target examples (Huang et al., 2024), while X-LoRA conditions token- and layer-wise expert scales on hidden states (Buehler & Buehler, 2024). FSA instead predicts task-level strength and allocation from label-aware support summaries and label-free support–query comparisons. It composes mechanism-specialized experts with coefficients shared across layers and query rows, without target-task optimization.

Task-conditioned model and parameter generation. Hypernetworks use one network to generate the parameters of another network (Ha et al., 2017). HyperTuning generates soft-prefix or LoRA parameters from few-shot examples (Phang et al., 2023), and Text-to-LoRA generates LoRA parameters from natural-language task descriptions (Charakorn et al., 2025). In tabular learning, HyperFast and MotherNet generate task-specific predictors from training data (Bonet et al., 2024; Müller et al., 2025). Feature to Dynamics similarly maps input features to forecasting strategies in time-series

forecasting (Wu et al., 2026). These works condition generated parameters, predictors, or strategies on task information. FSA instead retains the pretrained in-context predictor and a separately learned expert-adapter bank, and uses the tabular support–query task context to determine how the expert updates are applied.

3 METHOD

3.1 FEATURE-TO-STRATEGY ADAPTATION

FSA proceeds in two stages. It first learns an expert-adapter bank from target training tasks while keeping the pretrained backbone frozen. After specialization, both the backbone and expert-adapter bank remain frozen while FSA learns to predict an adaptation strategy from each task context. The first stage supplies reusable update directions; the second learns how the current prediction task should control their application.

For each prediction task, the observable task context consists of a labeled support set $\mathcal{D}_s = (X_s, Y_s) = \{(x_i, y_i)\}_{i=1}^{n_s}$ and an unlabeled query set $X_q = \{x_{q,i}\}_{i=1}^{n_q}$. We denote this context by $\mathcal{T} = (\mathcal{D}_s, X_q)$. For C classes, $\widehat{P}_q \in [0,1]^{n_q \times C}$ denotes the predicted class probabilities for the query examples.

Let $\mathcal{A}$ denote the expert-adapter bank learned in the first stage and F_{θ_0} the frozen backbone representation extractor. FSA constructs a strategy feature $\mathbf{z}$ from the task context and maps it to an adaptation strategy $\mathbf{w}$:

$$\mathbf{z} = \Phi_\eta(\mathcal{T}; F_{\theta_0}), \qquad \mathbf{w} = G_\phi(\mathbf{z}), \qquad \widehat{P}_q = f_{\theta_0,\mathcal{A}}(\mathcal{D}_s, X_q; \mathbf{w}). \tag{1}$$

Here Φ_η constructs the strategy feature, G_ϕ predicts the coefficients that compose the frozen expert updates, and $f_{\theta_0,\mathcal{A}}$ applies the resulting strategy during prediction with the frozen backbone and expert-adapter bank. Figure 1 summarizes how FSA derives an adaptation strategy from the current task and applies it to the frozen expert-adapter bank.

3.2 LEARNING EXPERT ADAPTERS FROM TARGET TASKS

FSA treats a target training task as a support–query prediction episode drawn from target data. In TabFSA, we instantiate this idea with synthetic target training tasks for four mechanism families: extended-IID, temporal, grouped, and high-cardinality. For each family, we generate support–query episodes that reflect the corresponding target setting and use these episodes to train one specialized expert adapter. We extend the structural causal model (SCM) prior used by TabICLv2 (Qu et al., 2026) to define the four family-specific specialization distributions.

A base task $\tau \sim p_0$ and mechanism family μ define a transformation T_μ with parameters ξ and a support–query split rule S_μ:

$$\widetilde{\tau} = T_\mu(\tau; \xi), \qquad (\mathcal{D}_s, \mathcal{D}_q) = S_\mu(\widetilde{\tau}), \tag{2}$$

where $\mathcal{D}_q = (X_q, Y_q)$ is labeled during training. Extended-IID tasks vary response and observation mechanisms under common-population sampling. Temporal tasks organize support and query across earlier and later observations, grouped tasks introduce group-dependent structure and transfer across groups, and high-cardinality tasks vary category frequencies and the evidence available for rare or query-only categorical levels. These constructions induce four specialization distributions $p_1, \ldots, p_4$, each of which supplies the target training tasks for one expert adapter. Appendix B gives the detailed transformation and split rules.

We use pretrained TabICLv2, with frozen parameters θ_0, as the backbone (Qu et al., 2026) and parameterize each expert adapter at selected ICL attention projections with LoRA (Hu et al., 2022). All four expert adapters share the same parameterization and insertion locations, so their specialization is determined by the target training distribution on which each adapter is optimized. Appendix A.3 gives the configuration.

Let $\mathcal{A}_k$ denote the trainable low-rank factors of expert adapter k. Expert adapter k is trained only on support–query episodes sampled from its corresponding specialization distribution p_k, while the TabICLv2 backbone remains frozen and the other expert adapters are inactive. We minimize the

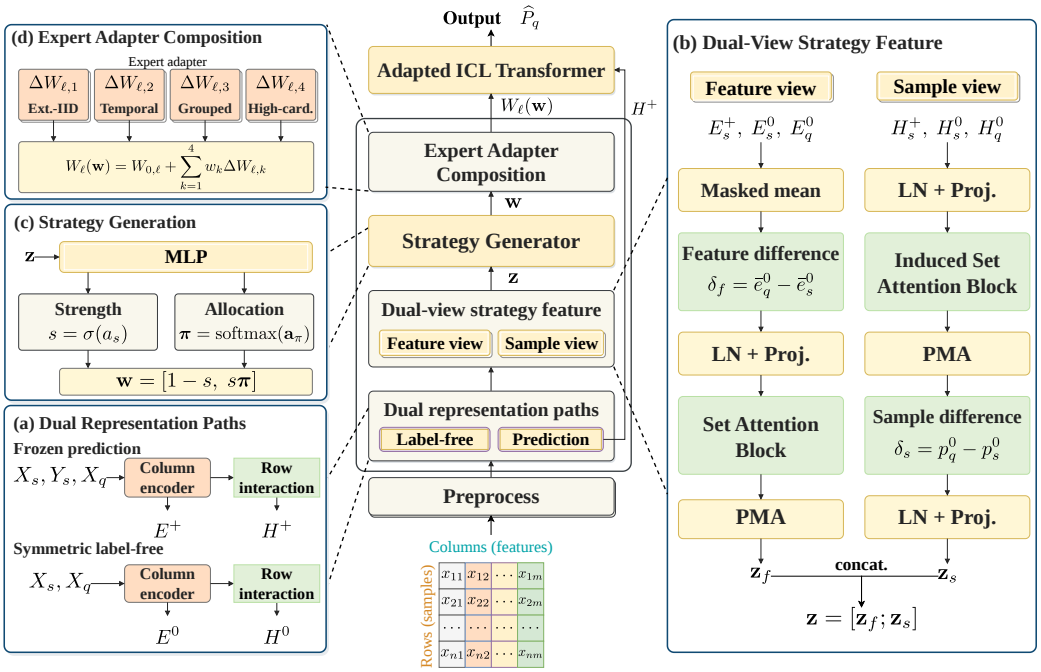


Figure 1: TabFSA inference architecture. Dual representation paths provide label-aware support information and a symmetric label-free support–query comparison. The feature and sample views form the dual-view strategy feature $\mathbf{z}$; the Strategy Generator predicts adaptation strength s and expert allocation $\boldsymbol{\pi}$, whose coefficients $\mathbf{w}$ compose the frozen expert-adapter updates for prediction. One ensemble view is shown.

query negative log-likelihood (NLL) $\ell_q(\widehat{P}_q, Y_q) = -n_q^{-1} \sum_i \log \widehat{P}_q[i, y_{q,i}]$:

$$\min_{\mathcal{A}_k} \ \mathbb{E}_{(\mathcal{D}_s, \mathcal{D}_q) \sim p_k} \ell_q(f_{\theta_0, \mathcal{A}_k}(\mathcal{D}_s, X_q), Y_q)\,. \tag{3}$$

The resulting bank $\mathcal{A} = \{\mathcal{A}_k\}_{k=1}^4$ therefore contains one expert adapter for each mechanism family. The backbone and expert-adapter bank are then frozen before strategy learning.

3.3 Adaptation Strategy and Expert-Adapter Composition

The frozen expert-adapter bank provides four update directions. An adaptation strategy controls their application through an adaptation strength $s \in [0, 1]$ and a nonnegative expert allocation $\boldsymbol{\pi}$, with $\sum_{k=1}^4 \pi_k = 1$. For an adapted projection ℓ, expert adapter k contributes the low-rank update $\Delta W_{\ell,k} = (\alpha/r) B_{\ell,k} A_{\ell,k}$ to the frozen matrix $W_{0,\ell}$; k indexes the extended-IID, temporal, grouped, and high-cardinality adapters, while r and α are the rank and scaling parameter. A zero-update component lets the strategy reduce total adaptation as well as redistribute it:

$$\mathcal{S} = \Delta^4 = \{\mathbf{w} \in \mathbb{R}^5_{\geq 0} : \textstyle\sum_{k=0}^4 w_k = 1\}, \qquad W_\ell(\mathbf{w}) = W_{0,\ell} + \textstyle\sum_{k=1}^4 w_k \Delta W_{\ell,k}. \tag{4}$$

The coefficient vector $\mathbf{w} = [1 - s,\ s\boldsymbol{\pi}]$ represents this strategy. Its expert coefficients $w_k = s\pi_k$ sum to s, and $\boldsymbol{\pi}$ specifies their relative allocation. The zero-update coefficient is $w_0 = 1 - s$, while $W_{0,\ell}$ retains coefficient one. A single task-conditioned strategy is shared across adapted matrices and query rows. Appendix A.3 details the low-rank implementation.

To examine how the same frozen expert-adapter bank responds to different applications, we hold the backbone and expert adapters frozen and vary only s and $\boldsymbol{\pi}$ by evaluating the same 64 sampled strategies on each task. Figure 2 summarizes the resulting query-NLL responses and the coefficients of the lowest-NLL candidate for each task. Appendix E.1 gives the details of the candidate construction and evaluation.

The lowest-NLL candidates use different strengths and expert allocations across tasks. The best sampled strategy attains lower NLL than the best direct expert on six tasks; on five of them it also

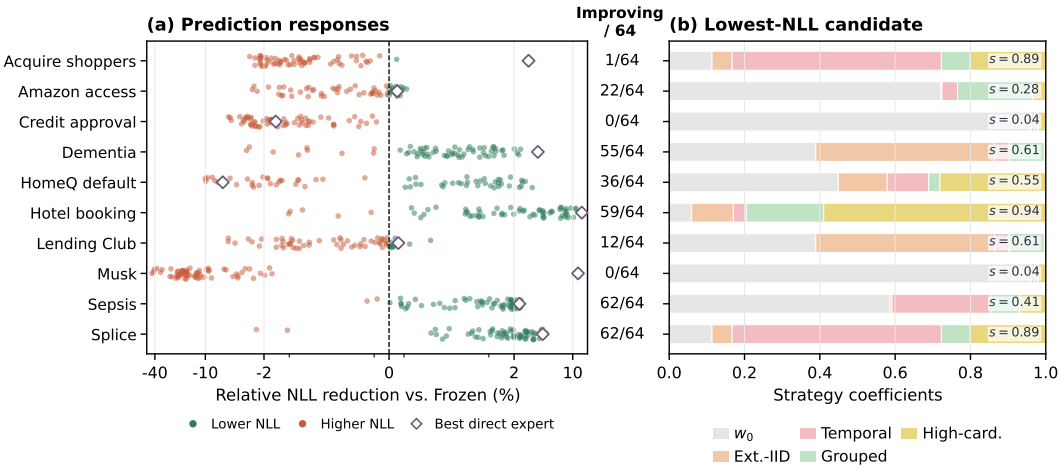


Figure 2: Prediction responses to adaptation strategies with a frozen backbone and expert-adapter bank. (a) Relative query NLL reduction versus frozen TabICLv2 for the same 64 sampled strategies on each task (symmetric-log scale). Diamonds mark the best direct expert, the lowest-NLL result from applying one expert alone at maximum adaptation strength. (b) Coefficients $\mathbf{w} = [1 - s, s\boldsymbol{\pi}]$ of the sampled strategy with the lowest query NLL on each task. Gray denotes $1 - s$; colored segments denote $s\pi_k$.

improves on frozen TabICLv2. For HomeQ default, for example, the best direct expert increases NLL by 6.20% relative to frozen TabICLv2, whereas a sampled combination reduces it by 3.29%. These comparisons separate learning update directions from deciding how to use them: the expert-adapter bank supplies the directions, but their predictive effect depends on the task and the coefficients assigned to them. FSA therefore predicts the adaptation strategy from the observed task context.

3.4 Dual-View Strategy Feature

Predicting this task-dependent strategy requires information from both the labeled support examples and the unlabeled query inputs. FSA combines a label-aware support summary with a label-free support–query comparison. Feature-level changes may be localized to particular groups before cross-feature interaction, whereas changes in the sample population may emerge after features have been combined into row representations. FSA therefore captures the comparison at both points: the feature view forms group-wise differences before aggregation, while the sample view forms a set-level difference after cross-feature interaction and set encoding.

Directly comparing label-aware support with unlabeled query representations can mix input differences with label-conditioning differences. We call the unchanged, label-aware TabICLv2 route the *Frozen prediction path*; it retains the backbone's target-aware (label-conditioned) front end for the support summary and prediction (Qu et al., 2026). We add a *Symmetric label-free path* that removes target embeddings and forms column context from support and query inputs together. TabICLv2 retains V ensemble views; let $v \in \{1, \ldots, V\}$ index one view. The two paths compute

$$(E^{+,(v)}, H^{+,(v)}) = F^{+}_{\theta_0}(X^{(v)}_s, Y^{(v)}_s, X^{(v)}_q), \qquad (E^{0,(v)}, H^{0,(v)}) = F^{0}_{\theta_0}(X^{(v)}_s, X^{(v)}_q). \tag{5}$$

Here E contains one embedding for every row–feature-group pair before cross-feature interaction, whereas H contains the resulting row representations. The label-aware $E^{+,(v)}$ and $H^{+,(v)}$ supply the support summaries and prediction path; their label-free counterparts provide comparable support and query representations. Each *feature group* combines processed columns using TabICLv2's repeated feature grouping, and prediction continues from $H^{+,(v)}$. Both paths use the same frozen weights and support-fitted preprocessing, and $Y^{(v)}_s$ includes class relabeling. Superscripts $+$ and 0 identify the label-aware and label-free paths. Appendix A.1 gives the representation shapes.

Feature view. Masked mean pooling produces a support and query summary for each feature group: $\bar{e}^{+,(v)}_{s,j}$ from label-aware support embeddings and $\bar{e}^{0,(v)}_{s,j}, \bar{e}^{0,(v)}_{q,j}$ from label-free embeddings.

For each group j, the signed feature-difference statistic is the label-free query-minus-support mean. We concatenate it with the label-aware support mean and aggregate the resulting group tokens using Set Transformer blocks (Lee et al., 2019): a Set Attention Block (SAB) relates the tokens, and one-seed Pooling by Multihead Attention (PMA) produces their summary. Layer normalization (LN) (Ba et al., 2016) precedes the learned projection:

$$\delta_{f,j}^{(v)} = \bar{e}_{q,j}^{0,(v)} - \bar{e}_{s,j}^{0,(v)}, \qquad r_{f,j}^{(v)} = P_f \,\mathrm{LN}_f([\bar{e}_{s,j}^{+,(v)}; \delta_{f,j}^{(v)}]),$$
$$\mathbf{z}_f^{(v)} = \mathrm{PMA}_{f,1}\big(\mathrm{SAB}(\{r_{f,j}^{(v)}\}_{j\in\mathcal{J}_v})\big). \tag{6}$$

Here $[\cdot;\cdot]$ denotes concatenation, and $\mathcal{J}_v$ contains feature groups with valid observations in both the support and query sets. Forming the support–query difference before pooling keeps each group's label-aware support summary and signed input difference together until the group tokens are aggregated into a table-level summary.

Sample view. A shared set encoder g_η projects the row representations, applies an Induced Set Attention Block (ISAB), and pools the resulting set with PMA (Lee et al., 2019). The same encoder is applied separately to the label-aware support and the two label-free sets:

$$p_s^{+,(v)} = g_\eta(H_s^{+,(v)}), \qquad p_s^{0,(v)} = g_\eta(H_s^{0,(v)}), \qquad p_q^{0,(v)} = g_\eta(H_q^{0,(v)}). \tag{7}$$

The signed sample-difference statistic is formed after set encoding, when each row has incorporated cross-feature information. Concatenating this label-free statistic with the label-aware support summary and projecting gives $\mathbf{z}_s^{(v)}$:

$$\delta_s^{(v)} = p_q^{0,(v)} - p_s^{0,(v)}, \qquad \mathbf{z}_s^{(v)} = P_r \,\mathrm{LN}_r([p_s^{+,(v)}; \delta_s^{(v)}]). \tag{8}$$

The feature and sample branches have separate parameters; each branch reuses its parameters across ensemble views. Appendix A.2 gives the detailed attention and masking configuration.

We average each branch output across ensemble views and concatenate the feature-view and sample-view summaries:

$$\mathbf{z}_f = \frac{1}{V}\sum_v \mathbf{z}_f^{(v)}, \qquad \mathbf{z}_s = \frac{1}{V}\sum_v \mathbf{z}_s^{(v)}, \qquad \mathbf{z} = [\mathbf{z}_f; \mathbf{z}_s] \in \mathcal{Z}. \tag{9}$$

The resulting $\mathbf{z}$ is the dual-view strategy feature used by the Strategy Generator.

3.5 Strategy Generation and Learning

The Strategy Generator G_ϕ uses a multilayer perceptron (MLP) (Rumelhart et al., 1986) with separate output heads to map the strategy feature $\mathbf{z}$ to a scalar strength logit a_s and allocation logits $\mathbf{a}_\pi$. The logits define the strategy through

$$s = \sigma(a_s), \qquad \boldsymbol{\pi} = \mathrm{softmax}(\mathbf{a}_\pi), \qquad \mathbf{w} = [1-s,\ s\boldsymbol{\pi}]. \tag{10}$$

Appendix A.3 gives the Strategy Generator configuration. With the backbone and expert-adapter bank frozen, the strategy-feature constructor and Strategy Generator are trained jointly on episodes drawn from a mixed task distribution p_{mix}:

$$\min_{\eta,\phi}\ \mathbb{E}_{(\mathcal{D}_s,\mathcal{D}_q)\sim p_{\mathrm{mix}}} \ell_q(f_{\theta_0,\mathcal{A}}(\mathcal{D}_s, X_q; G_\phi(\Phi_\eta(\mathcal{T}; F_{\theta_0}))), Y_q)\,. \tag{11}$$

This distribution combines episodes sampled from TabICLv2's original prior with single-mechanism episodes and compatible mechanism compositions, including temporal and group-based split constraints. Because the backbone and expert updates remain frozen in this stage, minimizing query NLL trains the strategy-feature constructor and Strategy Generator to determine how those updates should be applied. Gradients pass through the generated strategy and the composed updates. Query labels provide the training loss, while the strategy feature and adaptation strategy are computed from $(\mathcal{D}_s, X_q)$. Appendix C gives optimization and gradient-flow details.

4 EXPERIMENTS

We evaluate TabFSA on 98 classification datasets from BeyondArena (Purucker et al., 2026) and 200 from TALENT (Liu et al., 2025). BeyondArena covers independent and identically distributed (IID), grouped, and temporal splits. We compare against eleven baselines using their released default results and report Elo, Improvability, and average rank. On TALENT, we compare test accuracy with frozen TabICLv2 using paired wins, ties, and losses; Appendix D gives the evaluation protocol.

We analyze expert specialization and strategy composition, ablate task information and training data composition, and fit symbolic strategy surrogates. We also test cross-backbone transfer to TabPFN. Appendix E provides detailed results, including query-context sensitivity, population response, and inference cost.

4.1 GENERALIZATION TO UNSEEN TARGET TASKS

BeyondArena. Across the 98 BeyondArena classification datasets, TabFSA increases aggregate Elo from 1178 to 1216, reduces Improvability from 0.133 to 0.106, and improves average rank from 4.40 to 3.90 relative to frozen TabICLv2 (Appendix Table 8). Appendix E.2 reports the full baseline comparison together with uncertainty and rank analyses.

The split-specific results locate these gains (Table 1). The aggregate IID result is nearly unchanged from frozen TabICLv2. On grouped splits, Elo rises from 1085 to 1262 and average rank improves from 6.73 to 4.00. On temporal splits, Elo rises from 780 to 906 and average rank improves from 7.50 to 5.70. The highest-Elo released boosted-tree baseline nevertheless retains higher Elo on the temporal subset. The largest subset-level differences occur on the grouped and temporal splits, while the IID result remains nearly unchanged.

Table 1: BeyondArena enhancement over frozen TabICLv2 by split. Each row uses an independently fitted Elo scale over all twelve methods.

		Elo ↑			Average rank ↓	
Split	Datasets	TabICLv2	TabFSA	Δ	TabICLv2	TabFSA
All	98	1178	**1216**	+38	4.40	**3.90**
IID	77	1258	1259	+1	3.66	3.65
Grouped	11	1085	**1262**	+177	6.73	**4.00**
Temporal	10	780	906	+126	7.50	5.70

Figure 4 breaks down BeyondArena Elo by dataset size, dimensionality, and feature type. Relative to frozen TabICLv2, gains on several size groups and high-cardinality datasets coexist with lower Elo on medium-sized datasets and datasets with text columns.

TALENT. TabFSA records 136 wins, 15 ties, and 49 losses against frozen TabICLv2 across 200 paired datasets (71.8% tie-adjusted paired score). Across the additional pairwise comparisons, TabFSA attains at least 79% tie-adjusted pairwise wins against every non-TabICLv2 comparator shown, using the datasets available for each method pair (Figure 3). This paired result extends the learned adaptation from the synthetic training tasks to a broad collection of real classification datasets. Appendix D.3 gives method attribution and score provenance.

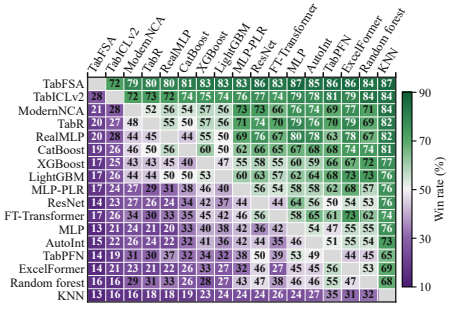


Figure 3: Pairwise TALENT win rates (%). Rows versus columns use common datasets and half credit for ties; TabFSA–TabICLv2 uses 200 datasets.

4.2 ENHANCEMENT MECHANISM ANALYSIS

Expert updates from target training tasks. We evaluate each expert on an independent panel of 160 synthetic episodes, 40 per mechanism family (Table 2).

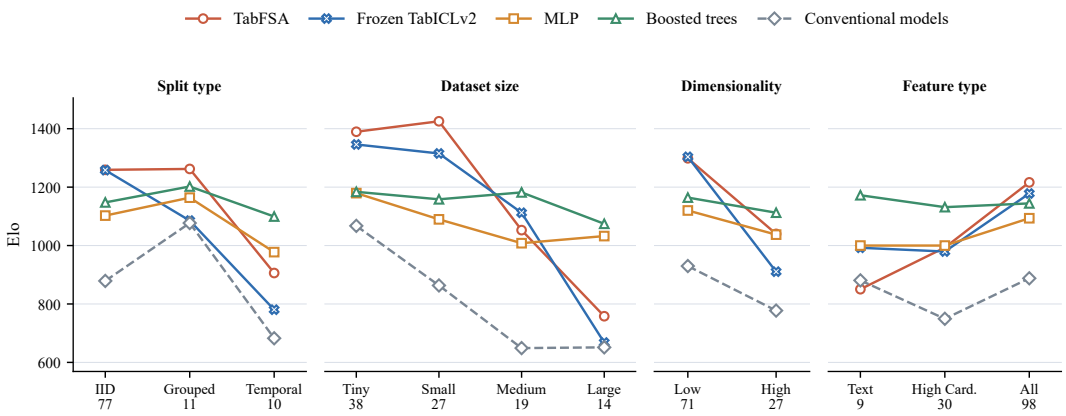


Figure 4: BeyondArena Elo profiles across dataset subsets. Counts indicate datasets. Elo is fitted separately within each subset; each comparator family uses its highest-Elo released default for that subset.

Table 2: Expert specialization across mechanism families: change in mean query NLL from frozen TabICLv2. Negative values favor adaptation; bold marks the lowest NLL in each column.

Expert adapter	Ext.-IID	Temporal	Grouped	High-card.
Ext.-IID	$\mathbf{-0.0631}$	-0.0313	$+0.0727$	-0.0116
Temporal	$+0.0247$	$\mathbf{-0.0800}$	$+0.0925$	$+0.0407$
Grouped	-0.0372	$+0.0174$	$\mathbf{-0.0903}$	-0.0317
High-card.	-0.0278	-0.0063	-0.0400	$\mathbf{-0.0366}$

The matching-family expert gives the lowest mean NLL for every mechanism family. Several off-family applications increase NLL. Together, these results show that the target training distributions produce specialized expert updates whose predictive effects depend on the task family.

Ablation study on learning task-conditioned strategies. The fixed-strategy exploration in Figure 2 shows that favorable expert compositions differ across tasks. Table 3 next asks whether learning this task dependence is more effective than using shared, fixed, or discrete strategies for the same bank. On 40 held-out synthetic tasks and 40 selected BeyondArena datasets, the task-independent baselines comprise fixed single-adapter choices, equal allocation (Uniform), and one learned strategy shared across tasks (Learned global mixture). Hard selection + strength retains the predicted strength and selects only the expert with the largest allocation weight. Appendix D.4 gives the panel protocol. We report the relative reduction in aggregate mean query NLL against frozen TabICLv2, $R_m = 100(1 - \overline{L}_m/\overline{L}_{\text{Frozen}})$, where $\overline{L}$ gives each task or dataset equal weight; higher is better, and negative values indicate higher mean NLL than frozen TabICLv2.

Table 3: Relative query-NLL reduction (%): adaptation strategies.

Strategy	Synth. (40)	Real (40)
TabFSA	**2.25**	**2.24**
Hard selection + strength	1.45	1.24
Learned global mixture	2.05	0.08
Uniform allocation	-0.34	-0.98
Fixed Ext.-IID	2.07	0.10
Fixed temporal	1.51	-6.63
Fixed grouped	0.59	-0.43
Fixed high-cardinality	0.65	1.59

Table 4: Relative query-NLL reduction (%): strategy-feature ablations.

Variant	Synth. (40)	Real (40)
TabFSA	**2.25**	**2.24**
Support summary only	0.88	1.00
Support–query only	0.76	0.71
Feature view only	0.78	1.10
Sample view only	1.37	1.37
Frozen-path comparison	1.46	1.29

On the real panel, the task-conditioned strategy reaches 2.24%, compared with 0.08% for the learned global mixture shared across tasks. Restricting the task-conditioned allocation to its highest-weight expert gives 1.24%, while the best fixed expert reaches 1.59%. On the synthetic panel, the global mixture is closer (2.05% versus 2.25%). These comparisons favor task-conditioned, continuous allocation. On Covertype, swapping strategies between query subsets raises NLL with support fixed (Appendix E.5).

Ablation study on task information for strategy prediction. Using the same tasks and examples, Table 4 asks which task information supports strategy prediction. Support summary only and Support–query only retain one of the two information sources; Feature view only and Sample view only retain one aggregation view; and Frozen-path comparison replaces the Symmetric label-free path with the Frozen prediction path. Each variant is trained separately under the same configuration. On the real panel, retaining only one information source reduces the gain from 2.24% to 1.00% with the support summary or 0.71% with the support–query difference. Retaining only the feature or sample view gives 1.10% and 1.37%, respectively. These results support retaining both information sources and both aggregation views. The Symmetric label-free path also gives a larger reduction than the Frozen prediction path (2.24% versus 1.29%; Appendix E.3.2).

Ablation study on training data composition. Table 5 changes the training data composition under the main experimental configuration. The full mixture gives the largest NLL reductions, supporting the inclusion of both original-prior and extended tasks. Original prior only training yields low strength on most tasks, with applied updates concentrated on the extended-IID direction and little real-panel gain. New tasks only training collapses to a single high-strength, fixed expert combination, identical on both panels: it improves average NLL but applies the same update regardless of task. The full mixture learns task-dependent strengths and allocations, with a larger real-panel gain (2.24% versus 0.65%; Appendix E.3.3).

Table 5: Training data composition: relative query-NLL reduction (%).

Training data	S-40	R-40
Full mixture	**2.25**	**2.24**
Original prior only	0.47	−0.02
New tasks only	0.79	0.65

Symbolic analysis of the learned strategy. We use PySR (Cranmer, 2023), a symbolic-regression method, to fit compact surrogates of the strategy generator outputs. The surrogates are fitted from standardized strategy features $\mathbf{x}$, obtained by per-fold z-score normalization of the dual-view features $\mathbf{z}$. The best strength fold (fold 0, validation mean absolute error (MAE) 0.0365) gives, with $\mathbf{x} = \mathbf{x}^{(0)}$,

$$\widehat{s}^{(0)} = \tanh\left(x_{183}\right) + \tanh\left(x_{316}\right). \tag{12}$$

Here x_{183} belongs to the feature view and x_{316} to the sample view, and strength increases with either coordinate, with diminishing sensitivity as each term saturates. The allocation surrogates show the same pattern: every expert's fit combines both views (Appendix E.6), consistent with the ablation finding that both views are needed for the full gain (Table 4).

Cross-backbone transfer to TabPFN. To test whether FSA is tied to one base model, we instantiate FSA on TabPFN. The strategy-feature constructor, Strategy Generator, and expert adapters are all retrained from scratch on TabPFN. After training, we evaluate TabPFN-FSA on both the synthetic and real panels: relative to the frozen TabPFN, it reduces query NLL by 4.6% on the 40 synthetic datasets and by 2.2% on the 40 real datasets (Appendix E.7), showing that the framework transfers across backbones and can enhance the generalization of an existing tabular foundation model.

5 Conclusion

FSA learns specialized expert updates from target training tasks and controls their strength and allocation through a dual-view strategy feature. TabFSA improves over frozen TabICLv2 on BeyondArena and TALENT, with controlled analyses supporting expert specialization and task-conditioned composition. Retraining the adapters and strategy modules on frozen TabPFN also lowers query NLL on synthetic and real panels, extending FSA beyond one backbone. FSA thus turns target data into reusable, task-conditioned adaptation without backbone retraining or parameter optimization on unseen tasks. However, TabFSA incurs substantial computational overhead and does not support regression tasks. Addressing these limitations is a direction for future work.

AI use statement. Generative AI tools, including OpenAI Codex, were used for text polishing and code review. The authors reviewed the assistance and take responsibility for the final text, claims, implementation, artifacts, and results.

Ethics statement. The study is computational and uses public benchmarks together with synthetic tasks. Tabular predictors can inherit bias and privacy risks; all benchmark data remain subject to their original licenses and access conditions.

Reproducibility statement. Appendices A–C describe the model and training procedure. Appendices D–E document the evaluation and additional experiments.

REFERENCES

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

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation, 2013. URL https://arxiv.org/abs/1308.3432.

David Bonet, Daniel Mas Montserrat, Xavier Giró-i Nieto, and Alexander G Ioannidis. HyperFast: Instant classification for tabular data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(10):11114–11123, 2024. doi: 10.1609/aaai.v38i10.28988. URL https://doi.org/10.1609/aaai.v38i10.28988.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. doi: 10.1023/A:1010933404324. URL https://doi.org/10.1023/A:1010933404324.

Eric L. Buehler and Markus J. Buehler. X-LoRA: Mixture of low-rank adapter experts, a flexible framework for large language models with applications in protein mechanics and molecular design, 2024. URL https://arxiv.org/abs/2402.07148.

Rujikorn Charakorn, Edoardo Cetin, Yujin Tang, and Robert Tjarko Lange. Text-to-LoRA: Instant transformer adaption. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 7485–7514. PMLR, 2025. URL https://proceedings.mlr.press/v267/charakorn25a.html.

Jintai Chen, Jiahuan Yan, Qiyuan Chen, Danny Z. Chen, Jian Wu, and Jimeng Sun. Can a deep learning model be a sure bet for tabular prediction? In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, pp. 288–296, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704901. doi: 10.1145/3637528.3671893. URL https://doi.org/10.1145/3637528.3671893.

Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 785–794, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939785. URL https://doi.org/10.1145/2939672.2939785.

D. R. Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, 20(2):215–232, 07 1958. ISSN 0035-9246. doi: 10.1111/j.2517-6161.1958.tb00292.x. URL https://doi.org/10.1111/j.2517-6161.1958.tb00292.x.

Miles Cranmer. Interpretable machine learning for science with pysr and symbolicregression.jl, 2023. URL https://arxiv.org/abs/2305.01582.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(1):1–30, 2006. URL http://jmlr.org/papers/v7/demsar06a.html.

Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006. doi: 10.1007/s10994-006-6226-1. URL https://doi.org/10.1007/s10994-006-6226-1.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 18932–18943. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/9d86d83f925f2149e9edb0ac3b49229c-Paper.pdf.

Yury Gorishniy, Ivan Rubachev, and Artem Babenko. On embeddings for numerical features in tabular deep learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 24991–25004. Curran Associates, Inc., 2022. doi: 10.52202/068431-1812. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/9e9f0ffc3d836836ca96cbf8fe14b105-Paper-Conference.pdf.

Yury Gorishniy, Ivan Rubachev, Nikolay Kartashev, Daniil Shlenskii, Akim Kotelnikov, and Artem Babenko. TabR: Tabular deep learning meets nearest neighbors. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 18209–18249, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/4ef594af0d9a519db8fb292452c461fa-Paper-Conference.pdf.

Yury Gorishniy, Akim Kotelnikov, and Artem Babenko. TabM: Advancing tabular deep learning with parameter-efficient ensembling. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 77899–77935, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/c1ba41c694834aeef91ae161711d4939-Paper-Conference.pdf.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Mihir Manium, Shi Bin Hoo, Magnus Bühler, Anurag Garg, Dominik Safaric, Jake Robertson, Benjamin Jäger, Simone Alessi, Adrian Hayler, Vladyslav Moroshan, Lennart Purucker, Philipp Singer, Alan Arazi, Julien Siems, Jan Hendrik Metzen, Georg Grab, Nick Erickson, Siyuan Guo, Eliott Kalfon, Simon Bing, David Salinas, Clara Cornu, Lilly Charlotte Wehrhahn, Diana Kriuchkova, Kursat Kaya, Lydia Sidhoum, Marie Salmon, Jerry Chen, Madelon Hulsebos, Yann LeCun, Samuel Müller, Bernhard Schölkopf, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. Tabpfn-3: Technical report, 2026. URL https://arxiv.org/abs/2605.13986.

David Ha, Andrew M. Dai, and Quoc V. Le. Hypernetworks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rkpACe1lx.

Rundong He and Jieming Shi. Prior-free tabular test-time adaptation. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 69478–69494, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/718dd6fa696c6303ca13ad81836014ee-Abstract-Conference.html.

Kai Helli, David Schnurr, Noah Hollmann, Samuel Müller, and Frank Hutter. Drift-resilient TabPFN: In-context learning temporal distribution shifts on tabular data. *Advances in Neural Information Processing Systems*, 37:98742–98781, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/b2e2774c8e76afe191b5bf518f5cb727-Paper-Conference.pdf.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs), 2016. URL https://arxiv.org/abs/1606.08415.

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_.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025. doi: 10.1038/s41586-024-08328-6. URL https://doi.org/10.1038/s41586-024-08328-6.

David Holzmüller, Léo Grinsztajn, and Ingo Steinwart. Better by default: Strong pre-tuned mlps and boosted trees on tabular data. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 26577–26658. Curran Associates, Inc., 2024. doi: 10.52202/079017-0837. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/2ee1c87245956e3eaa71aaba5f5753eb-Paper-Conference.pdf.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. LoraHub: Efficient cross-task generalization via dynamic LoRA composition. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=TrloAXEJ2B.

Benjamin Jäger, Nick Erickson, Léo Grinsztajn, Felix Birkel, Klemens Flöge, Oscar Key, Kürşat Kaya, Jonas Kübler, Adèle Frankel, Tobias Schröder, Anurag Garg, Jan Hendrik Metzen, David Salinas, Simon Bing, Kristina Collins, Tuana Çelik, Vahid Balazadeh, Lydia Sidhoum, Tomás Pereda, Brendan Roof, Andrej Tschalzev, Siyuan Guo, Philipp Singer, Lennart Purucker, Jake Robertson, Marie Salmon, Philipp Jund, Jerry Chen, Diana Kriuchkova, Arthur Cahu, Eliott Kalfon, Adrian Hayler, Georg Grab, Vitor Monteiro, Lilly Wehrhahn, Dominik Safaric, Clara Cornu, Alan Arazi, Rylee Grace, Simone Alessi, Mihir Manium, Bernhard Schölkopf, Yann LeCun, Madelon Hulsebos, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. Tabpfn-3.5: Technical report, 2026. URL https://arxiv.org/abs/2609.17895.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30, pp. 3146–3154. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

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 *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3744–3753. PMLR, 2019. URL https://proceedings.mlr.press/v97/lee19d.html.

Seunghan Lee. Mitigating label shift in tabular in-context learning via test-time posterior adjustment, 2026. URL https://arxiv.org/abs/2605.04363.

Si-Yang Liu, Hao-Run Cai, Qi-Le Zhou, Huai-Hong Yin, Tao Zhou, Jun-Peng Jiang, and Han-Jia Ye. TALENT: A tabular analytics and learning toolbox. *Journal of Machine Learning Research*, 26 (226):1–16, 2025. URL http://jmlr.org/papers/v26/25-0512.html.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Junwei Ma, Valentin Thomas, Rasa Hosseinzadeh, Alex Labach, Hamidreza Kamkari, Jesse C. Cresswell, Keyvan Golestan, Guangwei Yu, Anthony L. Caterini, and Maksims Volkovs. Tabdpt: Scaling tabular foundation models on real data, 2026. URL https://arxiv.org/abs/2410.18164.

Andreas Müller, Carlo Curino, and Raghu Ramakrishnan. MotherNet: Fast training and inference via hyper-network transformers. In *International Conference on Learning Representations*, volume 2025, pp. 76666–76686, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/bed8e95c6a99df485a6fc8b973e2d6cb-Abstract-Conference.html.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=KSugKcbNf9.

Duong Nguyen, Mohammed Jawhar, and Nicolas Chesneau. TFM-Retouche: A lightweight input-space adapter for tabular foundation models, 2026. URL https://arxiv.org/abs/2605.06047.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. AdapterFusion: Non-destructive task composition for transfer learning. In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty (eds.), *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 487–503, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.39. URL https://aclanthology.org/2021.eacl-main.39/.

Jason Phang, Yi Mao, Pengcheng He, and Weizhu Chen. HyperTuning: Toward adapting large language models without back-propagation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 27854–27875. PMLR, 2023. URL https://proceedings.mlr.press/v202/phang23a.html.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. CatBoost: unbiased boosting with categorical features. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31, pp. 6638–6648. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/14491b756b3a51daac41c24863285549-Paper.pdf.

Lennart Purucker, Andrej Tschalzev, Nick Erickson, Gioia Blayer, David Holzmüller, Alan Arazi, Alexander Pfefferle, Mustafa Tajjar, Gaël Varoquaux, and Frank Hutter. Beyond IID: How general are tabular foundation models, really?, 2026. URL https://arxiv.org/abs/2606.30410.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICL: A tabular foundation model for in-context learning on large data. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 50817–50847. PMLR, 2025. URL https://proceedings.mlr.press/v267/qu25d.html.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A better, faster, scalable, and open tabular foundation model. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=SxsyLjIfWB.

Ivan Rubachev, Akim Kotelnikov, Nikolay Kartashev, and Artem Babenko. On finetuning tabular foundation models, 2025. URL https://arxiv.org/abs/2506.08982.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986. doi: 10.1038/323533a0. URL https://doi.org/10.1038/323533a0.

Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. AutoInt: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, pp. 1161–1170, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450369763. doi: 10.1145/3357384.3357925. URL https://doi.org/10.1145/3357384.3357925.

Aditya Tanna, Pratinav Seth, Mohamed Bouadi, and Vinay Kumar Sankarapu. Exploring fine-tuning for tabular foundation models. In *Proceedings of the ACM Web Conference 2026*, pp. 8613–8616. Association for Computing Machinery, 2026. ISBN 9798400723070. doi: 10.1145/3774904.3792923. URL https://doi.org/10.1145/3774904.3792923.

Valentin Thomas, Junwei Ma, Rasa Hosseinzadeh, Keyvan Golestan, Guangwei Yu, Maksims Volkovs, and Anthony Caterini. Retrieval & fine-tuning for in-context tabular models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 108439–108467. Curran Associates, Inc., 2024. doi: 10.52202/079017-3442.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://papers.nips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Yifan Wu, Junjie Wu, Kai Wu, Xiaoyu Zhang, and Jian Lou. Feature to Dynamics: Feature-space to autoregression strategy for zero-shot time series forecasting. *arXiv preprint arXiv:2606.01289*, 2026. URL https://arxiv.org/abs/2606.01289.

Derek Xu, Olcay Cirit, Reza Asadi, Yizhou Sun, and Wei Wang. Mixture of in-context prompters for tabular PFNs. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 24630–24665, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/3de60dc2ff7ac8866e805bbf457e2291-Paper-Conference.pdf.

Han-Jia Ye, Huai-Hong Yin, De-Chuan Zhan, and Wei-Lun Chao. Revisiting nearest neighbor for tabular data: A deep tabular baseline two decades later. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 94917–94936, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/ecd294194d5155432aab533f7ce0f6fe-Paper-Conference.pdf.

Xiyuan Zhang, Danielle Maddix Robinson, Junming Yin, Nick Erickson, Abdul Fatir Ansari, Boran Han, Shuai Zhang, Leman Akoglu, Christos Faloutsos, Michael Mahoney, Tony Hu, Huzefa Rangwala, George Karypis, and Yuyang (Bernie) Wang. Mitra: Mixed synthetic priors for enhancing tabular foundation models. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 15795–15840. Curran Associates, Inc., 2025. doi: 10.52202/085713-0535. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/177d68f4adef163b7b123b5c5adb3c60-Paper-Conference.pdf.

A Architecture

A.1 TabICLv2 representations

The frozen TabICLv2 backbone follows a column, row, and dataset hierarchy (Qu et al., 2026). Repeated feature grouping combines each feature with cyclic offsets 1 and 3, retaining m groups. Support target embeddings enter before the column-wise embedding Transformer, whose three blocks produce a 128-dimensional vector for each row-feature-group pair. The row-wise interaction Transformer uses three layers and four summary tokens; concatenating their outputs gives a 512-dimensional row vector. The final dataset-wise ICL stack performs prediction across support and query rows and has 12 layers.

For ensemble view v, the Frozen prediction path and Symmetric label-free path have matching output shapes:

$$E^{+,(v)}, E^{0,(v)} \in \mathbb{R}^{n\times m\times 128}, \qquad H^{+,(v)}, H^{0,(v)} \in \mathbb{R}^{n\times 512}. \tag{13}$$

The Frozen prediction path follows TabICLv2's target-aware front end, and its support and query row representations provide the input to dataset-wise ICL prediction. The Symmetric label-free path differs in two ways: it disables target-aware support embedding, and its column-level inducing representations attend to support and query together in one unlabeled column context. Both paths otherwise use the same preprocessing and frozen front-end weights.

Here $n = n_s + n_q$ counts all rows in the task context, and $m = m_v$ counts feature groups after input preprocessing and repeated grouping. Support and query masks identify row subsets; validity masks exclude padding. Originally missing values are encoded as observations, while padding masks

identify absent positions. Define $\mathcal{J}_v$ to contain groups with at least one valid observation in each partition. For subset $a \in \{s, q\}$, mode $b \in \{+, 0\}$, and $j \in \mathcal{J}_v$, the feature mean is

$$\bar{e}^{b,(v)}_{a,j} = \frac{\sum_{i \in \mathcal{I}_a} M^{(v)}_{ij} E^{b,(v)}[i,j]}{\sum_{i \in \mathcal{I}_a} M^{(v)}_{ij}}. \tag{14}$$

Here $\mathcal{I}_a$ indexes the relevant row partition and $M^{(v)}_{ij} \in \{0, 1\}$ marks valid row-feature positions. The definition on $\mathcal{J}_v$ guarantees a positive denominator. Attention and pooling masks exclude other groups and padded positions. The feature-view input combines the label-aware support summary with the label-free support–query difference.

A.2 Feature and sample branches

Both encoders use width 256, four attention heads, feed-forward width 512, and GELU activations (Hendrycks & Gimpel, 2016). The feature branch normalizes the 256-dimensional concatenation of a label-aware support summary and a label-free feature-difference statistic. The 256-to-256 projection P_f is followed by one SAB and one-seed PMA from the Set Transformer construction (Lee et al., 2019):

$$\begin{gathered} \delta^{(v)}_{f,j} = \bar{e}^{0,(v)}_{q,j} - \bar{e}^{0,(v)}_{s,j}, \qquad r^{(v)}_{f,j} = P_f \,\mathrm{LN}_f\big([\bar{e}^{+,(v)}_{s,j}; \delta^{(v)}_{f,j}]\big), \\ \mathbf{z}^{(v)}_f = \mathrm{PMA}_{f,1}\big(\mathrm{SAB}(\{r^{(v)}_{f,j}\}_{j \in \mathcal{J}_v})\big) \in \mathbb{R}^{256}. \end{gathered} \tag{15}$$

Layer normalization precedes projection; attention then relates and pools the feature tokens.

The sample branch projects 512-dimensional row vectors to width 256 and uses one ISAB with 16 inducing points. We write $\mathrm{MAB}(Q, K)$ for a multi-head attention block with Q as queries and K providing both keys and values. The ISAB operations are

$$\begin{aligned} U &= \mathrm{MAB}_1(I, \widetilde{H}), \\ \mathrm{ISAB}(\widetilde{H}) &= \mathrm{MAB}_2(\widetilde{H}, U). \end{aligned} \tag{16}$$

Here, $I \in \mathbb{R}^{16 \times 256}$ is learned and $\widetilde{H} = P_s \,\mathrm{LN}_s(H)$. Linear projections and normalizations act on the final dimension. The first block summarizes the row tokens through the inducing points, whereas the second propagates the induced representation back to the row tokens, following the ISAB construction of Lee et al. (2019).

In our configuration, each MAB uses pre-normalization with residual connections around its multi-head attention and feed-forward sublayers. Let $\mathrm{MHA}(Q, K, V)$ denote standard multi-head attention (Vaswani et al., 2017). Then

$$\begin{aligned} U_Q &= Q + \mathrm{MHA}\big(\mathrm{LN}_Q(Q), \mathrm{LN}_K(K), \mathrm{LN}_V(K)\big), \\ \mathrm{MAB}(Q, K) &= U_Q + \mathrm{FFN}\big(\mathrm{LN}_{\mathrm{FF}}(U_Q)\big). \end{aligned} \tag{17}$$

The feed-forward sublayer maps $256 \to 512 \to 256$, with GELU between the two linear layers. Each block has its own normalization parameters.

SAB is self-attention over the feature-token set, $\mathrm{SAB}(R) = \mathrm{MAB}_f(R, R)$. For branch $b \in \{f, s\}$, one-seed PMA includes its output normalization:

$$\mathrm{PMA}_{b,1}(R) = \mathrm{LN}_{\mathrm{out},b}\big(\mathrm{MAB}_{b,\mathrm{pool}}(S_b, R)\big), \qquad S_b \in \mathbb{R}^{1 \times 256}. \tag{18}$$

The singleton seed dimension is removed after pooling. The branches have separate seeds and attention parameters, reused across ensemble views. Validity masks exclude padding from attention keys and pooled means; padded rows are masked from subsequent pooling.

The three evaluations of g_η use the same parameters. Each produces a 256-dimensional pooled vector with output normalization:

$$g_\eta(H) = \mathrm{PMA}_{s,1}\left(\mathrm{ISAB}(P_s \,\mathrm{LN}_s(H))\right). \tag{19}$$

Output normalization precedes subtraction of the label-free summaries. The final sample-view projection in Equation 8 is

$$\delta^{(v)}_s = p^{0,(v)}_q - p^{0,(v)}_s, \qquad \mathbf{z}^{(v)}_s = P_r \,\mathrm{LN}_r([p^{+,(v)}_s; \delta^{(v)}_s]) \in \mathbb{R}^{256}, \tag{20}$$

where P_r maps the 512-dimensional concatenation to width 256. For a row set of size n_a, ISAB uses $16n_a$ attention pairs in each direction, giving linear row-count scaling at fixed width.

A.3 Strategy Generator and updates

The Strategy Generator in Equation 10 applies input layer normalization and two hidden linear layers of width 256 with GELU activations. Separate output heads produce $a_s \in \mathbb{R}$ and $\mathbf{a}_\pi \in \mathbb{R}^4$. The numerical parameterization approaches the simplex boundary as logits become large in magnitude, and $s = 0$ recovers frozen TabICLv2. These outputs define the continuous adaptation strength and expert allocation used in Equation 4.

Every expert adapter has rank $r = 8$ and scale $\alpha = 8$. Updates act on the packed query-key-value projection and attention output projection in the final four ICL layers, numbered 9–12. The four adapters share these locations and dimensions. Let ℓ index these adapted weight matrices, matching Figure 1. For $W_{0,\ell} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, expert adapter k has factors and low-rank update

$$A_{\ell,k} \in \mathbb{R}^{r \times d_{\text{in}}}, \qquad B_{\ell,k} \in \mathbb{R}^{d_{\text{out}} \times r}, \qquad \Delta W_{\ell,k} = (\alpha/r) B_{\ell,k} A_{\ell,k}. \tag{21}$$

For each adapted matrix, the task-conditioned expert-adapter composition adds $\sum_{k=1}^{4} w_k \Delta W_{\ell,k}$ to the frozen weight matrix $W_{0,\ell}$. The pretrained matrix retains coefficient one, and w_0 contributes the zero update. The expert coefficients sum to s, so s controls the total coefficient assigned to the expert updates. Factors remain separate across expert adapters; their weighted matrix products define the combined update. The packed query-key-value matrix has shape 1536×512 and the output matrix 512×512. Each per-matrix update $\Delta W_{\ell,k}$ has rank at most eight; the weighted sum for matrix ℓ has rank at most 32.

A.4 Conditional risk

For a fixed strategy space, the task-specific optimum minimizes the conditional prediction risk:

$$\mathbf{w}^\star(\mathcal{T}) \in \arg\min_{\mathbf{w} \in \mathcal{S}} R_{\mathcal{T}}(\mathbf{w}). \tag{22}$$

For classification, the conditional risk is

$$R_{\mathcal{T}}(\mathbf{w}) = \mathbb{E}_{Y_q|\mathcal{T}} \left[-\frac{1}{n_q} \sum_{i=1}^{n_q} \log \widehat{P}_q(\mathcal{T}; \mathbf{w})[i, y_{q,i}] \right]. \tag{23}$$

Here, $\mathcal{T} = (\mathcal{D}_s, X_q)$ is the task context defined in Section 3.1, and $\widehat{P}_q(\mathcal{T}; \mathbf{w})$ denotes the predicted class probabilities in Equation 27 of Appendix C.3. The expectation is over the query labels conditioned on the task context. Equation 22 defines the task-specific optimum within the strategy space.

Feature-to-Strategy Adaptation learns a shared mapping for the task-specific strategy:

$$\mathbf{w} = G_\phi(\Phi_\eta(\mathcal{T}; F_{\theta_0})), \tag{24}$$

where Φ_η constructs the strategy feature and the Strategy Generator G_ϕ maps it to the strategy space. Equation 11 trains this mapping directly from sampled query losses over p_{mix}, resulting in a continuous adaptation strategy for each unseen task.

B Synthetic Mechanisms

B.1 Base tasks and transformations

Synthetic construction begins with the graph-based structural causal model prior used by TabICLv2 (Qu et al., 2026), which samples variables, a directed computation graph, response mechanisms, and latent variation. A mechanism family turns a sampled base task into a target training task by specifying a transformation T_μ and a support–query split rule S_μ. The transformation can modify the response, covariate distribution, dependence across observations, categorical frequencies, or missingness. The split determines the evidence available in support and the prediction relation evaluated in query.

Across episodes, task size, feature count, class count, noise, missingness, class imbalance, and categorical cardinality vary. Sampling the base task and mechanism parameters changes the instance and



Figure 5: Synthetic task construction. Examples illustrate response structure, temporal change, group effects, and categorical variation. Colors denote class labels.

severity of each problem. The family names index the corresponding specialization distributions, and the following sections specify their transformation and split logic. Figure 5 illustrates representative tasks from these families.

B.2 Original-prior and extended-IID

Original-prior episodes leave the sampled task and its original support and query partition unchanged. They retain the original prior alongside the specialized mechanisms in the training distribution for the feature-to-strategy mapping.

Extended-IID episodes vary the prediction task while keeping support and query sampling within a common population. Response transformations include sparse linear combinations, monotone additive functions, low-rank interactions, piecewise thresholds, and tree-based rules. Noise mechanisms vary tail behavior or make variance depend on the inputs.

Observation mechanisms vary missingness, class imbalance, and label noise. Feature transformations add irrelevant, redundant, or collinear variables and mix numerical, binary, and categorical inputs. Missingness can be independent of observed values, depend on other observed features, or depend on the value being hidden.

B.3 Temporal episodes

Temporal mechanisms represent prediction from earlier observations to later ones. An ordered index controls the changing process. Abrupt change replaces parameters at a change point; gradual change interpolates between processes over an interval; recurrent change revisits a previous process.

Pure covariate shift changes the input marginal $p(X)$ while preserving $p(Y \mid X)$; pure label shift changes $p(Y)$ while preserving $p(X \mid Y)$. Concept drift changes the conditional relationship $p(Y \mid X)$. Family names identify the primary targeted change, while compositional episodes can change several distributions simultaneously. Time-dependent missingness changes the observation process.

Task sampling varies the source process and temporal mechanism. Support observations precede query observations.

B.4 GROUPED EPISODES

Grouped mechanisms create dependence among observations through shared group factors. A group-level variable can shift several features, correlate noise, or alter the response function. Other mechanisms vary class balance or missingness across groups. Sampling a new base task and group effects changes both within-group similarity and between-group differences.

For unseen-group transfer, entire groups are assigned to query and the remaining groups to support. For mechanisms studying within-group correlation, group structure is retained in both partitions. These constructions pose different inference problems: the former requires transfer beyond observed groups, while the latter changes dependence among the examples available to the predictor.

B.5 HIGH-CARDINALITY EPISODES

High-cardinality mechanisms control the number, frequency, and predictive role of categorical levels. Long-tailed frequency distributions create a few common levels and many rare levels. Other transformations couple category identities to numerical variables or make category effects vary by group or time.

Category identities are constructed before partitioning. In shared-level episodes, both partitions sample from the categorical population, with rare levels receiving limited support evidence. Query-novelty episodes instead reserve selected levels for query, making their support count zero. Thus, high cardinality, rarity, and novelty are separate properties: a high-cardinality feature can have all its categories observed in support, and a lower-cardinality feature can still contain unseen query levels. These levels belong to categorical input features; target classes retain their task-level definition. Sampling the task and category mechanism varies these properties across episodes.

B.6 COMPOSITIONAL EPISODES

Compatible transformations are applied to one base task before the split. Examples include extended-IID tasks with nuisance features and imbalance, temporal tasks with high-cardinality categorical features or missingness, and grouped tasks with categorical interactions. When temporal or grouped mechanisms are included, the final split preserves chronological order or group separation, respectively. Combinations can therefore vary response, observation, and category structure while retaining the relevant transfer constraint.

C TRAINING DETAILS

C.1 GRADIENT FLOW

Each expert adapter is trained on its corresponding specialization distribution with the other adapters inactive and the TabICLv2 parameters frozen. The backbone remains in the forward path, and query negative log-likelihood updates the active adapter's low-rank factors. The four expert adapters have equal capacity but are trained on different specialization distributions.

After expert-adapter learning, the TabICLv2 backbone and all four adapters remain frozen. The strategy-feature constructor and Strategy Generator are optimized using query NLL from the adapted predictor. The constructor comprises the feature-branch encoder, the shared sample-branch encoder, and the sample-branch output projection P_r with its normalization.

The learned ablations use the same strategy-learning configuration as TabFSA, and each variant is trained separately. The straight-through estimator is used only by the separately trained hard-output variant in Appendix E.3.2; it is distinct from the post-hoc Hard selection + strength transformation in Table 3. Appendix D.4 gives the controlled evaluation panels, and Appendix E.3.2 gives the variant training protocol.

For the continuous strategy parameterization, the query loss depends on the generated strategy through the frozen low-rank updates. For encoder parameters η,

$$\frac{\partial \ell_q}{\partial \eta} = \frac{\partial \ell_q}{\partial \mathbf{w}} \frac{\partial G_\phi(\mathbf{z})}{\partial \mathbf{z}} \frac{\partial \Phi_\eta}{\partial \eta}. \tag{25}$$

In particular, $\partial W_\ell(\mathbf{w})/\partial w_k = \Delta W_{\ell,k}$ for $k = 1, \ldots, 4$; the frozen adapter matrices provide the update directions for these strategy gradients. The query NLL trains the strategy-feature constructor and Strategy Generator; the strategy itself is computed from $\mathcal{D}_s$ and X_q.

For the generator's output logits, the derivatives of an adapted matrix are

$$\begin{aligned} \frac{\partial W_\ell}{\partial a_s} &= s(1-s)\sum_{k=1}^{4}\pi_k \Delta W_{\ell,k}, \\ \frac{\partial W_\ell}{\partial a_{\pi,k}} &= s\pi_k\left(\Delta W_{\ell,k} - \sum_{h=1}^{4}\pi_h \Delta W_{\ell,h}\right). \end{aligned} \tag{26}$$

C.2 Optimization

Both expert-adapter specialization and strategy learning use AdamW (Loshchilov & Hutter, 2019) with a learning rate of 3×10^{-4}, a batch size of 8 episodes, and gradient clipping at 1.0 on two NVIDIA RTX 3090 GPUs. Each expert adapter is specialized independently for 120,000 updates with the TabICLv2 backbone frozen. The resulting expert-adapter bank is then frozen. Strategy learning updates the strategy-feature constructor and Strategy Generator for 60,000 updates, and the 60,000-step checkpoint is evaluated. For the training-progression analysis in Appendix E.4, we continue the same strategy-learning trajectory beyond the main 60k checkpoint and evaluate additional checkpoints up to 100k updates; all main benchmark and ablation results use the 60k model.

Training-distribution sampling. Each expert adapter is trained exclusively on episodes from its corresponding specialization distribution: extended-IID, temporal, grouped, or high-cardinality. Strategy learning uses a mixture of original-prior, single-mechanism, and compositional episodes. Original-prior, extended-IID, temporal, grouped, and high-cardinality episodes are each sampled with probability 0.15, while compositional episodes are sampled with probability 0.25.

Episode sizes. The episode-size distribution is unchanged throughout training. With probability 0.70, an episode contains 256–4,096 examples; with probability 0.25, it contains more than 4,096 and at most 10,240; and with probability 0.05, it contains more than 10,240 and at most 20,480. Size is sampled logarithmically within each range, after which the mechanism-specific rule divides it into support and query.

C.3 Inference protocol

For an unseen target task $\mathcal{T} = (\mathcal{D}_s, X_q)$, TabFSA constructs the strategy feature and predicts $\mathbf{w}$ with the Strategy Generator. The pretrained backbone, expert-adapter factors, feature and sample branches, and Strategy Generator remain frozen, while $\mathbf{w}$ controls the expert updates for the task. The adapted predictor returns

$$\widehat{P}_q = f_{\theta_0,\mathcal{A}}(\mathcal{D}_s, X_q; \mathbf{w}). \tag{27}$$

At evaluation, the strategy and predictions are computed from $(\mathcal{D}_s, X_q)$ with frozen parameters; the query labels then score those predictions. Predictions from ensemble views are aligned by class and combined using the retained TabICLv2 ensemble protocol (Qu et al., 2026).

Let $L_q^{(v)} \in \mathbb{R}^{n_q \times C}$ be view-specific prediction logits and $\rho_v(c)$ the column corresponding to original class c. Alignment gives $\widetilde{L}_q^{(v)}[i, c] = L_q^{(v)}[i, \rho_v(c)]$. With logit averaging, the backbone ensemble computes

$$\widehat{P}_q = \mathrm{softmax}_{\mathrm{class}}\left(\frac{1}{VT_{\mathrm{soft}}}\sum_{v=1}^{V}\widetilde{L}_q^{(v)}\right), \tag{28}$$

where $T_{\mathrm{soft}} > 0$ is the backbone's softmax temperature. If probability averaging is selected instead, softmax is applied within each view before averaging. The backbone's ensemble setting is retained; class alignment is required in either case.

D Evaluation Protocols

D.1 BeyondArena

The BeyondArena classification subset contains 98 datasets: 73 binary and 25 multiclass. Its split types comprise 77 IID, 11 grouped, and ten temporal datasets. We use the benchmark's released profile metadata. Scale sub-benchmarks use the training size of the first released split: tiny (100–999), small (1,000–9,999), medium (10,000–99,999), and large ($\geq 100{,}000$). Dimensionality uses the column count in the dataset index, before model-specific transformations: low is at most 100 columns and high exceeds 100. The text subset contains a non-target column marked as text; the high-cardinality subset contains a categorical feature with more than 50 distinct levels (Purucker et al., 2026). Sub-benchmarks overlap across these axes.

Inputs. We use BeyondArena's benchmark-defined preprocessing, fitted on the training partition and applied to query inputs (Purucker et al., 2026). Consequently, the feature groups supplied to TabFSA can differ from the raw input columns. For grouped tasks, we retain the benchmark protocol: label-per-group datasets use train-selected aggregation features computed from the observed covariates of each intact query group, whereas group indices are removed for per-row prediction. Strategy features are constructed from the processed task inputs without using benchmark split metadata.

Backbone behavior. Tasks with more than ten classes use TabICLv2's original many-class inference procedure (Qu et al., 2026). Appendix C.3 describes class alignment and ensemble aggregation. BeyondArena evaluations use eight TabICLv2 ensemble estimators, reduced to one for the five largest datasets following the benchmark protocol (Purucker et al., 2026).

Strategy-analysis settings. The temporal context-sensitivity analysis partitions each query set into ten consecutive blocks. The Covertype swapping analysis uses the same 2,048 labeled support examples for 124 head and 132 tail queries, with the query subsets defined by category occurrence counts in support.

Baselines. The BeyondArena comparison imports released dataset-level scores and default configurations from the benchmark (Purucker et al., 2026). The compared neural methods are TabICLv2 (Qu et al., 2026), TabPFN-2.6 (using the benchmark's version label), TabDPT (Ma et al., 2026), RealMLP (Holzmüller et al., 2024), and TabM (Gorishniy et al., 2025). The remaining baselines are CatBoost (Prokhorenkova et al., 2018), LightGBM (Ke et al., 2017), XGBoost (Chen & Guestrin, 2016), random forest (Breiman, 2001), extra trees (Geurts et al., 2006), and logistic regression (Cox, 1958). We retain each released default configuration.

D.2 Metrics and tests

BeyondArena errors are $1 - \mathrm{ROC\text{-}AUC}$ for binary classification and mean negative log-likelihood (log-loss) for multiclass classification. Let e_{dj} denote error for method j on dataset d and $e_d^\star = \min_j e_{dj}$. Each dataset has equal weight in the aggregate summaries. Improvability is

$$\mathrm{Improvability}(j) = \frac{1}{D}\sum_{d=1}^{D}\left(1 - \frac{e_d^\star}{e_{dj}}\right). \tag{29}$$

A zero contribution is assigned when both errors are zero. Lower values indicate less remaining improvement relative to the best error on each dataset. Average rank averages within-dataset ranks, using average ranks for ties. Rank-1 share counts every exact tie for best, so its percentages need not sum to 100. For a common set of D datasets and K methods, the aggregate win rate is

$$\mathrm{WinRate}(j) = \frac{1}{D(K-1)}\sum_{d=1}^{D}\sum_{k\neq j}\left(\mathbf{1}[e_{dj} < e_{dk}] + \tfrac{1}{2}\mathbf{1}[e_{dj} = e_{dk}]\right). \tag{30}$$

Pairwise matrices use the same half-credit rule for their two methods. Improvability intervals are 95% percentile intervals from 100 dataset-bootstrap resamples with random seed 0.

For Elo, pairwise outcomes are fitted with a logistic win-probability model. On the usual Elo scale, the predicted probability that method j defeats method k is $[1 + 10^{(r_k - r_j)/400}]^{-1}$. Ratings are

translated so that default XGBoost has score 1,000. Point estimates use maximum likelihood. Elo intervals use the 2.5th and 97.5th percentiles from 200 dataset-bootstrap resamples with random seed 0. Point estimates are translated to the XGBoost reference while interval widths are retained from the uncalibrated fit. Each resample selects the same datasets for all methods. A subset receives its own fit and comparison population. Each subset therefore has its own Elo scale; comparisons use within-subset orderings and gaps.

Critical-difference diagrams use the Nemenyi threshold at $\alpha = 0.05$ with dataset-level ranks (Demšar, 2006). Their bars identify methods whose average-rank differences fall within that threshold. These tests use datasets as the units of comparison and characterize dataset-level variation. Training-seed and repeated-inference variation are separate axes.

D.3 TALENT

TALENT (Liu et al., 2025) uses fixed 64% training, 16% validation, and 20% test partitions. The training partition supplies support and the test partition supplies queries. We use 32 TabICLv2 ensemble estimators, matching the benchmark configuration used for the frozen backbone. We report the 200 datasets with paired TabFSA and frozen TabICLv2 measurements. Each dataset has equal weight, and paired outcomes use test accuracy. W/T/L denotes wins, ties within 10^{-9}, and losses. The tie-adjusted paired score is $(W + 0.5T)/(W + T + L)$. For the broader comparison in Figure 3, external baseline accuracies are the reported means in TALENT's released classification results, while TabFSA and frozen TabICLv2 use our paired measurements. Each pair uses the datasets available for both methods. Method attribution follows ModernNCA (Ye et al., 2025), TabR (Gorishniy et al., 2024), MLP-PLR (Gorishniy et al., 2022), the tabular ResNet and FT-Transformer (Gorishniy et al., 2021), AutoInt (Song et al., 2019), and ExcelFormer (Chen et al., 2024). The MLP and KNN labels follow the TALENT release (Liu et al., 2025); TabPFN denotes the method of Hollmann et al. (2023) with scores from that release.

D.4 Controlled evaluation panels

Expert specialization in Table 2 uses a separate 160-episode synthetic panel, with 40 episodes from each mechanism family. The strategy-application and strategy-feature analyses use two matched panels. The synthetic panel contains 40 held-out tasks: ten extended-IID, ten temporal, ten grouped, and ten high-cardinality tasks. The real panel contains all 11 grouped and all ten temporal BeyondArena classification datasets together with 19 IID datasets selected across classification type and dataset-size strata, with at most three datasets per stratum, giving 40 selected real datasets. Labeled support is capped at 2,048 examples and query at 1,024 examples. Each task or dataset receives equal weight in its panel mean. Tables 3 and 4 use the same panels and the same support and query examples.

E Additional Experiments

E.1 Fixed Strategy-Space Exploration

This analysis characterizes prediction responses to adaptation strategies after the expert adapters have been learned. The preselected stratified panel contains 10 real tasks. We evaluate a common set of 64 fixed candidate adaptation strategies on these tasks. The preset coefficient vectors were generated once using $s \sim \mathrm{Uniform}(0, 1)$ and $\boldsymbol{\pi} \sim \mathrm{Dirichlet}(1, 1, 1, 1)$. All candidates satisfy $0 < s < 1$ and $\pi_k > 0$. The backbone and expert adapters remain frozen, while the support/query budgets (2,048/1,024) and four ensemble views are fixed. We evaluate each candidate by query-set NLL. The candidates provide controlled probes of the frozen bank; learned strategy prediction is evaluated separately in Tables 3 and 4.

The coefficient vector of each taskwise best candidate appears in Figure 2(b). Table 6 reports the coefficients rounded to four decimal places.

Across the ten evaluated tasks, the lowest-NLL sampled strategy improves over frozen TabICLv2 on eight tasks and over the best direct expert on six. Five tasks satisfy both comparisons. Seven different strategies attain the lowest observed NLL across the ten tasks; Credit Approval and Musk have no improving candidate in the sampled set.

Table 6: Coefficients of the lowest-NLL sampled strategy for each task in Figure 2(b), rounded to four decimal places.

Task	w_0	w_{Ext}	w_{Temp}	w_{Group}	w_{High}
Acquire shoppers	0.1137	0.0535	0.5556	0.0766	0.2006
Amazon access	0.7216	0.0026	0.0415	0.1993	0.0350
Credit Approval	0.9610	0.0093	0.0071	0.0017	0.0209
Dementia	0.3894	0.4747	0.0385	0.0934	0.0040
HomeQ default	0.4491	0.1298	0.1102	0.0297	0.2812
Hotel booking	0.0597	0.1106	0.0344	0.2057	0.5896
Lending Club	0.3894	0.4747	0.0385	0.0934	0.0040
Musk	0.9610	0.0093	0.0071	0.0017	0.0209
Sepsis	0.5862	0.0051	0.2623	0.0775	0.0688
Splice	0.1137	0.0535	0.5556	0.0766	0.2006

Table 7: Task-level relative NLL reduction versus frozen TabICLv2 (%) for 64 sampled strategies and four direct expert baselines. n_+ and n_- count improving and degrading sampled strategies.

Task	Split	Best sampled	Best direct	Worst sampled	n_+	n_-
Acquire shoppers	Temporal	$+0.05$	$+2.97$	-2.96	1	63
Amazon access	IID	$+0.12$	$+0.05$	-2.75	22	42
Credit Approval	IID	-0.08	-1.45	-5.28	0	64
Dementia	Grouped	$+2.39$	$+3.84$	-3.23	55	9
HomeQ default	IID	$+3.29$	-6.20	-10.04	36	28
Hotel booking	Temporal	$+10.92$	$+12.85$	-1.02	59	5
Lending Club	Temporal	$+0.28$	$+0.06$	-5.34	12	52
Musk	Grouped	-1.61	$+11.57$	-43.15	0	64
Sepsis	Grouped	$+2.45$	$+2.31$	-0.15	62	2
Splice	IID	$+4.75$	$+4.39$	-2.46	62	2

E.2 BeyondArena

Table 8 gives the released-default aggregate comparison.

Table 8: BeyondArena classification results for TabFSA and the eleven released default baseline configurations on 98 datasets.

Method	Elo ↑	Improv. ↓	Avg. rank ↓	Rank-1 (%) ↑
TabFSA	**1216**	**0.106**	**3.90**	**37.8**
TabPFN-2.6	1212	0.119	3.95	19.4
TabICLv2	1178	0.133	4.40	9.2
CatBoost	1144	0.181	4.87	12.2
TabM	1094	0.193	5.61	7.1
RealMLP	1019	0.218	6.74	3.1
TabDPT	1002	0.225	6.99	3.1
XGBoost	1000	0.233	7.02	1.0
LightGBM	983	0.244	7.28	1.0
Random forest	888	0.281	8.63	1.0
Linear / logistic	839	0.330	9.27	4.1
Extra trees	832	0.296	9.35	1.0

Figure 6 reports the overall comparison, Figure 7 separates binary and multiclass classification, and Figure 8 separates the three split types. Figure 9 reports pairwise win rates, and Figure 10 compares average ranks with the Nemenyi critical difference.

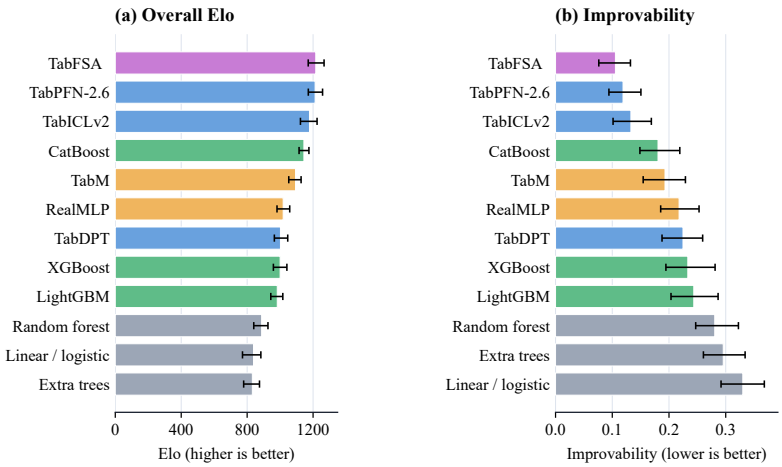


Figure 6: BeyondArena classification comparison on 98 datasets against released default baseline configurations. Left: Elo (higher is better). Right: Improvability (lower is better). Error bars show 95% dataset-bootstrap intervals; colors identify model families.

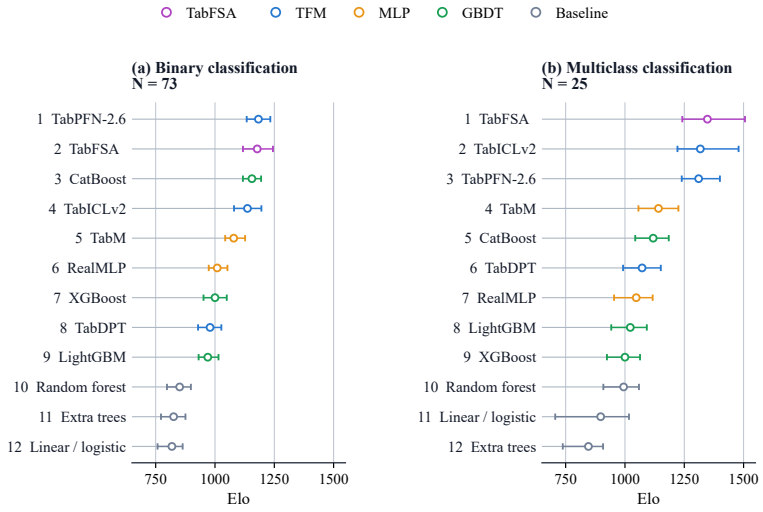


Figure 7: BeyondArena Elo comparisons on binary and multiclass tasks against released default baseline configurations. Methods are ordered within each classification type; error bars show 95% dataset-bootstrap intervals.

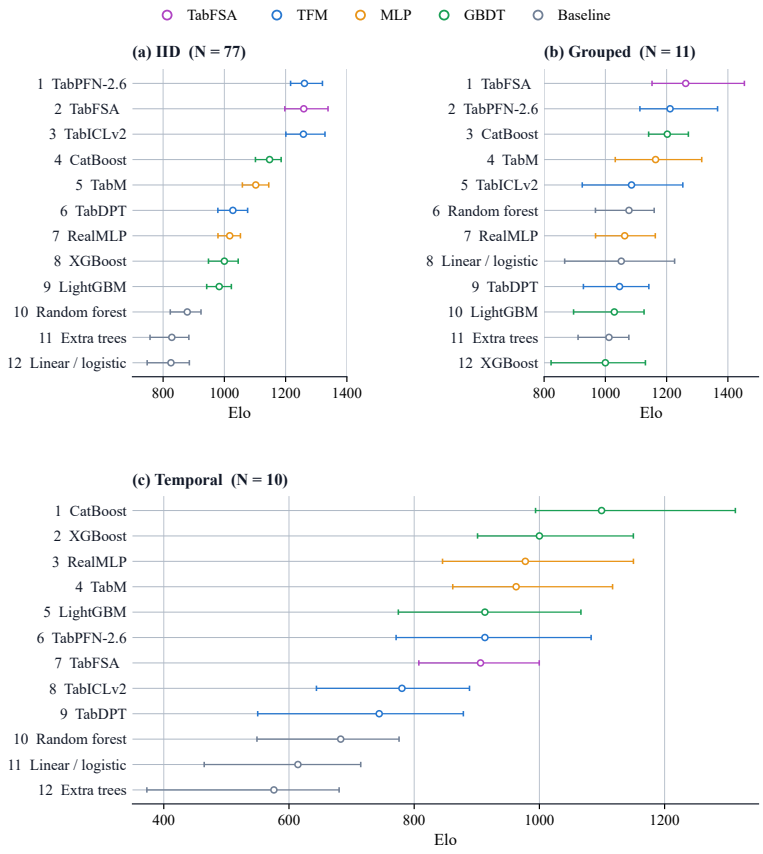


Figure 8: Elo rankings under IID, grouped, and temporal splits, using the default-configuration comparison. Intervals represent 95% dataset-bootstrap uncertainty. Methods are ordered independently within each subset.

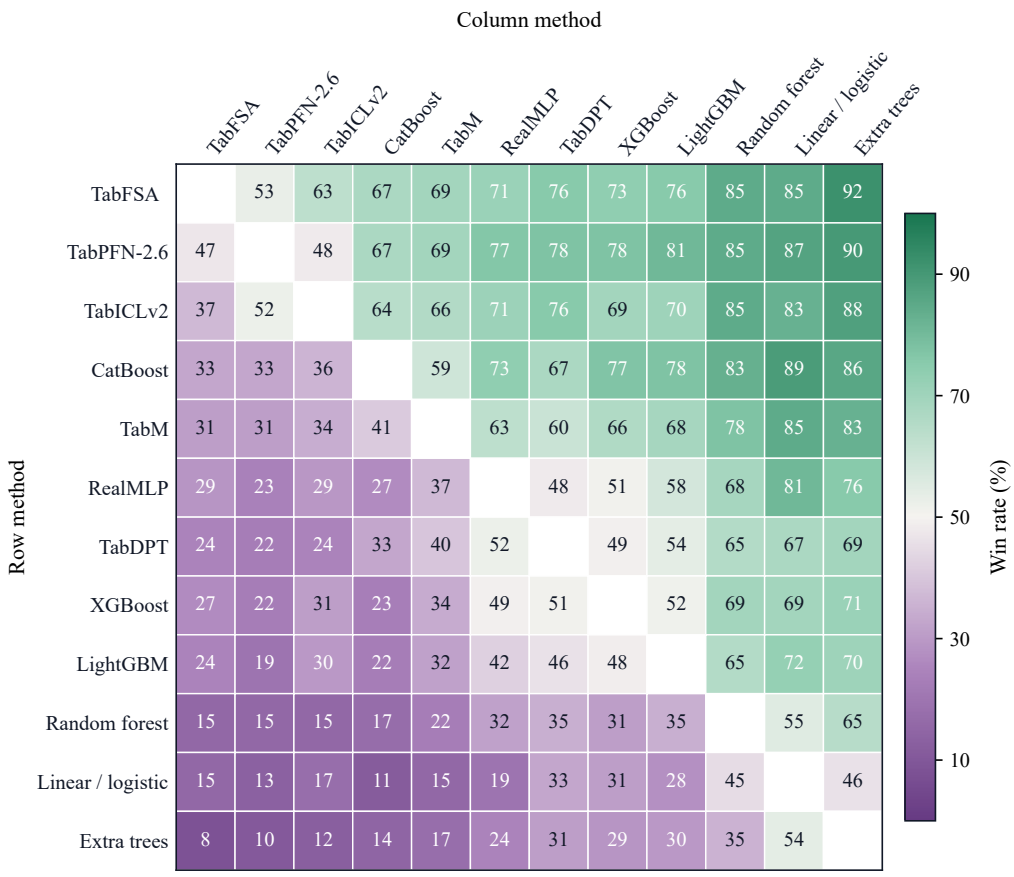
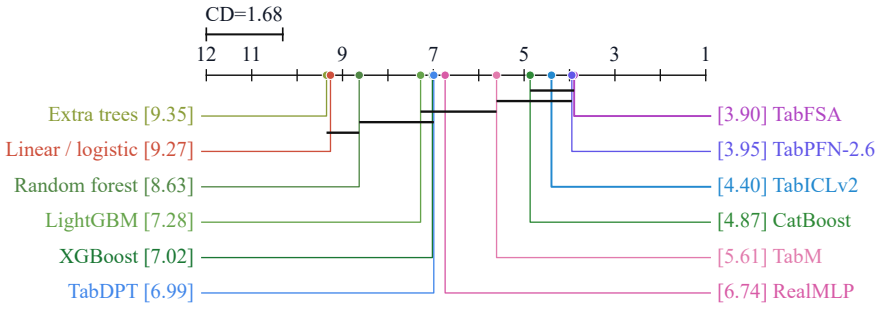
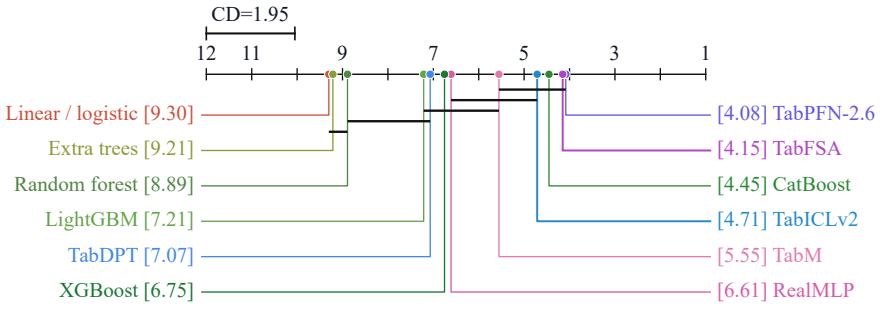


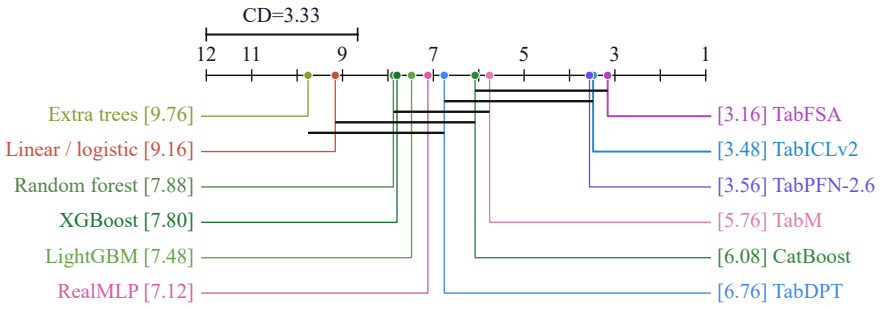
Figure 9: Pairwise win rates (%) on 98 BeyondArena datasets. Rows are compared against columns, with half credit for ties.



(a) All BeyondArena datasets (N=98)



(b) Binary classification datasets (N=73)



(c) Multiclass classification datasets (N=25)

Figure 10: BeyondArena average-rank comparisons on all classification tasks and on the binary and multiclass subsets. Lower ranks are better. Bars connect methods within the Nemenyi critical difference at $\alpha = 0.05$.

E.3 Controlled Strategy Application and Feature Ablations

E.3.1 Adaptation-Strategy Applications

Table 3 compares alternative adaptation strategies for the same frozen expert-adapter bank. Hard selection + strength retains its adaptation strength s and replaces the expert allocation $\boldsymbol{\pi}$ with the one-hot vector $\mathbf{e}_{\arg\max_k \pi_k}$. For Fixed single-adapter strategies, the same expert is used for every task with $s = 1$ and one-hot expert allocation. Uniform allocation also uses $s = 1$ but assigns equal weight to all four experts. Learned global mixture uses the same task distribution, number of sampled training tasks, and 60,000-update optimization budget as TabFSA, but replaces task-conditioned strategy prediction with an input-independent strength logit and four expert-allocation logits shared across tasks. Table 9 reports the resulting global strategy parameters.

Table 9: Global adaptation strength and expert allocation.

Strategy	s	π_{ext}	π_{temp}	π_{group}	π_{card}
Uniform allocation	1.000	0.250	0.250	0.250	0.250
Learned global mixture	0.746	0.160	0.441	0.258	0.141

E.3.2 Strategy-Feature Ablations

Each learned variant in Table 4 is trained separately for 60,000 updates, with the TabICLv2 backbone and expert adapters frozen. All use the matched panels and examples in Appendix D.4.

The variants isolate the information sources and comparison paths used in the main text. Support summary only retains the label-aware support summary but omits the support–query difference; Support–query only retains the label-free difference but omits the label-aware support summary. The Feature view only and Sample view only variants retain the feature or sample view, respectively. Frozen-path comparison forms the support–query comparison from the Frozen prediction path, whereas the full dual-view strategy feature uses the Symmetric label-free path.

A separately trained hard-output variant uses a straight-through estimator (Bengio et al., 2013) to select one expert and omits adaptation strength. Its relative NLL reductions are 1.61% on Synthetic-40 and 1.00% on Real-40, compared with 2.25% and 2.24% for the full dual-view strategy feature and continuous adaptation strategy.

E.3.3 Training Data Composition

All variants in Table 5 train the task-feature constructor and Strategy Generator for 60,000 updates with the same frozen backbone, expert bank, and optimization settings in Appendix C.2. The full mixture samples original-prior, extended-IID, temporal, grouped, and high-cardinality tasks with probability 0.15 each, and compositional tasks with probability 0.25. Original prior only retains the original prior. New tasks only samples the four extended families and compositional tasks uniformly. All variants use the same evaluation panels, four ensemble views, and support/query examples as Tables 3 and 4, following Appendix D.4. Table 10 reports the strategy distributions.

Table 10: Strategy outputs under different training data composition. Strength is reported as the taskwise mean and standard deviation; allocation columns report unweighted taskwise means of π_k. H_n is the mean of $H(\boldsymbol{\pi})/\log 4$.

Training data	Panel	s	π_{ext}	π_{temp}	π_{group}	π_{card}	H_n
Full mixture	Synthetic	0.791 ± 0.161	0.403	0.311	0.153	0.133	0.800
	Real	0.667 ± 0.199	0.320	0.155	0.312	0.213	0.801
Original prior only	Synthetic	0.122 ± 0.241	0.157	0.161	0.059	0.624	0.361
	Real	0.256 ± 0.338	0.371	0.054	0.127	0.447	0.238
New tasks only	Synthetic / Real	0.935 ± 0.000	0.120	0.315	0.335	0.231	0.954

For Original prior only, the high-cardinality expert has the largest unweighted mean allocation in Table 10. After weighting each task's allocation by its adaptation strength, the extended-IID expert accounts for 79.6% and 94.3% of the total applied expert coefficients on the synthetic and real panels, respectively, computed as $\sum_t s_t \pi_{t,\text{ext}} / \sum_t s_t$. Thus, the unweighted allocation favors high-cardinality, whereas the strength-weighted allocation is dominated by extended-IID. For New tasks only, the generator produces identical outputs on both panels, so they are reported on a single row. The taskwise standard deviations of s and each π_k are below 2×10^{-5}, i.e. the strategy is constant across tasks; the displayed zero standard deviations reflect rounding. Its high taskwise entropy ($H_n = 0.954$) reflects mixing experts within each task, not varying their allocation across tasks. For Full mixture, the mean allocation has normalized entropy approximately 0.927/0.971 on the synthetic/real panels, exceeding the mean taskwise entropy (0.800/0.801). This gap shows that expert allocation varies across tasks, alongside strength.

E.4 STRATEGY-LEARNING PROGRESSION

We evaluate all checkpoints on the fixed panels in Appendix D.4. Training updates increase both optimization budget and cumulative exposure to sampled synthetic tasks. The checkpoint results use the relative NLL reduction defined in Section 4.2.

Table 11: Checkpoints from one strategy-learning trajectory on the fixed panels.

	Relative NLL reduction (%)	
Training updates	Synthetic-40	Real-40
6k	−2.10	−1.45
12k	−0.05	−0.01
30k	+1.71	+1.87
60k	+2.25	+2.24
80k	+2.30	+2.22
100k	+2.32	+2.23

The trajectory moves from degradation at 6k to approximately neutral at 12k and positive transfer by 30k. From 60k to 100k, synthetic reduction changes from 2.25% to 2.32%, while the real result stays between 2.22% and 2.24%; the additional 66.7% updates yield little further improvement.

E.5 QUERY-CONTEXT SENSITIVITY AND POPULATION RESPONSE

The strategy is computed from labeled support and unlabeled query inputs. Three analyses separate temporal context sensitivity, population-level coefficient response, and predictive strategy swapping while holding labeled support fixed.

Temporal context sensitivity. We divide each temporal query set into consecutive blocks. All-query context uses every query input, prefix context uses inputs through the evaluated block, and block context uses only that block. All settings score the same current block.

Table 12: Temporal query-context sensitivity: mean NLL (↓).

Panel	Frozen	All-query	Prefix	Block
Synthetic	1.145	**1.121**	1.127	1.129
Real	0.498	**0.482**	0.484	0.485

All-query context gives the lowest mean NLL on both panels; prefix and block contexts also improve on the frozen backbone.

Population response and predictive swapping. For temporal tasks, groups A and B correspond to query regions closer to and farther from the support period; for high-cardinality tasks, they correspond to head and tail or out-of-vocabulary query populations. On Covertype, head and tail queries share the same 2,048 labeled support examples; support category counts define the two subsets. We evaluate each subset with its own strategy and with the strategy generated for the other subset.

Expert allocation changes in both the temporal and high-cardinality cases, while query group B receives lower mean adaptation strength. On Covertype, each fixed query subset attains lower NLL with its own generated strategy than with the strategy generated for the other subset. With labeled support held fixed, this result shows the predictive value of conditioning expert composition on the current query context.

Table 13: Population response and predictive strategy swapping. Panel (a) reports mean strength changes $s_B - s_A$ and expert-allocation distances $\|\boldsymbol{\pi}_B - \boldsymbol{\pi}_A\|_1$ across 18 real cases. Panel (b) reports Covertype query NLL (↓) with the same labeled support; Swapped uses the other query subset's strategy.

(a) Strategy response across real cases

Regime	Cases	Δs	$\|\Delta\boldsymbol{\pi}\|_1$
Temporal	4	-0.107	0.549
High-cardinality	14	-0.111	0.608

(b) Covertype predictive strategy swapping

Strategy	Head (124)	Tail (132)
TabFSA: own subset	**0.681**	**0.577**
TabFSA: swapped subset	0.692	0.599
Learned global mixture	0.700	0.598
Uniform	0.720	0.630
Frozen TabICLv2	0.713	0.621

E.6 Symbolic Approximation of Strategy Outputs

PySR (Cranmer, 2023) is a symbolic-regression package that evolves a population of expression trees built from basic operators and the input coordinates, retaining a Pareto front that trades off fit error against expression complexity.

Fitting protocol. We analyze the TabFSA strategy generator on Synthetic-40 and Real-40 using five dataset-grouped folds, each with 64 fitting and 16 validation tasks. For fold r, per-coordinate z-score normalization gives standardized strategy features $x_i^{(r)} = (z_i - \mu_i^{(r)})/\tau_i^{(r)}$, where $\mu_i^{(r)}$ and $\tau_i^{(r)}$ are means and standard deviations computed only on the fitting set; constant coordinates are set to zero. Indices are unchanged: 0–255 denote the feature view and 256–511 the sample view. We write x_i for $x_i^{(r)}$ within each fold. Standardization, expression search, and coefficient fitting use only fitting tasks; validation tasks select the final expressions by MAE. PySR directly fits s and each π_k with 200 search iterations per fit. Across the five folds, strength MAE is 0.0653 ± 0.0226; Table 14 summarizes errors against the generator outputs.

Table 14: Direct symbolic approximation: validation MAE (mean ± standard deviation across five folds). Errors are in strategy units, not query NLL.

Target	s	π_{ext}	π_{temp}	π_{group}	π_{card}
MAE	0.0653 ± 0.0226	0.0426 ± 0.0199	0.0414 ± 0.0282	0.0851 ± 0.0400	0.0356 ± 0.0157

Complete fold-0 surrogates. Table 15 reports the five direct fits on fold 0. The corresponding expressions, with constants rounded to three decimal places, are

$$
\begin{aligned}
\widehat{s}^{(0)} &= \tanh(x_{183}) + \tanh(x_{316}),\\
\widehat{\pi}^{(0)}_{\text{ext}} &= 1.695\,\tanh(5.182\,x_{380} + 0.751) - 0.526 - \tanh\big(\tanh(x_{426}) - \tanh(\tanh(x_{175}))\big),\\
\widehat{\pi}^{(0)}_{\text{temp}} &= \big[\tanh(x_{186} + x_5 - 1.914) + 0.541\big] \cdot \big[\tanh(\tanh(6.598\,x_{379})) + 1.216\big],\\
\widehat{\pi}^{(0)}_{\text{group}} &= 1.490\,\tanh\big((x_{252} + 1.047)(41.295\,x_{438} - 5.199)\big) + 1.063 + 0.358\,\tanh(x_{193}),\\
\widehat{\pi}^{(0)}_{\text{card}} &= 0.041\Big[x_{395}^4 - \big(8.368\,(\tanh(x_{179}) - \tanh(5.348\,x_{352})) + 7.883\big)\Big].
\end{aligned}
\tag{31}
$$

Each output is fitted independently. The extended-IID and high-cardinality expressions each use one feature-view and two sample-view coordinates; the temporal and grouped expressions each use two feature-view and one sample-view coordinate. Thus, all four combine both views through distinct coordinate subsets. Table 16 lists all five strength fits.

Table 15: Complete fold-0 surrogates. Nodes count the exported expression trees; MAE uses the original full-precision fits.

	s	π_{ext}	π_{temp}	π_{group}	π_{card}
Nodes	5	18	16	19	17
MAE	0.0365	0.0229	0.0111	0.0188	0.0147

Table 16: Strength surrogates by fold in standardized strategy features. Constants are rounded to three decimal places; MAE uses the original full-precision fits. Nodes count the exported expression trees before algebraic simplification.

Fold r	$\widehat{s}^{(r)}(\mathbf{x})$	Nodes	MAE
0	$\tanh(x_{183}) + \tanh(x_{316})$	5	0.0365
1	$x_{111} + \tanh(0.242 - x_{20}) + \tanh(x_{445})$	9	0.0528
2	$1.423\tanh(x_{314}x_{242}x_{42}^2 + x_{338} + \tanh(x_{152}))$	14	0.0921
3	$0.739x_{224} - 0.294x_{319}$ $- \tanh(0.610x_{250}x_{454} + x_{296}) + 0.214$	18	0.0920
4	$\tanh(x_{183}) + \tanh(x_{445})$	5	0.0533

In Eq. 12, x_{183} is a standardized feature-view coordinate and x_{316} is standardized sample-view coordinate 60. Folds 0 and 4 use five-node sums of two saturating terms, with MAE 0.0365 and 0.0533. Fold 1 adds a linear term (nine nodes, MAE 0.0528). Folds 2 and 3 include coordinate interactions (14 and 18 nodes), with MAE 0.0921 and 0.0920 on their respective validation folds. Fold-0 allocation errors range from 0.0111 to 0.0229. These expressions describe the generator outputs through small subsets of the 512 standardized strategy features, rather than causal effects of raw input features.

E.7 Cross-Backbone Transfer to TabPFN-2.5

We use the official TabPFN-2.5 classification checkpoint, frozen throughout. Four LoRA expert adapters (rank 8, $\alpha = 8$) are attached to the column- and row-attention linear layers of the final four Transformer blocks and trained independently for 120k updates with the same configuration as TabFSA; the strategy-feature constructor and Strategy Generator are retrained as well. At inference the composed update follows Eq. 4 with the base-specific experts. Evaluation is on closed-set tasks with at most ten classes (a TabPFN-2.5 constraint), using support at most 2048 and query at most 1024 rows; the real panel consists of 40 BeyondArena datasets (21 IID, 10 temporal, 9 grouped).

Table 17: Relative query-NLL reduction over the frozen TabPFN-2.5 (%): adaptation strategies. The frozen base is the zero reference and is not listed.

Strategy	Synth. (40)	Real (40)
Fixed Ext.-IID	0.57	−0.67
Fixed temporal	−2.46	−2.07
Fixed grouped	−0.96	−5.84
Fixed high-cardinality	−3.27	−0.75
Uniform allocation	0.84	−0.71
TabPFN-FSA	**4.64**	**2.18**

E.8 Inference Cost

Under matched single-view inference on one NVIDIA RTX 3090, TabICLv2 and TabFSA use the same precision, and TabFSA reuses the label-aware representation computed during strategy-feature construction. All measured settings use 256 query examples. We report end-to-end latency and peak CUDA-allocated memory. Across the measured settings, TabFSA takes $1.91\times$–$2.13\times$ the latency

and 9.4%–14.3% more peak memory; the overhead comes from constructing the strategy feature and applying the composed expert updates.

Table 18: Matched single-view inference cost on one RTX 3090.

		Latency			Peak GPU memory		
Support	Feat.	TabICLv2 (ms)	TabFSA	×	TabICLv2 (MB)	TabFSA	Inc.
1,024	8	30.5	63.4	2.08	175.4	199.1	13.5%
1,024	30	37.9	77.9	2.06	275.5	303.2	10.0%
1,024	100	66.9	130.1	1.94	580.1	662.8	14.3%
4,096	8	53.8	102.6	1.91	300.6	328.7	9.4%
4,096	30	75.2	160.4	2.13	621.1	681.1	9.6%
4,096	100	145.0	290.1	2.00	1,654.7	1,869.6	13.0%