WHEN DOES DEPTH HELP? UNDERSTANDING LAYERWISE INFERENCE IN TABULAR FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Transformer-based tabular foundation models (TFMs) achieve state-of-the-art performance on small to medium-sized tabular prediction tasks while processing diverse inputs through a shared stack of layers. Recent layerwise analyses identify strong predictive performance in early TFM representations and substantial overlap in computation across layers. However, early predictive accessibility alone does not establish whether later layers are dispensable for every input, and aggregate accuracy can obscure which inputs benefit from additional computation. We study all 24 in-context-learning layers of TabPFN-3 across 272 classification datasets, combining representation geometry, prediction trajectories, truncation, and structural interventions, while also examining related representation dynamics in four additional TFMs. We uncover a pronounced *expansion-compression* trajectory, with stronger late dimensional contraction associated with larger increases in class-prototype separation. A targeted analysis of early-stabilizing samples shows that substantial representational changes persist after predicted classes stabilize, while locally ambiguous queries obtain larger accuracy gains from shallow to late layers than well-separated queries. Repeating a single pretrained block or perturbing layer order reduces predictive performance on average, constraining simple post-hoc substitutions in the frozen model. These findings distinguish predictive accessibility, class stability, and the benefits of continuing computation, supporting an input-dependent assessment of depthwise redundancy in TFMs. Code and evaluation pipelines are available at `https://anonymous.4open.science/r/TFMAnalysis-47E8/`.

1 INTRODUCTION

Tabular foundation models (TFMs) (Gupta et al., 2026; Hollmann et al., 2025; Qu et al., 2025; Garg et al., 2025; Zhang et al., 2026; Ma et al., 2026) use in-context learning to predict on new datasets from labeled support examples, without task-specific parameter updates. A single pretrained model processes tasks with different feature spaces and class structures through the same stack of in-context-learning layers. Understanding how much different inputs benefit from this depth is important for assessing computational redundancy and identifying opportunities to reduce inference cost while preserving prediction quality (Elbayad et al., 2018; Fan et al., 2019).

Recent layerwise analyses identify strong predictive performance in early TFM representations and substantial overlap in computation across layers (Balef et al., 2026). TabSwift (Liu & Ye, 2026) further demonstrates per-query early exiting and observes that deeper exits concentrate in ambiguous regions of representation space. These findings motivate examining what subsequent computation still contributes. Early predictive accessibility does not establish that later layers are dispensable for every input, and aggregate accuracy can obscure which inputs benefit from them.

Our layerwise evaluation of TabPFN-3 makes this distinction concrete. When queries are grouped by local class separation within each dataset, locally ambiguous groups show larger absolute accuracy gains from L9 to L23 on average than well-separated groups (Figure 1 (a)). Meanwhile, a targeted analysis of early-stabilizing samples shows that their representations continue to change substantially after their predicted classes stabilize, including pronounced late dimensional contrac-

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.

tion (Figure 1 (b)). These observations highlight the need to distinguish predictive accessibility, class stability, and the accuracy gains from additional computation. They motivate our central question: *When does later computation remain predictively useful, and how does its value vary across inputs?*

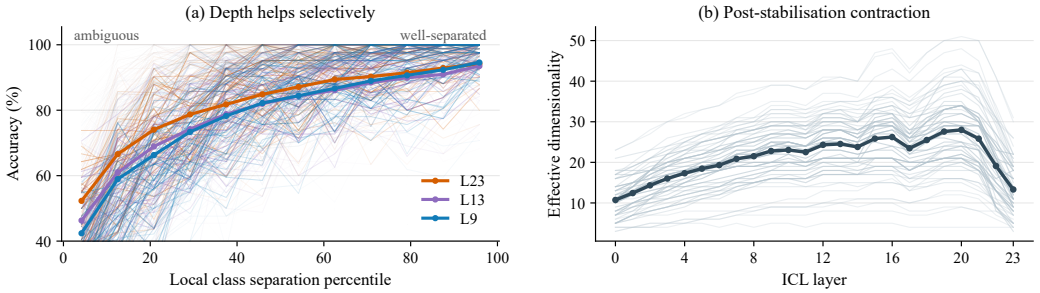

Figure 1: **Predictive accessibility, class stability, and the value of depth are distinct.** (a) L9/L13/L23 accuracy vs. local class separation: ambiguous queries gain more from depth. (b) Effective dimensionality per layer for early-stabilizing queries: representations keep changing after the class stabilises, then contract sharply.

We study all 24 ICL layers of TabPFN-3 using 272 classification datasets from OpenML-CC18 (Bischl et al., 2019) and TALENT (Liu et al., 2025). We complement its native attention-based retrieval decoder (Grinsztajn et al., 2026) with layer-matched classifiers and decoder-independent probes, combining representation geometry, prediction trajectories, truncation, and structural interventions. We also examine related representation dynamics in four additional TFMs (Hollmann et al., 2025; Grinsztajn et al., 2025; Qu et al., 2025; 2026).

We find that TabPFN-3 representations follow a pronounced *expansion-compression* trajectory: early expansion is followed by a relatively stable middle regime and substantial reorganization near the output. Class-relevant information is readable at shallow depth, while class-prototype separation strengthens mainly at later layers. Across datasets, stronger late dimensional contraction is associated with greater increases in prototype separation. These findings distinguish the early availability of predictive information from the subsequent evolution of representation geometry.

At the task level, datasets with later prediction stabilization tend to incur larger accuracy losses under truncation. The overall depthwise accuracy gain also cannot be attributed entirely to mismatch with the final decoder: across 72 OpenML-CC18 tasks, accuracy increases from 82.2% at L2 to 85.6% at L23 when each layer is evaluated using its own fitted readout. Additional computation therefore retains predictive value beyond what is accessible through shallow representations under these readouts.

Finally, interventions show that the frozen model is sensitive to changes in its learned layer composition. Replacing distinct blocks with repeated applications of a single pretrained block, or perturbing their execution order, alters representation dynamics and reduces predictive performance on average. These results constrain simple post-hoc substitutions within the pretrained computation. Our findings support evaluating depthwise redundancy through input-dependent predictive gains alongside the accompanying representation dynamics, and motivate future work on adaptive-depth TFM inference.

2 Related Work

Tabular Foundation Models. Tabular foundation models (TFMs) build on prior-data fitted networks (PFNs) to perform in-context learning without task-specific parameter updates (Müller et al., 2021). TabPFN-v1 performs ICL over row-level representations, whereas TabPFN-v2 moves to cell-level representations with alternating row- and feature-wise attention; TabPFN-2.5 largely retains this design while increasing depth and scaling to larger tables (Gupta et al., 2026; Hollmann et al., 2025; Grinsztajn et al., 2025). In parallel, TabICL introduces a compression-then-ICL architecture that first constructs fixed-dimensional row representations, with TabICLv2 further extending this design through repeated feature grouping and target-aware representations (Qu et al., 2025;

2026). LimiX expands structured-data foundation modeling through joint conditional prediction and retrieval-based ensemble inference (Zhang et al., 2025), while TabPFN-3 combines feature grouping and row compression in a three-stage architecture (Grinsztajn et al., 2026). Despite this architectural progress, mechanistic understanding of TFM inference remains limited. Existing studies have examined spectral adaptivity (Zheng et al., 2025), learned feature-space structure (Ye et al., 2025), intermediate-representation decodability (Gupta et al., 2026), and layerwise predictive refinement and redundancy (Balef et al., 2026). Our work complements these studies by jointly examining representation geometry, sample-level prediction dynamics, and the functional role of depth.

Layerwise Analysis of Transformer Representations. Layerwise analysis has been widely used to trace how information and computation evolve across Transformer depth. Probing and tuned readouts characterize when task-relevant information becomes accessible (Alain & Bengio, 2017; Hewitt & Manning, 2019; Tenney et al., 2019; Rogers et al., 2020; Belinkov, 2022; Belrose et al., 2023), while representation-similarity and geometric analyses characterize how internal representations change across layers (Kornblith et al., 2019; Ansuini et al., 2019; Skean et al., 2025). Complementary intervention-based studies test whether individual layers or their ordering are functionally consequential (Lad et al., 2025; Sun et al., 2025). Most directly, Balef et al. (2026) identify early predictive accessibility, iterative refinement, and substantial depthwise redundancy in TFMs. We extend this layerwise perspective beyond predictive readability to characterize how geometry and individual prediction trajectories evolve, and whether later transformations retain functional value.

Views of Computation Across Depth. Depth has been interpreted as both iterative refinement and structured representation transformation. Residual networks can implement successive refinements of existing representations (Jastrzębski et al., 2017), whereas geometric studies report nontrivial depthwise changes in dimensionality and information structure (Ansuini et al., 2019; Skean et al., 2025). Related work on shared transformations and adaptive-depth inference further asks whether computation can be repeated, truncated, or selectively allocated across inputs (Dehghani et al., 2018; Lan et al., 2019; Elbayad et al., 2018; Xin et al., 2020; Zhou et al., 2020). These perspectives leave open whether TFM depth primarily reflects repetitive refinement or an ordered sequence of distinct, selectively useful transformations.

3 Experimental Setup and Analysis Framework

3.1 Experimental Setup

Models. We use TabPFN-3 as the primary model for layerwise analysis (Grinsztajn et al., 2026). To assess cross-model generality, we additionally evaluate TabPFN-v2, TabPFN-2.5, TabICL, and TabICLv2 (Hollmann et al., 2025; Grinsztajn et al., 2025; Qu et al., 2025; 2026). Architectural details are provided in Appendix A.1, with cross-model results reported in Appendix C.

Datasets and Evaluation Protocol. We evaluate binary and multiclass classification on 72 tasks from OpenML-CC18 (Bischl et al., 2019) and 200 tasks from TALENT (Liu et al., 2025). We restrict the study to classification to maintain a common output space for analyzing layerwise prediction dynamics and representation geometry. All models use their standard preprocessing and inference pipelines. The layerwise representation-extraction protocol is detailed in Appendix A.2.

3.2 Layerwise Analysis Framework

Analysis framework. We analyze layerwise inference from three complementary perspectives: representation geometry, sample-dependent prediction dynamics, and the functional role of depth, building on prior layerwise probing, readout, and TFM analyses (Alain & Bengio, 2017; Belrose et al., 2023; Kornblith et al., 2019; Balef et al., 2026). For query i, we denote its hidden representation after layer ℓ by $h_i^{(\ell)}$ and track its representation and prediction across depth. The representation-extraction protocol is detailed in Appendix A.2.

Predictive readout. TabPFN-3 differs from earlier TabPFN variants in replacing the fixed-width classification head with an attention-based retrieval decoder (Grinsztajn et al., 2026). The decoder predicts directly from the relationship between query and support representations: support representations serve as keys, their one-hot labels as values, and query representations as queries. Because all 24 ICL layers preserve the same 512-dimensional hidden size and the decoder uses layer-

independent projections, we can apply the same pretrained model-native decoder at every depth to obtain a consistent predictive readout across layers. To control for possible mismatch between this decoder and intermediate representations, we complement the native readout with layer-matched readouts. Full decoder definitions and validation results are provided in Appendix E.1.

Representation and functional analyses. We characterize representation geometry using effective dimensionality (Ansuini et al., 2019), PCA displacement, linear CKA (Kornblith et al., 2019), and cosine similarity, capturing complementary aspects of dimensionality, movement, global structural similarity, and sample-wise directional alignment. We trace sample-level prediction trajectories and apply structural interventions to test when additional depth becomes functionally consequential (Lad et al., 2025; Sun et al., 2025; Balef et al., 2026). Metric definitions are provided in Appendix B, with supplementary prediction-dynamics and functional analyses in Appendices G and H, respectively.

4 Expansion–Compression Emerges Across Depth

4.1 Representational Expansion Precedes Late Compression

We first examine how representation geometry evolves across depth. We measure this evolution using PCA-based effective dimensionality (ED) (Ansuini et al., 2019). For the centered query representation matrix $\widetilde{H}^{(\ell)} \in \mathbb{R}^{n \times d}$ with singular values $\sigma_1 \geq \sigma_2 \geq \cdots$, we define

$$\mathrm{ED}(H^{(\ell)}) = \min \left\{ k : \frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_i \sigma_i^2} \geq 0.9 \right\}.$$

ED measures the number of principal directions required to explain most of the representation variance. Full details are provided in Appendix B.2.

Figure 2 reveals a pronounced non-monotonic trajectory. Effective dimensionality rises sharply at shallow depth, remains broadly elevated through the middle layers, and contracts near the output, with a brief local rebound preceding the final decline. We summarize these stages as *expansion*, *plateau*, and *compression*; L9 and L17 serve only as descriptive anchors for subsequent analyses.

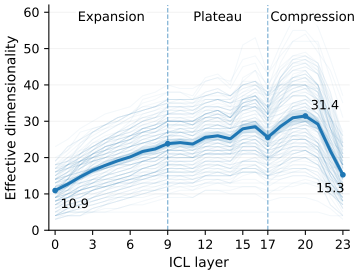

Figure 2: **Representational expansion followed by late compression.** Thin curves denote individual datasets and the thick curve denotes the dataset-level mean.

This pattern reveals a characteristic geometric trajectory: early layers broaden the effective representational space, whereas substantial contraction emerges only near the output. We refer to this trajectory as *Expansion–Compression*. The pattern persists across retained-variance thresholds, normalization and covariance controls, and nearby regime boundaries. Random-weight initialization produces a qualitatively different depthwise profile (Appendix E.3; Appendix D.1).

This late contraction occurs alongside continued representational change. We therefore next examine how strongly representations are reorganized within and across these regimes.

4.2 Late Compression Involves Concentrated Reorganization

The ED trajectory characterizes how broadly representation variance is distributed across depth. To quantify the strength of transformation within each regime, we complement ED with pairwise layer similarity, measured by linear CKA (Kornblith et al., 2019) and cosine similarity, together with adjacent-layer displacement in a shared PCA space. Metric details are provided in Appendix B.3.

Figure 3(a) reveals distinct depthwise transformation dynamics. During expansion, representations progressively depart from earlier layers. The plateau forms a pronounced high-similarity block across both nearby and distant layers, indicating relative mid-depth stabilization. Near the output, this block breaks down as the final layers increasingly diverge from the plateau representations.

Representation displacement exhibits the same structure (Figure 3(b)). Layer-to-layer movement remains substantial during expansion, reaches its minimum through the plateau, and rises sharply

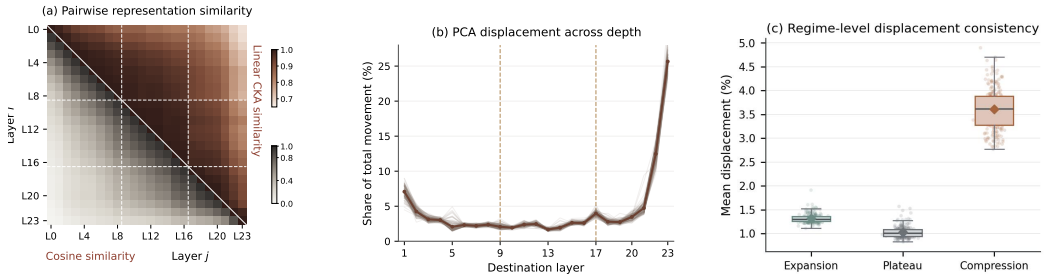

Figure 3: **Representation transformation dynamics across depth.** (a) Mean pairwise layer similarity, with linear CKA above the diagonal and cosine similarity below. (b) Adjacent-layer displacement measured in a shared PCA basis. (c) Dataset-level displacement across the expansion, plateau, and compression regimes.

again at late depth.The final layer transitions exhibit the largest concentration of PCA displacement, indicating intensified representational reorganization near the output. This concentration is consistent across datasets, with displacement lowest during the plateau and highest in the compression regime (Figure 3(c)).

Late compression therefore coincides with a concentrated phase of geometric reorganization near the output. Representation transformation intensifies again after the mid-depth plateau, showing that substantial internal reorganization persists into the final layers.

4.3 Early Predictive Accessibility Precedes Candidate Resolution

To characterize the decision structure associated with this late reorganization, we analyze the geometry among competing class candidates. For each support class c, we construct a layer-specific prototype $\mu_c^{(\ell)}$ and compute its cosine similarity with query i, $s_{i,c}^{(\ell)} = \cos(h_i^{(\ell)}, \mu_c^{(\ell)})$. We quantify candidate competition by the margin between the two most similar class prototypes, $\Delta_i^{(\ell)} = s_{i,(1)}^{(\ell)} - s_{i,(2)}^{(\ell)}$, and use the rank of the correct-class prototype to measure alignment with the target class. The analysis covers all 105 multiclass datasets from OpenML-CC18 and TALENT. Full diagnostic details are provided in Appendix F.

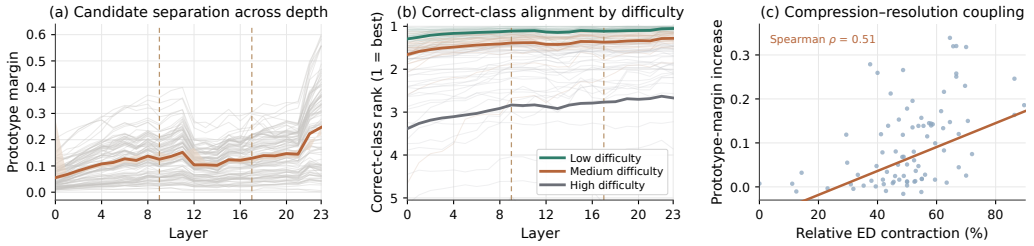

Figure 4: **Candidate separation strengthens at late depth.** (a) Prototype margins remain small through the early and middle layers and widen most strongly near the output. (b) Correct-class alignment emerges earlier on easier tasks, whereas harder tasks retain candidate ambiguity deeper into the network. (c) Stronger late effective-dimensionality contraction is associated with larger increases in prototype margin.

Figure 4(a) shows that prototype margins remain small through the early and middle layers and widen most strongly near the output. Figure 4(b) reveals a corresponding progression in correct-class alignment: easier tasks establish favorable alignment earlier, whereas harder tasks retain candidate ambiguity deeper into the network.

Class-relevant information is already accessible well before strong candidate separation emerges. The attn@V probe exhibits strong linear class decodability by L2, whereas prototype margins widen primarily at later depth (Appendix F.3). Predictive accessibility therefore precedes strong geometric resolution among competing class candidates.

We next relate late candidate resolution to the compression identified in Section 4.1. For each dataset d, we define relative late compression as

$$C_d = \frac{\mathrm{ED}_d^{(20)} - \mathrm{ED}_d^{(23)}}{\mathrm{ED}_d^{(20)}},$$

and the corresponding increase in candidate separation as $R_d = \Delta_d^{(23)} - \Delta_d^{(20)}$. Figure 4(c) shows that datasets with stronger late compression exhibit larger increases in prototype margin. The same association persists across alternative specifications (Appendix F.4). Late dimensional contraction and candidate resolution therefore covary systematically across datasets.

These results separate predictive accessibility from geometric resolution. Class-relevant information becomes readable at shallow depth while competing candidates remain weakly separated. Near the output, concentrated reorganization and compression coincide with stronger candidate separation and target-aligned geometry. Predictive accessibility therefore precedes geometric resolution.

5 Predictions Commit at Different Depths

5.1 Prediction Commitment Is Sample-Dependent

Section 4.3 shows that predictive accessibility can precede geometric resolution. We next turn to the sample level and ask when the predicted class itself becomes stable. For sample i, we denote its layerwise prediction trajectory as $p_i = (\hat{y}_i^{(0)}, \ldots, \hat{y}_i^{(23)})$ and define the stabilization depth as the earliest layer after which the predicted class remains identical to the final-layer prediction: $s_i = \min\{\ell : \hat{y}_i^{(k)} = \hat{y}_i^{(23)}, \forall k \geq \ell\}$. We use *predictive commitment* to denote this stabilization of class identity. The resulting stabilization depth s_i is a retrospective trajectory-level quantity. Confidence and representation geometry may continue to evolve after commitment.

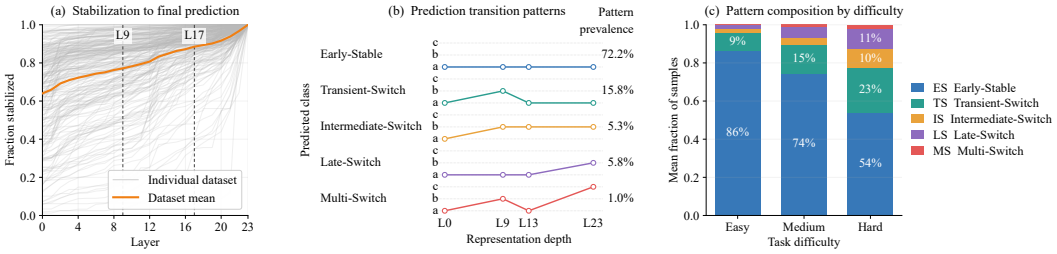

Figure 5: **Sample-level predictive commitment across depth.** (a) Cumulative fraction of samples stabilized to the final prediction across layers, shown per dataset and in aggregate. (b) Five recurring prediction-transition patterns and their pooled prevalence. (c) Pattern composition across tertiles of chance-normalized task difficulty; statistical analyses use difficulty as a continuous variable.

Figure 5(a) reveals substantial heterogeneity in predictive commitment. Many predictions stabilize at shallow depth, while others continue to change through intermediate or late layers, with pronounced variation across datasets. Predictions therefore commit at markedly different depths.

Stabilization depth captures *when* a prediction becomes stable, while the preceding trajectory reveals *how* its class identity evolves before commitment. We characterize this trajectory using prediction identities at L9, L13, and L23 together with the total number of class transitions across all 24 layers, $n_{\mathrm{chg},i} = \sum_{\ell=1}^{23} \mathbb{I}\left[\hat{y}_i^{(\ell)} \neq \hat{y}_i^{(\ell-1)}\right]$. Pattern assignment uses only the prediction trajectory and does not depend on ground-truth correctness. Full assignment rules are provided in Appendix G.1, with sensitivity to nearby representative-layer choices reported in Appendix G.3.

Figure 5(b) summarizes five recurring trajectory patterns. **Early-Stable (ES)** predictions reach the final predicted class early and remain there; **Transient-Switch (TS)** predictions temporarily depart from it before returning; **Intermediate-Switch (IS)** and **Late-Switch (LS)** predictions reach the final class at intermediate and late depth, respectively; and **Multi-Switch (MS)** predictions undergo repeated class changes. ES is dominant at 72.2%, followed by TS (15.8%), LS (5.8%), IS (5.3%), and MS (1.0%). Although early commitment is predominant, more than one quarter of samples

exhibit non-early prediction dynamics. Similar pattern distributions are observed under dataset-level aggregation and nearby representative-layer choices (Appendices G.2–G.3).

Representation dynamics continue beyond early predictive commitment. Early-Stable samples have stabilized by L9, while their representations still exhibit the characteristic Expansion–Compression trajectory, including pronounced late reorganization (Appendix G.5). Predictive commitment can therefore precede substantial later representational transformation. More broadly, all samples traverse the same sequence of layers, while their predictions commit at substantially different depths.

5.2 Commitment Depth Tracks Task Difficulty and Local Ambiguity

Commitment depth varies systematically with task difficulty and local ambiguity. At the task level, we define model-relative, chance-normalized difficulty as

$$D_d = 1 - \frac{a_d - 1/C_d}{1 - 1/C_d},$$

where a_d is final-layer accuracy and C_d is the number of classes. Harder tasks stabilize later on average, with a consistent association across alternative definitions of difficulty (Appendix G.4).

Correspondingly, Figure 5(c) shows a systematic shift from ES toward non-ES trajectories as task difficulty increases. At the sample level, we quantify local ambiguity using class separation in the input space:

$$r_i = \frac{d(x_i, \mathrm{NN}_{\mathrm{diff}})}{d(x_i, \mathrm{NN}_{\mathrm{same}})},$$

where $\mathrm{NN}_{\mathrm{same}}$ and $\mathrm{NN}_{\mathrm{diff}}$ denote the nearest same- and different-class neighbors, respectively. Smaller r_i indicates greater local ambiguity. Median separation decreases from 1.52 for ES to 0.81 for MS (Figure 6). Dataset-aware comparisons further show higher separation for ES than for every non-ES pattern in approximately 95% of shared datasets (Appendix G.6). For cross-dataset analyses, we convert r_i to a within-dataset local-separation percentile to place datasets with different raw distance scales on a common reference. Lower percentiles correspond to more ambiguous samples, whereas higher percentiles indicate more clearly separated local neighborhoods.

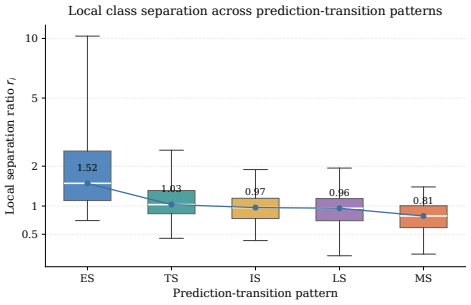

Figure 6: **Local class separation across prediction-transition patterns.** Separation decreases from ES to MS, indicating local ambiguity for more dynamic trajectories.

Commitment depth therefore varies systematically across both tasks and samples. Harder tasks commit later on average, while locally ambiguous samples exhibit more dynamic prediction trajectories. Thus, inputs traversing the same network can differ substantially in how much depth is required before their predictions stabilize.

6 The Functional Value of Depth

6.1 Additional Depth Has Selective Utility

We next examine whether the heterogeneous commitment identified above is reflected in the functional value of additional computation at both the sample and task levels. Lower local-separation percentiles indicate greater ambiguity, whereas higher percentiles correspond to more clearly separated samples.

At the sample level, Figure 7(a) shows that depthwise benefit depends strongly on local ambiguity. Accuracy is already high at L9 for well-separated samples and changes only modestly toward L23, whereas the gap between shallow and final depth grows as local separation decreases. Locally ambiguous inputs therefore obtain larger aggregate gains from additional computation. To distinguish aggregate depthwise gains from the fate of individual shallow predictions, we separately measure retention of shallow successes and rescue of shallow errors:

$$\mathrm{Retain} = P(\hat{y}^{(23)} = y \mid \hat{y}^{(9)} = y), \qquad \mathrm{Rescue} = P(\hat{y}^{(23)} = y \mid \hat{y}^{(9)} \neq y).$$

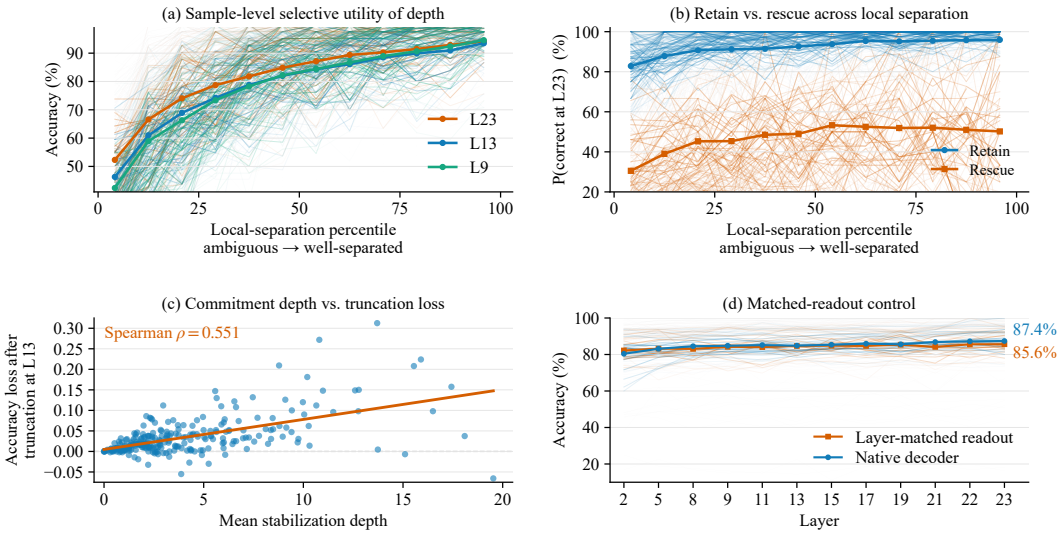

Figure 7: **Depth has selective utility across samples and tasks.** (a) Accuracy at L9, L13, and L23 across within-dataset local-separation percentiles; locally ambiguous samples show larger gains with depth. (b) Retention of L9-correct predictions and rescue of L9 errors across local-separation percentiles; both increase with separation. (c) Mean stabilization depth is positively associated with L13 truncation loss. (d) Accuracy increases with depth under layer-matched readouts.

Figure 7(b) shows that both retention and rescue increase with local separation. Thus, ambiguous groups show larger aggregate gains overall, even though errors already present at L9 are less frequently rescued. The same trends hold with L8 or L10 as the shallow reference (Appendix H.2).

The same selectivity appears at the task level. We define the loss from truncating inference at layer ℓ as $\Delta\mathrm{Acc}_d^{(\ell)} = \mathrm{Acc}_d^{(23)} - \mathrm{Acc}_d^{(\ell)}$. Using L13 as the representative truncation point, Figure 7(c) shows that datasets with later mean stabilization depth incur greater accuracy loss. The association remains positive for later truncation points and with balanced accuracy (Appendix H.1). Later-committing tasks are therefore more sensitive to truncation. The depthwise advantage also persists under depth-specific readouts. With independently trained layer-matched readouts, Figure 7(d) shows that accuracy increases from 82.2% at L2 to 85.6% at L23. Greater depth therefore remains predictively beneficial when each layer is evaluated through its own matched readout.

The functional value of depth is therefore selective: locally ambiguous samples obtain larger aggregate gains from additional computation, while later-committing tasks are more sensitive to truncation. Well-separated inputs, by contrast, gain comparatively little from the remaining layers.

6.2 Later Computation Is Ordered Rather Than Repetitive

We next characterize the structure of this later computation through interventions on the ICL stack, comparing repeated shared transformations with the sequence of distinct learned blocks.

Replacing the sequence of distinct ICL blocks with repeated applications of a shared block substantially weakens both late compression and class-discriminative growth (Figure 8(a)). The effective-dimensional compression ratio decreases from $2.85\times$ to $1.05\times$, while the FDR gain decreases from 2.53 to 0.84. The repeated-block trajectory remains qualitatively distinct even after 24 applications (Figure 8(b)), with the same pattern observed for representative blocks selected from different stages (Appendix H.3).

These representational differences are accompanied by predictive differences. Figure 8(c) shows a positive average advantage for the original model under clean evaluation, label noise, feature perturbation, and reduced sample availability. Under 20% label noise, this advantage holds on every evaluated task (Appendix H.4). The learned sequence of distinct blocks therefore yields both the characteristic representation trajectory and stronger predictive behavior than repeated generic computation.

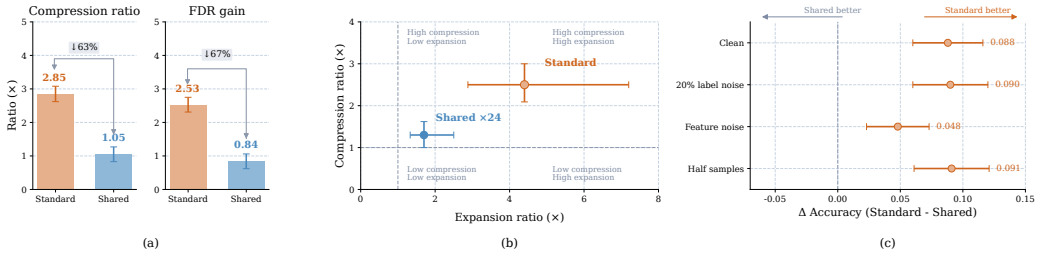

Figure 8: **Shared-weight repetition produces a distinct depthwise trajectory.** (a) Shared-weight repetition weakens compression and class-discriminative growth. (b) The repeated-block trajectory remains distinct at matched computational depth. (c) The full model retains a predictive advantage under perturbation.

A complementary intervention preserves the learned blocks while changing only their execution order. Shuffling or reversing the learned sequence largely removes the characteristic late compression in the representation trajectory (Figure 9(a)), while also reducing predictive performance on most evaluated tasks (Figure 9(b)). Because the learned transformations are preserved, this intervention isolates execution order as a functional component of depth.

Later computation is therefore structured as an ordered sequence of distinct learned operations. Its representational and predictive effects depend on both transformation identity and execution order. This computation path is shared across inputs, while their resolution requirements vary substantially.

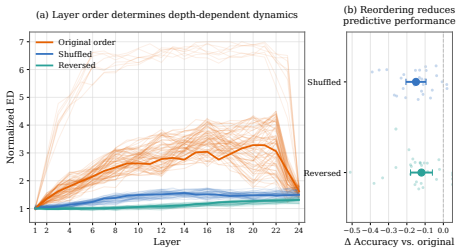

Figure 9: **Layer order is functionally consequential.** (a) Shuffling or reversing the learned sequence disrupts the depthwise representation dynamics. (b) Reordering reduces predictive performance on average.

7 Conclusion

Interpretation. Our study clarifies when later computation remains predictively useful and how its value varies across inputs. Class-relevant information becomes accessible at shallow depth, while representation geometry continues to evolve through an *Expansion–Compression* trajectory, with predictive accessibility preceding geometric resolution. At the sample level, predictions commit at substantially different depths, varying systematically with task difficulty and local ambiguity. This heterogeneity is reflected in the functional value of depth: locally ambiguous samples obtain larger aggregate gains from additional computation, later-committing tasks are more sensitive to truncation, and depthwise predictive gains remain under layer-matched readouts. Structural interventions further show that later computation depends on an ordered sequence of distinct learned transformations rather than generic repetition. These findings show that TFMs follow a shared computation path whose functional value varies substantially across inputs, motivating an input-dependent view of depthwise redundancy.

Limitations and Future Work. Our analysis focuses primarily on TabPFN-3 and classification, although complementary experiments reveal related representation dynamics in several additional TFMs. The strength and temporal organization of *Expansion–Compression* may vary across architectures and model scales. A practical early-exit mechanism will require reliable online signals that estimate whether additional computation is likely to alter or improve a prediction, rather than relying on the retrospective stabilization depth used in our analysis. Future work could investigate inference-time signals derived from prediction confidence, trajectory consistency, or representation geometry, and determine how reliably they identify inputs that benefit from continued computation. Extending these analyses to broader TFM architectures and regression settings is another important direction, particularly because regression will require suitable continuous notions of predictive commitment rather than discrete class-identity stabilization.

Reproducibility Statement

We have made every effort to ensure that the results presented in this paper are reproducible. The experimental setup, including the evaluated models, datasets, layerwise evaluation protocols, representation metrics, prediction analyses, and structural interventions, is described in detail in the main paper and appendices. We additionally provide the definitions of all major evaluation metrics, dataset-level aggregation procedures, statistical analyses, and robustness checks used throughout the study. The datasets used in our experiments are drawn from publicly available benchmark collections, including OpenML-CC18 and TALENT, enabling consistent and reproducible evaluation. Detailed descriptions of model architectures and layerwise representation extraction are provided in Appendix A, while Appendix B specifies the representation metrics and analytical procedures. Additional analyses of candidate geometry, prediction dynamics, truncation, matched readouts, shared-weight controls, data perturbations, and layer-order interventions are documented in the subsequent appendices. We believe these details provide sufficient information to facilitate replication and verification of our results.

AI Use Statement

Large language models (LLMs) were used to assist with the writing and polishing of the manuscript. Specifically, we used LLMs to refine language, improve readability, enhance clarity, and support sentence rephrasing, grammar checking, and the overall flow of the text. LLMs were also used for literature retrieval and discovery to help identify relevant prior work and reduce the risk of overlooking closely related studies. In addition, LLMs were used to support research ideation, primarily to refine the research questions. All AI-assisted outputs were reviewed and verified by the authors, who take full responsibility for the final content of the work.

References

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. In *International Conference on Learning Representations*, 2017.

Alessio Ansuini, Alessandro Laio, Jakob H. Macke, and Davide Zoccolan. Intrinsic dimension of data representations in deep neural networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Amir Rezaei Balef, Mykhailo Koshil, and Katharina Eggensperger. Is one layer enough? understanding inference dynamics in tabular foundation models. *arXiv preprint arXiv:2605.06510*, 2026.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022. doi: 10.1162/coli_a_00422.

Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023.

Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Pieter Gijsbers, Frank Hutter, Michel Lang, Rafael G. Mantovani, Jan N. van Rijn, and Joaquin Vanschoren. OpenML benchmarking suites. *arXiv preprint arXiv:1708.03731*, 2019.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. *arXiv preprint arXiv:1807.03819*, 2018.

Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. Depth-adaptive transformer. *arXiv preprint arXiv:1807.03819*, 2018.

Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*, 2019.

Anurag Garg, Muhammad Ali, Noah Hollmann, Lennart Purucker, Samuel Müller, and Frank Hutter. Real-tabpfn: Improving tabular foundation models via continued pre-training with real-world data. *arXiv preprint arXiv:2507.03971*, 2025.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Benjamin Jäger, Dominik Safaric, Simone Alessi, Adrian Hayler, Mihir Manium, Rosen Yu, Felix Jablonski, Shi Bin Hoo, Anurag Garg, Jake Robertson, Magnus Bühler, Vladyslav Moroshan, Lennart Purucker, Clara Cornu, Lilly Charlotte Wehrhahn, Alessandro Bonetto, Bernhard Schölkopf, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. TabPFN-2.5: Advancing the state of the art in tabular foundation models, 2025.

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.

Aviral Gupta, Armaan Sethi, and Dhruv Kumar. TabPFN through the looking glass: An interpretability study of tabpfn and its internal representations. *arXiv preprint arXiv:2601.08181*, 2026. URL `https://openreview.net/forum?id=cp5PvcI6w8_`.

John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word representations. In *Proceedings of NAACL-HLT*, pp. 4129–4138, 2019. doi: 10.18653/v1/N19-1419.

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.

Stanisław Jastrzębski, Devansh Arpit, Nicolas Ballas, Vikas Verma, Tong Che, and Yoshua Bengio. Residual connections encourage iterative inference. *arXiv preprint arXiv:1710.04773*, 2017.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3519–3529. PMLR, 2019.

Vedang Lad, Jin Hwa Lee, Wes Gurnee, and Max Tegmark. The remarkable robustness of LLMs: Stages of inference? In *Advances in Neural Information Processing Systems*, volume 38, pp. 144216–144249, 2025. doi: 10.52202/085713-4332.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

Si-Yang Liu and Han-Jia Ye. TabSwift: An efficient tabular foundation model with row-wise attention. *arXiv preprint arXiv:2606.07345*, 2026.

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.

Junwei Ma, Valentin Thomas, Rasa Hosseinzadeh, Alex Labach, Jesse Cresswell, Keyvan Golestan, Guangwei Yu, Anthony L Caterini, and Maks Volkovs. Tabdpt: Scaling tabular foundation models on real data. *Advances in Neural Information Processing Systems*, 38:172692–172722, 2026.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. *arXiv preprint arXiv:2112.10510*, 2021. URL `https://openreview.net/forum?id=KSugKcbNf9`.

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.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A better, faster, scalable, and open tabular foundation model. *arXiv preprint arXiv:2602.11139*, 2026.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020. doi: 10.1162/tacl_a_00349.

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Nikul Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. Layer by layer: Uncovering hidden representations in language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 55854–55875. PMLR, 2025.

Qi Sun, Marc Pickett, Aakash Kumar Nain, and Llion Jones. Transformer layers as painters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 25219–25227, 2025. doi: 10.1609/aaai.v39i24.34708.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4593–4601, 2019. doi: 10.18653/v1/P19-1452.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. DeeBERT: Dynamic early exiting for accelerating BERT inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2246–2251, 2020. doi: 10.18653/v1/2020.acl-main.204.

Han-Jia Ye, Si-Yang Liu, and Wei-Lun Chao. A closer look at TabPFN v2: Understanding its strengths and extending its capabilities. In *Advances in Neural Information Processing Systems*, volume 38, pp. 150356–150388, 2025. doi: 10.52202/085713-4527.

Xingxuan Zhang, Gang Ren, Han Yu, Hao Yuan, Hui Wang, Jiansheng Li, Jiayun Wu, Lang Mo, Li Mao, Mingchao Hao, Ningbo Dai, Renzhe Xu, Shuyang Li, Tianyang Zhang, Yue He, Yuanrui Wang, Yunjia Zhang, Zijing Xu, Dongzhe Li, Fang Gao, Hao Zou, Jiandong Liu, Jiashuo Liu, Jiawei Xu, Kaijie Cheng, Kehan Li, Linjun Zhou, Qing Li, Shaohua Fan, Xiaoyu Lin, Xinyan Han, Xuanyue Li, Yan Lu, Yuan Xue, Yuanyuan Jiang, Zimu Wang, Zhenlei Wang, and Peng Cui. LimiX: Unleashing structured-data modeling capability for generalist intelligence, 2025.

Xiyuan Zhang, Danielle Maddix Robinson, Junming Yin, Nick Erickson, Abdul Fatir Ansari, Boran Han, Shuai Zhang, Leman Akoglu, Christos Faloutsos, Michael Mahoney, et al. Mitra: Mixed synthetic priors for enhancing tabular foundation models. *Advances in neural information processing systems*, 38:15795–15840, 2026.

Jianqiao Zheng, Cameron Gordon, Yiping Ji, Hemanth Saratchandran, and Simon Lucey. From tables to signals: Revealing spectral adaptivity in TabPFN. *arXiv preprint arXiv:2511.18278*, 2025.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. BERT loses patience: Fast and robust inference with early exit. In *Advances in Neural Information Processing Systems*, volume 33, pp. 18330–18341, 2020.

A Model Architectures and Layerwise Representation Extraction

This appendix provides the architectural and representation-extraction details underlying our layerwise analyses. All empirical findings are reported in the corresponding main-text and supplementary analyses.

A.1 Model Architectures

Primary architecture. We use TabPFN-3 as the primary model for layerwise analysis (Grinsztajn et al., 2026). The pretrained model remains frozen throughout all representation and inference analyses.

TabPFN-3 maps heterogeneous tabular features into fixed-dimensional row representations before performing in-context learning over rows. Its architecture consists of three stages. Stage 1 performs column-wise feature-distribution embedding through three induced-attention blocks. Stage 2 aggregates feature information within each row using three transformer blocks and four learned CLS tokens. The resulting four 128-dimensional vectors are flattened into a 512-dimensional row representation. Stage 3 applies 24 in-context-learning transformer blocks. All layerwise analyses in this paper concern these 24 Stage-3 blocks.

Table 1: **Architecture of TabPFN-3 used in the primary analysis.** Layer indices L0–L23 refer exclusively to the 24 Stage-3 ICL blocks.

Component	Hyperparameter	Value
Stage 1: Feature-distribution embedding		
	Base embedding dimension	128
	Features per circular-shift group	3
	Induced-attention blocks	3
	Attention heads	8
	Inducing points per column	128
Stage 2: Feature aggregation		
	Transformer blocks	3
	Attention heads	8
	CLS tokens per row	4
	Positional encoding	RoPE
	RoPE base	100,000
Stage 3: In-context learning		
	Row representation dimension	512
	Transformer blocks	24
	Query attention heads	8
	KV heads for support rows	8
	KV heads for query rows	1
	Feed-forward expansion	2×
Classification decoder		
	Decoder type	Attention retrieval
	Decoder heads	6
	Head dimension	64
	Pretraining class limit	160

TabPFN-3 uses RMSNorm throughout the network. Its classification decoder is an attention-based retrieval module: support representations serve as keys, their one-hot labels as values, and query representations as queries. The decoder operates on the 512-dimensional Stage-3 representations and provides the model-native predictive readout used in our layerwise analyses.

Cross-model architectures. To examine cross-model generality, we additionally consider TabPFN-v2, TabPFN-2.5, TabICL, and TabICLv2 (Hollmann et al., 2025; Grinsztajn et al., 2025; Qu et al., 2025; 2026). Table 2 summarizes the architectural properties most relevant to comparing their layerwise representations. All models are evaluated using their pretrained parameters without task-specific backbone fine-tuning.

Table 2: **Architectures used in the cross-model analyses.** "ICL depth" denotes the sequence of transformer blocks over which layerwise representations are analyzed. Parameter counts correspond to the classification models where available.

Model	Parameters	ICL depth	ICL dim.	Pre-ICL stages	Normalization
TabPFN-v2	7.24M	12	192	Alternating feature/item attention	Post-LN
TabPFN-2.5	10.72M	24	192	Alternating feature/item attention	Post-LN
TabPFN-3	53M	24	512	Column embedding + row aggregation	RMSNorm
TabICL	27.05M	12	512	TFcol (3) + TFrow (3)	Pre-LN
TabICLv2	–	12	512	TFcol (3) + TFrow (3)	Pre-LN

Table 3: **Component-level architecture of TabPFN-v2 and TabPFN-2.5.**

Component	TabPFN-v2	TabPFN-2.5
Total parameters	7,244,554	10,718,218
Feature encoder	768	1,152
Label encoder	576	576
Positional embedding	9,408	9,408
Transformer depth	12	24
Transformer-block parameters	7,077,888	10,616,832
Parameters per block	589,824	442,368
Feature-attention module	147,456	147,456
Item-attention module	147,456	147,456
MLP	147,456 + 147,456	73,728 + 73,728
Decoder	148,224 + 7,690	74,112 + 3,850

TabPFN-v2 performs in-context computation through alternating attention over features and items. TabPFN-2.5 retains this representation strategy while increasing the transformer depth from 12 to 24 blocks. Their component-level architectures are summarized in Table 3.

TabICL. TabICL separates feature encoding from dataset-level in-context learning. A column-embedding module first processes individual features, followed by a row-interaction module and a 12-layer ICL predictor. Table 4 summarizes its component-level architecture.

TabICLv2. TabICLv2 follows a related compression-then-ICL organization while modifying feature grouping, target injection, and attention scaling. Features are repeatedly grouped through circular shifts $(0, 1, 3)$, followed by a column-wise Set Transformer ($\mathrm{TF}_{\mathrm{col}}$), a row-wise transformer ($\mathrm{TF}_{\mathrm{row}}$), and a dataset-level ICL transformer ($\mathrm{TF}_{\mathrm{icl}}$). Table 5 summarizes the architectural configuration.

The TabICLv2 paper does not report a single total parameter count for the classification model. We therefore leave the parameter count unspecified rather than infer an unreported value.

A.2 Layerwise Representation Extraction

For TabPFN-3, we extract the output representation after each of the 24 Stage-3 ICL blocks. For a query set containing n_q examples, the post-block query representation matrix at depth ℓ is

$$H_q^{(\ell)} = \left[h_{q,1}^{(\ell)}, \ldots, h_{q,n_q}^{(\ell)}\right]^\top \in \mathbb{R}^{n_q \times 512}.$$

Each layer is represented by the row-level residual-stream output after the corresponding ICL block. The preceding feature-distribution and feature-aggregation stages are excluded from the depth index. Consequently, L0 denotes the output of the first ICL block and L23 denotes the output of the twenty-fourth and final ICL block.

Support and query representations are retained at every depth. Query representations form the primary object of the representation-geometry and sample-level analyses. Support representations are used when required for class-conditioned diagnostics and model-native decoding.

Table 4: **Component-level architecture of TabICL.**

Stage	Component	Parameters
Column embedder	Total	877,824
	Input linear	256
	Transformer blocks (3)	844,032
	Each block	281,344
	Attention 1	$66{,}048 + 16{,}512$
	MLP 1	$33{,}024 + 32{,}896$
	Attention 2	$132{,}480 + 66{,}048$
	Output linear	$16{,}512 + 256$
Row interaction	Total	398,216
	Rotary embedding	8
	Transformer blocks (3)	397,440
	Each block	132,480
	Attention	$66{,}048 + 16{,}512$
ICL predictor	Total	25,775,626
	Transformer blocks (12)	25,233,408
	Each block	2,102,784
	Attention	$1{,}050{,}624 + 262{,}656$
	MLP	$525{,}312 + 524{,}800$
Prediction head	Linear decoder	535,562
Total model parameters		27,051,666

Table 5: **Architecture of TabICLv2 used in the cross-model analysis.**

Stage	Hyperparameter	Value
$\mathrm{TF}_{\mathrm{col}}$	Layers	3
	Inducing vectors	128
	Model dimension	128
	Attention heads	8
$\mathrm{TF}_{\mathrm{row}}$	Layers	3
	Model dimension	128
	Attention heads	8
	CLS tokens	4
$\mathrm{TF}_{\mathrm{icl}}$	Layers	12
	Model dimension	512
	Attention heads	8
Target embedding	$\mathrm{Embed}_{\mathrm{TAE}}$	Lookup, 10 classes
	$\mathrm{Embed}_{\mathrm{ICL}}$	Lookup, 10 classes
Prediction head	Hidden dimension	1024
	Output dimension	10
Other	LayerNorm	Pre-LN with bias
	FFN expansion	$2\times$
	Activation	GELU
	QASSMax MLP hidden dim.	64

For the cross-model analyses, we analogously extract the post-block representation after every block in each model's dataset-level ICL sequence. Layer indices are therefore model-specific rather than matched by absolute index across architectures.

B Definitions and Representation-Space Measurements

This appendix formalizes the representation-space quantities used throughout our layerwise analyses. Unless otherwise specified, all measurements are computed independently within each dataset

using the same set of query examples across layers. Cross-dataset aggregation is specified by the corresponding analysis.

We characterize representation geometry through complementary quantities: PCA-based effective dimensionality describes how representation variance is distributed across principal directions; shared-PCA displacement measures sample-wise movement across depth; linear CKA and cosine similarity quantify pairwise representational similarity; and the Fisher discriminant ratio measures label-conditioned class separation.

B.1 NOTATION

For a dataset containing n query examples, let

$$H^{(\ell)} = \begin{bmatrix} (h_1^{(\ell)})^\top \\ \vdots \\ (h_n^{(\ell)})^\top \end{bmatrix} \in \mathbb{R}^{n \times d} \tag{1}$$

denote the query representation matrix after layer ℓ, where $h_i^{(\ell)} \in \mathbb{R}^d$ is the representation of query i. For TabPFN-3, $\ell \in \{0, \ldots, 23\}$ and $d = 512$.

We define the sample-centered representation matrix as

$$\widetilde{H}^{(\ell)} = H^{(\ell)} - \mathbf{1}\big(\bar{h}^{(\ell)}\big)^\top, \qquad \bar{h}^{(\ell)} = \frac{1}{n}\sum_{i=1}^{n} h_i^{(\ell)}. \tag{2}$$

All layerwise comparisons use matched query examples across depth.

B.2 PCA-BASED EFFECTIVE DIMENSIONALITY

We quantify effective dimensionality using the number of principal components required to explain a fixed fraction of the representation variance. Let

$$\sigma_1^{(\ell)} \geq \sigma_2^{(\ell)} \geq \cdots \tag{3}$$

denote the singular values of the centered representation matrix $\widetilde{H}^{(\ell)}$.

The PCA-based effective dimensionality is

$$\mathrm{ED}\Big(H^{(\ell)}\Big) = \min\left\{ k : \frac{\sum_{j=1}^{k}\left(\sigma_j^{(\ell)}\right)^2}{\sum_j \left(\sigma_j^{(\ell)}\right)^2} \geq 0.90 \right\}. \tag{4}$$

A larger ED indicates that representation variance is distributed across a broader set of principal directions, whereas a smaller ED indicates greater concentration of variance in fewer directions. We use the term *PCA-based effective dimensionality* to emphasize that this quantity is defined from the variance spectrum rather than from a nonlinear manifold estimator.

The 90% retained-variance threshold is used throughout the primary analysis. Sensitivity to alternative thresholds is reported in the robustness analyses.

Expansion and compression ratios. To summarize the non-monotonic effective-dimensionality trajectory within each dataset, let

$$\ell^* = \arg\max_{\ell} \mathrm{ED}\Big(H^{(\ell)}\Big) \tag{5}$$

denote a layer at which effective dimensionality reaches its maximum.

We define the expansion ratio as

$$\rho_{\mathrm{exp}} = \frac{\mathrm{ED}\big(H^{(\ell^*)}\big)}{\mathrm{ED}\big(H^{(0)}\big)}, \tag{6}$$

and the compression ratio as

$$\rho_{\mathrm{com}} = \frac{\mathrm{ED}\big(H^{(\ell^*)}\big)}{\mathrm{ED}\big(H^{(L)}\big)}, \tag{7}$$

where L denotes the final analyzed layer, with $L = 23$ for TabPFN-3.

Thus, ρ_{exp} measures the increase from the initial representation to the maximum effective dimensionality reached across depth, whereas ρ_{com} measures the subsequent contraction from that maximum to the final representation. Both quantities are computed from the complete layerwise trajectory within each dataset.

B.3 Representation Displacement in a Shared PCA Space

Effective dimensionality characterizes the distribution of representation variance but does not quantify how far individual sample representations move across depth. We therefore measure adjacent-layer displacement in a shared PCA coordinate system.

For each dataset, we concatenate the query representations from all analyzed layers,

$$H_{\mathrm{joint}} = \begin{bmatrix} H^{(0)} \\ H^{(1)} \\ \vdots \\ H^{(L)} \end{bmatrix}, \tag{8}$$

and fit PCA once to the resulting joint representation matrix.

We retain

$$k_{\mathrm{PCA}} = \min(10, d, n-1) \tag{9}$$

principal directions. Let P denote the resulting projection matrix and μ_{joint} the mean of H_{joint}. The shared-PCA representation of query i at layer ℓ is

$$z_i^{(\ell)} = P^\top \left(h_i^{(\ell)} - \mu_{\mathrm{joint}} \right) \in \mathbb{R}^{k_{\mathrm{PCA}}}. \tag{10}$$

Adjacent-layer displacement is defined as

$$D_{\mathrm{PCA}}^{(\ell)} = \frac{1}{n} \sum_{i=1}^{n} \left\| z_i^{(\ell+1)} - z_i^{(\ell)} \right\|_2, \qquad \ell = 0, \ldots, L-1. \tag{11}$$

Using a single PCA basis places all layers in the same coordinate system and makes adjacent-layer movement directly comparable across depth.

To quantify where representation movement is concentrated, for a set of adjacent-layer transitions $\mathcal{I}$ we define

$$R_{\mathrm{disp}}(\mathcal{I}) = \frac{\sum_{\ell \in \mathcal{I}} D_{\mathrm{PCA}}^{(\ell)}}{\sum_{\ell=0}^{L-1} D_{\mathrm{PCA}}^{(\ell)}}. \tag{12}$$

For the late-displacement statistic reported for TabPFN-3, $\mathcal{I}$ contains the final six adjacent-layer transitions. This quantity therefore measures the fraction of total depthwise representation movement concentrated near the output.

B.4 Pairwise Representation Similarity

We characterize the similarity between representations at arbitrary layers ℓ and m using two complementary measures: linear centered kernel alignment and sample-wise cosine similarity.

Linear CKA. For centered representation matrices $\widetilde{H}^{(\ell)}$ and $\widetilde{H}^{(m)}$, linear CKA is

$$\mathrm{CKA}\left(H^{(\ell)}, H^{(m)}\right) = \frac{\left\| \left(\widetilde{H}^{(\ell)}\right)^\top \widetilde{H}^{(m)} \right\|_F^2}{\left\| \left(\widetilde{H}^{(\ell)}\right)^\top \widetilde{H}^{(\ell)} \right\|_F \left\| \left(\widetilde{H}^{(m)}\right)^\top \widetilde{H}^{(m)} \right\|_F}. \tag{13}$$

Larger CKA indicates greater similarity in the global linear representational structure of the two layers.

Cosine similarity. For the same pair of layers, we compute the mean cosine similarity between corresponding query representations,

$$S_{\cos}^{(\ell,m)} = \frac{1}{n} \sum_{i=1}^{n} \frac{\left(h_i^{(\ell)}\right)^\top h_i^{(m)}}{\left\|h_i^{(\ell)}\right\|_2 \left\|h_i^{(m)}\right\|_2}. \tag{14}$$

This quantity measures sample-wise directional alignment independently of vector magnitude. Final-layer alignment is obtained as the special case $S_{\cos}^{(\ell,L)}$.

Linear CKA and cosine similarity are used jointly in the pairwise layer-similarity matrices: CKA characterizes similarity in global representational structure, while cosine similarity characterizes the directional alignment of corresponding sample representations.

B.5 Fisher Discriminant Ratio

To quantify label-conditioned class separation, we compute the Fisher discriminant ratio (FDR) from query representations and query labels.

For class c, let n_c denote the number of query examples belonging to that class, $\mu_c^{(\ell)}$ their mean representation at layer ℓ, and $\mu^{(\ell)}$ the global query mean.

The between-class scatter matrix is

$$S_b^{(\ell)} = \sum_c n_c \left(\mu_c^{(\ell)} - \mu^{(\ell)}\right) \left(\mu_c^{(\ell)} - \mu^{(\ell)}\right)^\top, \tag{15}$$

and the within-class scatter matrix is

$$S_w^{(\ell)} = \sum_c \sum_{i:y_i=c} \left(h_i^{(\ell)} - \mu_c^{(\ell)}\right) \left(h_i^{(\ell)} - \mu_c^{(\ell)}\right)^\top. \tag{16}$$

We define

$$\mathrm{FDR}^{(\ell)} = \frac{\mathrm{tr}\left(S_b^{(\ell)}\right)}{\mathrm{tr}\left(S_w^{(\ell)}\right)}. \tag{17}$$

Larger FDR indicates greater between-class variation relative to within-class variation and therefore stronger geometric class separation. Query labels are used for this retrospective representation-space diagnostic.

In the structural-intervention analyses, *FDR gain* denotes the ratio between final- and initial-layer FDR, $\mathrm{FDR}^{(L)} / \mathrm{FDR}^{(0)}$.

B.6 Representative Analytical Layers

All primary representation-space measurements are computed across the complete layerwise trajectory. For TabPFN-3, we use L9 and L17 as descriptive anchors for regime-level summaries. The corresponding transition intervals are L0–L9 for expansion, L9–L17 for the plateau, and L17–L23

Table 6: **Representative TabPFN-3 layers used in subsequent analyses.** The Expansion–Compression trajectory is characterized from the complete layerwise measurements; these depths serve as recurring analytical anchors.

Layer	Role in subsequent analyses
L9	Early-depth anchor; end of the expansion summary interval
L13	Intermediate plateau anchor
L17	Late-depth anchor; start of the compression summary interval
L20	Late-depth anchor near the aggregate ED maximum
L23	Final-layer representation

for compression. These intervals summarize the aggregate effective-dimensionality trajectory rather than defining separate dataset-specific change points.

The late interval contains the local rebound near L20 followed by the pronounced contraction toward L23. Accordingly, L20 is used as a representative late-depth anchor near the aggregate effective-dimensionality maximum, while L23 denotes the final representation.

Several representative depths recur in subsequent analyses, as summarized in Table 6.

L9 and L13 are used as representative depths in the sample-level prediction-trajectory analyses. L13, L17, and L20 are additionally used as truncation points in the functional-depth experiments. Robustness to nearby representative-layer choices is evaluated separately.

C Cross-Model Generality of Expansion–Compression

The main analyses focus on TabPFN-3 because its 24-layer ICL stack provides a particularly clear setting in which to study depth-dependent representation dynamics. We next examine whether the broad Expansion–Compression pattern also appears in other pretrained tabular foundation models.

We consider four additional models: TabPFN-v2, TabPFN-2.5, TabICL, and TabICLv2. TabPFN-3 is omitted from the figures in this appendix because its representation trajectory is analyzed extensively in the main text. The purpose of the present analysis is instead to examine whether related geometric dynamics extend beyond the primary model.

Across the four auxiliary architectures, the cross-dataset mean effective-dimensionality trajectories exhibit the same broad qualitative ordering: effective dimensionality increases from its initial state, reaches an intermediate maximum, and subsequently decreases. The magnitude and temporal organization of this trajectory nevertheless differ substantially across models. We therefore use Expansion–Compression to describe this recurring representation-space pattern across the models examined here, while allowing its strength and temporal organization to remain architecture dependent.

C.1 Cross-Model Evaluation Protocol

For the trajectory comparison, we restrict the analysis to classification datasets for which all four auxiliary models have valid layerwise measurements. Within each dataset, the same evaluation examples are used across the compared representation measurements.

Architectural details of the individual models are reported in Appendix A.1, and all geometric quantities follow the definitions in Appendix B.

Because the models differ in both representation width and initial effective dimensionality, raw ED values are not directly comparable across architectures. For visualization, we therefore normalize effective dimensionality separately within each dataset–model pair by its first-layer value,

$$\widehat{\mathrm{ED}}^{(\ell)} = \frac{\mathrm{ED}(H^{(\ell)})}{\mathrm{ED}(H^{(0)})}. \tag{18}$$

Thus, every normalized trajectory begins at one, and values above one indicate expansion relative to that model's own initial representation. This is not a min–max normalization. Consequently, ar-

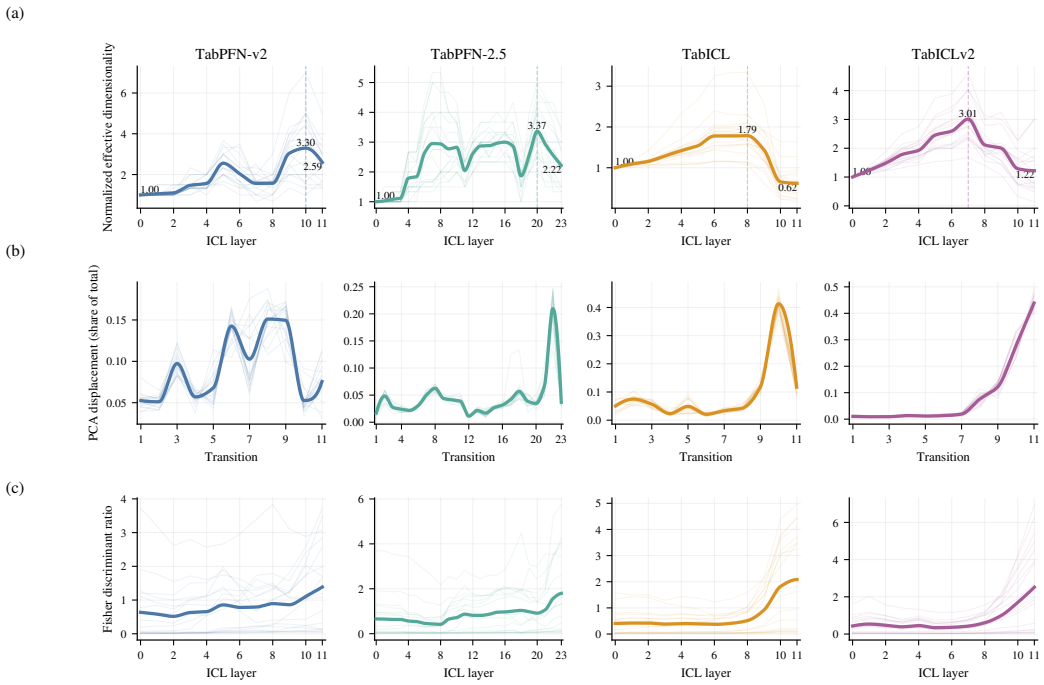

Figure 10: **Cross-model representation dynamics.** Results are shown for TabPFN-v2, TabPFN-2.5, TabICL, and TabICLv2 on matched datasets. **(a)** Effective dimensionality normalized by the first-layer value within each dataset–model pair. **(b)** Adjacent-layer PCA displacement expressed as a fraction of the total trajectory displacement. **(c)** Fisher discriminant ratio across depth. Thin curves denote individual datasets and thick curves denote the cross-dataset mean. Mean curves are interpolated only for visual smoothness; all statistics use the original layerwise measurements. Peak annotations in panel (a) refer to the cross-dataset mean trajectories and therefore need not equal the mean within-dataset expansion ratios reported in Table 7.

chitectures can attain different normalized peak values when their representations undergo different relative amounts of expansion.

The expansion and compression ratios reported below are computed from the unnormalized within-dataset ED trajectories using the definitions in Appendix B.2. For each dataset–model pair, the ratios are computed from its complete layerwise trajectory and are then averaged acros datasets. Because the layer attaining maximum ED can differ across datasets, the mean expansion ratio need not equal the maximum of the cross-dataset mean normalized trajectory.

Final normalized ED is likewise computed within each dataset–model pair before cross-dataset averaging. Expansion, compression, and final normalized ED are therefore averaged as separate dataset-level quantities and need not satisfy an exact algebraic relation after aggregation.

For PCA displacement, each adjacent-layer displacement is divided by the total displacement of the corresponding dataset–model trajectory. This allows the temporal concentration of representation movement to be compared across architectures with different overall displacement scales.

The curves in Figure 10 show the original layerwise measurements together with a shape-preserving interpolation of their cross-dataset means for visualization. The interpolation affects only the rendering of the curves; all quantitative statistics are computed from the original discrete layer values.

C.2 Representation Trajectories Across Architectures

Figure 10 compares effective dimensionality, PCA displacement, and label-conditioned class separation across the four auxiliary models.

Table 7: **Summary of cross-model representation dynamics.** Expansion and compression ratios are first computed within each dataset and then averaged across matched datasets. Final ED denotes the final effective dimensionality normalized by the first-layer value and is likewise averaged from dataset-level values. Because these quantities are averaged independently, their cross-dataset means need not satisfy exact algebraic ratio identities. Late PCA reports the fraction of total PCA displacement occurring in the final quarter of layer transitions.

Model	Depth	Expansion	Compression	Final ED	Late PCA
TabPFN-v2	12	$3.61\times$	$1.43\times$	2.59	27.8%
TabPFN-2.5	24	$3.87\times$	$1.87\times$	2.22	44.4%
TabICL	12	$1.82\times$	$3.88\times$	0.62	64.9%
TabICLv2	12	$3.01\times$	$3.59\times$	1.22	83.8%

The normalized ED trajectories show that non-monotonic Expansion–Compression behavior is not restricted to TabPFN-3. All four auxiliary models reach an intermediate representation with higher effective dimensionality than their respective initial state.

The strength of expansion and subsequent compression nevertheless differs markedly across architectures. Table 7 summarizes these differences.

TabPFN-v2 and TabPFN-2.5 exhibit strong expansion but comparatively modest peak-to-final contraction. Their final representations therefore remain well above their initial effective dimensionality.

The two TabICL models exhibit a different organization. TabICL shows weaker expansion but much stronger subsequent contraction, ending below its initial normalized ED on average. TabICLv2 combines pronounced expansion with similarly strong late contraction.

The PCA displacement trajectories reinforce this distinction. Only 27.8% of the total displacement in TabPFN-v2 occurs in the final quarter of transitions, increasing to 44.4% for TabPFN-2.5. The corresponding fractions are substantially larger for TabICL (64.9%) and TabICLv2 (83.8%), indicating that representation movement is considerably more concentrated near the end of their ICL stacks.

The label-conditioned geometry follows a related but non-identical trajectory. FDR generally increases toward later depth, indicating stronger class separation in later representations, but the timing and magnitude of this increase differ across architectures. ED, displacement, and label-conditioned class separation therefore capture complementary aspects of the depthwise representation trajectory.

C.3 Pairwise Representation Similarity

The preceding analysis characterizes how individual geometric statistics change with depth. We additionally examine the complete pairwise relationship between representations from different layers.

Figure 11 combines two complementary similarity measures. The upper triangular portion of each matrix reports linear CKA, which measures similarity in global relational structure, whereas the lower triangular portion reports mean cosine similarity, which measures directional alignment of corresponding representation vectors.

The similarity matrices reveal substantial architecture-dependent structure that is not visible from ED alone. TabPFN-v2 and TabPFN-2.5 exhibit pronounced block-like regions in which nearby groups of layers remain highly similar, separated by transitions with lower cross-block similarity. This pattern is especially visible in the CKA matrices.

TabICL and TabICLv2 show broader regions of high CKA similarity across depth. At the same time, their cosine similarity decreases more strongly between temporally distant layers. Thus, two layers can preserve a relatively similar global relational structure while their individual representation vectors undergo substantial directional reorientation.

The distinction between CKA and cosine similarity is therefore informative for interpreting representation change. High CKA can coexist with substantial directional reorientation of individual vectors, while reduced cosine similarity can occur without a comparable loss of global relational

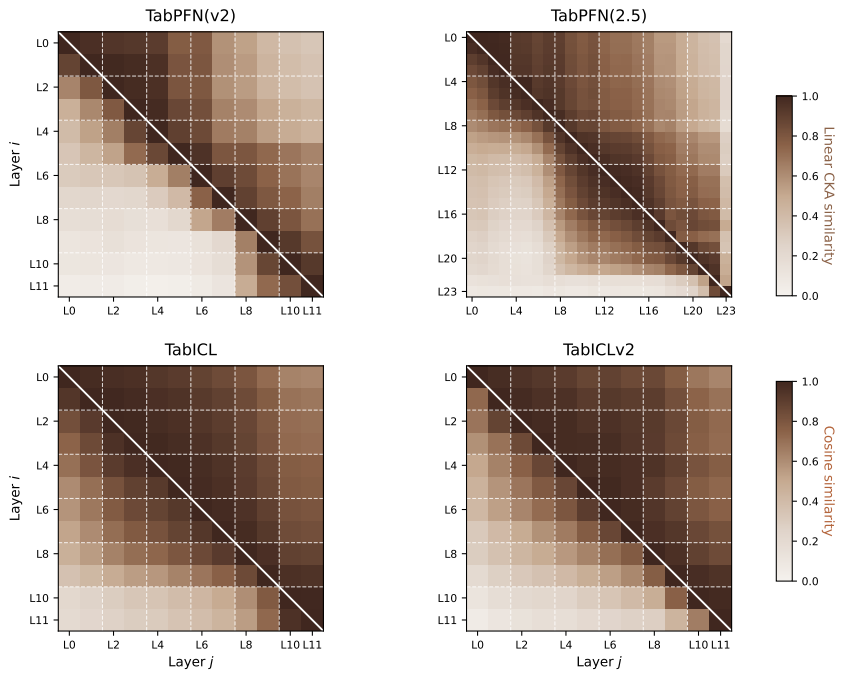

Figure 11: **Pairwise representation similarity across architectures.** For each model, the upper triangular matrix reports mean linear CKA between every pair of layers, while the lower triangular matrix reports mean cosine similarity. Darker entries indicate greater similarity.

structure. The two measurements provide complementary views of the same depth-dependent trajectory.

C.4 Scope of the Cross-Model Evidence

Taken together, the cross-model results show that the Expansion–Compression pattern observed in the primary TabPFN-3 analysis is not unique to that architecture. Related non-monotonic ED trajectories are also present in TabPFN-v2, TabPFN-2.5, TabICL, and TabICLv2.

At the same time, the models differ substantially in expansion strength, compression strength, the temporal concentration of representation movement, and the continuity of their pairwise representation structure. The broad geometric pattern is therefore shared across the examined architectures, whereas its detailed organization is model dependent.

The present comparison does not isolate a single architectural determinant of these differences. Model depth, hidden dimension, attention organization, decoder design, and pretraining procedure vary jointly across the model families. Identifying which of these factors determines the precise form of the trajectory therefore remains separate from the cross-model generality established here.

Accordingly, the appropriate scope of the result is: *the Expansion–Compression phenomenon extends beyond TabPFN-3 within the tabular foundation models examined here, while its strength and temporal organization remain architecture dependent.*

D Architectural and Training Controls

To characterize the conditions under which Expansion–Compression emerges, we conduct three complementary controls. We first randomize the learned weights of TabPFN-3 while preserving its architecture. We then vary computational depth within a lightweight TabPFN-style model while holding the model family and training objective fixed. Finally, we compare the resulting geometry with that of a conventionally supervised FT-Transformer. Together, these experiments separate

Table 8: **Random-weight control for TabPFN-3.** Each entry reports expansion ratio / compression ratio. The pretrained model exhibits substantial Expansion–Compression, whereas randomizing either the ICL stack or the complete model yields flat effective-dimensionality trajectories on the tested datasets.

Dataset	Pretrained	Randomized ICL blocks	Fully randomized
vehicle	$4.00\times\,/2.33\times$	$1.00\times\,/1.00\times$	$1.00\times\,/1.00\times$
segment	$4.67\times\,/2.33\times$	$1.00\times\,/1.00\times$	$1.00\times\,/1.00\times$
satimage	$7.50\times\,/3.33\times$	$1.00\times\,/1.00\times$	$1.00\times\,/1.00\times$
diabetes	$2.88\times\,/2.09\times$	$1.00\times\,/1.00\times$	$1.00\times\,/1.00\times$

Table 9: **Controlled-depth experiment with nanoTabPFN.** The three- and four-layer variants exhibit representation expansion without a subsequent compression phase, whereas the five-layer variant exhibits both expansion and compression.

Depth	ED trajectory	Expansion	Compression
3 layers	$1 \to 2 \to 2$	$2.00\times$	$1.00\times$
4 layers	$1 \to 1 \to 2 \to 2$	$2.00\times$	$1.00\times$
5 layers	$1 \to 2 \to 2 \to 2 \to 1$	$2.00\times$	$2.00\times$

the roles of learned parameterization, computational depth, and training paradigm in shaping the observed representation trajectory.

D.1 Randomized Weights Do Not Reproduce the Learned Trajectory

We first isolate the contribution of learned parameterization in TabPFN-3. We evaluate the same architecture under three weight conditions: the original pretrained model, a model in which only the 24 ICL blocks are randomized while the surrounding embedding and decoder components remain pretrained, and a model in which all weights are randomized.

Table 8 reports the expansion and compression ratios defined in Appendix B.2.

The pretrained network exhibits substantial expansion followed by compression on every tested dataset. In contrast, both randomized conditions yield expansion and compression ratios of one, corresponding to flat effective-dimensionality trajectories.

Preserving the TabPFN-3 architecture while removing its learned parameterization therefore eliminates the observed non-monotonic trajectory in these controls. The Expansion–Compression profile is consequently tied to the learned transformations of the pretrained model rather than to the architecture alone.

D.2 Controlled Depth in nanoTabPFN

The random-weight control establishes the importance of learned transformations but does not isolate how computational depth contributes to the trajectory. We therefore use nanoTabPFN, a lightweight trainable TabPFN-style model for which variants with different numbers of transformer layers can be trained under the same ICL objective.

We train variants with three, four, and five layers and examine their layerwise effective-dimensionality trajectories on synthetic data drawn from the training distribution. The model family, training objective, and evaluation procedure are held fixed while depth is varied.

Expansion is already present in the shallower variants: both the three- and four-layer models reach twice their initial effective dimensionality, but neither subsequently contracts. The five-layer model instead follows $1 \to 2 \to 2 \to 2 \to 1$, returning to its initial effective dimensionality after reaching an expanded intermediate state.

Within this controlled setting, increasing depth changes whether the representation trajectory proceeds from expansion into compression. Representation expansion therefore emerges at relatively shallow depth, while additional computation can enable completion of the compression phase.

Table 10: **Representation geometry in the supervised FT-Transformer control.** The supervised model exhibits substantial effective-dimensionality contraction without the pronounced preceding expansion characteristic of the Expansion–Compression trajectory.

Measurement	FT-Transformer
Mean expansion ratio	$1.11\times$
Mean compression ratio	$2.33\times$
Characteristic trajectory	Predominantly contraction

D.3 Supervised FT-Transformer Exhibits Contraction Without Pronounced Expansion

We finally examine whether the same geometric organization appears in a conventionally supervised transformer for tabular data. We train an FT-Transformer consisting of a feature tokenizer and eight transformer encoder blocks with hidden dimension 64 and apply the same layerwise effective-dimensionality analysis to CC18 classification datasets.

Across these datasets, the mean expansion ratio is only $1.11\times$, whereas the mean compression ratio is $2.33\times$. The dominant geometric trend is therefore contraction, with only weak preceding expansion.

The supervised control therefore separates late contraction from the full Expansion–Compression profile. Substantial contraction can arise under conventional supervised transformer training, whereas the pronounced preceding expansion observed in the pretrained TFMs is largely absent in this control.

Taken together, the three controls delineate complementary properties of the observed trajectory. Learned parameterization is required to reproduce Expansion–Compression in TabPFN-3; additional depth enables the compression phase to emerge in the controlled nanoTabPFN setting; and the supervised FT-Transformer exhibits contraction without pronounced expansion. Within the models examined here, these results support interpreting Expansion–Compression as a learned, depth-dependent organization of ICL representations rather than an automatic consequence of transformer depth.

E Methodological Validation and Robustness

This appendix provides additional methodological validation for our layerwise analysis. We first establish why the native TabPFN-3 decoder can be used as a common readout of intermediate representations, and then separate native-decoder alignment from predictive information availability using layer-matched and decoder-independent probes. Finally, we report robustness checks for the geometric quantities underlying the Expansion–Compression analysis.

E.1 Validating the Native TabPFN-3 Decoder as a Layerwise Readout

Native many-class decoder. TabPFN-3 replaces the fixed-width classification head used in earlier TabPFN variants with an attention-based retrieval decoder (Grinsztajn et al., 2026). For classification, training-set representations act as keys, their corresponding one-hot labels act as values, and query representations act as queries. Prediction is therefore implemented as a learned soft nearest-neighbor retrieval over the in-context support set.

Let $h_{m,\mathrm{q}}^{(\ell)} \in \mathbb{R}^{512}$ denote the representation of query sample m after ICL layer ℓ, and let $h_{n,\mathrm{s}}^{(\ell)} \in \mathbb{R}^{512}$ denote the corresponding representation of support sample n. For decoder head r, the projected query and key representations are

$$q_m^{(\ell,r)} = W_Q^{(r)} h_{m,\mathrm{q}}^{(\ell)}, \tag{19}$$

$$k_n^{(\ell,r)} = W_K^{(r)} h_{n,\mathrm{s}}^{(\ell)}. \tag{20}$$

For notational simplicity, we write the attention weights using the standard scaled dot-product form,

$$\alpha_{m,n}^{(\ell,r)} = \frac{\exp\left({q_m^{(\ell,r)}}^\top k_n^{(\ell,r)} / \sqrt{d_h} \right)}{\sum_{j=1}^{N_s} \exp\left({q_m^{(\ell,r)}}^\top k_j^{(\ell,r)} / \sqrt{d_h} \right)}, \tag{21}$$

where the actual TabPFN-3 implementation additionally applies its pretrained QASSMax query scaling (Grinsztajn et al., 2026). For class c, the decoder aggregates the attention mass assigned to support samples of that class and averages it across heads,

$$p_{m,c}^{(\ell)} = \frac{1}{H} \sum_{r=1}^{H} \sum_{n:\, y_n = c} \alpha_{m,n}^{(\ell,r)}. \tag{22}$$

The class logits are then

$$z_{m,c}^{(\ell)} = \log\Big(\mathrm{clip}\Big(p_{m,c}^{(\ell)} \Big) \Big). \tag{23}$$

This is equivalent to the report's description of a head-averaged, attention-weighted average of one-hot labels, written explicitly in class-wise form. TabPFN-3 uses $H = 6$ decoder heads with head dimension $d_h = 64$.

Why the decoder is well-defined across depth. All 24 ICL layers preserve the same 512-dimensional hidden representation, and the decoder applies the same learned query and key projections independently of layer index. The native decoder is therefore dimensionally well-defined at every ICL layer and provides a fixed, model-native readout across depth. This does *not* imply a fixed decision boundary: both query and support representations evolve with ℓ, changing their relative geometry and the resulting attention weights.

Decoder-alignment concern. Dimensional compatibility does not imply that native-decoder performance measures all predictive information in an intermediate representation. Earlier layers may contain label-relevant information while remaining imperfectly aligned with the final decoder, motivating tuned or layer-specific readouts (Belrose et al., 2023; Balef et al., 2026). We therefore distinguish *predictive decodability* from *native-decoder alignment*.

E.1.1 Layer-Matched Readout Control

To control for decoder mismatch, we train an independent linear readout for each analyzed layer. Only query hidden states are used for this experiment: within each dataset, query examples are split 50/50 into probe-training and probe-evaluation subsets, and a separate classifier is fitted to $h_{\mathrm{q}}^{(\ell)}$ for every layer ℓ. This procedure removes the requirement that an intermediate representation be aligned with the native decoder while preserving the amount of predictive information available at that layer.

We evaluate both the native decoder and the layer-matched readout on all OpenML-CC18 classification tasks. Figure 12 shows the dataset-level mean accuracy and negative log-likelihood (NLL); error bars denote 95% bootstrap confidence intervals obtained by resampling datasets.

Accuracy. At L2, the native decoder reaches 0.804 accuracy and the layer-matched readout reaches 0.822, indicating partial early-layer decoder misalignment. From L2 to L23, matched-readout accuracy increases from 0.822 to 0.856 ($+3.4$ pp; 95% CI $[+2.1, +4.8]$ pp), while native accuracy increases from 0.804 to 0.874 ($+7.0$ pp; 95% CI $[+5.2, +8.9]$ pp). Thus, the overall depthwise accuracy increase is not explained by output-head mismatch alone.

Probability quality. Native-decoder NLL decreases from 0.490 at L2 to 0.284 at L23 ($\Delta = -0.205$; 95% CI $[-0.259, -0.157]$), whereas the matched-readout change is small and statistically unresolved ($\Delta = +0.007$; 95% CI $[-0.059, +0.077]$). We use NLL as a measure of probabilistic prediction quality rather than as a standalone measure of calibration.

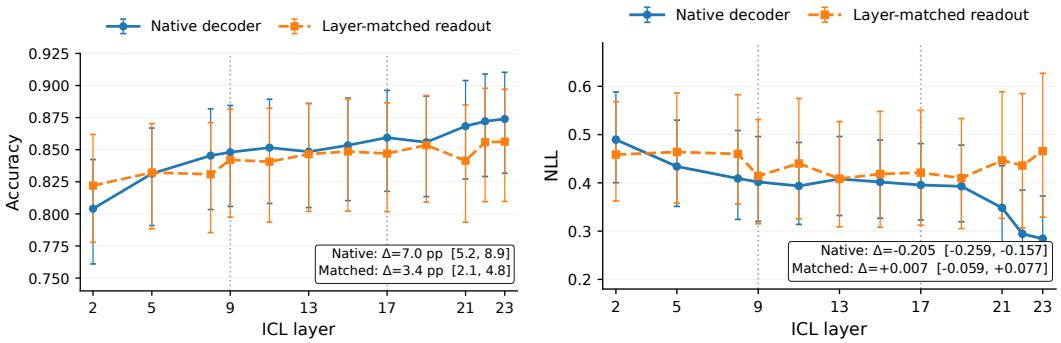

Figure 12: **Native-decoder and layer-matched readout performance across depth. Left:** classification accuracy averaged over 72 OpenML-CC18 tasks. **Right:** negative log-likelihood. Error bars denote 95% bootstrap confidence intervals over datasets. Vertical dotted lines at L9 and L17 indicate representative analytical boundaries rather than estimated change points.

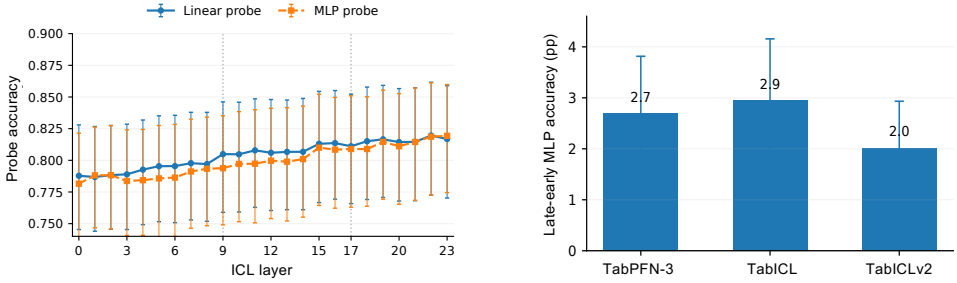

Figure 13: **Decoder-independent predictive decodability across depth. (a)** Linear and capacity-controlled MLP probe accuracy across the 24 ICL layers of TabPFN-3, averaged over 72 OpenML-CC18 tasks. Error bars denote 95% bootstrap confidence intervals over datasets; dotted vertical lines indicate the representative L9 and L17 analytical boundaries. **(b)** Change in MLP-probe accuracy from the first to the final third of the ICL layers for TabPFN-3, TabICL, and TabICLv2. Differences are computed within each dataset before aggregation, and error bars denote 95% bootstrap confidence intervals over the 72 tasks.

E.2 Decoder-Independent Probe Control

The matched-readout experiment controls specifically for alignment with TabPFN-3's final decoder. We additionally perform a broader decoder-independent probing analysis to test whether the representation dynamics identified in the main paper persist without using the native decoder at all.

Following the general probing setup used in prior TFM analyses (Balef et al., 2026), we extract query hidden states from each layer and train both a linear classifier and a capacity-controlled MLP probe with hidden dimension 64. Because support rows explicitly incorporate in-context labels, only query representations are used as probe examples. For each dataset, query examples are divided equally into probe-training and probe-evaluation subsets. We report accuracy, NLL, and ECE. The experiment is run on all 72 OpenML-CC18 tasks for TabPFN-3, TabICL, and TabICLv2. Across the 72 tasks, all three tested models exhibit higher MLP-probe accuracy in later than in earlier layers:

Figure 13 shows an overall increase in decoder-independent predictive accuracy across depth in TabPFN-3, despite local fluctuations, together with positive late-over-early MLP differences for TabPFN-3, TabICL, and TabICLv2.

Table 11 summarizes the corresponding cross-model decoder-independent probe results.

For TabPFN-3, a phase-aligned analysis for the detailed geometric study further separates changes in effective dimensionality from changes in predictive decodability, as summarized in Table 12:

Table 11: Decoder-independent probing across the 72 OpenML-CC18 tasks. Δ denotes the mean difference between the final and initial thirds of the layer sequence. "Positive" denotes the fraction of datasets with a positive MLP-accuracy difference.

Model	Δ MLP Acc.	Positive	Δ ECE	Δ MLP NLL
TabPFN-3	+0.027	64%	-0.005	-0.056
TabICL	+0.029	66%	-0.017	-0.179
TabICLv2	+0.020	59%	-0.008	-0.104

Table 12: Phase-wise changes in effective dimensionality and decoder-independent probe performance for TabPFN-3. The phase boundaries are representative analytical boundaries derived from the aggregate ED trajectory.

Phase	ΔED	Δ Linear Acc.	Δ MLP Acc.	Δ MLP NLL
Expansion	+10.95	+0.0274	+0.0278	-0.291
Plateau	+0.95	+0.0093	+0.0150	-0.011
Compression	-10.55	-0.0040	+0.0045	+0.023
Full depth	+1.40	+0.0327	+0.0473	-0.279

Across the full depth of TabPFN-3, MLP-probe accuracy increases by 4.73 percentage points (95% CI $= [+2.1, +8.1]$ pp), while MLP-probe NLL decreases by 0.279 (95% CI $= [-0.462, -0.144]$). ED, by contrast, returns close to its initial magnitude after expanding and then compressing.

Within the compression regime, the MLP-accuracy change is small ($+0.45$ pp) and its bootstrap interval includes zero. Thus, the control does not establish a significant compression-phase increase in decodability; it shows that the large reduction in ED occurs without a corresponding loss of decoder-independent predictive performance.

E.3 Robustness of the Geometric Analysis

We test the geometric analysis against ED-threshold choice, representation normalization, random-weight controls, representative phase boundaries, and PCA coordinate artifacts. For a layerwise ED trajectory, we define the expansion and compression ratios as

$$\rho_{\text{exp}} = \frac{\max_\ell \text{ED}(H^{(\ell)})}{\text{ED}(H^{(0)})}, \qquad \rho_{\text{com}} = \frac{\max_\ell \text{ED}(H^{(\ell)})}{\text{ED}(H^{(L)})}. \tag{24}$$

Values substantially above one indicate expansion from the initial representation and subsequent compression toward the final representation, respectively.

Sensitivity to the ED threshold. Our primary definition of ED uses the minimum number of principal components required to explain 90% of the variance. To test whether the resulting trajectory depends on this choice, we recompute ED using variance thresholds of 80%, 85%, 90%, and 95% on OpenML-CC18 datasets. Table 13 reports the resulting expansion and compression ratios.

Across all four thresholds, the mean expansion ratio remains at least $2.40\times$ and the mean compression ratio at least $1.84\times$; at least 95% of datasets satisfy each corresponding criterion.

Normalization control. Representation norms vary substantially across depth, raising the possibility that changes in ED could be induced by scale variation rather than by a change in the spectrum of representational degrees of freedom. We therefore recompute the complete ED trajectory after three transformations: row normalization, feature-wise standardization, and whitening. The experiment is conducted on six datasets using the same layerwise analysis pipeline. Table 14 summarizes the resulting expansion and compression ratios.

Table 13: **Sensitivity to the effective-dimensionality threshold.** Expansion and compression remain pronounced across variance thresholds from 80% to 95%. The last two columns report the fraction of datasets satisfying $\rho_{\mathrm{exp}} > 1.5$ and $\rho_{\mathrm{com}} > 1.3$, respectively.

Threshold	ρ_{exp}	ρ_{com}	Exp. > 1.5	Comp. > 1.3
80%	2.41×	2.80×	95%	100%
85%	2.40×	2.57×	95%	100%
90%	2.45×	2.24×	95%	95%
95%	2.54×	1.84×	100%	100%

Table 14: **Effective-dimensionality robustness to representation normalization.** The expansion–compression structure persists after removing sample-level norm, feature-wise scale, or covariance effects.

Representation	ρ_{exp}	ρ_{com}
Raw	4.00×	2.41×
Row normalization	3.70×	2.35×
Column standardization	3.16×	1.69×
Whitening	4.00×	2.41×

All six tested datasets retain expansion and compression under every normalization control. Row normalization, in particular, leaves the ratios close to the raw values ($3.70\times$ /$2.35\times$ versus $4.00\times$ /$2.41\times$), showing that the late ED reduction is not explained by representation-norm growth alone.

Random-weight control. We next test whether expansion followed by compression arises automatically from repeatedly applying the TabPFN-3 architecture, independent of learned weights. On representative datasets, we compare the pretrained model with two controls: one in which only the ICL blocks are randomly reinitialized while the surrounding embedding and decoder components remain pretrained, and one in which all model weights are randomly initialized. For the pretrained model, the four datasets exhibit expansion ratios between $2.88\times$ and $7.50\times$ and compression ratios between $2.09\times$ and $3.33\times$. In both randomized conditions, all four datasets instead yield

$$\rho_{\mathrm{exp}} = \rho_{\mathrm{com}} = 1.00, \tag{25}$$

corresponding to essentially flat ED trajectories. On these four datasets, randomizing either the ICL blocks or all weights therefore removes the expansion–compression trajectory. This control is limited to the tested settings and is not a claim about arbitrary randomly initialized Transformers.

Robustness of the representative phase boundaries. L9 and L17 are representative analytical boundaries derived from the aggregate ED trajectory, not estimated change points. For the sample-level trajectory analysis, we also replace the default $(L9, L13)$ pair with $(L8, L12)$, $(L10, L14)$, $(L9, L14)$, and $(L8, L13)$; the qualitative trajectory structure is preserved under these nearby choices.

Joint-fit PCA displacement. PCA coordinates are identifiable only up to rotations and sign changes. Fitting PCA independently at each layer could therefore cause a change in coordinate basis to be mistaken for movement of the underlying representations. To avoid this artifact, our PCA-displacement analysis uses a single shared basis. We concatenate query representations from all 24 layers and fit PCA once, retaining

$$k = \min(10, d, n-1) \tag{26}$$

components. Every layer is then projected into this same coordinate system. If $z_t^{(\ell)}$ denotes the resulting projection of sample t at layer ℓ, adjacent-layer displacement is computed as

$$d_{\mathrm{PCA}}\left(H^{(\ell)}, H^{(\ell+1)}\right) = \mathbb{E}_{t\in\mathcal{T}}\left[\left\| z_t^{(\ell+1)} - z_t^{(\ell)} \right\|_2\right]. \tag{27}$$

Displacement is computed within each dataset before equal-weight cross-dataset aggregation. Using one shared PCA basis prevents independent layerwise basis rotations from contributing to the measured displacement.

F CANDIDATE COMPETITION, EARLY READABILITY, AND LATE RESOLUTION

This appendix provides the full diagnostic analyses supporting Section 4.3. We first define the multi-class task pool and the prototype-based candidate measures used throughout the analysis, then report supplementary trajectories of candidate competition across depth, examine when class-relevant information becomes linearly readable from the attention-aggregated value stream, and finally quantify the coupling between late dimensional compression and candidate resolution.

F.1 TASK POOL AND CANDIDATE DIAGNOSTICS

The analysis covers all 105 multiclass classification tasks used in the main-text candidate study, including 25 tasks from OpenML-CC18 and 80 tasks from TALENT. All tasks are evaluated with the same frozen TabPFN-3 model and the same layerwise representation-extraction protocol.

All diagnostic quantities are first computed at the query level, then averaged within each dataset, and finally aggregated across datasets with equal dataset weight, so that datasets with more query examples do not dominate the pooled statistics.

For the difficulty-stratified analysis in Figure 4(b), tasks are grouped according to final full-model classification accuracy. Tasks with final accuracy at least 90% are labeled *Low difficulty*, tasks with final accuracy between 70% and 90% are labeled *Medium difficulty*, and tasks below 70% are labeled *High difficulty*.

For each layer ℓ and support class c, we construct a layer-specific prototype by averaging the support representations belonging to that class:

$$\mu_c^{(\ell)} = \frac{1}{|\mathcal{S}_c|} \sum_{n \in \mathcal{S}_c} h_{n,\mathrm{s}}^{(\ell)}, \tag{28}$$

where $\mathcal{S}_c$ denotes the support samples of class c.

For each query i, we compute the cosine similarity between its layer-ℓ representation and every class prototype:

$$s_{i,c}^{(\ell)} = \cos\left(h_i^{(\ell)}, \mu_c^{(\ell)}\right). \tag{29}$$

Our primary measure of candidate separation is the margin between the two most similar class prototypes:

$$\Delta_i^{(\ell)} = s_{i,(1)}^{(\ell)} - s_{i,(2)}^{(\ell)}, \tag{30}$$

where $s_{i,(1)}^{(\ell)} \geq s_{i,(2)}^{(\ell)}$ are the largest and second-largest prototype similarities. A small margin indicates that the two leading class-aligned candidates remain closely matched, whereas a large margin indicates more decisive separation.

We additionally record the rank of the correct-class prototype and the *correct-class top-rank rate*, defined as the fraction of queries for which the correct prototype occupies rank one.

To characterize the broader similarity structure across all class-aligned candidates, we also retain two supplementary diagnostics. First, we convert the prototype similarities into a normalized candidate distribution

$$\pi_{i,c}^{(\ell)} = \frac{\exp(s_{i,c}^{(\ell)})}{\sum_{c'} \exp(s_{i,c'}^{(\ell)})}, \tag{31}$$

and define the normalized candidate entropy

$$\widetilde{\mathcal{H}}_i^{(\ell)} = -\frac{\sum_c \pi_{i,c}^{(\ell)} \log \pi_{i,c}^{(\ell)}}{\log C_i}, \tag{32}$$

where C_i is the number of classes in the corresponding task.

Table 15: **Candidate diagnostics across the 105 multiclass tasks.** Entries are equal-weight dataset means. The final column counts tasks whose L23–L0 change matches the expected direction under candidate resolution.

Metric	L0	L9	L13	L17	L20	L23	Consistent
Normalized candidate entropy	.997	.994	.995	.993	.992	.978	104/105
Active candidate fraction	.932	.599	.486	.529	.465	.465	103/105
Prototype margin	.055	.125	.104	.129	.147	.247	104/105
Mean correct-class rank	2.082	1.753	1.803	1.727	1.689	1.645	104/105
Correct-class top-rank rate	.634	.731	.710	.729	.739	.770	104/105

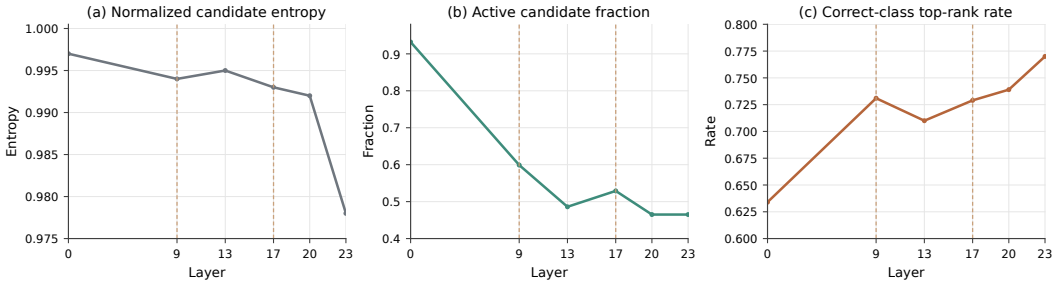

Figure 14: **Supplementary diagnostics of candidate structure across depth. (a)** Normalized candidate entropy remains high through intermediate depth and decreases most visibly near the output. **(b)** The active candidate fraction contracts overall but shows a local rebound in the middle regime. **(c)** Correct-class top-rank rate improves overall, but its intermediate trajectory is non-monotonic. Dashed vertical lines mark the representative analytical boundaries at L9 and L17.

Second, we define the active candidate fraction

$$A_i^{(\ell)} = \frac{1}{C_i} \sum_c \mathbf{1}\left[s_{i,c}^{(\ell)} > 0.6 \max_{c'} s_{i,c'}^{(\ell)} \right]. \tag{33}$$

This quantity measures the fraction of class prototypes that remain competitive under a fixed relative-similarity threshold. These two quantities serve as supplementary consistency checks rather than the primary basis of the candidate-resolution conclusion.

F.2 Candidate Competition Across Depth

Table 15 summarizes the candidate diagnostics at representative layers.

Across tasks, the endpoint change from L0 to L23 is highly consistent with candidate resolution: normalized candidate entropy decreases, active candidate fraction decreases, prototype margin increases, mean correct-class rank improves, and correct-class top-rank rate increases. The intermediate-depth trajectories, however, are not strictly monotonic.

Prototype margin increases from 0.055 at L0 to 0.125 at L9, then decreases to 0.104 at L13 before increasing again and reaching 0.247 at L23. Similarly, the active candidate fraction decreases from 0.932 at L0 to 0.486 at L13, but rises to 0.529 at L17 before decreasing again. Intermediate depth is therefore characterized by persistent candidate competition together with local reorganization rather than by smooth monotonic separation.

Figure 14 visualizes the three supplementary diagnostics not shown directly in the main text. Candidate entropy remains close to its maximum through much of the network and drops most clearly near the output. The active candidate fraction contracts substantially at early depth but shows a local rebound in the middle layers. Correct-class top-rank rate also exhibits a mild mid-depth reversal before improving again toward L23. These observations reinforce the main text conclusion that candidate resolution is strongly depth-dependent but not smoothly monotonic.

The strongest candidate separation is concentrated near the output. Between L20 and L23, prototype margin increases from 0.147 to 0.247, while normalized candidate entropy decreases from 0.992 to

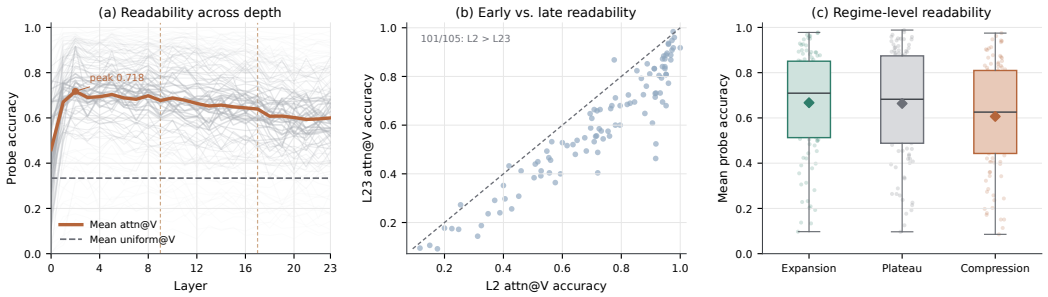

Figure 15: **Class information becomes linearly readable before candidate resolution is complete. (a)** Mean attn@V probe accuracy across depth, with 95% bootstrap confidence intervals over datasets; uniform@V serves as a matched aggregation baseline. The mean attn@V trajectory peaks at L2. **(b)** Dataset-level comparison of attn@V readability at L2 and L23. Most datasets lie below the diagonal (101/105), indicating higher readability at L2 than at L23. **(c)** Distribution of mean probe accuracy across datasets within the Expansion, Plateau, and Compression regimes; diamonds indicate equal-weight dataset means.

0.978. This late-depth transition coincides with the late effective-dimensionality contraction and increased PCA displacement reported in Section 4.2. Candidate geometry therefore remains only partially resolved through intermediate depth and becomes substantially more decisive near the output.

The difficulty-stratified correct-class rank curves in Figure 4(b) provide a complementary view. Lower-difficulty tasks establish favorable correct-class alignment earlier, whereas higher-difficulty tasks retain worse correct-class ranks deeper into the network. Candidate ambiguity therefore varies with both depth and task difficulty.

F.3 Early Readability of the Attention-Aggregated Value Stream

We next examine whether persistent candidate ambiguity reflects the absence of class-relevant information at shallow depth. To do so, we probe the attention-aggregated value representation at every layer.

For each layer, we train a linear probe on the value representation produced by learned attention-weighted aggregation, denoted *attn@V*. As a matched baseline, we apply the same readout to value representations aggregated with uniform attention weights, denoted *uniform@V*. This analysis asks when class-relevant information becomes linearly readable, rather than whether the probe reproduces the model's native decision rule.

Figure 15(a) shows that linear readability emerges at shallow depth. Mean attn@V accuracy increases from 0.460 at L0 to 0.718 at L2, then gradually decreases to 0.600 at L23. The mean uniform@V baseline remains near 0.334 throughout. Class information is therefore highly linearly readable in the value stream well before candidate geometry reaches its strongest separation.

This early-readability pattern is also prevalent at the dataset level. Figure 15(b) compares attn@V accuracy at L2 and L23 for each of the 105 tasks. In 101 of 105 datasets, attn@V accuracy is higher at L2 than at L23. Figure 15(c) shows the same pattern after regime-wise aggregation. Mean readability is 0.667 in the expansion regime, 0.664 in the plateau regime, and 0.607 in the compression regime. Accordingly, 101 of 105 tasks satisfy Plateau $>$ Compression, whereas no systematic ordering exists between Expansion and Plateau.

These results separate *information availability* from *candidate resolution*: class-relevant information becomes linearly readable early, while candidate geometry continues to reorganize and reaches its strongest separation substantially later.

F.4 Coupling Late Compression to Candidate Resolution

We next quantify whether the late geometric contraction identified in Section 4.2 systematically corresponds, across datasets, to the late increase in candidate separation.

Table 16: **Summary statistics for attn@V readability across the 105 multiclass tasks.** Unless otherwise stated, values are equal-weight dataset means.

Statistic	Value
L0 attn@V accuracy	.460
L2 attn@V accuracy	.718
L23 attn@V accuracy	.600
Mean uniform@V accuracy	.334
Expansion mean	.667
Plateau mean	.664
Compression mean	.607
Tasks with L2 > L23	101/105
Tasks with Plateau > Compression	101/105
Tasks with Expansion > Compression	98/105

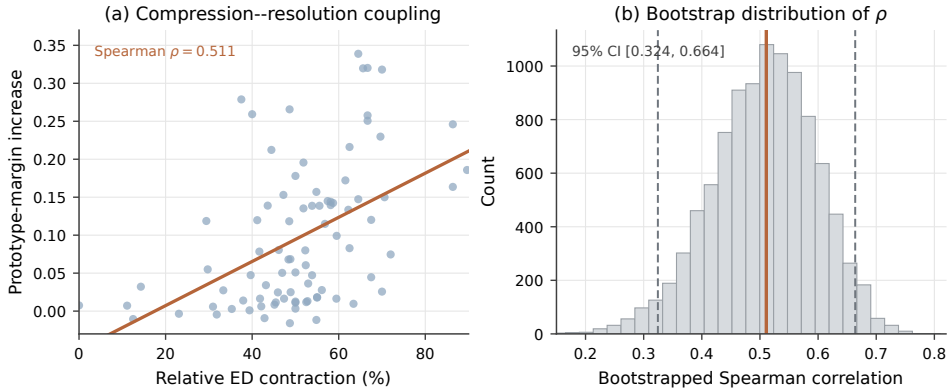

Figure 16: **Robustness of the compression–resolution relationship.** **(a)** Relative effective-dimensionality contraction from L20 to L23 versus the corresponding increase in prototype margin across matched datasets. **(b)** Bootstrap distribution of the Spearman correlation. The solid line marks the observed correlation and dashed lines denote the 95% bootstrap confidence interval.

For datasets with both effective-dimensionality and prototype-margin measurements available over L20–L23, we define relative late compression as

$$C_d = \frac{\mathrm{ED}_d^{(20)} - \mathrm{ED}_d^{(23)}}{\mathrm{ED}_d^{(20)}}, \tag{34}$$

and late candidate resolution as

$$R_d = \Delta_d^{(23)} - \Delta_d^{(20)}. \tag{35}$$

Across all 105 datasets, stronger late compression is associated with larger late increases in prototype margin, with Spearman $\rho = 0.511$, $p = 5.85 \times 10^{-7}$, and bootstrap 95% CI $[0.324, 0.664]$, as shown in Figure 16(a). Late compression and candidate separation therefore systematically covary across datasets.

The relationship remains stable under additional checks. In a leave-one-dataset-out recomputation, the Spearman correlation remains in the range $[0.495, 0.546]$. When the late margin increase is normalized by the L20 prototype margin, the association remains positive, with $\rho = 0.440$ and $p = 2.56 \times 10^{-5}$. The bootstrap distribution in Figure 16(b) is likewise concentrated above zero.

These analyses establish a systematic association between late compression and late candidate resolution across datasets; the analysis is correlational and does not identify a causal direction.

Table 17: **Compression–resolution coupling and robustness checks.**

Statistic	Value
Matched datasets	105
Analysis interval	L20–L23
Spearman $\rho(C_d, R_d)$	.511
p-value	5.85×10^{-7}
Bootstrap 95% CI	[.324, .664]
Leave-one-dataset-out ρ range	[.495, .546]
Relative-margin normalization ρ	.440
Relative-margin normalization p-value	2.56×10^{-5}

G Supplementary Analyses of Sample-Level Predictive Commitment

This appendix provides the definitions, aggregation details, and robustness analyses underlying the sample-level results in Sections 5.1–5.2. The prediction-trajectory analyses cover 272 datasets. All trajectory quantities are computed from the complete sequence of 24 native-decoder predictions rather than inferred from a small set of representative layers.

G.1 Stabilization Depth and Prediction-Trajectory Assignment

For sample i, let

$$\mathbf{p}_i = (\hat{y}_i^{(0)}, \ldots, \hat{y}_i^{(23)}) \tag{36}$$

denote the full prediction trajectory. We define stabilization depth as

$$s_i = \min \left\{ \ell : \hat{y}_i^{(k)} = \hat{y}_i^{(23)},\ \forall k \geq \ell \right\}. \tag{37}$$

Because the final layer always satisfies this condition, $s_i \in [0, 23]$. Samples whose prediction equals the final class at every layer have $s_i = 0$.

We separately define the total number of adjacent-layer class transitions as

$$n_{\mathrm{chg},i} = \sum_{\ell=1}^{23} \mathbb{I}\left[\hat{y}_i^{(\ell)} \neq \hat{y}_i^{(\ell-1)} \right]. \tag{38}$$

The five-pattern taxonomy used in the main paper is label-free. With $p_i^{(\ell)} = \hat{y}_i^{(\ell)}$, the assignment is

$$\mathrm{Pattern}(i) = \begin{cases} \mathrm{ES}, & p_i^{(9)} = p_i^{(23)} \ \wedge\ n_{\mathrm{chg},i} \leq 1, \\ \mathrm{TS}, & p_i^{(9)} = p_i^{(23)} \ \wedge\ n_{\mathrm{chg},i} > 1, \\ \mathrm{IS}, & p_i^{(9)} \neq p_i^{(23)} \ \wedge\ p_i^{(13)} = p_i^{(23)}, \\ \mathrm{LS}, & p_i^{(9)} = p_i^{(13)} \neq p_i^{(23)}, \\ \mathrm{MS}, & \text{otherwise.} \end{cases} \tag{39}$$

The conditions are mutually exclusive and exhaustive. Final correctness is not used anywhere in the five-pattern assignment. The trajectories shown in Figure 5(b) are schematic representatives. Pattern assignment uses $n_{\mathrm{chg},i}$ from all 24 layers together with the representative predictions at L9, L13, and L23; L0 is shown only for visual context.

G.2 Aggregation Protocol and Pattern Prevalence

Figure 5(b) reports pooled prevalence across all 105,801 test samples. Dataset-level analyses instead assign equal weight to each of the 272 datasets. Table 19 shows that the two aggregation schemes yield similar central prevalences, indicating that the dominance of ES and the relative rarity of MS are also present under dataset-level aggregation.

Table 18: **Five prediction-transition patterns.** The rules are applied to the complete 24-layer trajectory and do not use the ground-truth label.

Pattern	Representative form	Assignment criterion
ES	$A \to A \to A \to A$	$p^{(9)} = p^{(23)},\ n_{\text{chg}} \leq 1$
TS	$A \to B \to A \to A$	$p^{(9)} = p^{(23)},\ n_{\text{chg}} > 1$
IS	$A \to B \to B \to B$	$p^{(9)} \neq p^{(23)},\ p^{(13)} = p^{(23)}$
LS	$A \to A \to A \to B$	$p^{(9)} = p^{(13)} \neq p^{(23)}$
MS	$A \to B \to A \to C$	otherwise

Table 19: **Pattern prevalence under pooled and dataset-macro aggregation.** Macro values are mean $\pm$ standard deviation across 272 datasets.

Pattern	Pooled samples	Dataset macro-average
ES	72.2%	71.6% $\pm$ 22.3%
TS	15.8%	15.9% $\pm$ 10.9%
IS	5.3%	5.3% $\pm$ 5.2%
LS	5.8%	6.1% $\pm$ 8.0%
MS	1.0%	1.1% $\pm$ 3.5%

For the difficulty visualization in Figure 5(c), we first compute chance-normalized difficulty continuously for every dataset and then divide datasets into equal-sized tertiles. Pattern fractions are averaged within each tertile with equal dataset weight. Table 20 gives the exact values underlying the stacked bars.

G.3 Robustness to Representative-Layer Choice

Stabilization depth is computed from the full trajectory and therefore does not depend on L9 or L13. The discrete taxonomy uses these depths only to localize the timing of a transition. To test sensitivity to this choice, we recompute the five-pattern assignment after shifting the representative early/intermediate depths by one layer.

Table 21 shows that every pattern changes by less than one percentage point from the default prevalence under all tested alternatives. ES remains dominant, and the relative prevalence of the remaining categories is qualitatively stable.

G.4 Task-Difficulty Definition and Class-Imbalance Robustness

The main text defines model-relative task difficulty using chance-normalized final accuracy,

$$D_d^{\text{chance}} = 1 - \frac{a_d - 1/C_d}{1 - 1/C_d}, \tag{40}$$

where a_d is final-layer accuracy and C_d is the number of classes. This normalization removes the first-order dependence of raw accuracy on class count. All correlations treat difficulty as continuous; tertiles are used only for visualization.

Because $1/C_d$ is not an imbalance-aware baseline, we additionally test three alternative definitions: raw error $1 - a_d$, balanced error $1 - \text{BA}_d$, and majority-normalized difficulty

$$D_d^{\text{maj}} = 1 - \frac{a_d - b_d}{1 - b_d}, \tag{41}$$

where b_d is the majority-class frequency in the evaluation split.

Table 22 and Figure 17 report the association of each measure with mean stabilization depth. The association remains positive under all four definitions. Its magnitude is strongest for raw and class-count-normalized difficulty and remains statistically resolved under both imbalance-aware alternatives.

Table 20: **Pattern composition across chance-normalized difficulty tertiles.** Tertiles are used only for visualization.

Difficulty	ES	TS	IS	LS	MS
Easy	86.3%	9.5%	2.1%	2.0%	0.2%
Medium	74.5%	15.0%	4.0%	5.4%	1.1%
Hard	54.2%	23.2%	9.8%	10.9%	1.9%

Table 21: **Sensitivity of pooled pattern prevalence to representative layers.** L23 is fixed as the final layer.

Representative depths	ES	TS	IS	LS	MS
Default	72.2%	15.8%	5.3%	5.8%	1.0%
$L8, L12$	72.1%	15.5%	4.9%	6.5%	1.1%
$L10, L14$	72.3%	16.7%	5.0%	5.0%	1.0%
$L9, L14$	72.2%	15.8%	6.1%	4.9%	1.1%
$L8, L13$	72.1%	15.5%	5.9%	5.5%	1.0%

G.5 Representation Dynamics After Early Predictive Commitment

The main representation analyses characterize geometry over the full set of query samples. We further examine whether substantial representation dynamics persist after predicted class identity has already stabilized. To this end, we repeat the geometric analysis after restricting the query set to *Early-Stable (ES)* samples.

Under the trajectory definition in Appendix G.1, an ES sample satisfies $p_i^{(9)} = p_i^{(23)}$ and $n_{\mathrm{chg},i} \leq 1$. These conditions imply that its predicted class cannot change after L9: a departure from the L9 prediction followed by a return to the final prediction would require at least two additional class transitions. L9 therefore provides a common reference point after which all ES predictions have committed to their final class identity.

For this targeted analysis, we consider the 60 datasets with the highest ES prevalence, providing well-populated ES subsets for representation-space estimation. ES prevalence ranges from 88.2% to 100% across these datasets, corresponding to 23,737 ES queries among 25,426 evaluated samples. This analysis specifically characterizes post-commitment representation dynamics in datasets dominated by early predictive commitment.

For each dataset, we recompute effective dimensionality using only ES query representations while retaining the same definition and 90% variance threshold as in Appendix B.2. We additionally measure adjacent-layer representation displacement using the shared-PCA procedure defined in Appendix B.3.

Figure 18(a) shows that restricting the analysis to ES samples preserves the characteristic Expansion–Compression trajectory. Across the 60 datasets, mean ES-only effective dimensionality is 22.8 at L9, increases to 28.0 at L20, and then contracts to 13.3 at L23. In 58 of the 60 datasets, effective dimensionality rises above its L9 value at a later layer, and all 60 datasets exhibit a post-L9 dimensional maximum above their final-layer effective dimensionality.

The displacement analysis provides a complementary view (Figure 18(b)). Despite stable class identity, ES representations continue to move substantially after L9. Averaged at the dataset level, 69.2% of total adjacent-layer displacement occurs after L9, while the final six transitions alone account for 43.8%. Mean transition-wise displacement is also larger in the late regime than in the intermediate regime (5.47 versus 2.37), with this ordering observed in all 60 analyzed datasets.

These results separate predictive commitment from representational stasis. Even after predicted class identity has stabilized, ES representations continue to undergo substantial depth-dependent transformation, including further expansion, late reorganization, and final compression. Predictive commitment can therefore precede substantial later representational change.

Table 22: **Robustness of the difficulty–commitment association.** Bootstrap intervals resample datasets (5,000 replicates). Positive ρ means harder datasets stabilize later on average.

Difficulty definition	n	Spearman ρ	95% bootstrap CI	p
Raw error	272	0.775	[0.702, 0.831]	3.8×10^{-52}
Chance-normalized	272	0.694	[0.614, 0.761]	8.2×10^{-38}
Balanced error	272	0.505	[0.389, 0.608]	7.5×10^{-18}
Majority-normalized	272	0.361	[0.240, 0.469]	3.1×10^{-9}

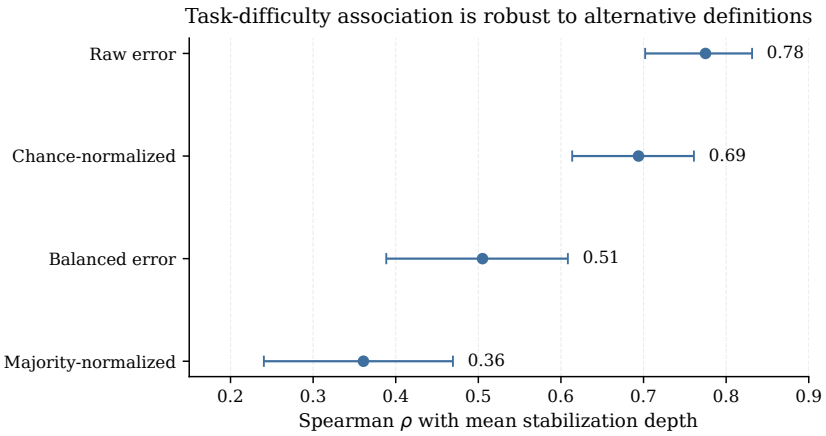

Figure 17: **Difficulty–commitment association under alternative task difficulty definitions.** Points show Spearman correlations with mean stabilization depth; error bars are 95% dataset-bootstrap intervals.

G.6 Local Class Separation and Dataset-Aware Statistics

The local-topology analysis is performed in the preprocessed input feature space, not in a hidden representation. Numerical features follow the common training-fitted standardization used by the data pipeline, and categorical features use the same encoded input representation used by the topology analysis.

For each analyzed sample x_i, the diagnostic compares the nearest same-class and different-class distances,

$$r_i = \frac{d(x_i, \mathrm{NN}_{\mathrm{diff}})}{d(x_i, \mathrm{NN}_{\mathrm{same}})}. \tag{42}$$

Larger r_i indicates clearer local class separation. Class labels are used only to identify same- versus different-class neighbors for this retrospective diagnostic; they are not used by the five-pattern trajectory classifier.

The pooled medians shown in the main text are descriptive. Because samples are nested within datasets, we additionally perform dataset-aware inference. For each dataset and trajectory pattern, we first average the available sample-level separation ratios. Figure 19 shows the resulting distribution of dataset-level pattern means. ES has the largest median dataset-level separation, whereas MS has the smallest.

We further compare ES with each non-ES pattern only within datasets for which both estimates are available. Table 25 reports one-sided paired Wilcoxon tests with Holm correction across the four comparisons. ES has higher local separation in approximately 95% of shared datasets for every comparison, including MS. This dataset-aware analysis reproduces the pooled ordering without treating individual samples from the same dataset as independent experimental units.

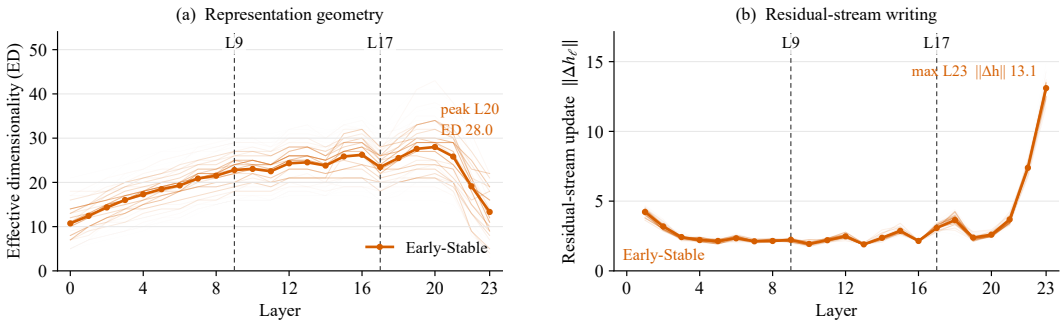

Figure 18: **Representation dynamics continue after predictive commitment.** Results are computed using Early-Stable samples only; faded curves denote individual datasets and bold curves denote the dataset-level mean. **(a)** Effective dimensionality continues to increase after L9, rising from 22.8 at L9 to 28.0 at L20 before contracting to 13.3 at L23. **(b)** Representation movement remains substantial after L9 and is concentrated toward later depth.

Table 23: **Summary of ES-only post-commitment representation dynamics.** All statistics are computed after restricting the query representations to Early-Stable samples. Displacement fractions are first computed within each dataset and then macro-averaged across datasets.

Statistic	ES-only result
Analyzed datasets	60
ES queries	23,737
ES prevalence range	88.2–100%
Mean ED at L9	22.8
Mean ED at L20	28.0
Mean ED at L23	13.3
Datasets with ED increase after L9	58/60
Datasets with post-L9 contraction	60/60
Displacement occurring after L9	69.2%
Displacement in final six transitions	43.8%

Table 24: **Local separation at sample and dataset levels.** Sample entries report pooled median [IQR]. Dataset entries report the median [IQR] of within-dataset pattern means.

Pattern	Pooled median [IQR]	Datasets	Dataset median [IQR]
ES	1.52 [1.11, 2.50]	270	1.90 [1.34, 6.26]
TS	1.03 [0.85, 1.34]	263	1.15 [1.05, 1.39]
IS	0.97 [0.76, 1.16]	251	1.03 [0.96, 1.17]
LS	0.96 [0.72, 1.16]	251	1.04 [0.97, 1.17]
MS	0.81 [0.61, 1.01]	93	0.87 [0.71, 1.02]

Table 25: **Dataset-aware paired comparisons of local separation.** Δ is the median within-dataset difference in mean separation ratio; p values use a one-sided paired Wilcoxon test with Holm correction.

Comparison	Shared datasets	Median Δ	ES higher	Holm p
ES > TS	263	0.57	94.7%	$< 10^{-11}$
ES > IS	251	0.78	96.5%	$< 10^{-11}$
ES > LS	251	0.72	94.8%	$< 10^{-11}$
ES > MS	93	0.86	94.5%	$< 10^{-11}$

H Supplementary Analyses of Late-Layer Functional Value

This appendix provides additional analyses supporting Section 6. We first examine the robustness of the relation between predictive commitment and truncation sensitivity across alternative truncation

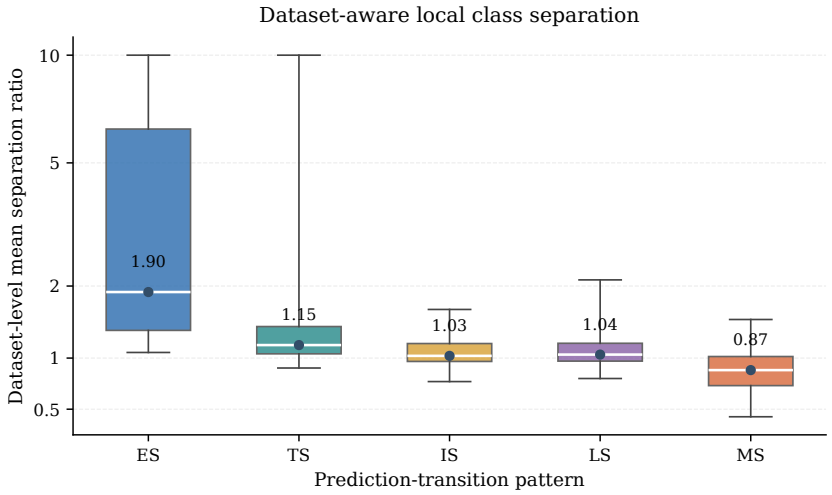

Figure 19: **Dataset-aware local class separation.** Each box summarizes within-dataset mean separation ratios for one trajectory pattern. The vertical axis is log-compressed and values above 10 are clipped for display; all reported statistics use the uncapped ratios. Tick labels remain on the original ratio scale. Numbers below the axis indicate datasets containing a finite estimate for that pattern.

depths and evaluation metrics. We then test the stability of the retention–rescue analysis to the choice of shallow reference layer, followed by geometric, compute-matched, predictive, and layer-order controls for the functional role of later computation.

H.1 Robustness of Late-Layer Utility

The main text examines whether tasks with later predictive commitment are more sensitive to removing subsequent computation. For dataset d, the accuracy loss induced by truncation at layer ℓ is defined as

$$\Delta\mathrm{Acc}_d^{(\ell)} = \mathrm{Acc}_d^{(23)} - \mathrm{Acc}_d^{(\ell)}. \tag{43}$$

We repeat the analysis at L13, L17, and L20 and additionally replace accuracy with balanced accuracy to reduce sensitivity to class imbalance.

Table 26 shows that mean stabilization depth is positively associated with truncation loss under all six variants. The association is strongest for earlier truncation points, where a larger portion of the remaining computation is removed, and remains positive at L20.

Table 26: **Robustness of the commitment–truncation association.** Spearman correlations relate dataset-level mean stabilization depth to performance loss under truncation. Confidence intervals are obtained by dataset bootstrap.

Truncation metric	ρ	95% CI	p
Accuracy, L13	.551	[.443, .643]	1.55×10^{-21}
Balanced accuracy, L13	.468	[.365, .561]	3.25×10^{-15}
Accuracy, L17	.481	[.366, .583]	3.95×10^{-16}
Balanced accuracy, L17	.408	[.300, .504]	1.32×10^{-11}
Accuracy, L20	.344	[.220, .456]	1.80×10^{-8}
Balanced accuracy, L20	.308	[.191, .419]	5.72×10^{-7}

Later computation also improves probabilistic prediction quality. Comparing L13 with L23, NLL decreases in 58 of 60 datasets (96.7%), while ECE decreases in 57 of 60 datasets (95.0%). Top-1 accuracy increases in 54 datasets, remains unchanged in three, and decreases in three. Improvements in NLL and ECE are therefore more consistent across tasks than changes in the discrete class prediction.

We additionally evaluate each depth using a layer-specific matched readout. Table 27 reports the resulting trajectory alongside the native decoder.

Table 27: **Native and layer-specific matched readouts across depth.** A separate matched readout is trained at each layer to evaluate predictive information available in the corresponding representation.

Depth	Native Acc.	Native NLL	Matched Acc.	Matched NLL
L2	.804	.490	.822	.459
L5	.831	.434	.832	.464
L8	.845	.409	.831	.460
L9	.848	.402	.842	.414
L13	.848	.408	.847	.409
L17	.859	.396	.847	.421
L19	.856	.393	.854	.410
L21	.868	.348	.842	.446
L23	.874	.284	.856	.466

Matched-readout accuracy increases from .822 at L2 to .856 at L23, a 3.4 percentage-point gain. Native accuracy increases from .804 to .874 over the same interval. Later representations therefore provide greater classification utility even when each layer is evaluated with its own matched readout.

The probabilistic trajectories differ from the accuracy trajectories. Native NLL decreases substantially with depth, whereas matched-readout NLL remains comparatively stable. The native decoder therefore extracts increasingly better probabilistic predictions from later representations, while the matched-readout trajectory shows a smaller but persistent increase in classification accuracy.

H.2 Sensitivity to the Shallow Reference Layer

The retention–rescue analysis in the main text uses L9 as the shallow reference layer. We repeat the analysis with L8 and L10 while holding the remaining protocol fixed.

Table 28: **Retention and rescue under different shallow reference layers.** Values are cohort means over 272 datasets. The first two columns summarize the endpoint changes across local-separation percentiles, and the final two columns report sample-level Spearman correlations with local separation ($n \approx 9.6 \times 10^4$; all $p < 10^{-230}$).

Shallow reference	Retain (%)	Rescue (%)	ρ_{retain}	ρ_{rescue}
L8	83.6 → 95.8	32.0 → 53.7	+0.203	+0.253
L9	83.2 → 95.9	30.4 → 50.2	+0.187	+0.244
L10	85.0 → 96.1	28.1 → 51.9	+0.194	+0.240

Across all three reference layers, retention and rescue increase with local separation, with similar correlation magnitudes and response profiles. The relation between local separation and later predictive change is therefore stable across nearby shallow reference depths.

H.3 Shared-Weight and Matched-Compute Controls

The shared-weight intervention compares the standard sequence of distinct ICL blocks with repeated application of a fixed transformation. We extract a single ICL block and repeatedly apply it to the evolving residual stream, using the same geometric analysis as for the standard 24-layer network.

For a representation trajectory with effective dimension $\mathrm{ED}^{(\ell)}$, we define the compression ratio as

$$R_{\text{comp}} = \frac{\max_\ell \mathrm{ED}^{(\ell)}}{\mathrm{ED}^{(\text{final})}}, \tag{44}$$

and define FDR gain as the ratio between the final and initial class-discriminative Fisher ratio.

Table 29 summarizes the primary comparison. Repeated application of the shared block produces substantially weaker late compression and weaker growth of class-discriminative structure.

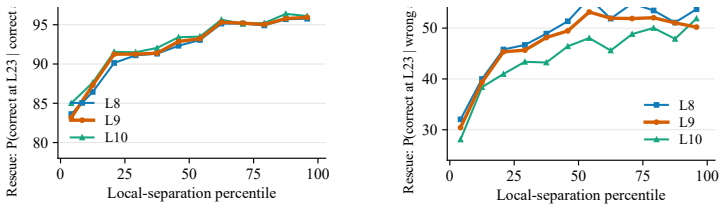

Figure 20: **Retention and rescue across alternative shallow reference layers.** **(a)** Retain, $P(\text{correct at L23} \mid \text{correct at the reference layer})$, and **(b)** Rescue, $P(\text{correct at L23} \mid \text{wrong at the reference layer})$, plotted against within-dataset local-separation percentile. Results are shown for L8, L9, and L10, with L9 used as the main-text reference. Curves denote cohort means over 272 datasets.

Table 29: **Standard depth-specific computation versus shared-weight iteration.** Values are dataset means $\pm$ standard deviations. Wilcoxon tests are one-sided for the hypothesis that the standard model yields the larger quantity.

Metric	Standard	Shared	p
Compression ratio	2.85 ± 1.01	1.05 ± 0.08	9.5×10^{-7}
FDR gain	2.53 ± 1.94	0.84 ± 0.26	2.9×10^{-6}

All shared-weight runs have compression ratios below 1.3. The standard depth-specific sequence therefore exhibits substantially stronger late compression than repeated application of a fixed block. The FDR comparison shows a parallel difference in class-discriminative structure, which grows strongly across the standard sequence but remains much smaller under shared weights.

We next match the number of computational transformations by repeating the shared block for 24 steps. Expansion is defined as

$$R_{\text{exp}} = \frac{\max_\ell \text{ED}^{(\ell)}}{\text{ED}^{(0)}}, \tag{45}$$

with compression defined as above.

Table 30: **Matched-step shared-weight control.** Each entry reports expansion/compression ratio. The shared block is iterated for 24 steps, matching the number of transformations in the standard network.

Dataset	Standard 24-layer	Shared $\times 24$
vehicle	4.00/2.33	1.57/1.22
segment	4.67/2.33	1.33/1.14
satimage	7.50/3.33	2.50/1.00
diabetes	2.88/2.09	1.62/1.62

Across all four tasks, the 24-step shared-weight trajectory remains substantially different from the standard model. Expansion is weaker in every case, and compression remains below the corresponding standard value. Distinct depth-specific transformations therefore produce a different representation trajectory from repeated application of the same operation, even at matched computational depth.

H.4 Predictive Robustness Under Data Perturbations

We further compare the standard and shared-weight models under perturbed evaluation conditions. For each dataset and condition, we report

$$\Delta\text{Acc} = \text{Acc}_{\text{standard}} - \text{Acc}_{\text{shared}}, \tag{46}$$

so that positive values favor the standard model.

Table 31 reports results on ten representative datasets. The clean condition is included alongside 20% label noise, feature noise, and reduced training-sample availability.

Table 31: **Predictive advantage of the standard model under data perturbations.** Entries are $\Delta\mathrm{Acc} = \mathrm{Acc}_{\mathrm{standard}} - \mathrm{Acc}_{\mathrm{shared}}$. Positive values indicate higher accuracy for the standard model.

Dataset	Clean	20% label noise	Feature noise	Half samples
adult	+.005	+.030	.000	+.050
cmc	+.030	+.065	+.035	+.090
credit-g	+.030	+.015	+.040	+.085
diabetes	-.019	+.045	+.013	.000
balance-scale	+.232	+.288	+.056	+.256
vehicle	+.312	+.194	+.153	+.171
car	+.205	+.180	+.070	+.235
bank-marketing	+.015	+.005	+.010	-.015
credit-approval	+.007	+.072	+.043	+.029
electricity	+.060	+.010	+.055	+.010
Mean	+.088	+.090	+.048	+.091
Standard > Shared	9/10	10/10	9/10	8/10

The standard model has a positive mean accuracy advantage in all four conditions. Under 20% label noise, the advantage occurs on all ten datasets. Under feature noise, it occurs on nine datasets with one tie; with half the training samples, it occurs on eight datasets with one tie.

The shared-weight and standard models therefore remain predictively distinct under clean and perturbed evaluation conditions, with the largest cross-dataset consistency observed under label noise.

H.5 Layer-Order Perturbation Statistics

Section 6.2 changes the execution order of the learned layers while preserving the blocks themselves and the number of computational steps. We consider both a shuffled ordering and complete reversal of the learned sequence.

Table 32 summarizes the dataset-level effects on effective-dimensional compression and predictive accuracy. Wilcoxon tests are paired and one-sided for the hypothesis that the learned ordering produces the larger value.

Table 32: **Dataset-level effects of layer-order perturbation.** "Learned better" reports the number of datasets on which the learned ordering yields the larger metric. Ties are omitted from this count.

Metric	Learned	Perturbed	Learned better	p
Compression, shuffled	1.935	1.053	60/60	8.65×10^{-7}
Compression, reversed	1.935	1.012	60/60	8.64×10^{-7}
Accuracy, shuffled	.806	.650	58/60	1.42×10^{-6}
Accuracy, reversed	.806	.682	56/60	5.53×10^{-6}

Effective-dimensional compression is lower after both shuffling and reversal on all 60 datasets. The learned ordering also yields higher accuracy than the shuffled ordering on 58 datasets and higher accuracy than the reversed ordering on 56 datasets.

Changing layer order therefore alters both the representation trajectory and predictive behavior. Together with the shared-weight controls, these results show that later computation is organized as an ordered sequence of distinct learned transformations.