CONFORMAL MAPS FOR LLM EVALUATION: LEARNING BEHAVIOR-AWARE INSTRUCTION GEOMETRY

Anonymous authors
Paper under double-blind review

ABSTRACT

Aggregate benchmark scores collapse heterogeneous patterns of model success and failure into a single number. A more informative evaluation should characterize this variation at the level of individual instructions, relating a held-out instruction to previously observed model behavior while quantifying uncertainty in that prediction. Conformal prediction provides a natural framework for the latter through finite-sample coverage guarantees, but the informativeness of its prediction sets depends on the score being calibrated. We introduce *conformal maps*, learned instruction embedding spaces whose local support is optimized for informative label-conditional conformal prediction while remaining anchored to a pretrained semantic representation. The same measure of local support organizes historical successes and failures, defines the conformity score, and exposes the evidence surrounding a new instruction. Across five QA and reasoning benchmarks and three target models, the learned maps induce measurable behavioral organization, retain a controllable amount of pretrained structure, and improve conformal singleton rate over the pretrained representation in most dataset–model pairs while maintaining label-conditional coverage. The learned geometry also exposes the amount, direction, and provenance of local behavioral evidence, providing an inspectable account of which prior instructions support a prediction.

1 INTRODUCTION

From aggregate evaluation to instruction-level behavior. Benchmark evaluation typically summarizes a model by an average accuracy or pass rate. Such summaries are useful for comparing systems, but they hide how success and failure are distributed across individual instructions (Liang et al., 2023; Hernández-Orallo et al., 2022). A model may perform well overall while failing systematically on particular kinds of requests, and two models with similar aggregate accuracy may fail on very different examples. Manually defined slices provide a finer view (Ribeiro et al., 2020), but they are restricted to categories specified in advance and may miss structure that is specific to the behavior of the model being evaluated.

This limitation is especially important when evaluation must generalize beyond the instructions already tested. Assessing a new instruction requires relating it to previously observed model behavior (Schellaert et al., 2025), which in turn requires a notion of which evaluated examples are relevant. Semantic representations offer one such notion, organizing instructions by meaning. However, semantic similarity need not predict whether the target model will behave similarly on two instructions (Rabinovich et al., 2023). An instruction-level evaluator should therefore learn a behavior-aware notion of neighborhood without discarding the semantic structure that makes local comparisons meaningful. It should also quantify uncertainty with a clear statistical interpretation, resolving success from failure when the available evidence permits.

Conformal prediction (Vovk et al., 2005) is particularly appealing in this setting because it produces prediction sets with finite-sample coverage under exchangeability. Its validity, however, does not by itself make the resulting sets informative (Pekala et al., 2012; Stutz et al., 2022; Wang et al., 2026). In our setting, a conformity score that does not separate success from failure leaves both labels in many prediction sets. This suggests coupling uncertainty quantification with the representation itself: the geometry should reflect model behavior strongly enough to yield informative conformity scores, while retaining pretrained structure for meaningful local comparison.

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.

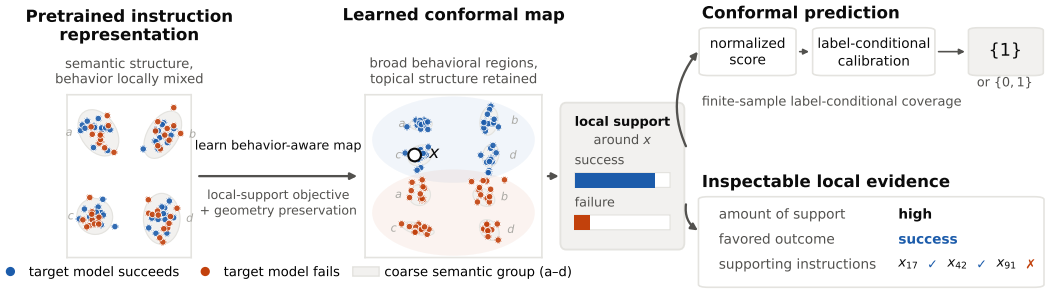


Figure 1: **Conceptual overview.** The pretrained representation organizes instructions semantically, but nearby instructions need not share the target model's outcome (left). The learned map increases behavioral separation while retaining local topical structure (center). For a query, the same local support for success and failure is shaped during training, normalized and calibrated into a conformal prediction set (top right), and read directly as the magnitude, direction, and provenance of local behavioral evidence (bottom right).

Learning conformal maps of instruction space. We introduce *conformal maps*: behavior-aware instruction representations learned to reduce ambiguity in label-conditional conformal prediction. Starting from a pretrained instruction encoder, we fine-tune the geometry so that kernel-smoothed local support better distinguishes observed success from failure. The defined local support measure subsequently defines the conformity score, directly affecting whether a calibrated prediction set is singleton or ambiguous. A geometry-preserving regularizer controls how much of the pretrained semantic organization is retained as behavioral structure is introduced. After model selection, label-conditional split conformal calibration provides finite-sample coverage, while the learned map shapes the informativeness of the resulting sets. The learned local structure also exposes how much evidence surrounds an instruction, which outcome it favors, and which previously evaluated instructions contribute to it. Once the historical outcome bank is available, evaluating a new instruction requires only its text and the reference bank, without target-model logits, activations, or additional generations. Figure 1 summarizes this construction.

We evaluate conformal maps across five QA and reasoning benchmarks and three target models. Across these settings, the learned embedding spaces introduce model-dependent behavioral organization while retaining a controllable amount of the pretrained semantic geometry, improve conformal singleton rate over the original raw representation in most dataset–model pairs, and expose local behavioral evidence that can be inspected through the learned geometry.

Contributions. ▶ We introduce *conformal maps* of instruction space: behavior-aware representations learned around a local support measure that also defines the conformity score, with a geometry-preserving regularizer that controls distortion of the pretrained representation. ▶ We characterize the trade-off between behavioral organization and retention of the pretrained semantic ordering, and relate the learned local-support margin to the ambiguity region of label-conditional conformal prediction. ▶ Across five QA and reasoning benchmarks and three target models, we show that the learned maps improve conformal set efficiency over the original pretrained geometry and expose the amount, direction, and provenance of local behavioral evidence.

2 Related Work

Instance-level behavioral assessment and error discovery. Beyond aggregate benchmark scores, prior work has sought to characterize how model performance varies across individual inputs. Assessor models learn to predict the behavior of a system on a given problem from previously observed evaluations (Hernández-Orallo et al., 2022), with recent work investigating the predictability of language model performance at the instance level (Schellaert et al., 2025; Pacchiardi et al., 2025). A complementary line of work discovers systematic error patterns through automated slice discovery (Eyuboglu et al., 2022). Our work builds on the instance-level assessment perspective, but learns an explicit instruction geometry organized around a target model's observed successes and failures. This geometry supports label-conditional conformal prediction for previously unevaluated instructions while exposing the local behavioral evidence underlying each assessment.

LLM uncertainty estimation and conformal prediction. A large body of work estimates LLM uncertainty using signals available during or after generation, including token probabilities and internal activations (Shrivastava et al., 2023; Heo et al., 2025; Liu et al., 2024), consistency across sampled responses (Yang et al., 2025), and verbalized uncertainty (Tian et al., 2023). Conformal prediction has also been applied to LLMs to provide finite-sample guarantees for prediction sets constructed from model outputs, sampled answers, or other response-derived uncertainty scores (Kumar et al., 2023; Quach et al., 2024; Wang et al., 2024; Su et al., 2024). Related approaches use internal representations for conformal uncertainty estimation and abstention (Xu et al., 2026). Our setting differs in its prediction target and access requirements: we construct prediction sets over whether a fixed target model will succeed or fail on a new instruction, using only its text and a bank of previously evaluated instructions. The target model need not be queried on the new instruction, and its logits, activations, and generated responses are not required at assessment time.

Instruction representations and behavior-aware geometry. Pretrained and instruction-conditioned text representations provide useful geometric structure for comparing instructions and retrieving relevant examples (Su et al., 2023; Li et al., 2024; Shi et al., 2024). However, semantic similarity need not imply similarity in a target model's behavior. We adapt a pretrained instruction representation using the target model's observed correctness, while regularizing changes in pairwise distances to retain pretrained structure. Unlike representations learned primarily for semantic similarity or retrieval, the resulting geometry is organized around local behavioral support. The same support defines the conformity score and exposes the amount, direction, and provenance of the evidence surrounding a new instruction.

3 PRELIMINARIES

Problem setup. Let $\mathcal{X}$ denote the space of instructions and let π be a fixed target model policy. Evaluating π on an instruction $x \in \mathcal{X}$ produces a binary outcome $Y \in \{0, 1\}$, where $Y = 1$ denotes success under the evaluation criterion and $Y = 0$ denotes failure. We observe previously evaluated instruction–outcome pairs and seek to characterize the outcome of a new instruction drawn from the same distribution of queries.

Label-conditional conformal prediction. Conformal prediction (Vovk et al., 2005; Angelopoulos & Bates, 2022) turns any score into prediction sets with finite-sample coverage. Given a conformity score $s_y(x)$ for each candidate outcome y, with larger values indicating greater compatibility, its label-conditional split variant calibrates a threshold q_y from calibration examples with outcome y. The set is

$$C_\alpha(x) = \{y \in \{0, 1\} : s_y(x) \geq q_y\}, \qquad \Pr(y \in C_\alpha(X) \mid Y = y) \geq 1 - \alpha, \tag{1}$$

under exchangeability of calibration and test examples within each outcome class. We use the standard finite-sample order statistic for q_y; its exact form is given in Appendix A.3.

For binary outcomes, the prediction set may be $\emptyset$, $\{0\}$, $\{1\}$, or $\{0, 1\}$. We call $\{0, 1\}$ an *ambiguous* set. Since an empty set necessarily incurs miscoverage, its probability is bounded by the target miscoverage level α under the conformal coverage guarantee. Among the remaining sets, singletons are the only ones that resolve the outcome to a single label. We therefore use the singleton rate $\Pr(|C_\alpha(X)| = 1)$ as our primary measure of conformal efficiency.

For a fixed α, coverage does not determine the singleton rate, which depends on how well the conformity score separates the two outcomes. Our method learns the representation underlying this score to increase the singleton rate, while calibration preserves coverage.

4 METHOD

We learn an instruction geometry from the target model's observed outcomes, anchored to a pretrained representation. A single local support defines the training loss and the conformity score, and geometries are selected by the singleton rate they achieve. The evaluated instruction–outcome pairs are divided into four disjoint subsets: $\mathcal{D}_{\text{train}}$ trains the representation and serves as the reference bank of previously evaluated instructions; $\mathcal{D}_{\text{val}}$ is used only for model selection; $\mathcal{D}_{\text{cal}}$ is used only to compute the final conformal thresholds; and $\mathcal{D}_{\text{test}}$ is held out for evaluation.

Local support in instruction space. Let $E_\theta : \mathcal{X} \to \mathbb{R}^d$ be an instruction encoder initialized from a pretrained encoder E_0, and let $d_\theta(x, x')$ denote cosine distance between two learned embeddings. Given the reference bank $\mathcal{R} = \{(x_i, y_i)\} = \mathcal{D}_{\text{train}}$ and a locality parameter $\eta > 0$, we define the support for outcome y as

$$g_{\theta,\eta}(x, y) = \sum_{i:y_i=y} \exp[-\eta\, d_\theta(x, x_i)], \qquad p_{\theta,\eta}(y \mid x) = \frac{g_{\theta,\eta}(x, y)}{g_{\theta,\eta}(x, 0) + g_{\theta,\eta}(x, 1)}. \tag{2}$$

The parameter η controls the scale at which evaluated instructions contribute evidence. We will use the normalized support $p_{\theta,\eta}(y \mid x)$ as the conformity score $s_y(x)$ of Eq. 1.

For binary outcomes, the relative support can be summarized by the margin $m_{\theta,\eta}(x) = \log g_{\theta,\eta}(x, 1) - \log g_{\theta,\eta}(x, 0)$. Positive margins indicate greater local support for success and negative margins greater support for failure. The total support and the two class-specific components are properties of the map; Section 5 uses them to study the evidence exposed by the learned geometry.

Conformal ambiguity and the learning objective. For binary outcomes, ambiguity occurs when both labels are included. We note that, $p_{\theta,\eta}(1 \mid x) = \sigma(m_{\theta,\eta}(x))$ and $p_{\theta,\eta}(0 \mid x) = \sigma(-m_{\theta,\eta}(x))$, so the success score is a logistic transform of the local-support margin. Let q_0 and q_1 denote the label-conditional conformity thresholds obtained after calibration. Both outcomes are included when $\text{logit}(q_1) \le m_{\theta,\eta}(x) \le \text{logit}(1 - q_0)$, with the interval empty when the lower endpoint exceeds the upper endpoint. Ambiguous prediction sets therefore correspond to a bounded interval of the representation-induced margin.

Direct optimization of the empirical ambiguity rate is not suitable for representation learning because the conformal thresholds are discrete functions of a held-out calibration sample. We therefore optimize a smooth, threshold-free surrogate induced by the same local support that later defines the conformity score, $-\log p_{\theta,\eta}(y \mid x) = \log(1 + \exp(-(2y - 1)m_{\theta,\eta}(x)))$, which is the logistic loss on the signed margin. This objective has the form of the neighborhood log-likelihood used in neighborhood components analysis (NCA; Goldberger et al., 2004). Appendix A.7 characterizes calibrated ambiguity through the class-conditional ranking of the learned margin and establishes sufficient conditions for reducing ambiguity after population and finite-sample recalibration.

Optimizing outcome separation alone can substantially alter the pretrained instruction geometry. We therefore penalize changes in pairwise distance from the initialization. The resulting objective is

$$\mathcal{L}(\theta) = \mathbb{E}_{(x,y),\,\eta}[-\log p_{\theta,\eta}(y \mid x)] + \lambda\, \mathbb{E}_{x,x'}[d_\theta(x, x') - d_0(x, x')]^2, \tag{3}$$

where $\lambda \ge 0$ controls geometry preservation; varying it traces the behavioral–pretrained-geometry trade-off studied in Section 5.1. During training, locality scales are drawn independently from a log-uniform distribution, exposing the representation to neighborhoods at multiple scales; the inference-time scale is selected on $\mathcal{D}_{\text{val}}$.

Model selection and final calibration. Hyperparameters, checkpoints, λ and the inference-time scale η are selected on $\mathcal{D}_{\text{val}}$ alone, by stratified cross-validation: temporary label-conditional thresholds are estimated on the training folds and the singleton rate is measured on the held-out fold. The selected score is then calibrated once on $\mathcal{D}_{\text{cal}}$ (Eq. 1). Details of this procedure are given in Appendix A. Keeping selection and calibration on separate subsets preserves the coverage guarantee for the selected representation.

5 Experiments

Experimental setting. We evaluate conformal maps on five QA and reasoning benchmarks—MMLU (Hendrycks et al., 2021), TriviaQA (Joshi et al., 2017), CommonsenseQA (CQA; Talmor et al., 2019), HellaSwag (Zellers et al., 2019), and MedMCQA (Pal et al., 2022)—against three target LLMs: Qwen3-4B-Thinking, Gemma-4-E4B-it, and GPT-OSS-20B. For each benchmark–model pair, we query the target model once per instruction with greedy decoding (temperature 0) and set $Y = 1$ when its answer is correct.

We initialize all learned representations from Qwen3-Embedding-0.6B (Zhang et al., 2025) and train them with AdamW (Loshchilov & Hutter, 2019). Each dataset is split into $\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{val}}$, $\mathcal{D}_{\text{cal}}$ and

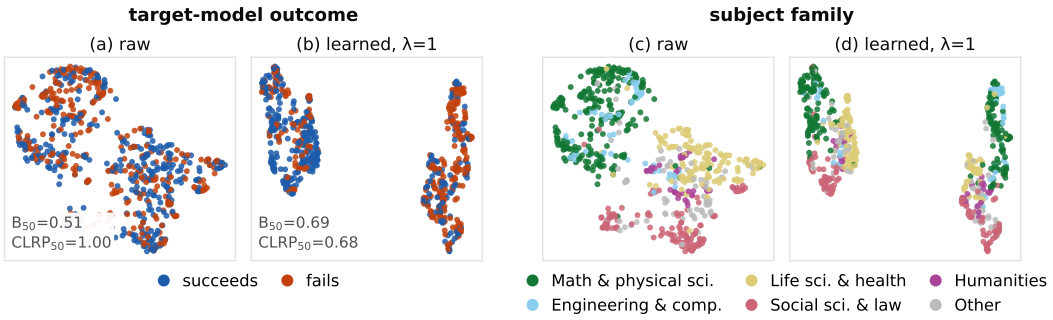


Figure 2: **Held-out instructions before and after learning.** MMLU/Qwen3-4B-Thinking ($\lambda = 1$); all panels show the same test queries, colored by target-model outcome (a, b) and by coarse subject family (c, d). Raw and learned spaces are projected separately with identical UMAP settings.

$\mathcal{D}_{\text{test}}$ (Section 4) in proportions $80/6.7/6.7/6.7$, and $\mathcal{D}_{\text{train}}$ also serves as the reference bank. Each training seed draws its own random partition, so results averaged over seeds reflect variation in both the data split and the optimization.

We organize our experiments around three main questions: **Q1)** What behavioral structure emerges in the learned geometry, and how much pretrained semantic organization is retained? **Q2)** Does this geometry improve the efficiency of label-conditional conformal prediction? **Q3)** What local behavioral evidence does the learned map expose beyond the final prediction set?

5.1 How does the learned geometry organize model behavior?

Before testing downstream conformal prediction, we ask whether training produces the intended geometry. A useful map should organize instructions around target-model behavior while retaining meaningful pretrained structure. We assess both behavioral organization and preservation of the pretrained semantic ordering within the resulting outcome branches.

Visualizing the learned geometry. Figure 2 provides a qualitative view of the learned reorganization on MMLU/Qwen3-4B-Thinking which agrees with our intuition. The raw embedding is organized primarily by subject while outcomes are locally mixed; after fine-tuning, outcome structure becomes more pronounced while coarse subject organization remains visible.

Quantifying behavioral organization and pretrained-geometry retention. Let $\mathcal{N}_K^0(x)$ denote the K nearest training instructions to a held-out query x under the pretrained embedding. Within it, let $\mathcal{N}_K^+(x)$ and $\mathcal{N}_K^-(x)$ denote neighbors whose outcome respectively matches or differs from that of x. We define behavioral ordering as

$$B_K(x) = \Pr\left[d_\theta(x, X^+) < d_\theta(x, X^-) \;\middle|\; X^+ \in \mathcal{N}_K^+(x),\, X^- \in \mathcal{N}_K^-(x)\right], \tag{4}$$

with ties assigned weight $1/2$. The benchmark-level B_K is the mean of its averages over succeeding and over failing queries. Thus $B_K = 0.5$ corresponds to no systematic association between learned distance and outcome agreement, irrespective of the benchmark's base success rate; larger values indicate that learning has reordered the fixed raw neighborhoods around target-model behavior.

Behavioral ordering alone does not ensure that pretrained local structure is retained. As a complementary metric, we therefore define *Conditional Local Rank Preservation* (CLRP), which measures how much of the pretrained local ordering survives within outcome-consistent neighborhoods. For each query x and outcome branch $c \in \{0, 1\}$, we compute the Spearman correlation between pretrained and learned distances to members of that branch within $\mathcal{N}_K^0(x)$,

$$\text{CLRP}_K = \mathbb{E}_{x,c}\left[\text{Spearman}_{x_i \in \mathcal{N}_{K,c}^0(x)}\big(d_0(x, x_i), d_\theta(x, x_i)\big)\right]. \tag{5}$$

Unlike a global distance correlation, CLRP permits success and failure examples to separate; it asks whether the pretrained distance ordering *within* the resulting branches survives. By construction, the raw embedding has $\text{CLRP}_K = 1$.

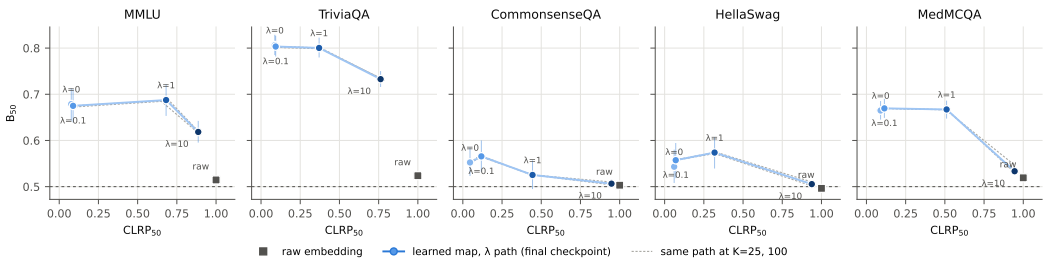


Figure 3: **The regularizer traces a behavioral–semantic trade-off.** B_{50} against CLRP_{50} for Qwen3-4B-Thinking. The raw embedding has $\mathrm{CLRP}_{50} = 1$ and the dashed line marks $B_{50} = 0.5$. Points trace $\lambda \in \{0, 0.1, 1, 10\}$ at matched final checkpoints; intervals are 95% query-bootstrap intervals. Dashed grey paths repeat the same computation at $K = 25$ and $K = 100$.

Geometry regularization controls a behavioral–semantic trade-off. Figure 3 evaluates both properties for varying strengths of the semantic regularization parameter λ. Learning most strongly increases behavioral ordering on MMLU, TriviaQA, and MedMCQA, with smaller changes on CQA and HellaSwag. Increasing λ moves the representation back toward the pretrained semantic geometry: CLRP rises while the added behavioral ordering generally declines. Notably, small to moderate regularization recovers a substantial amount of local semantic structure while preserving high behavioral ordering. We next ask whether the induced restructuring improves conformal prediction.

5.2 Does the learned geometry support efficient conformal prediction?

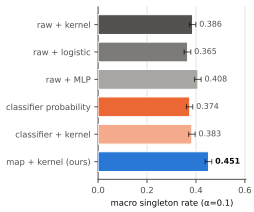
In this section we measure: (i) whether the learned geometry yields more informative sets than the original pretrained embedding and (ii) how its efficiency compares with direct supervised predictors trained on the same success/failure labels.

Evaluation protocol. All configuration selection occurs inside $\mathcal{D}_{\mathrm{val}}$ by stratified 5-fold cross-validation: four folds provide temporary label-conditional thresholds and the held-out fold measures singleton rate. We select at $\alpha = 0.1$, then use $\mathcal{D}_{\mathrm{cal}}$ once to compute the final thresholds and evaluate $\mathcal{D}_{\mathrm{test}}$ once. Fine-tuned checkpoints are selected in the same way.

Baselines. Our primary pipeline applies the kernel-support score of Section 4 in the learned map. We compare against baselines that separate the effects of representation learning, supervised prediction, and neighborhood-based scoring. ▶ *Raw + kernel* uses the same kernel-support conformity score as our method, $p_\eta(y \mid x)$, but computes it in the frozen pretrained embedding; comparison with our learned map therefore isolates the effect of learning the geometry. ▶ *Raw + logistic* and *raw + MLP* instead fit, respectively, a linear logistic readout and a nonlinear MLP on the frozen embedding by minimizing binary cross-entropy. Their predicted probability for candidate label y is used directly as the conformity score. These baselines test whether supervised prediction on the pretrained representation is sufficient without modifying its geometry. ▶ *Classifier probability* and *classifier + kernel* fine-tune the same encoder end to end with a binary classification head and binary cross-entropy. The first uses the predicted label probability as the conformity score; the second discards the head and applies our kernel-support score to the embeddings of the fine-tuned encoder. These baselines provide an end-to-end discriminative alternative that is trained for correctness prediction but does not explicitly expose neighborhood structure. Classifier + kernel and our map apply the same kernel-support

Table 1: **Learning the map improves conformal efficiency.** Relative change in singleton rate at $\alpha = 0.1$, learned vs. original representation under the same kernel-support score, averaged over three training seeds. Shading gives the magnitude, blue for improvement. Second line: learned vs. original rates.

	Qwen3	Gemma	GPT-OSS
MMLU	+20% 0.46 vs 0.39	+1% 0.37 vs 0.36	+41% 0.43 vs 0.31
TriviaQA	+11% 0.70 vs 0.63	+7% 0.70 vs 0.66	+26% 0.42 vs 0.33
CQA	+4% 0.25 vs 0.24	+12% 0.25 vs 0.23	+1% 0.25 vs 0.24
HellaSwag	+13% 0.30 vs 0.27	+6% 0.24 vs 0.23	-10% 0.23 vs 0.25
MedMCQA	+20% 0.51 vs 0.42	+17% 0.46 vs 0.39	+29% 0.54 vs 0.42



	raw + kernel	raw + logistic	raw + MLP	classifier prob.	classifier + kernel	map + kernel
MMLU	0.34	0.37	0.41	**0.48**	**0.48**	0.46
TriviaQA	0.65	0.54	0.71	0.59	0.59	**0.72**
CommonsenseQA	**0.26**	0.25	0.22	0.21	0.24	**0.26**
HellaSwag	0.26	**0.29**	0.25	0.25	0.23	0.27
MedMCQA	0.42	0.38	0.45	0.35	0.38	**0.56**
macro	0.39	0.37	0.41	0.37	0.38	**0.45**

Figure 4: **Conformal efficiency against raw and supervised alternatives.** Singleton rate at $\alpha = 0.1$ on Qwen3-4B-Thinking. Left: five-benchmark macro average with a query-bootstrap 95% interval. Right: per-benchmark rates. See Appendix C.3 for other target models.

score to two embedding spaces learned from the same outcome labels but with different objectives. Their comparison therefore isolates the quality of the learned representation. All learned baselines use the same outcome labels, validation protocol, and label-conditional conformal calibration.

Representation learning improves conformal efficiency. Table 1 compares the learned map with the pretrained embedding under the same kernel-support score. This is a direct test of whether learning the instruction geometry improves conformal efficiency. The map has the higher singleton rate in 14 of the 15 dataset–model pairs. Gains occur across all three target-model families and are largest on MMLU/GPT-OSS (+41%). Averaged over the 15 pairs, the singleton rate rises from 0.36 to 0.41, a 14% relative increase. The gains also generalize beyond the training benchmark: calibrated on a benchmark it was not trained on, a map is no less efficient than the pretrained encoder in 18 of 20 benchmark pairs and significantly more efficient in 9, and a single map trained on all five benchmarks improves significantly on the pretrained encoder (Appendix C.4).

The learned map is competitive with direct supervised prediction. Because the map is itself supervised, we compare it with linear and nonlinear probes on the frozen embedding and an end-to-end classifier that fine-tunes the same encoder. Figure 4 reports this matched comparison for Qwen3-4B-Thinking. The map has the highest five-benchmark macro singleton rate (0.451), versus 0.408 for the MLP and 0.374 for the classifier. Although scalar predictors can be competitive in conformal efficiency on individual tasks, they provide only a prediction score and do not preserve an explicit neighborhood geometry from which local evidence can be inspected. Section 5.3 therefore evaluates complementary properties of conformal maps.

Efficiency improves while label-conditional validity is preserved. We verify coverage separately for $Y = 0$ and $Y = 1$. Across the full grid in Appendix Figure 7, empirical label-conditional coverage closely tracks the target; the median signed deviation from the target has magnitude below 10^{-5}, and the largest shortfall is 0.067, at the scale of finite-test fluctuations. At $\alpha = 0.1$, empty sets are negligible (at most 4×10^{-4}), so the singleton-rate differences in Figure 4 are not driven by systematic undercoverage or empty prediction sets.

Conformal efficiency increases with local support. The aggregate singleton-rate gains of Figure 4 do not show where in the learned geometry informative predictions arise. We stratify test queries by their total local support $S(x) = g(x, 0) + g(x, 1)$. Within each dataset–model pair, queries are divided into equal-frequency support bins, and singleton rates are then aggregated across all fifteen pairs and three training seeds. Figure 5 shows that singleton rate increases with total support for both the learned and pretrained geometries, but substantially more sharply for the learned map. In particular, the highest-support regions of the learned geometry contain a much larger fraction of singleton predictions. Representation learning concentrates informative predictions in regions with strong local support. See Appendix C.9 for detailed results per each model-dataset pair.

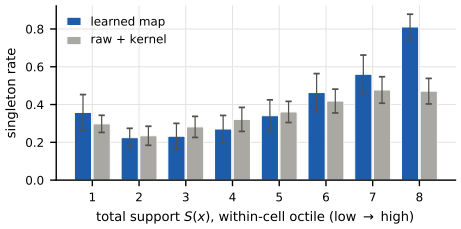


Figure 5: **Informative predictions concentrate where local support is high.** Singleton rate at $\alpha = 0.1$ against the total support $S(x)$, averaged over the fifteen dataset–target-model pairs and three training seeds.

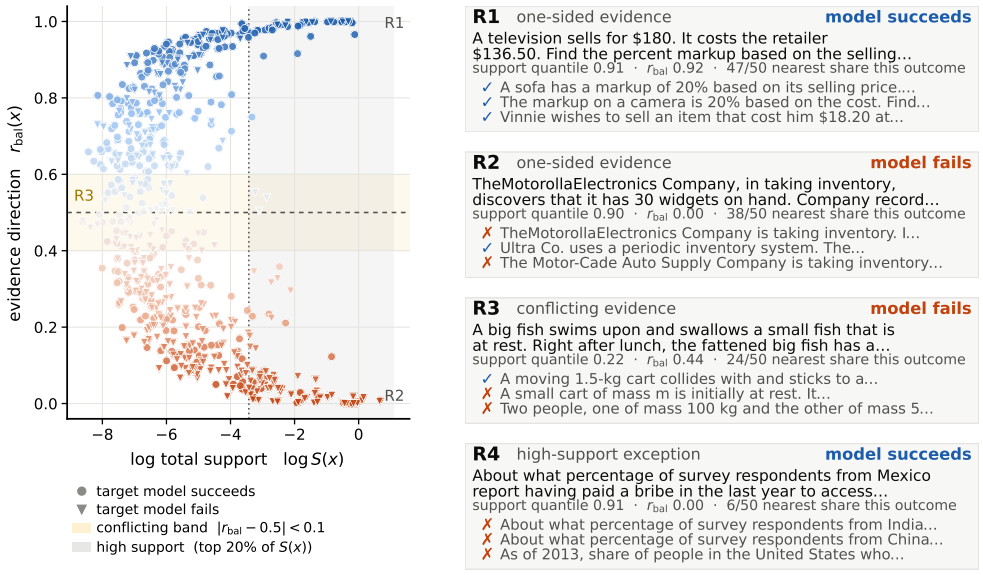


Figure 6: **Local evidence and the instructions that constitute it.** MMLU/Qwen3-4B-Thinking, test held-out queries. Left: total support $S(x)$ against base-rate-corrected evidence direction $r_{\text{bal}}(x)$. Right: one example per regime with its three nearest training instructions. Markers coloured by $r_{\text{bal}}(x)$ (blue towards success, orange towards failure). Regimes R1 and R2 are labeled on the plane and R3 is the shaded band; R4 is the high-support minority of R1 and R2 whose outcome runs against the evidence and by design has no region of its own.

5.3 What does an explicit map expose that a scalar predictor does not?

Beyond prediction-set efficiency, conformal maps are designed to expose the local evidence underlying an instruction-level prediction. For each query, the learned geometry reveals how much evidence is available, which outcome that evidence favors, and which previously evaluated instructions contribute to it. We next examine the inspectability of this local evidence.

The map separates the amount, direction, and provenance of local evidence. The two supports $g(x, 0)$ and $g(x, 1)$ separate evidence amount from direction: $S(x) = g(x, 0) + g(x, 1)$ measures total local support, while the base-rate-corrected fraction $r_{\text{bal}}(x)$ records which outcome the neighborhood favors (0.5 is the benchmark baseline; see Appendix A.4 for exact definitions). Both are computed from the representation and training bank; the held-out outcome is used only afterwards to assess alignment. Figure 6 shows the MMLU/Qwen3-4B-Thinking evidence plane. High-support queries are typically one-sided, whereas conflicting evidence concentrates at lower support. We name four regimes of this plane: dense evidence aligned with success (R1) or with failure (R2), conflicting evidence (R3), and high-support exceptions whose outcome runs against the evidence (R4). The cards show one example drawn from each regime.

The learned map adds behavioral structure while retaining more topical structure. Quantitatively, we characterize the learned neighborhoods using four statistics. *Behavioral coherence* measures how strongly a neighborhood is dominated by one outcome after correcting for the outcome base rate. *Topical retention* is the fraction of neighbors sharing the query's subject label. We call a neighborhood *homogeneous* when its behavioral coherence reaches 0.8, and report the fraction of queries that fall in such regions. Finally, *outcome AUC* measures how well the base-rate-corrected fraction of successful neighbors ranks held-out outcomes: the probability that a succeeding query has a higher such fraction than a failing one. Outcome AUC is the only statistic using the held-out outcome. Exact definitions are given in Appendix B.5.

Table 2 reports each neighborhood statistic for the pretrained encoder, together with its change under the learned map and under the classifier representation. Averaged across five benchmarks and

Table 2: **Learned map against classifier geometry.** Qwen3-4B-Thinking, averaged over three training seeds. Each group gives the neighborhood statistic under the base pretrained encoder (raw) and the change the learned map and the end-to-end classifier representation induce. The more favourable of the two macro differences is bolded. See Table 10 for every target model.

	behavioral coherence			topical retention			homogeneous share			outcome AUC		
	raw	Δmap	Δclf	raw	Δmap	Δclf	raw	Δmap	Δclf	raw	Δmap	Δclf
MMLU	0.62	+0.22	+0.25	0.63	-0.14	-0.28	0.05	+0.62	+0.78	0.678	+0.057	+0.031
TriviaQA	0.66	+0.32	+0.14	–*	–*	–*	0.12	+0.85	+0.45	0.738	+0.106	+0.061
CQA	0.57	+0.31	+0.01	0.12	-0.05	-0.09	0.00	+0.68	+0.00	0.536	+0.009	-0.011
HellaSwag	0.59	+0.10	+0.06	0.47	-0.08	-0.18	0.00	+0.22	+0.14	0.609	+0.015	-0.011
MedMCQA	0.64	+0.02	+0.07	0.83	-0.04	-0.18	0.04	+0.05	+0.15	0.663	+0.018	+0.031
macro	0.62	**+0.20**	+0.11	0.51	**-0.08**	-0.18	0.04	**+0.48**	+0.30	0.645	**+0.041**	+0.020

*Topical retention is undefined for TriviaQA, which carries only one subject label.

three training seeds, behavioral coherence increases from 0.62 in the pretrained representation to 0.81 in the learned map and 0.72 in the classifier representation. At the same time, topical retention decreases from 0.51 in the pretrained representation to 0.43 for the map and 0.33 for the classifier. The map therefore introduces stronger behavioral organization while retaining more of the topical structure present in the pretrained geometry. This difference is also reflected in the prevalence of homogeneous neighborhoods. The pretrained encoder places only 4% of queries in regions with behavioral coherence of at least 0.8; the learned map places 53% and the classifier 35%. Neighborhood composition is also more informative about the held-out outcome in the learned map: outcome AUC rises from 0.645 in the pretrained representation to 0.686 for the map and 0.665 for the classifier. The additional behavioral organization of the learned map is therefore accompanied by less distortion of the pretrained topical structure, and by neighborhoods whose success fraction ranks held-out outcomes more accurately.

The learned evidence structure is reproducible across retraining. The evidence quantities used in our interpretation are not specific to a particular training seed. Across the fifteen dataset–target-model cells and three seed pairs, query rankings by margin and total support remain substantially aligned beyond the agreement attributable to the raw encoder. Coarse evidence regimes likewise agree beyond their raw-geometry baseline. Thus, the higher-level properties on which our interpretation relies are reproducible across independently trained maps. Appendix C.14 reports the full neighborhood and λ-stability analyses.

6 Discussion

Our results show that previously observed successes and failures can be organized into a behavior-aware instruction map for evaluating new queries. The representation is learned to make local support more discriminative, improving the efficiency of the resulting conformal prediction sets, while label-conditional calibration provides finite-sample coverage guarantees. A geometry-preserving regularizer controls how much of the pretrained local semantic organization is retained as behavioral structure is introduced. Scalar supervised predictors remain natural alternatives when only predictive efficiency is required. An explicit map additionally retains a neighborhood structure that identifies how much evidence is available, which outcome it favors, and which prior instructions contribute to it.

Limitations. The conformal guarantee requires exchangeability between calibration and test examples; under distribution shift, realized coverage need not remain at the nominal level. Our main results use random within-benchmark splits. Coverage on a new benchmark can nevertheless be retained by calibrating on labeled examples from it, even when the representation and reference bank were learned elsewhere, and efficiency then largely carries over: maps transferred between benchmarks, and a single map trained on all five, maintain or improve singleton rate relative to the pretrained encoder on most benchmarks (Appendix C.4). The method requires labeled instruction–outcome data: enough to train the map and, on each benchmark to which it is applied, enough for a reference bank and separate validation and calibration splits.

AI USE STATEMENT

In this work, we used generative AI tools as coding and writing assistants. Among tasks with required disclosure, we used them to implement parts of the experimental pipeline, to draft interpretations of results and to validate our theoretical results. Additionally, we used generative AI tools to create and modify figures and tables, to write and edit analysis code and to draft and edit parts of the paper. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Code for training the conformal maps and the baselines, for model selection and conformal calibration, and for obtaining main table and figure results is available at `https://anonymous.4open.science/r/conformal-maps-DC87`, together with the configuration files of all reported runs. The training objective and the conformity score are defined in Section 4, and the experimental setup in Section 5. Appendix B gives the data splits, architectures, and training hyperparameters; Appendix A.2 the cross-validated model-selection procedure, with the candidate grids of every pipeline in Table 7; Appendix A.3 the exact form of the label-conditional calibration; and Appendix B.5 the definitions of all reported metrics. The assumptions and derivation of the theoretical result are given in Appendix A.7. All benchmarks are publicly available, and splits are determined by fixed random seeds. Results are reported with query-level bootstrap intervals and, where indicated, averaged over three training seeds; the variability across seeds is analyzed in Appendix C.14.

REFERENCES

Anastasios N. Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification, 2022. URL https://arxiv.org/abs/2107.07511.

Sabri Eyuboglu, Maya Varma, Khaled Saab, Jean-Benoit Delbrouck, Christopher Lee-Messer, Jared Dunnmon, James Zou, and Christopher Ré. Domino: Discovering systematic errors with cross-modal embeddings, 2022. URL https://arxiv.org/abs/2203.14960.

Jacob Goldberger, Geoffrey E Hinton, Sam Roweis, and Russ R Salakhutdinov. Neighbourhood components analysis. In L. Saul, Y. Weiss, and L. Bottou (eds.), *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004. URL https://proceedings.neurips.cc/paper_files/paper/2004/file/42fe880812925e520249e808937738d2-Paper.pdf.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

Juyeon Heo, Miao Xiong, Christina Heinze-Deml, and Jaya Narain. Do llms estimate uncertainty well in instruction-following?, 2025. URL https://arxiv.org/abs/2410.14582.

José Hernández-Orallo, Wout Schellaert, and Fernando Martínez-Plumed. Training on the test set: Mapping the system-problem space in ai. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 12256–12261, 2022. doi: 10.1609/aaai.v36i11.21487. URL https://doi.org/10.1609/aaai.v36i11.21487.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. triviaqa: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. *arXiv e-prints*, art. arXiv:1705.03551, 2017.

Bhawesh Kumar, Charlie Lu, Gauri Gupta, Anil Palepu, David Bellamy, Ramesh Raskar, and Andrew Beam. Conformal prediction with large language models for multi-choice question answering, 2023. URL https://arxiv.org/abs/2305.18404.

Yiwei Li, Jiayi Shi, Shaoxiong Feng, Peiwen Yuan, Xinglin Wang, Boyuan Pan, Heda Wang, Yao Hu, and Kan Li. Instruction embedding: Latent representations of instructions towards task identification. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=3Yrfx7oYMF.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Alexander Cosgrove, Christopher D Manning, Christopher Re, Diana Acosta-Navas, Drew Arad Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue WANG, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri S. Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Andrew Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic evaluation of language models. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=iO4LZibEqW. Featured Certification, Expert Certification.

Linyu Liu, Yu Pan, Xiaocheng Li, and Guanting Chen. Uncertainty estimation and quantification for llms: A simple supervised approach, 2024. URL https://arxiv.org/abs/2404.15993.

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

Lorenzo Pacchiardi, Konstantinos Voudouris, Ben Slater, Fernando Martínez-Plumed, Jose Hernandez-Orallo, Lexin Zhou, and Wout Schellaert. PredictaBoard: Benchmarking LLM score predictability. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 15245–15266, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.790. URL https://aclanthology.org/2025.findings-acl.790/.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In Gerardo Flores, George H Chen, Tom Pollard, Joyce C Ho, and Tristan Naumann (eds.), *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pp. 248–260. PMLR, 07–08 Apr 2022. URL https://proceedings.mlr.press/v174/pal22a.html.

Michael J. Pekala, Ashley J. Llorens, and I-Jeng Wang. Local distance metric learning for efficient conformal predictors. In *2012 IEEE International Workshop on Machine Learning for Signal Processing*, pp. 1–6, 2012. doi: 10.1109/MLSP.2012.6349813.

Victor Quach, Adam Fisch, Tal Schuster, Adam Yala, Jae Ho Sohn, Tommi S. Jaakkola, and Regina Barzilay. Conformal language modeling, 2024. URL https://arxiv.org/abs/2306.10193.

Ella Rabinovich, Samuel Ackerman, Orna Raz, Eitan Farchi, and Ateret Anaby Tavor. Predicting question-answering performance of large language models through semantic consistency. In Sebastian Gehrmann, Alex Wang, João Sedoc, Elizabeth Clark, Kaustubh Dhole, Khyathi Raghavi Chandu, Enrico Santus, and Hooman Sedghamiz (eds.), *Proceedings of the Third Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pp. 138–154, Singapore, December 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.gem-1.12/.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of NLP models with CheckList. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4902–4912, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.442. URL https://aclanthology.org/2020.acl-main.442/.

Wout Schellaert, Fernando Martínez-Plumed, and José Hernández-Orallo. Analysing the predictability of language model performance. *ACM Trans. Intell. Syst. Technol.*, 16(2), February 2025. ISSN 2157-6904. doi: 10.1145/3706118. URL https://doi.org/10.1145/3706118.

Fobo Shi, Peijun Qing, Dong Yang, Nan Wang, Youbo Lei, Haonan Lu, Xiaodong Lin, and Duantengchuan Li. Prompt space optimizing few-shot reasoning success with large language models, 2024. URL https://arxiv.org/abs/2306.03799.

Vaishnavi Shrivastava, Percy Liang, and Ananya Kumar. Llamas know what gpts don't show: Surrogate models for confidence estimation, 2023. URL https://arxiv.org/abs/2311.08877.

David Stutz, Krishnamurthy Dj Dvijotham, Ali Taylan Cemgil, and Arnaud Doucet. Learning optimal conformal classifiers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=t8O-4LKFVx.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. One embedder, any task: Instruction-finetuned text embeddings, 2023. URL https://arxiv.org/abs/2212.09741.

Jiayuan Su, Jing Luo, Hongwei Wang, and Lu Cheng. Api is enough: Conformal prediction for large language models without logit-access, 2024. URL https://arxiv.org/abs/2403.01216.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D. Manning. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback, 2023. URL https://arxiv.org/abs/2305.14975.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*, volume 29. Springer, 2005.

Tao Wang, Yan Sun, and Edgar Dobriban. Singleton-optimized conformal prediction. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=mO3nEGibLA.

Zhiyuan Wang, Jinhao Duan, Lu Cheng, Yue Zhang, Qingni Wang, Xiaoshuang Shi, Kaidi Xu, Hengtao Shen, and Xiaofeng Zhu. Conu: Conformal uncertainty in large language models with correctness coverage guarantees, 2024. URL https://arxiv.org/abs/2407.00499.

Rui Xu, Yi Chen, Sihong Xie, and Hui Xiong. Geometry-calibrated conformal abstention for language models, 2026. URL https://arxiv.org/abs/2604.27914.

Yongjin Yang, Haneul Yoo, and Hwaran Lee. Maqa: Evaluating uncertainty quantification in llms regarding data uncertainty, 2025. URL https://arxiv.org/abs/2408.06816.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models, 2025.

APPENDIX CONTENTS

A Method details **13**
A.1 Training objective in practice . . . 13
A.2 Model selection . . . 13
A.3 Label-conditional calibration . . . 14
A.4 Local evidence quantities . . . 14
A.5 Computational cost . . . 14
A.6 Stochastic target policies . . . 15
A.7 Calibration-aware ambiguity and the logistic surrogate . . . 15

B Experimental details **18**
B.1 Datasets, labels, and splits . . . 18
B.2 Models and training . . . 18
B.3 Candidate sets and baselines . . . 19
B.4 Evaluation statistics . . . 19
B.5 Geometry and neighborhood metrics . . . 19

C Additional results **20**
C.1 Label-conditional coverage across the coverage grid . . . 20
C.2 Efficiency across the coverage grid . . . 21
C.3 Efficiency against raw and supervised alternatives, other target models . . . 21
C.4 Generalization across benchmarks . . . 22
C.5 Stochastic target policies . . . 24
C.6 Model-selection and candidate-budget sensitivity . . . 25
C.7 Multi-scale training . . . 26
C.8 Representation–score decomposition . . . 26
C.9 Efficiency by local support . . . 27
C.10 Evidence planes across datasets and target models . . . 28
C.11 Locality-scale sensitivity . . . 29
C.12 Full population neighborhood statistics . . . 30
C.13 Raw versus learned neighborhood examples . . . 30
C.14 Stability across seeds . . . 31

A METHOD DETAILS

A.1 TRAINING OBJECTIVE IN PRACTICE

During optimization, the support in Eq. 2 is estimated from pairwise interactions within the optimization minibatch, while the full $\mathcal{D}_{\mathrm{train}}$ bank is used as the reference bank for selection, calibration and inference. The training loss is therefore a stochastic minibatch approximation to the full-bank neighborhood score. Training is multi-scale: each anchor i receives its own $\eta_i \sim \log U[10^{-1}, 10^{2})$, held constant across that anchor's pairwise interactions within the batch. The geometry penalty in Eq. 3 is evaluated on the same minibatch pairs, with the pretrained distances d_0 computed from embeddings of the frozen encoder E_0 that are stored before training.

A.2 MODEL SELECTION

Every candidate configuration (for our method: the regularization strength λ, the training checkpoint and the inference-time scale η) is scored on $\mathcal{D}_{\mathrm{val}}$ alone by stratified K-fold cross-validation at the target level α, with $K = 5$. Within each fold, the remaining $K - 1$ folds supply temporary label-conditional conformal thresholds, computed as in Appendix A.3, and the held-out fold is scored by its singleton rate. A candidate's score is the fold-size-weighted mean singleton rate; ties are broken by the lower cross-validated empty-set rate and then by a fixed candidate order. The fold assignment is drawn once per dataset and shared by all candidates, so that candidates are compared on identical partitions.

Once selection is frozen, $\mathcal{D}_{\mathrm{cal}}$ is used only once to compute the final thresholds and $\mathcal{D}_{\mathrm{test}}$ is evaluated once. Neither $\mathcal{D}_{\mathrm{cal}}$ nor $\mathcal{D}_{\mathrm{test}}$ influences any model, hyperparameter or checkpoint choice; selection

routines are restricted to $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{val}}$, so calibration and test examples are inaccessible during model selection.

A.3 Label-conditional calibration

Let $s_y(x)$ be the score a representation assigns to label y. For each label y we collect the scores of the calibration points carrying that label, $\mathcal{S}_y = \{s_y(x_i) : (x_i, y_i) \in \mathcal{D}_{\text{cal}},\ y_i = y\}$, and take the exact finite-sample order statistic: with $n_y = |\mathcal{S}_y|$ and $\mathcal{S}_y$ sorted increasingly, q_y is its $\lfloor \alpha(n_y+1) \rfloor$-th element. The prediction set is

$$C_\alpha(x) = \{\, y : s_y(x) \geq q_y \,\}, \tag{6}$$

which satisfies $P\big(y \in C_\alpha(X) \mid Y = y\big) \geq 1 - \alpha$ provided the calibration and test points of class y are exchangeable, and conditional on the training and selection procedure that produced s. Nothing requires s_y to be a calibrated probability.

A.4 Local evidence quantities

For a representation E and bandwidth η,

$$\begin{aligned} g(x, c) &= \sum_{i\,:\,y_i = c} \exp\big(-\eta\, d(E(x), E(x_i))\big), \\ S(x) = g(x,0) + g(x,1), &\qquad m(x) = \log g(x,1) - \log g(x,0), \end{aligned} \tag{7}$$

summing over $\mathcal{D}_{\text{train}}$. Both $g(x,1)/S(x)$ and $m(x)$ inherit the bank's base rate, so for interpretation we normalize each class by its size in the bank, $\bar{g}_c = g(x,c)/n_c$, giving

$$r_{\text{bal}}(x) = \frac{\bar{g}_1}{\bar{g}_0 + \bar{g}_1}, \qquad \text{equivalently} \qquad \operatorname{logit} r_{\text{bal}}(x) = m(x) - \log(n_1/n_0), \tag{8}$$

so that $r_{\text{bal}} = 0.5$ means "as the benchmark overall"; it does not mean "equally many successes and failures nearby". The bandwidth used for the evidence analyses of Section 5.3 is fixed as described in Appendix B.5.

A.5 Computational cost

We write $n = |\mathcal{R}|$ for the size of the reference bank, d for the embedding dimension, C_E for the cost of one forward pass of the encoder on one instruction, B for the minibatch size and T for the number of optimization steps. For a transformer encoder with P parameters and instructions of L tokens, $C_E = O(PL)$. In our experiments $d = 1024$, $P \approx 0.6 \times 10^9$, $B \leq 200$ and n ranges from 8,032 (HellaSwag) to 27,203 (MedMCQA).

Training. The regularizer compares learned distances with pretrained ones, so the pretrained embeddings $E_0(x)$ of the training instructions are computed once, at cost $O(n\, C_E)$, and stored. Each optimization step then encodes a minibatch, at cost $O(B\, C_E)$ for the forward and backward passes, and evaluates the support of Eq. 2 and the penalty of Eq. 3 from the $B \times B$ matrices of learned and pretrained pairwise distances, at cost $O(B^2 d)$ with $O(B^2)$ additional memory. Drawing a separate scale η_i for each anchor adds $O(B)$. Training one map therefore costs

$$O\big(n\, C_E + T\,(B\, C_E + B^2 d)\big). \tag{9}$$

Because $C_E \gg Bd$ for the encoders considered here, the encoder passes dominate, and a training step costs essentially the same as one step of fine-tuning the encoder with a per-example loss. The support is computed within the minibatch and never against the full bank, so the cost of a step does not grow with n.

Model selection. Selection requires the embeddings of $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{val}}$ under every candidate representation. With $|\Lambda|$ values of λ and S saved checkpoints per run, encoding costs $O\big(|\Lambda|\, S\, (n + n_{\text{val}})\, C_E\big)$. For each representation the matrix of distances between validation queries and the bank is computed once, at cost $O(n_{\text{val}}\, n\, d)$, and reused for every value in the grid H of kernel scales. Each scale requires the two kernel sums, at cost $O(n_{\text{val}}\, n)$, and K-fold cross-validation, whose thresholds

are exact order statistics obtained by sorting, at cost $O(n_{\mathrm{val}} \log n_{\mathrm{val}})$ per fold. Selection therefore costs

$$O\Big(|\Lambda|\, S\, \big[(n + n_{\mathrm{val}})\, C_E + n_{\mathrm{val}}\, n\, d + |H|\, n_{\mathrm{val}}\, (n + K \log n_{\mathrm{val}})\big]\Big). \tag{10}$$

The raw-embedding baseline is the case $|\Lambda| = S = 1$.

Calibration. Once a configuration is selected, the conformity scores of the calibration sample cost $O\big(n_{\mathrm{cal}}(C_E + nd)\big)$, and the two label-conditional thresholds are order statistics, at cost $O(n_{\mathrm{cal}} \log n_{\mathrm{cal}})$. Calibration is performed once.

Inference. For a new instruction, the prediction set requires one encoder pass, the distances to the n bank instructions, the two kernel sums and two comparisons with the stored thresholds,

$$O(C_E + nd), \tag{11}$$

with $O(nd)$ memory for the stored bank embeddings. No call to the target model is made. The local evidence of Section 5.3 is obtained from the same distances: the total support and the balanced success fraction are by-products of the kernel sums, and the k instructions contributing most to the support are found by partial sorting in $O(n + k \log k)$. Compared with a classifier that maps the embedding to a probability in $O(d)$, the kernel score adds $O(nd)$ operations. For the largest bank, $nd \approx 2.8 \times 10^7$, several orders of magnitude below the $O(PL)$ cost of the encoder pass that both methods share.

Scaling the bank. The inference cost grows linearly with n. For banks much larger than ours, the exact sums can be replaced by sums over the approximate nearest neighbors of the query, since the kernel weights $\exp(-\eta\, d_\theta)$ of distant instructions are negligible at the selected scales. The coverage guarantee is preserved as long as the calibration scores are computed with the same approximation, because split conformal validity holds for any fixed conformity score. For the same reason, adding newly evaluated instructions to the bank changes the score, and the calibration scores must then be recomputed, at cost $O(n_{\mathrm{cal}}\, n\, d)$ given the stored calibration embeddings.

A.6 Stochastic target policies

We briefly discuss how the framework can be extended to settings in which the target policy π is stochastic. A single evaluation of π on an instruction $x \in \mathcal{X}$ then yields a random correctness indicator $R_\pi(x) \in \{0, 1\}$. The relevant quantity becomes the success probability

$$p_\pi(x) := \mathbb{E}[R_\pi(x)] \in [0, 1],$$

where the expectation is over the generation randomness of π. The problem can then be viewed as one of estimating or calibrating a real-valued target rather than a deterministic binary label.

A simple reduction to the binary setting is to introduce a threshold $\tau \in [0, 1]$ and define

$$Y_\tau := \mathbf{1}\,[\,p_\pi(x) > \tau\,]\,.$$

The simplification is practically meaningful, since deployment decisions are typically threshold-based. Because p_π is a fixed function of the instruction, so is Y_τ: the label is again deterministic, the randomness of π having been averaged out. This converts the problem back to binary classification, in which case the framework of the main paper applies directly. An experiment with stochastic target policies is reported in Appendix C.5.

A.7 Calibration-aware ambiguity and the logistic surrogate

We characterize the ambiguity of label-conditional conformal prediction in terms of the class-conditional ranking induced by the learned margin. The characterization applies to each fitted representation with its own calibrated thresholds and extends to expected ambiguity under finite-sample calibration.

Throughout, we condition on a fitted margin function $m(x) = m_{\theta,\eta}(x)$, obtained independently of the final calibration sample. Let

$$F_y(t) = \Pr(m(X) \le t \mid Y = y), \qquad \pi_y = \Pr(Y = y), \quad y \in \{0, 1\}.$$

For the results below, assume that the class-conditional margin distributions are continuous and strictly increasing on a common support. This ensures that the relevant quantiles and inverse ROC curve are well-defined.

A.7.1 AMBIGUITY AFTER POPULATION CALIBRATION

The conformity scores induced by the margin are

$$s_1(x) = \sigma(m(x)), \qquad s_0(x) = \sigma(-m(x)).$$

At population level, label-conditional calibration at miscoverage level α gives the margin thresholds

$$a_\alpha = F_1^{-1}(\alpha), \qquad b_\alpha = F_0^{-1}(1-\alpha).$$

The prediction set is ambiguous precisely when $a_\alpha \le m(X) \le b_\alpha$.

Proposition 1 (calibrated ambiguity). The population probability of an ambiguous prediction set is

$$\begin{aligned} A_\alpha &:= \Pr(C_\alpha(X) = \{0,1\}) \\ &= \pi_1 \big[F_1(b_\alpha) - \alpha\big]_+ + \pi_0 \big[1 - \alpha - F_0(a_\alpha)\big]_+, \end{aligned} \tag{12}$$

where $[z]_+ = \max\{z, 0\}$.

Proof. Conditional on $Y = 1$, the probability of ambiguity is

$$\big[F_1(b_\alpha) - F_1(a_\alpha)\big]_+ = \big[F_1(b_\alpha) - \alpha\big]_+.$$

Similarly, conditional on $Y = 0$, it is

$$\big[F_0(b_\alpha) - F_0(a_\alpha)\big]_+ = \big[1 - \alpha - F_0(a_\alpha)\big]_+.$$

Weighting by the class probabilities gives Eq. equation 12. □

Both thresholds in Eq. equation 12 are quantiles of the margin distribution induced by the fitted representation. The expression therefore characterizes ambiguity after calibrating that representation.

A.7.2 CONNECTION TO CLASS-CONDITIONAL RANKING

Let

$$R(v) = 1 - F_1\big(F_0^{-1}(1-v)\big)$$

be the ROC curve of the margin, expressed as true-positive rate at false-positive rate v. Let $R^{-1}(t)$ be the false-positive rate corresponding to true-positive rate t. equation 12 can then be written as

$$A_\alpha = \pi_1 \big[1 - \alpha - R(\alpha)\big]_+ + \pi_0 \big[R^{-1}(1-\alpha) - \alpha\big]_+. \tag{13}$$

Thus calibrated ambiguity depends on two operating points of the class-conditional ranking: the true-positive rate at false-positive rate α, and the false-positive rate at true-positive rate $1 - \alpha$.

Corollary 1 (ROC dominance). Consider two fitted margins m and $\widetilde{m}$, with ROC curves R and $\widetilde{R}$. If

$$\widetilde{R}(v) \ge R(v) \qquad \text{for all } v \in [0,1],$$

then their separately calibrated ambiguity probabilities satisfy

$$\widetilde{A}_\alpha \le A_\alpha \qquad \text{for every } \alpha \in (0,1).$$

Proof. ROC dominance implies $\widetilde{R}(\alpha) \ge R(\alpha)$ and $\widetilde{R}^{-1}(1-\alpha) \le R^{-1}(1-\alpha)$. Both terms in Eq. equation 13 are therefore no larger for $\widetilde{m}$. □

The corollary gives a sufficient condition for reducing calibrated ambiguity through representation learning. It depends on class-conditional ranking and allows the two representations to have different calibration thresholds.

A.7.3 FINITE-SAMPLE CALIBRATION

We next consider the empirical thresholds used by label-conditional split conformal prediction. Conditional on the fitted score and the calibration class sizes, assume that calibration examples are independent draws from their respective class-conditional distributions.

Let n_y be the number of calibration examples with label y, and define

$$k_y = \lfloor (n_y + 1)\alpha \rfloor.$$

For simplicity, suppose $1 \le k_y \le n_y$. Under the ascending conformity-score convention, the threshold for class y is its k_y-th order statistic. Equivalently, the empirical margin thresholds are

$$\widehat{a} = m_{1,(k_1)}, \qquad \widehat{b} = m_{0,(n_0+1-k_0)},$$

where $m_{y,(j)}$ denotes the j-th smallest calibration margin among examples with label y.

Define the class-conditional probability scales of these thresholds by

$$U = F_1(\widehat{a}), \qquad V = 1 - F_0(\widehat{b}).$$

The probability integral transform and the distributions of order statistics give

$$U \sim \mathrm{Beta}(k_1, n_1 + 1 - k_1), \qquad V \sim \mathrm{Beta}(k_0, n_0 + 1 - k_0). \tag{14}$$

Moreover, U and V are independent conditional on the calibration class sizes. Their joint distribution consequently depends on the calibration order-statistic rule and class sizes, but not on the fitted margin function.

For realized values $U = u$ and $V = v$, the probability of ambiguity on an independent test example is

$$A(u, v; R) = \pi_1 \big[1 - R(v) - u\big]_+ + \pi_0 \big[R^{-1}(1 - u) - v\big]_+. \tag{15}$$

Proposition 2 (finite-calibration comparison). Consider two fixed, fitted margin functions satisfying the regularity conditions above. Suppose they are calibrated using the same class-specific calibration sizes and order-statistic rule, with calibration data independent of the fitted margins. If $\widetilde{R}(v) \ge R(v)$ for all $v \in [0, 1]$, then

$$\begin{aligned} &\mathbb{E}_{D_{\mathrm{cal}}} \left[\Pr\Big(\widetilde{C}_\alpha(X) = \{0, 1\} \mid D_{\mathrm{cal}} \Big) \right] \\ &\qquad \le \mathbb{E}_{D_{\mathrm{cal}}} \left[\Pr(C_\alpha(X) = \{0, 1\} \mid D_{\mathrm{cal}}) \right]. \end{aligned} \tag{16}$$

Proof. For any fixed (u, v), ROC dominance gives

$$1 - \widetilde{R}(v) - u \le 1 - R(v) - u$$

and

$$\widetilde{R}^{-1}(1 - u) - v \le R^{-1}(1 - u) - v.$$

Hence $A(u, v; \widetilde{R}) \le A(u, v; R)$ by Eq. equation 15. Equation equation 14 shows that (U, V) has the same joint distribution for either fitted margin. Integrating the pointwise inequality over this distribution proves Eq. equation 16. □

Proposition 2 compares the complete procedures after finite-sample recalibration. The inequality concerns expected ambiguity over calibration samples; it does not assert an ordering for every realized calibration sample. When $k_y = 0$, the corresponding conformity threshold is $-\infty$, and the same comparison follows using the associated extended margin threshold.

A.7.4 CONNECTION TO THE LOGISTIC TRAINING OBJECTIVE

The preceding results identify class-conditional ranking as a property governing calibrated ambiguity. Logistic loss provides a population surrogate for learning this ranking.

Write

$$\rho(x) = \Pr(Y = 1 \mid X = x).$$

The conditional logistic risk of a margin m is

$$\rho(x)\log(1+e^{-m(x)}) + (1-\rho(x))\log(1+e^{m(x)}).$$

Where $0 < \rho(x) < 1$, its minimizer is

$$m^*(x) = \mathrm{logit}(\rho(x)).$$

By Bayes' rule, this margin differs from the log class-conditional likelihood ratio only by the constant log prior odds:

$$m^*(x) = \log \frac{p(x \mid Y=1)}{p(x \mid Y=0)} + \log \frac{\pi_1}{\pi_0}.$$

The likelihood-ratio ranking attains the optimal ROC curve by the Neyman–Pearson lemma. Consequently, when the population logistic-risk minimizer exists and satisfies the regularity conditions above, it minimizes calibrated ambiguity at every α, including expected ambiguity under the finite-calibration procedure of Proposition 2.

The same ranking interpretation extends to the fixed deterministic policy considered in our experiments. In that setting, $\rho(x) \in \{0, 1\}$, and the unrestricted logistic risk generally has no finite minimizer: its infimum is approached as the correctly signed margin diverges. The limiting margin separates the two outcome classes. Thus, the population connection concerns the ranking targeted by logistic loss, without requiring a finite unrestricted optimum for deterministic labels.

Our learned margins belong to the restricted family induced by kernel support in the instruction geometry. Training also averages the logistic loss over locality scales and includes a geometry-preserving regularizer. Accordingly, the population characterization does not imply that every reduction in empirical training loss improves conformal efficiency. It establishes the relationship between the ranking targeted by the surrogate and ambiguity after recalibration.

Finally, singleton rate also depends on the probability of an empty set:

$$\Pr(|C_\alpha(X)| = 1) = 1 - \Pr(C_\alpha(X) = \{0, 1\}) - \Pr(C_\alpha(X) = \varnothing).$$

The results above characterize the ambiguous component. In the experiments of Section 5.2, empty sets occur infrequently, so changes in ambiguity closely track changes in singleton rate.

B Experimental details

B.1 Datasets, labels, and splits

We evaluate on five benchmarks — MMLU (Hendrycks et al., 2021), TriviaQA (Joshi et al., 2017), CommonsenseQA (CQA; Talmor et al., 2019), HellaSwag (Zellers et al., 2019) and MedMCQA (Pal et al., 2022) — against three target models: Qwen3-4B-Thinking, Gemma-4-E4B-it and GPT-OSS-20B.

Labels are constructed by querying each target model once per instruction under greedy decoding at temperature 0, and setting $Y = 1$ if the produced answer is correct and $Y = 0$ otherwise.

Each benchmark is split into four disjoint parts in the proportions $80/6.7/6.7/6.7$:

- $\mathcal{D}_{\mathrm{train}}$ trains the representation and every supervised baseline, and is the reference bank of previously evaluated instructions for the kernel support score.
- $\mathcal{D}_{\mathrm{val}}$ carries all selection: hyperparameters, training checkpoint, λ and the inference-time kernel scale η.
- $\mathcal{D}_{\mathrm{cal}}$ supplies the final label-conditional conformal thresholds and nothing else.
- $\mathcal{D}_{\mathrm{test}}$ is evaluated once.

B.2 Models and training

The base encoder is Qwen3-Embedding-0.6B (Zhang et al., 2025) with embedding dimension 1024, and all distances are cosine distances between ℓ_2-normalized embeddings. Training uses AdamW (Loshchilov & Hutter, 2019) with learning rate 2×10^{-5}, a linear decay schedule with a warmup of

at most 50 steps, and bf16 precision, on a machine with four NVIDIA A100 GPUs; for each run we use the largest batch that fits in memory, between 70 and 200 samples per GPU, and validate every 5 training steps. For Table 1 and Section 5.3, the checkpoint with the highest cross-validated singleton rate on $\mathcal{D}_{\mathrm{val}}$ (Appendix A.2) is retained during training. For Figure 4 and Section 5.1, maps are trained for a fixed length, their embeddings are retained at ten evenly spaced steps and at the end of training, and the checkpoint is selected on $\mathcal{D}_{\mathrm{val}}$ together with λ and η.

B.3 Candidate sets and baselines

The candidate sets are: for the map, $\lambda \in \{0, 0.1, 1, 10\} \times$ the training checkpoint $\times\ \eta \in \{0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100\}$; for the classifier, the training checkpoint $\times$ the same η grid when scored by kernel support, or the checkpoint alone when scored by its own head; for the logistic probe, the inverse regularization strength $C \in \{0.01, 0.1, 1, 10, 100\}$; and for the MLP probe, width $\in \{128, 256, 512\} \times$ the training epoch. For Table 1, where each run keeps a single checkpoint, the map candidates are $\lambda \in \{0, 0.1, 0.5, 1, 5, 10\} \times$ the same η grid. Representations that were trained by us are retained at ten evenly spaced checkpoints, and the MLP probe is trained for a fixed 60 epochs with the model retained at ten evenly spaced epochs, so that its training length is selected on $\mathcal{D}_{\mathrm{val}}$ in the same way as ours; no internal early-stopping rule is used.

B.4 Evaluation statistics

The primary efficiency statistic is the singleton rate $P(|C_\alpha(X)| = 1)$; we also report the empty-set rate $P(|C_\alpha(X)| = 0)$ and, where relevant, the average set size, because at loose targets an increasing empty-set rate can reduce the singleton rate.

B.5 Geometry and neighborhood metrics

Throughout, $\mathcal{N}_K^0(x)$ is the K-nearest-neighbor set of x in the *raw* embedding against $\mathcal{D}_{\mathrm{train}}$, and is held fixed across all representations so that the metrics describe how learning reorganizes a fixed set of instructions.

Behavioral ordering B_K. Defined in Eq. 4. Within $\mathcal{N}_K^0(x)$, split the neighbors into those sharing x's outcome and those not, and measure how often the learned distance d_θ ranks a same-outcome neighbor closer than a different-outcome one, counting ties as one half. Queries are averaged within each outcome and the two outcome means are then averaged, so that a benchmark's success rate cannot move the statistic: under no association $B_K = 0.5$ whatever the base rate. Queries whose raw neighborhood contains a single outcome have no cross-pairs and are excluded.

Conditional local rank preservation CLRP_K. Defined in Eq. 5. Within each outcome branch of $\mathcal{N}_K^0(x)$ separately, CLRP_K is the Spearman correlation between raw and learned distances to x, so it asks whether the original ordering survives inside the region the outcome occupies. A branch with fewer than five neighbors is undefined; a branch that collapses to a single distance value contributes zero, since a collapsed branch has not preserved the ordering. Branch values are averaged within a query and then balanced over query outcomes as for B_K. The raw embedding attains $\mathrm{CLRP}_K = 1$ by construction.

Effective neighborhood size. With weights $w_i = \exp(-\eta\, d(E(x), E(x_i)))$,

$$\mathrm{ESS}(x) = \frac{\left(\sum_i w_i\right)^2}{\sum_i w_i^2}, \tag{17}$$

the number of bank items effectively contributing to x's evidence. The diagnostic bandwidth is fixed per run by bisection on $\log \eta$ until the median ESS over the validation queries equals the target (50 unless stated otherwise). Median ESS falls very steeply in η, so a grid cannot separate targets of 25, 50 and 100; and per-query ESS is heavy-tailed, so the median is fitted using all validation queries. This diagnostic bandwidth is used only for the analyses in Section 5.3; it is not the η selected for conformal prediction.

Neighborhood composition. Let $\mathcal{N}_K(x)$ be the K nearest bank items of x in the representation being scored, of which k_1 have outcome 1 and $k_0 = K - k_1$ have outcome 0, and let n_0 and n_1 be the bank's class sizes. The base-rate-corrected fraction of the neighborhood and its *behavioral coherence* are

$$f_{\mathrm{bal}}(x) = \frac{k_1/n_1}{k_1/n_1 + k_0/n_0}, \qquad H_{\mathrm{bal}}(x) = \max\big(f_{\mathrm{bal}}(x),\, 1 - f_{\mathrm{bal}}(x)\big), \tag{18}$$

so H_{bal} measures how one-sided the neighborhood is, without reference to x's own outcome. A neighborhood is *homogeneous* when $H_{\mathrm{bal}}(x) \geq 0.8$, and the *homogeneous share* of a cell is the fraction of its held-out queries that are. *Topical retention* is the fraction of the K neighbors carrying the query's own subject label s_x,

$$T_K(x) = \frac{1}{K} \sum_{i \in \mathcal{N}_K(x)} \mathbf{1}[s_i = s_x], \tag{19}$$

a coarse topical proxy rather than a measure of semantics. It is undefined for TriviaQA, which carries a single subject label for every instruction.

Outcome AUC. How well the balanced success fraction f_{bal} of a query's neighborhood ranks held-out outcomes. Writing $\mathcal{Q}_1$ and $\mathcal{Q}_0$ for the held-out queries with outcome 1 and 0,

$$\mathrm{AUC} = \frac{1}{|\mathcal{Q}_1||\mathcal{Q}_0|} \sum_{i \in \mathcal{Q}_1} \sum_{j \in \mathcal{Q}_0} \Big(\mathbf{1}[f_{\mathrm{bal}}(x_i) > f_{\mathrm{bal}}(x_j)] + \tfrac{1}{2}\mathbf{1}[f_{\mathrm{bal}}(x_i) = f_{\mathrm{bal}}(x_j)] \Big), \tag{20}$$

the probability that a succeeding query has a higher f_{bal} than a failing one, with 0.5 chance. It applies no threshold and uses every held-out query, so geometries are scored on identical queries regardless of how one-sided their neighborhoods happen to be.

C Additional results

C.1 Label-conditional coverage across the coverage grid

This subsection supports the Section 5.2 claim that the efficiency comparisons are made at comparable validity.

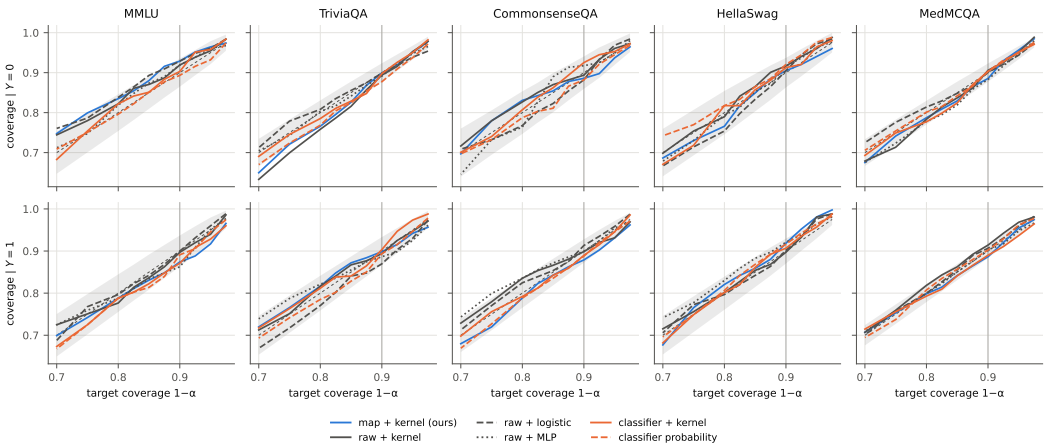


Figure 7: Realized coverage against the target $1-\alpha$ for Qwen3-4B-Thinking, separately for instructions the target model fails ($Y = 0$, top) and succeeds on ($Y = 1$, bottom), since the guarantee is per class and a marginal curve can hide a shortfall on one of them. Each curve is one fixed predictor: the configuration is selected once at $\alpha = 0.1$ (vertical line) and held fixed, and only the conformal threshold is recalibrated along the axis — no model is reselected at any other α. The dashed diagonal is exact calibration and the grey band is ± 2 standard errors around it implied by the per-class test counts.

C.2 Efficiency across the coverage grid

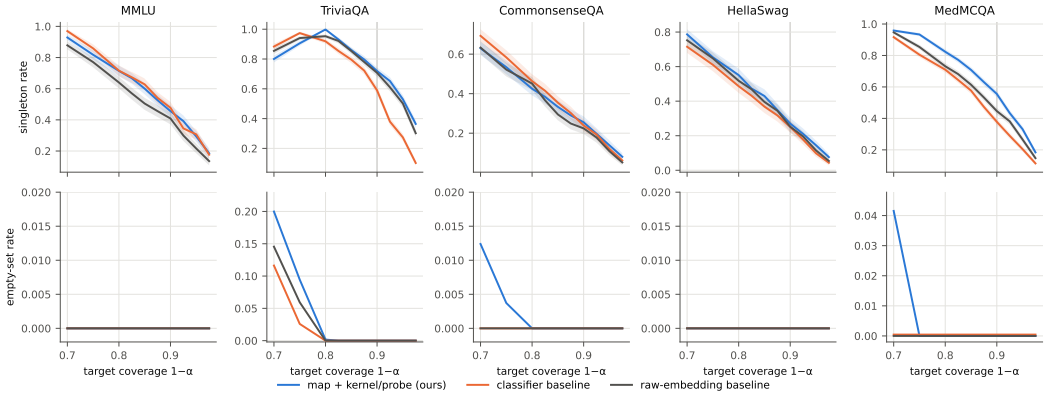


Figure 8: Singleton rate (top) and empty-set rate (bottom) against the coverage target for Qwen3-4B-Thinking, using the same frozen configurations as Figure 7: each pipeline's representation, score, λ, checkpoint and η or probe hyperparameter is chosen once on the validation split at $\alpha = 0.1$ and then held fixed, and only the conformal threshold is recalibrated along the axis. Bands are 95% bootstrap confidence intervals over test queries. The empty-set row explains the one non-monotonicity: on TriviaQA the singleton rate peaks at $\alpha = 0.2$ and then falls because the empty-set rate increases at looser targets. Empty sets are negligible at the headline $\alpha = 0.1$.

C.3 Efficiency against raw and supervised alternatives, other target models

Figure 4 compares the conformal pipelines on Qwen3-4B-Thinking. Figure 9 repeats that comparison for the other two target models, Gemma-4-E4B-it (a) and GPT-OSS-20B (b). The ordering is the same on both: the learned map has the highest macro singleton rate, ahead of the strongest baseline by 0.036 on Gemma-4-E4B-it and 0.044 on GPT-OSS-20B, with a joint bootstrap interval that excludes zero in both cases.



	raw + kernel	raw + logistic	raw + MLP	classifier prob.	classifier + kernel	map + kernel
MMLU	0.30	0.33	0.33	**0.36**	0.32	0.29
TriviaQA	0.62	0.43	0.63	0.53	0.55	**0.69**
CommonsenseQA	0.23	0.21	0.22	**0.26**	0.22	0.25
HellaSwag	0.24	0.26	0.27	0.21	0.27	**0.28**
MedMCQA	0.38	0.36	0.37	0.35	0.31	**0.48**
macro	0.35	0.32	0.36	0.34	0.33	**0.40**

(a) Gemma-4-E4B-it

	raw + kernel	raw + logistic	raw + MLP	classifier prob.	classifier + kernel	map + kernel
MMLU	0.28	0.34	0.40	0.34	0.31	**0.46**
TriviaQA	**0.34**	0.28	0.33	0.20	0.30	0.31
CommonsenseQA	0.22	0.22	0.21	0.17	0.21	**0.29**
HellaSwag	**0.28**	0.27	0.27	0.23	0.21	0.21
MedMCQA	0.45	0.42	0.39	0.31	0.35	**0.55**
macro	0.31	0.31	0.32	0.25	0.28	**0.36**

(b) GPT-OSS-20B

Figure 9: **Conformal efficiency against raw and supervised alternatives, other target models.** Singleton rate at $\alpha = 0.1$ on (a) Gemma-4-E4B-it and (b) GPT-OSS-20B. Left: five-benchmark macro average with a query-bootstrap 95% interval. Right: per-benchmark rates.

C.4 Generalization across benchmarks

The experiments in the main text use random splits of each benchmark, so the learned map, reference bank, and evaluated instructions all come from the same distribution. This section examines two departures from that setting: applying a map trained on one benchmark to another (Appendix C.4.1), and training a single map on all five benchmarks (Appendix C.4.2). In both, the label-conditional coverage guarantee is preserved, because it requires only that the calibration and test instructions be exchangeable; the representation and reference bank used to construct the conformity score need not come from the same distribution.

C.4.1 Transfer from one benchmark to another

We first examine whether a map trained on one benchmark can also be useful on another. We train the map on a source benchmark A and evaluate it on a target benchmark B, using B for the reference bank and all validation, calibration, and test instructions. The coverage guarantee on B remains unchanged, while any distribution shift between A and B may affect the efficiency of the prediction sets.

Protocol. For each ordered pair of distinct benchmarks, we encode the instructions from B using each of the six maps trained on A ($\lambda \in \{0, 0.1, 0.5, 1, 5, 10\}$; Qwen3-4B-Thinking). We select the configuration from the 6×9 combinations of λ and η using the cross-validated criterion of Appendix A.2 on the validation split of B. For the raw encoder, we select η on the same split. Both representations are then calibrated and evaluated on B, so the only difference between the two pipelines is the representation. The diagonal entries follow the same procedure, but use maps trained on B itself. All embeddings are recomputed from the saved map checkpoints. This changes the in-distribution singleton rates relative to Table 1 by at most 0.02.

Results. Table 3 shows the twenty transfer pairs and the five in-distribution maps. In 18 of the 20 transfer pairs, we find no significant decrease in singleton rate relative to the raw encoder, while 9 show a significant increase. The largest transfer gains occur on benchmarks where the in-distribution map is also most useful. On MedMCQA, for example, the in-distribution map improves the singleton rate by 0.14, and all four maps trained on other benchmarks also improve it, with gains ranging from 0.009 to 0.087.

The only two significant losses occur on CQA. Its in-distribution map provides no significant improvement over the raw encoder (+0.022), and its learned neighborhoods offer little improvement

Table 3: **Representation transfer.** Singleton rate at $\alpha = 0.1$ on each target benchmark (rows) for Qwen3-4B-Thinking. *raw* gives the singleton rate of the pretrained encoder. The remaining columns report the change in singleton rate relative to the raw encoder when the map is trained on the benchmark indicated by the column. The shaded diagonal corresponds to maps trained on the target benchmark itself. All pipelines use the target benchmark's reference bank and validation, calibration, and test splits. Bold entries indicate that the paired bootstrap 95% interval over the target's test instructions excludes zero.

		map trained on				
target	raw	MMLU	TriviaQA	CQA	HellaSwag	MedMCQA
MMLU	0.337	**+0.084**	**+0.058**	+0.029	+0.021	-0.009
TriviaQA	0.654	**+0.025**	**+0.101**	+0.004	**+0.050**	**+0.050**
CQA	0.261	-0.006	-0.032	+0.022	**-0.068**	**-0.032**
HellaSwag	0.261	-0.004	-0.001	**+0.040**	-0.003	+0.028
MedMCQA	0.417	**+0.045**	**+0.009**	**+0.059**	**+0.087**	**+0.138**

in outcome ranking over those of the pretrained encoder (Table 2). Across the transferred predictors, worst-class coverage ranges from 0.874 to 0.913 at the target level of 0.90, consistent with the range observed in distribution (Appendix C.1).

C.4.2 Training on the union of benchmarks

We next ask whether a single representation trained across all five benchmarks can replace benchmark-specific maps. We train one map on the union of the five training splits and evaluate it both benchmark by benchmark and on the pooled set of evaluation examples.

Protocol. The training set is the union of the five training splits for Qwen3-4B-Thinking ($66{,}825$ instructions); the validation, calibration, and test instructions of every benchmark are the same as elsewhere in the paper, and none of them occurs in the training set. We train maps with $\lambda \in \{0.1, 1, 5\}$ and retain, for each, the checkpoint with the highest validation criterion on the union of the validation splits. Runs with $\lambda \in \{0.1, 5\}$ are trained for 12 epochs; for $\lambda = 1$ the criterion peaked after four epochs and training was stopped after fourteen. When a benchmark B is evaluated separately, we follow the protocol of Appendix C.4.1: the reference bank and the validation, calibration, and test instructions come from B, and the configuration is selected from the 3×9 combinations of λ and η on the validation split of B. The benchmark-specific map selects from 6×9 combinations and corresponds to the diagonal of Table 3. When the union is evaluated, the reference bank and the validation, calibration, and test instructions are the unions of the corresponding splits, for both the pooled map and the raw encoder. As a reference, we also score every test instruction with the map of its own benchmark, calibrated on that benchmark; unlike the pooled map, this requires the benchmark of each instruction to be known.

Table 4: **A single map for all benchmarks.** Singleton rate at $\alpha = 0.1$ for Qwen3-4B-Thinking. *raw*: pretrained encoder; *benchmark-specific*: map trained on the evaluated benchmark; *pooled*: map trained on the union of the five benchmarks. Upper rows evaluate each benchmark separately. The last row evaluates the union of the benchmarks: the raw encoder and the pooled map are each calibrated once on the union, whereas the benchmark-specific entry scores every instruction with the map of its own benchmark, calibrated on that benchmark. Bold entries indicate that the paired bootstrap 95% interval over the test instructions excludes zero.

	raw	benchmark-specific	pooled	pooled − raw	pooled − specific
MMLU	0.337	0.421	0.450	**+0.112**	+0.029
TriviaQA	0.654	0.755	0.726	**+0.072**	**-0.029**
CQA	0.261	0.284	0.198	**-0.063**	**-0.086**
HellaSwag	0.261	0.258	0.315	**+0.054**	**+0.057**
MedMCQA	0.417	0.554	0.444	**+0.027**	**-0.111**
mixture	0.408	0.507	0.480	**+0.072**	**-0.027**

Results. Table 4 shows the results. Evaluated benchmark by benchmark, the pooled map increases the singleton rate relative to the raw encoder on four of the five benchmarks, with gains ranging from 0.027 on MedMCQA to 0.112 on MMLU, and decreases it on CQA (-0.063). Relative to the benchmark-specific maps, the pooled map is more efficient on HellaSwag ($+0.057$), where the benchmark-specific map does not improve on the raw encoder, and statistically indistinguishable on MMLU. It is less efficient on TriviaQA, CQA, and MedMCQA, with the largest gap on MedMCQA (-0.111).

On the union of the benchmarks, a single pooled predictor increases the singleton rate from 0.408 to 0.480 relative to the raw encoder, and remains 0.027 below the benchmark-specific maps, which in addition use the benchmark of each instruction. Worst-class coverage lies between 0.877 and 0.937 on the individual benchmarks, and the pooled predictor on the union attains a coverage of 0.909 and 0.905 for the two classes, at the target level of 0.90.

C.5 Stochastic target policies

We empirically examine the extension to stochastic target policies described in Appendix A.6. Specifically, we study whether conformal maps can be trained on thresholded success probabilities and whether maps learned for one target policy remain useful when recalibrated for another.

Protocol. We study the direct-answer policy of Qwen3-4B-Thinking. The reasoning phase is omitted, the response is prefixed with "The correct answer is:", and the next token is sampled from the model distribution restricted to the available option letters. Under this policy, $p_\pi(x)$ is available exactly as the normalized next-token probability assigned to the correct option, avoiding sampling error in the outcome definition.

We use the four multiple-choice benchmarks, excluding TriviaQA because it does not provide answer options. Correct option letters are recovered from the source datasets. All remaining data splits follow the main experiments. We use $\tau = 0.8$ as the primary threshold; at this level, the positive class comprises between 31% of test instructions on MMLU and 63% on CQA.

We consider two settings. First, we train maps directly on $Y_{0.8}$ using the main training protocol, with $\lambda \in \{0.1, 1, 5\}$ and validation-selected checkpoints, and compare them with the pretrained encoder. Second, we reuse maps trained on the outcomes of the reasoning policy, together with their reference bank and selected configuration, and change only the calibration target. These maps are recalibrated and evaluated on Y_τ for $\tau \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$; the pretrained encoder is recalibrated in the same way. This second setting tests whether behavioral geometry learned from the deterministic target policy remains useful for thresholded outcomes of another without retraining the representation.

Table 5: **Maps trained on stochastic-policy labels.** Singleton rate at $\alpha = 0.1$ for $Y_{0.8}$ under the direct-answer policy of Qwen3-4B-Thinking, using the pretrained encoder (*raw*) and maps trained directly on $Y_{0.8}$. The final column reports worst-class coverage for the learned map. Bold differences have paired-bootstrap 95% intervals over test instructions that exclude zero.

	raw	map	map − raw	coverage
MMLU	0.471	0.549	**+0.078**	0.920
MedMCQA	0.467	0.632	**+0.165**	0.882
CQA	0.226	0.295	**+0.069**	0.914
HellaSwag	0.263	0.210	**-0.052**	0.915

Results. Maps trained directly on $Y_{0.8}$ improve singleton rate over the pretrained encoder on MMLU ($+0.078$), MedMCQA ($+0.165$), and CQA ($+0.069$), while decreasing it on HellaSwag (-0.052) (Table 5). The gains on the first three benchmarks are comparable in magnitude to those obtained for the reasoning-policy outcomes. HellaSwag also differs in the direct-answer policy itself: the option letters account for a median of 56% of its next-token probability mass, compared with at least 80% on the other benchmarks. It is also the benchmark on which maps trained for the reasoning policy show the weakest improvement over the pretrained encoder.

Table 6: **Calibration-only reuse across target policies.** Change in singleton rate at $\alpha = 0.1$ relative to the pretrained encoder when maps trained on outcomes of the reasoning policy are recalibrated and evaluated on Y_τ from the direct-answer policy. The representation, reference bank, and selected map configuration are kept fixed. Bold entries have paired-bootstrap 95% intervals over test instructions that exclude zero.

	$\tau = 0.5$	$\tau = 0.6$	$\tau = 0.7$	$\tau = 0.8$	$\tau = 0.9$
MMLU	-0.020	-0.041	-0.041	-0.026	-0.020
MedMCQA	**+0.107**	**+0.129**	**+0.144**	**+0.148**	**+0.146**
CQA	**+0.050**	+0.017	**+0.055**	**+0.073**	**+0.062**
HellaSwag	+0.012	+0.024	+0.015	+0.009	-0.013

Calibration-only reuse gives a similar pattern across thresholds (Table 6). Of the twenty benchmark–threshold comparisons, nine show improvements whose paired-bootstrap intervals exclude zero, none show a decrease whose interval excludes zero, and eleven include zero. MedMCQA improves at all five thresholds, with gains from $+0.107$ to $+0.148$, while CQA improves at four of five thresholds. Differences on MMLU and HellaSwag are smaller and their intervals include zero throughout. The results vary relatively little across τ, suggesting that the transferred geometry is not tied to a single probability threshold. Across all settings, worst-class empirical coverage ranges from 0.874 to 0.930 at the nominal level of 0.90.

C.6 Model-selection and candidate-budget sensitivity

Our pipeline selects over $\lambda\times$ checkpoint $\times\eta$, and therefore searches a larger space than the raw baselines, which tune only a bandwidth or probe hyperparameter. Table 7 reports the selection budget for each pipeline in Figure 4. All model selection is performed on the validation split; the test split is used only for the final evaluation of the selected configuration. A larger search space can nevertheless benefit more from validation noise, so we quantify the effect of search budget directly.

Table 7: Selection budget for each pipeline in Figure 4. Candidates are ranked using validation performance only, and the configuration with the highest cross-validated singleton rate is evaluated once on test. The same grids are used for all five benchmarks. Checkpoints are the ten periodic snapshots plus the final model.

pipeline	search grid	candidates
raw + kernel	$\eta \in \{0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100\}$	9
raw + logistic	$C \in \{0.01, 0.1, 1, 10, 100\}$	5
raw + MLP	width $\in \{128, 256, 512\} \times 10$ epoch snapshots	30
classifier probability	11 checkpoints	11
classifier + kernel	11 checkpoints $\times 9$ bandwidths η	99
map + kernel (ours)	$\lambda \in \{0, 0.1, 1, 10\} \times 11$ checkpoints $\times 9$ bandwidths η	396

To assess how much of the gain comes from the larger search space, we first scored all 396 map+kernel configurations using the same cross-validated selection criterion and evaluated each configuration on test once. We then repeatedly sampled candidate subsets of fixed size, stratifying over λ so that every regularization strength was represented at every budget. Within each subset, we selected the configuration with the highest cross-validated singleton rate and recorded its test singleton rate. We used 500 random subsets for each budget.

budget B	8	24	48	100	full (396)
macro singleton rate	0.415	0.423	0.431	0.437	0.451

With a budget of eight candidates, comparable to the nine bandwidths searched by raw + kernel, map + kernel reaches a macro singleton rate of 0.415. This already exceeds the strongest raw baseline, raw + MLP, at 0.408 (Figure 4). These values are directly comparable because they are computed on the same five benchmarks and the same test split. Performance then increases gradually with

search budget, reaching 0.451 when all 396 configurations are available. Thus, the larger search space contributes to the final result, but it does not account for the entire advantage: the method remains ahead of the strongest raw baseline even under a closely matched selection budget.

C.7 MULTI-SCALE TRAINING

During training, each anchor is assigned a kernel scale $\eta_i \sim \log U[10^{-1}, 10^2)$ rather than using a single global value (Appendix A.1). We compare this multi-scale objective with fixed-scale variants trained under the same optimization procedure, with η held constant throughout training. All runs are evaluated using the same selection and calibration protocol: the inference-time scale is chosen from the same validation grid by the same cross-validated singleton-rate criterion on D_{val}, followed by a single label-conditional calibration on D_{cal}. Differences therefore reflect the training-time choice of kernel scale rather than differences in inference-time tuning.

Table 8: **Effect of the training-time kernel scale on singleton rate** at $\alpha = 0.1$. Rows indicate the scale used during training; the inference-time scale is selected on the validation split using the same procedure for every row. Results use $\lambda = 0$ and a single training seed. The sweep excludes CommonsenseQA and therefore covers twelve dataset–model cells.

target model	benchmark	$\eta = 0.1$	$\eta = 1.0$	$\eta = 10.0$	**multi**
Qwen3	MMLU	0.430	0.465	0.410	**0.474**
	TriviaQA	0.647	**0.735**	0.702	0.718
	HellaSwag	0.333	0.354	0.299	**0.363**
	MedMCQA	0.473	0.521	**0.540**	0.516
Gemma	MMLU	0.341	0.368	0.315	**0.369**
	TriviaQA	0.670	0.676	0.667	**0.683**
	HellaSwag	**0.321**	0.272	0.181	0.241
	MedMCQA	0.444	0.469	**0.484**	0.476
GPT-OSS	MMLU	0.389	0.461	0.355	**0.496**
	TriviaQA	0.494	0.411	0.352	**0.523**
	HellaSwag	0.291	**0.327**	0.248	0.275
	MedMCQA	0.513	0.536	**0.539**	0.535

The preferred fixed training scale varies across dataset–model pairs. Among the twelve cells, $\eta = 1$ gives the highest singleton rate in seven, $\eta = 10$ in three, and $\eta = 0.1$ in two. Averaged across cells, multi-scale training achieves a singleton rate of 0.472, compared with 0.466 for the strongest single fixed scale used uniformly across all cells ($\eta = 1$), 0.446 for $\eta = 0.1$, and 0.424 for $\eta = 10$. Empirical coverage is close to the 0.900 target and insensitive to the training scale, ranging from 0.890 to 0.891 across the four rows.

A per-cell oracle provides a stricter comparison. If the fixed training scale were selected separately for each dataset–model pair using knowledge of the test results, the resulting oracle would match or exceed multi-scale training in six of the twelve cells and improve the mean singleton rate by 0.008. Thus, multi-scale training does not uniformly outperform the best cell-specific fixed scale. Its advantage is that it achieves performance comparable to the strongest fixed-scale choices without requiring a training-time scale that transfers consistently across datasets and target models.

C.8 REPRESENTATION–SCORE DECOMPOSITION

The main comparisons vary both the representation and the conformity score. To separate these effects, we cross three representations—the pretrained encoder, the hidden representation of the end-to-end classifier, and the learned map—with five conformity scores. The *kernel* score is the local support of Eq. 2, with η selected on D_{val}. The *k-NN* score uses the base-rate-corrected vote f_{bal} of Eq. 18, with K selected on D_{val}. The *centroid* score uses the signed difference in cosine distance to the two class centroids of the reference bank. Finally, *logistic* and *MLP* fit linear and nonlinear probes on D_{tr} and use the predicted class probability as the conformity score.

All representation–score pairs are evaluated under the same protocol. Training length is matched, score-specific hyperparameters are selected by the same cross-validated singleton-rate criterion on D_{val}, and each score is calibrated label-conditionally once on D_{cal}. Across the fifteen

representation–score combinations, empirical coverage lies between 0.895 and 0.915 at the 0.900 target. We use the classifier trained under the same four-way split as the rest of the experiments; this analysis is restricted to Qwen3-4B-Thinking because a matched classifier representation is available only for that target model.

Table 9: **Representation–score decomposition.** Macro singleton rate at $\alpha = 0.1$ across the five benchmarks for Qwen3-4B-Thinking, at matched training length. Bold indicates the best score within each representation. The final rows report the number of validation candidates considered by each score.

representation	kernel	k-NN	centroid	logistic	MLP
pretrained encoder	0.386	0.296	0.295	0.365	**0.408**
classifier hidden	**0.376**	0.307	0.361	0.362	0.346
learned map (ours)	**0.421**	0.377	0.401	0.394	0.391
candidates: encoder, classifier	9	6	1	5	30
candidates: learned map	36	24	4	20	120

The learned map performs best with the kernel score used by the main method, reaching a macro singleton rate of 0.421. Its advantage is not confined to that score: the learned representation also exceeds the pretrained and classifier representations under k-NN, centroid, and logistic readouts, and outperforms the classifier representation under all five scores. The pretrained representation performs best with an MLP readout, where it slightly exceeds the map (0.408 versus 0.391). These results indicate that both representation and score contribute to conformal efficiency. In particular, the learned map supports several effective readouts, while its strongest performance occurs when paired with the same local-support construction around which the geometry was trained.

C.9 Efficiency by local support

Section 5.2 reports the aggregate relationship between total support and singleton rate. Figure 10 breaks it down by dataset–target-model pair. Queries are binned into within-cell octiles of $S(x)$, each geometry is stratified by its own support.

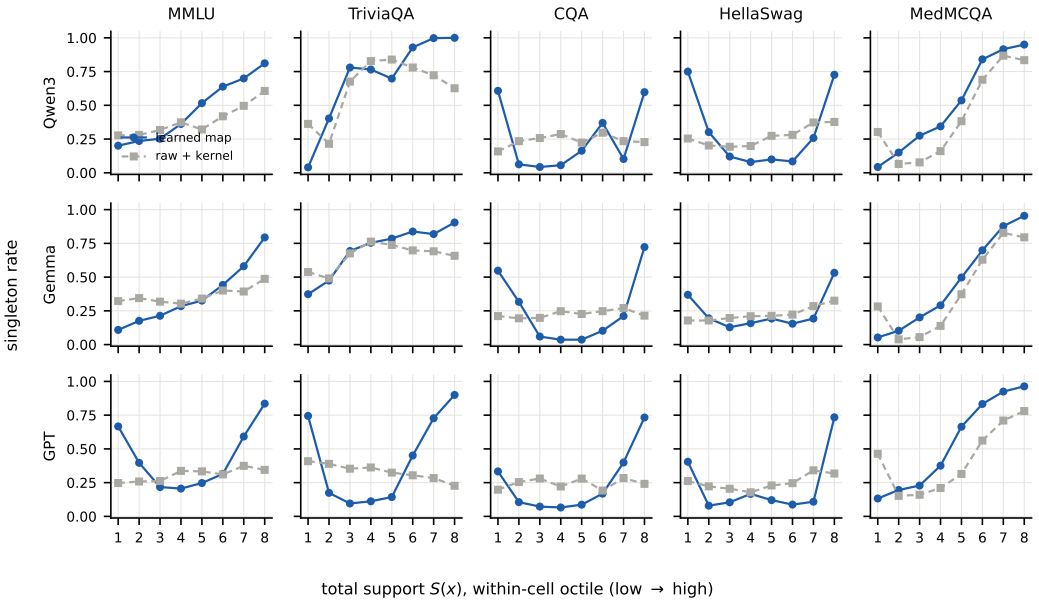


Figure 10: Total support vs. singleton rate for all fifteen dataset–target-model pairs, each averaged over three training seeds. Complementary to the aggregate results presented in Figure 4.

C.10 EVIDENCE PLANES ACROSS DATASETS AND TARGET MODELS

This subsection supports the Section 5.3 claim that the evidence structure of Figure 6 is not particular to one benchmark. Comparing Figure 11 with Figure 12, learned maps frequently develop one-sided high-support regions that are much less apparent in the raw embedding, and the strength of that effect differs substantially across dataset and target-model pairs.

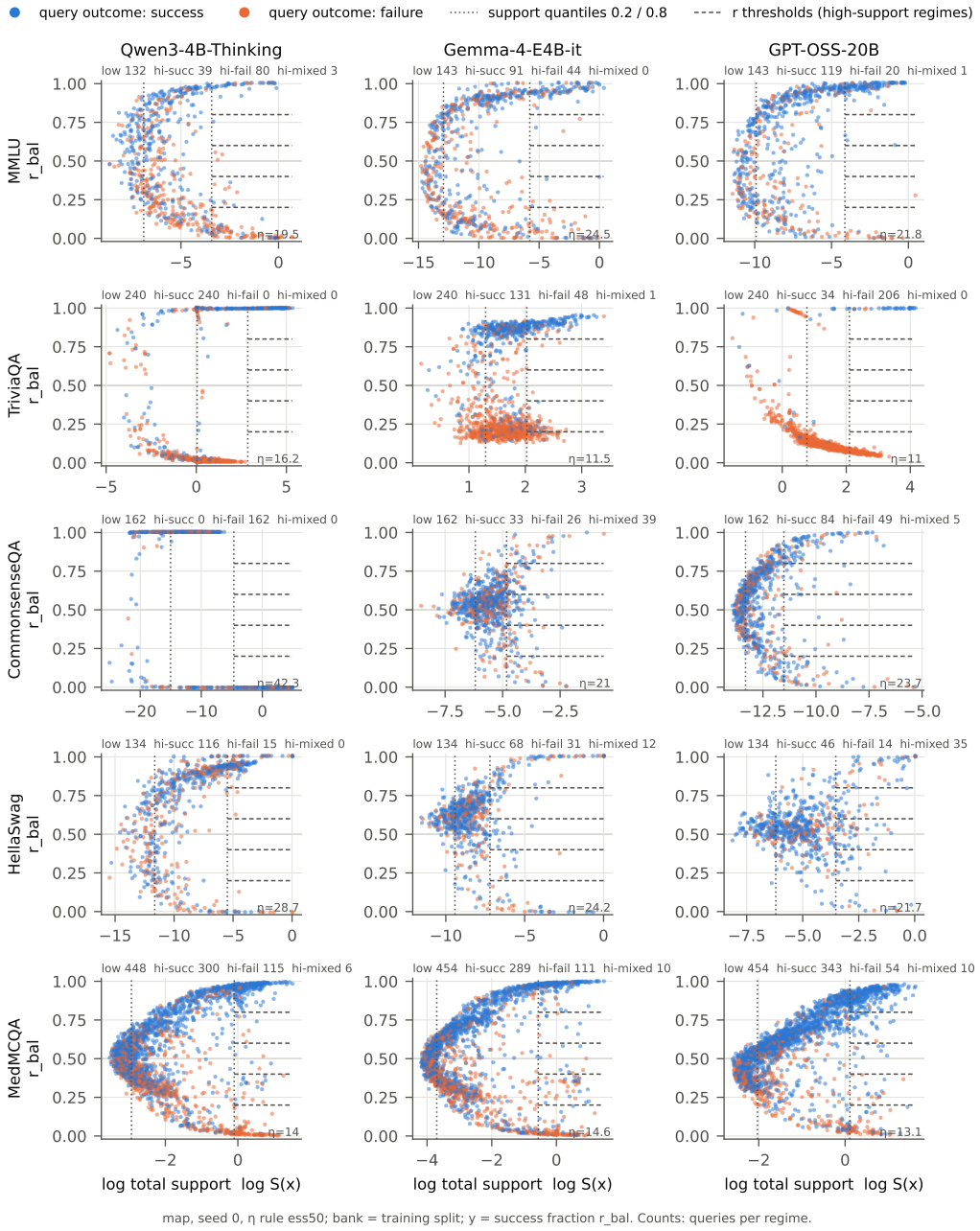


Figure 11: Learned map: held-out queries in the evidence plane for all five benchmarks (rows) and all three target models (columns), diagnostic bandwidth ESS $= 50$. Axes and regime boundaries as in Figure 6.

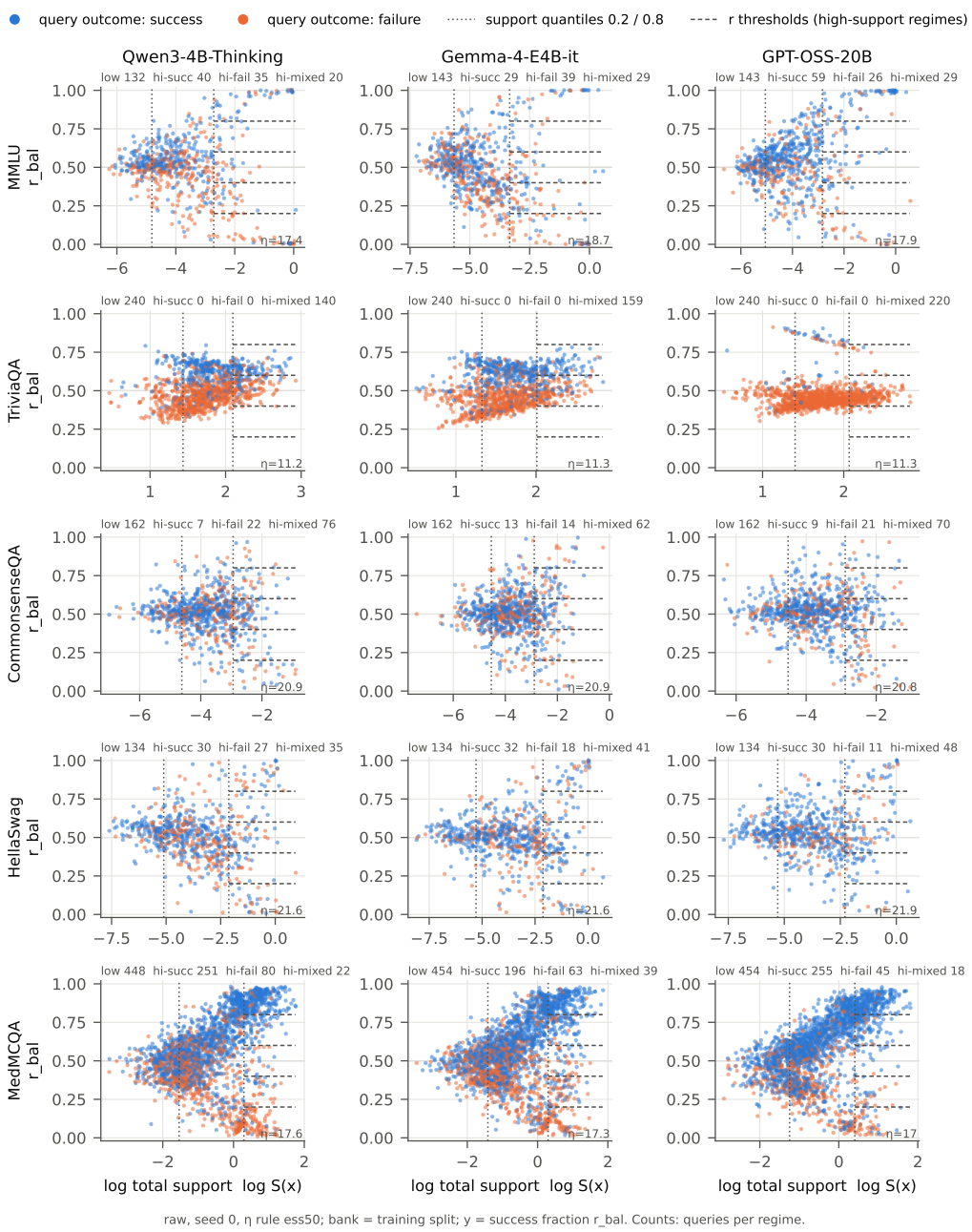


Figure 12: Raw embedding, the same queries and the same bandwidth policy.

C.11 Locality-scale sensitivity

The regime structure is not an artifact of one kernel bandwidth. Figure 13 repeats the planes at a tighter kernel bandwidth and under a bandwidth-free K-nearest-neighbor diagnostic, which replaces the kernel scale with a rank cutoff.

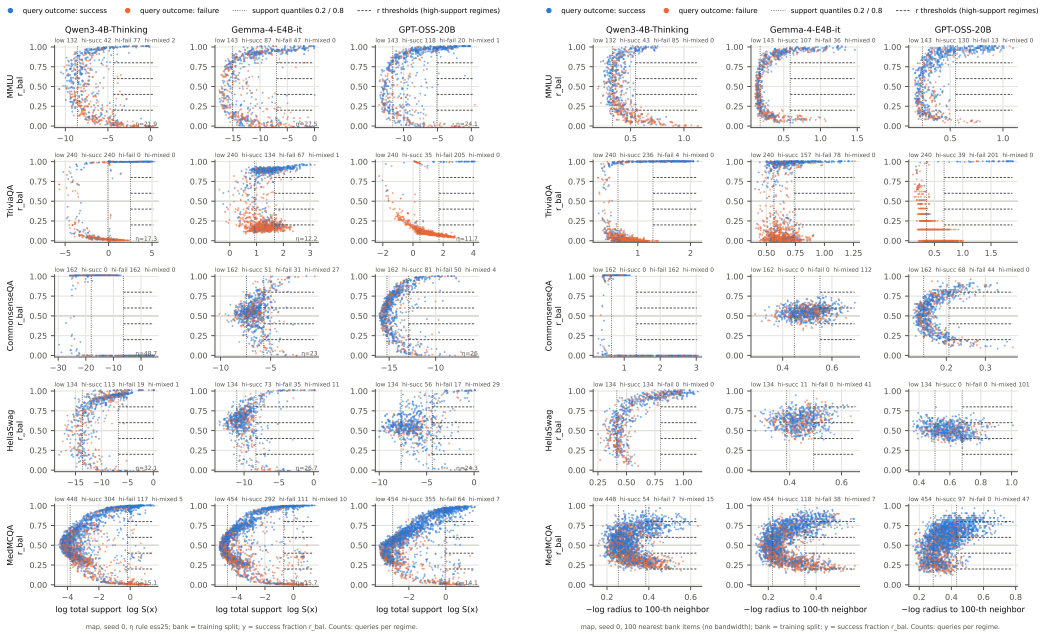


Figure 13: Left: kernel bandwidth at ESS $= 25$. Right: the bandwidth-free diagnostic at $K = 100$. Tighter locality sharpens the one-sided regions and wider locality softens them, but the arrangement of regimes is unchanged.

C.12 Full population neighborhood statistics

This subsection provides the population-level support for Section 5.3 across every dataset and target model, where the main text reports a single target model.

Table 10: **Neighborhood change for every dataset and target model.** $K = 50$, averaged over three training seeds.

	Δ coherence	Δ topical ret.	homog. share	Δ outcome AUC
Qwen3-4B-Thinking / MMLU	+0.224	-0.143	0.67	+0.057
Qwen3-4B-Thinking / TriviaQA	+0.319	–	0.97	+0.106
Qwen3-4B-Thinking / CommonsenseQA	+0.314	-0.054	0.68	+0.009
Qwen3-4B-Thinking / HellaSwag	+0.101	-0.078	0.23	+0.015
Qwen3-4B-Thinking / MedMCQA	+0.023	-0.041	0.09	+0.018
Gemma-4-E4B-it / MMLU	+0.165	-0.123	0.54	+0.032
Gemma-4-E4B-it / TriviaQA	+0.305	–	0.93	+0.107
Gemma-4-E4B-it / CommonsenseQA	+0.129	-0.011	0.28	+0.018
Gemma-4-E4B-it / HellaSwag	+0.092	-0.072	0.13	-0.020
Gemma-4-E4B-it / MedMCQA	+0.055	-0.069	0.19	+0.037
GPT-OSS-20B / MMLU	+0.167	-0.052	0.52	+0.041
GPT-OSS-20B / TriviaQA	+0.203	–	0.71	+0.064
GPT-OSS-20B / CommonsenseQA	+0.042	-0.012	0.07	-0.001
GPT-OSS-20B / HellaSwag	+0.002	-0.002	0.00	-0.003
GPT-OSS-20B / MedMCQA	+0.029	-0.034	0.24	+0.009

Δ columns are the learned map minus the raw encoder; topical retention is undefined for TriviaQA.

C.13 Raw versus learned neighborhood examples

Figure 14 shows the four MMLU/Qwen3-4B-Thinking anchors of Figure 6 in detail. Each row gives the held-out query, whether the target model succeeded on it, and its support quantile and r_{bal}. Beside it are the three nearest training instructions under the raw embedding and under the

learned map, each marked ✓ if the target model succeeded on it and ✗ if it failed, with its cosine distance d to the query. The count above each column is how many of the query's 50 nearest training instructions share the query's outcome. The anchors illustrate what each regime looks like; they do not measure an effect of learning. The regimes are defined on the learned map's own evidence plane, so the anchors' learned neighborhoods lean toward the query's outcome in R1 and R2, are mixed in R3, and lean against it in R4 because of how the anchors were selected. What the rows do show is what those neighborhoods contain. In R1 and R2 the nearest learned neighbors stay on the query's topic, retail markup problems and inventory accounting, so the map reorganizes the pretrained neighborhood around outcome rather than replacing it with unrelated instructions. The R4 query succeeds although all of its 50 nearest learned neighbors fail.

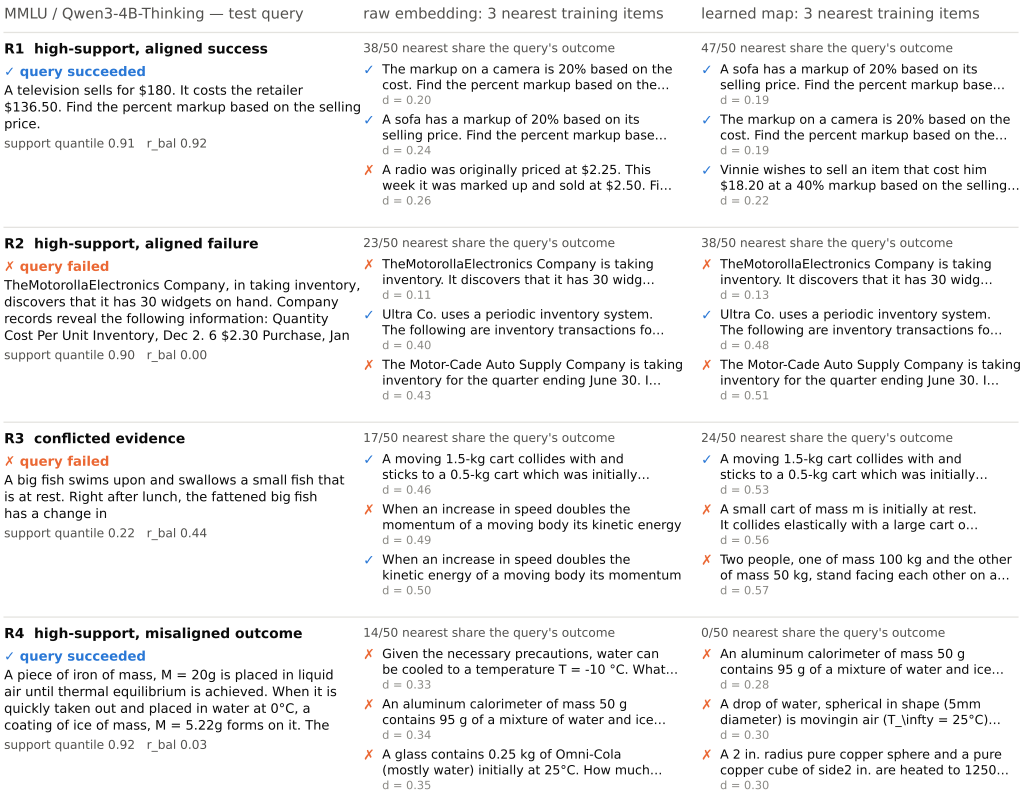


Figure 14: Raw against learned neighborhoods for the four MMLU/Qwen anchors of Figure 6, with the nearest training instructions in each geometry, their outcomes and their distances. Anchors are selected deterministically — within each regime, the query closest to the regime's median support rank.

C.14 Stability across seeds

Section 5.3 interprets the learned map through three properties of its evidence profile: query rankings by margin and support, evidence regimes, and local training neighborhoods. These interpretations are meaningful only if they are reproducible across independent training runs. Cross-seed agreement alone is insufficient, however, because two learned maps may agree simply by preserving the geometry of the same pretrained encoder. We therefore distinguish agreement inherited from the pretrained representation from agreement in the changes introduced by learning.

Setup. Let a and b denote maps trained with different seeds at the same λ. Both are evaluated on the same held-out queries and training bank. For each map $s \in \{a, b\}$, we define a raw counterpart raw_s using the pretrained encoder and the kernel bandwidth selected on seed s's validation split (ESS $= 50$). With three seeds, this gives three map pairs for each dataset, target model, and λ.

Table 11: **What is reproducible across retraining.** Agreement between maps trained with different seeds, averaged over the fifteen dataset × target-model cells, three seed pairs and all λ. The last column gives the range of the per-λ averages of the agreement beyond raw.

	cross-seed agreement	raw-geometry baseline	agreement beyond raw	across λ
margin ranking (Spearman)	0.84	0.80	0.61	[0.48, 0.71]
support ranking (Spearman)	0.75	0.69	0.51	[0.47, 0.56]
evidence regime (Cohen's κ)	0.51	0.26	0.25	[0.23, 0.30]
neighborhoods (Jaccard, $K = 50$)	0.48	0.30	0.24	[0.13, 0.36]

For each property, let $\phi_s(x)$ denote the output of map s and $\phi_{\mathrm{raw}_s}(x)$ the corresponding raw output. We report *cross-seed agreement*, measuring similarity between ϕ_a and ϕ_b; *raw inheritance*, measuring how much of each learned property is explained by its raw counterpart; and *agreement beyond raw*, measuring reproducible structure introduced by learning.

Our null model allows each learned map to depend arbitrarily on its own raw counterpart, but assumes that the two learned transformations otherwise contain no shared structure:

$$P(\phi_a, \phi_b \mid \phi_{\mathrm{raw}_a}, \phi_{\mathrm{raw}_b}) = P(\phi_a \mid \phi_{\mathrm{raw}_a}) P(\phi_b \mid \phi_{\mathrm{raw}_b}). \tag{21}$$

Agreement beyond raw should therefore be near zero when the changes introduced by the two training runs are independent. Because each map is adjusted relative to its own raw counterpart, changes aligned with the raw geometry may still be counted as inherited; the measure is therefore conservative. The estimators differ across properties, so their magnitudes are comparable across λ within a property, but not across properties.

Margin and support rankings. Queries are ranked by margin $m(x)$ and log total support $\log S(x)$. Let $r_s \in \mathbb{R}^n$ denote the learned rank vector and r_{raw_s} its raw counterpart. Cross-seed agreement is the Spearman correlation $\mathrm{corr}(r_a, r_b)$. Raw inheritance is the average correlation of each learned ranking with its own raw ranking, $\frac{1}{2}[\mathrm{corr}(r_a, r_{\mathrm{raw}_a}) + \mathrm{corr}(r_b, r_{\mathrm{raw}_b})]$.

Agreement beyond raw is the correlation between residual rank vectors,

$$\rho_{\perp\mathrm{raw}} = \mathrm{corr}(e_a, e_b), \qquad e_s = r_s - \hat{\alpha}_s \mathbf{1} - \hat{\beta}_s r_{\mathrm{raw}_s}, \tag{22}$$

where $(\hat{\alpha}_s, \hat{\beta}_s)$ are obtained by regressing r_s on r_{raw_s}. Thus $\rho_{\perp\mathrm{raw}}$ measures reproducible cross-seed variation not explained linearly by the raw rankings.

Evidence regimes. Each query is assigned an outcome-free regime from its support quantile $q(x)$ and balanced success fraction $r_{\mathrm{bal}}(x)$: high-support success-dominated ($q \geq 0.8$, $r_{\mathrm{bal}} \geq 0.8$), high-support failure-dominated ($q \geq 0.8$, $r_{\mathrm{bal}} \leq 0.2$), conflicted ($|r_{\mathrm{bal}} - 0.5| < 0.1$), or other.

Cross-seed agreement is Cohen's $\kappa = (p_o - p_e)/(1 - p_e)$. To estimate the agreement expected from the raw regimes alone, let $R_s(x)$ be the raw regime of query x, and let $P_s(\ell \mid r)$ be the fraction of queries with raw regime r that map s assigns to learned regime ℓ. The null agreement is $p_{\mathrm{null}} = n^{-1} \sum_x \sum_\ell P_a(\ell \mid R_a(x)) P_b(\ell \mid R_b(x))$, giving $\kappa_{\mathrm{null}} = (p_{\mathrm{null}} - p_e)/(1 - p_e)$. We report κ_{null} as raw inheritance and $\kappa - \kappa_{\mathrm{null}}$ as agreement beyond raw.

Neighborhoods. Let $N^s(x)$ be the $K = 50$ nearest training instructions to query x under map s, and let $N^{\mathrm{raw}}(x)$ be its K nearest neighbors under the pretrained encoder. Cross-seed agreement is the mean Jaccard index between $N^a(x)$ and $N^b(x)$ over held-out queries.

For query x, let $k_s = |N^s(x) \cap N^{\mathrm{raw}}(x)|$. If the two maps independently retain k_a and k_b elements from the same set of K raw neighbors, their expected retained intersection is $c = k_a k_b / K$. Assuming independently added neighbors do not coincide, the corresponding expected Jaccard agreement is $c/(2K - c)$. Averaged over queries, this gives the raw-inheritance reference.

Finally, let $A^s(x) = N^s(x) \setminus N^{\mathrm{raw}}(x)$ denote the neighbors introduced by learning. Agreement beyond raw is the mean Jaccard index $J(A^a(x), A^b(x))$ over queries for which $A^a(x) \cup A^b(x) \neq \emptyset$. Because the training bank contains at least 6,000 instructions, independently added neighbors rarely coincide, so this quantity is close to zero under the null.

Results. Table 11 reports the three quantities for each property. All properties agree across seeds beyond what the pretrained encoder predicts, but to different degrees. Rankings are the most reproducible: after the raw ranking of each map is partialled out, the margin rankings of two seeds correlate at 0.61 and the support rankings at 0.51. Regime assignments agree at $\kappa = 0.51$, of which 0.26 is expected from the raw regimes, an excess of 0.25. Neighborhood identity is the least reproducible: the introduced neighbors of two seeds overlap with a Jaccard index of 0.24. We therefore interpret the map through its support, the direction of its evidence and its regime structure, and treat the identity of individual neighbors as a lower-level diagnostic.

Figure 15 shows the same quantities as a function of λ. Cross-seed agreement increases steeply with λ, and the raw-geometry baseline increases with it. For neighborhoods, the overlap rises from 0.21 at $\lambda = 0$ to 0.85 at $\lambda = 10$ and the retention null from 0.07 to 0.71. Agreement beyond raw changes much less. It is nearly constant for support rankings and regimes, and rises from 0.48 to about 0.7 for margin rankings and from 0.13 to 0.36 for introduced neighbors. A larger λ therefore makes retrained maps more similar mainly by anchoring them to the pretrained encoder.

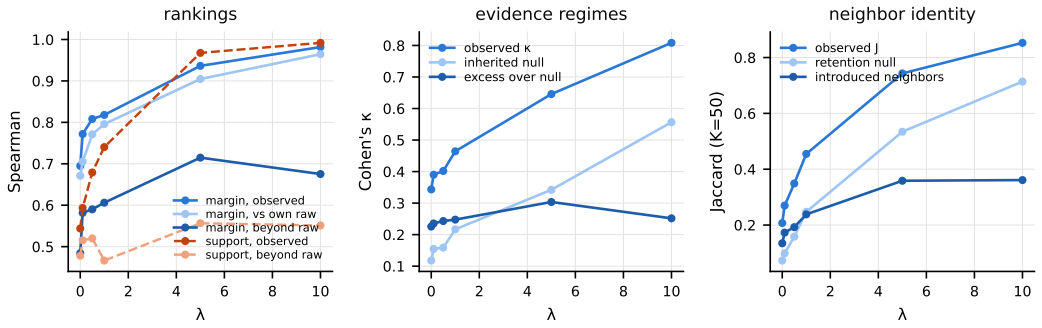


Figure 15: Cross-seed agreement as a function of λ, averaged over the fifteen dataset $\times$ target-model cells and three seed pairs. Left: margin and support rankings. Middle: evidence regimes. Right: neighborhoods ($K = 50$). Quantities are defined in Appendix C.14.