VERI-ERANK: VERILOG DATA SELECTION VIA STABLE STRUCTURAL ADAPTATION RESPONSES

Anonymous authors
Paper under double-blind review

ABSTRACT

Code language models are increasingly adapted for register-transfer-level (RTL) Verilog generation from natural-language specifications. Simulation-validated corpora provide abundant executable supervision, but functional correctness does not make every RTL example equally useful to a given model: an example's adaptation value depends on the model's current capabilities. Data selection should therefore identify examples that most effectively advance the model's specification-to-design ability. Existing methods, whether loss- or gradient-based, score candidates from signals aggregated over the full sequence, without explicitly assessing whether learning from an example strengthens the model's representations of the specification-relevant RTL structures. We introduce Veri-eRank, which ranks RTL examples by changes in specification-relevant structural representations from a base model to a shared reference-adapted model. Veri-eRank represents each RTL program using AST-derived structural spans, including declarations, assignments, event controls, and control flow, and weights each span by its relevance to the specification. Specifically, Veri-eRank measures the change in spectral entropy of these structural representations before and after adaptation, favoring examples with a reduction after adaptation. Theoretically, our analysis shows that preserving structural directions while reducing isotropic variation lowers spectral entropy after adaptation, motivating our spectral-response score. To avoid scores driven by a particular local observation, Veri-eRank evaluates resampled structural views and favors responses that are both strong and consistent. When selecting 20% of the available training data, Veri-eRank attains the highest mean Pass@1 among the six selection baselines across two models and three RTL benchmarks. On Qwen2.5-Coder and Seed-Coder, Veri-eRank improves five-seed macro-average Pass@1 over SPICE by 5.42 and 2.24 percentage points and over Full SFT by 1.05 and 1.98 points, reaching 50.83 ± 1.14 and 53.06 ± 0.57, respectively.

1 INTRODUCTION

Data selection is increasingly important for adapting pretrained language models as instruction corpora grow through human annotation, model-generated data, and task-specific collection (Wei et al., 2022; Ouyang et al., 2022; Wang et al., 2023). However, the value of a training example is not fixed. For a particular starting model, an example may teach a missing capability, reinforce a partially learned one, or offer little additional learning because the relevant pattern has already been acquired. Data selection should therefore identify the examples that are most useful for a model's next stage of learning (Zhou et al., 2023; Liu et al., 2024b; Chen et al., 2024).

This problem is particularly important for code generation. Generated code must implement the behavior requested by a specification, and its correctness depends on how multiple program components work together rather than on the plausibility of individual tokens alone. We study register-transfer-level (RTL) Verilog generation, where a model translates a natural-language specification into a functionally correct hardware design. RTL programs specify cycle-level behavior through structures such as declarations, assignments, event controls, and control flow. A mistake in any of these structures can invalidate the intended design. At the same time, simulation enables the construction of abundant corpora of executable, validated RTL examples (Wei et al., 2025). Passing simulation establishes that an example is usable supervision, but does not establish that it will teach

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.

a given model something new. RTL data selection should therefore identify examples that improve a model's ability to represent and generate the structures required by a specification.

Existing methods use prediction error (Paul et al., 2021), instruction-following difficulty (Li et al., 2024), or gradient-based signals (Xia et al., 2024; Chang et al., 2026) to estimate sample value. Because these signals aggregate over a full sequence, they do not directly assess whether adaptation strengthens the specification-relevant RTL structures that implement the desired behavior. This leaves an RTL-specific measurement gap: selection should account for how these structural representations change during adaptation.

We address this gap with Veri-eRank, a model-relative data selection method for RTL Verilog generation. Veri-eRank represents each program using syntax-derived spans, including declarations, assignments, event controls, and control flow, and weights the spans according to their relevance to the specification. It compares the representations of the same weighted structural observations in a base model and a shared reference-adapted model. Figure 1 illustrates these observations in a selected RTL example. The highlighted clock-edge indicator, reset/enable expressions, and assignments ground the adaptation measurement in the logic that governs when and how the register state is updated. Representation spectra provide a compact description of how these structural representations are organized (Roy & Vetterli, 2007; Garrido et al., 2023). Veri-eRank uses the change in spectral entropy to measure each example's structural adaptation response, favoring examples whose representations become more concentrated after adaptation.

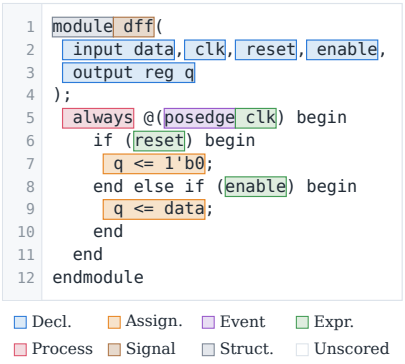


Figure 1: Selected RTL example with scored regions shaded by role.

Our theory establishes the score direction: when adaptation suppresses isotropic variation unrelated to the structural signal, representations become more concentrated and spectral entropy decreases. Veri-eRank evaluates resampled structural views, so high scores reflect responses that are strong and consistent rather than favorable local observations. We derive a finite-view robustness bound for this stability-aware ranking.

Our contributions are:

- **Model-relative RTL data selection.** We introduce Veri-eRank, which ranks Verilog supervision through adaptation-induced changes in representations of specification-relevant RTL structures.
- **Structure-aware spectral scoring with theoretical support.** We measure spectral response using matched weighted structural observations, establish the positive direction of the score, and derive a finite-view robustness bound for stability-aware ranking.
- **Consistent gains across models and benchmarks.** When selecting 20% of the available training data, Veri-eRank achieves the highest macro-average Pass@1 among the six selection baselines across two models and three RTL benchmarks. It improves macro-average Pass@1 over SPICE by 5.42 and 2.24 percentage points, and over Full SFT by 1.05 and 1.98 points, for Qwen2.5-Coder and Seed-Coder, respectively.

2 VERI-ERANK

Veri-eRank ranks each candidate through four steps. It extracts syntax-derived RTL spans and weights them by their relevance to the instruction; compares the same weighted spans in the base and shared reference-adapted models; measures the resulting change in spectral entropy; and retains examples whose responses remain strong across resampled structural views. These steps respectively determine what is observed, make the checkpoint comparison attributable to adaptation, and prevent rankings from depending on a favorable local observation. Figure 2 summarizes the procedure.

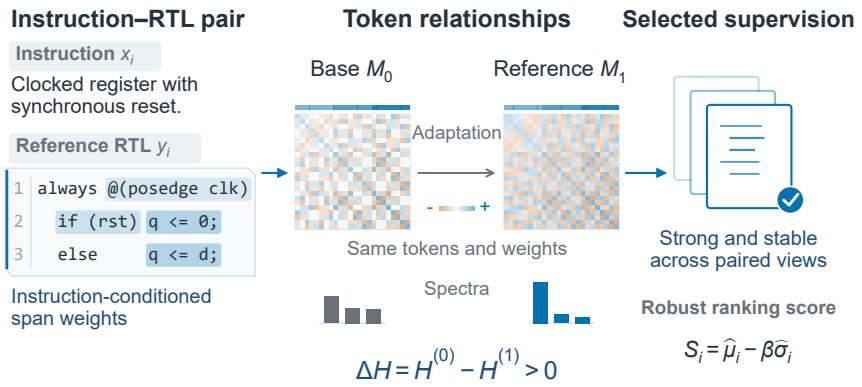


Figure 2: Overview of Veri-eRank. AST-derived RTL spans receive instruction-conditioned weights. The base and reference-adapted models are compared on matched weighted tokens to define spectral response. Resampled views preserve structural-role mass.

2.1 Preliminaries

Let $\mathcal{D} = \{d_i = (x_i, y_i)\}_{i=1}^{N}$ be a pool of instruction–RTL pairs, where x_i is a natural-language specification and y_i is its Verilog implementation. Let M_0 denote a pretrained code model. Given a retention fraction $\gamma \in (0, 1]$, Veri-eRank assigns each candidate d_i a score S_i and selects the top $K_\gamma = \lceil \gamma N \rceil$ examples. We evaluate the selected subset by the functional correctness achieved after applying a common fine-tuning procedure.

To score candidates, we construct one shared reference-adapted model,

$$M_1 = \mathcal{A}_{\text{ref}}(M_0, \mathcal{D}).$$

Veri-eRank compares the representations of each candidate under M_0 and M_1 to measure its response along this common adaptation trajectory. The reference adaptation is used only for selection. After ranking, final SFT starts from a fresh copy of M_0 with newly initialized adapter and optimizer states, so the selected subset is evaluated independently of the reference adaptation.

2.2 Instruction-conditioned structural observations

We first define the structural observations on which responses are measured. An AST parser partitions each RTL program into non-overlapping code spans $\mathcal{S}$, each assigned to a structural role in $\mathcal{R}$, such as declaration, assignment, event, or control. Tokenizer offsets identify the code tokens $T(s)$ owned by each span. These spans specify the structural units, and their token representations supply the geometry measured below (Appendix A.1).

We next estimate which structural observations are most informed by the instruction. M_0 evaluates the reference code y_i with its specification x_i (c^+) and without it (c^-). We use the instruction-induced reduction in uncertainty over structural roles as a proxy for a span's relevance to the specification. A lexical projection maps renormalized top-K_p next-token probabilities to structural-role probabilities $P_t^c(r)$. Let $J^c(s)$ contain the evaluable positions aligned to span s. The projected structural-role entropy and its span average are

$$E_t^c = -\sum_{r \in \mathcal{R}} P_t^c(r) \log P_t^c(r), \qquad \overline{E}^c(s) = \frac{1}{|J^c(s)|} \sum_{t \in J^c(s)} E_t^c. \tag{1}$$

The instruction gain

$$g(s) = [\overline{E}^-(s) - \overline{E}^+(s)]_+ \tag{2}$$

is positive when the instruction makes the predicted structural-role distribution more concentrated within a span. We convert this gain into a bounded positive observation weight. For $0 < a_{\min} < a_{\max}$, an increasing affine map sends the smallest and largest within-example gains, $g_{\min}$ and $g_{\max}$,

to these endpoints. When $g_{\max} > g_{\min}$, this gives

$$a(s) = a_{\min} + (a_{\max} - a_{\min})\frac{g(s) - g_{\min}}{g_{\max} - g_{\min}}, \qquad a_t = a(s) \quad (t \in T(s)). \tag{3}$$

The positive floor retains lower-gain regions, while the bounded range controls their relative influence. We compute the weights once with M_0 and fix them across checkpoints. Equal-gain and alignment conventions appear in Appendix A.2; all instantiated parameters appear in Table 3.

2.3 Paired spectral response

We measure response by comparing the spectral entropy of the same weighted structural observations before and after reference adaptation. Both checkpoints process the unchanged instruction–code sequence. Let $h_t^{(m)}$ be the last-layer state at code token t in model M_m, and let $\mathcal{T}$ be the shared eligible token support. Weighted centering and unit normalization give

$$w_t = \frac{a_t}{\sum_{u\in\mathcal{T}} a_u}, \qquad \mu^{(m)} = \sum_{t\in\mathcal{T}} w_t h_t^{(m)}, \qquad z_t^{(m)} = \frac{h_t^{(m)} - \mu^{(m)}}{\|h_t^{(m)} - \mu^{(m)}\|_2}. \tag{4}$$

Each checkpoint's full-support center stays fixed across views. For a view $V \subseteq \mathcal{T}$, stack its unit directions into $Z^{(m)}(V)$ and let $W(V) = \mathrm{diag}(w_t^V)$ contain normalized positive view weights. The token-space Gram matrix is

$$G^{(m)}(V) = W(V)^{1/2} Z^{(m)}(V) Z^{(m)}(V)^\top W(V)^{1/2}. \tag{5}$$

Its nonzero spectrum equals that of the directional second moment $C^{(m)}(V) = Z^{(m)}(V)^\top W(V) Z^{(m)}(V)$, whose trace is one. Identical support and weights make the comparison depend on the checkpoints' representation geometry rather than on a changed observation distribution.

We summarize the spectrum with entropy and effective rank (Roy & Vetterli, 2007; Sanchez Giraldo et al., 2015).

Definition 1 (Paired spectral response) *Let $p_j^{(m)}(V)$ be the normalized spectrum, with numerical processing specified in Appendix A.3. Define*

$$H^{(m)}(V) = -\sum_j p_j^{(m)}(V) \log p_j^{(m)}(V), \qquad \mathrm{eRank}^{(m)}(V) = \exp H^{(m)}(V), \tag{6}$$

and the response of a shared view by

$$\Delta H(V) = H^{(0)}(V) - H^{(1)}(V) = \log \frac{\mathrm{eRank}^{(0)}(V)}{\mathrm{eRank}^{(1)}(V)}. \tag{7}$$

The log-ratio builds on the paired spectral perspective of Wei et al. (2024). It measures relative concentration after adaptation and is invariant to orthogonal changes of feature coordinates.

2.4 Paired structural views and robust ranking

A response measured on one local set of spans may be favorable by chance. To test whether it is a property of the example, we construct paired views by sampling whole spans within each structural role from cached states of the intact program. Each view retains $\max(1, \lfloor \alpha |\mathcal{S}_r| \rfloor)$ spans without replacement, where $\alpha \in (0, 1]$. To vary local observations without changing structural composition, we preserve each role's full-support mass:

$$\pi_r = \sum_{t\in\mathcal{T}_r} w_t, \qquad w_t^{(b)} = \pi_r \frac{w_t}{\sum_{u\in V_r^{(b)}} w_u} \quad (t \in V_r^{(b)}), \qquad V^{(b)} = \bigcup_r V_r^{(b)}. \tag{8}$$

Both checkpoints use the same view and weights.

Lemma 1 (Role-composition control) *The weights in Equation* (8) *satisfy* $\sum_{t\in V_r^{(b)}} w_t^{(b)} = \pi_r$. *Hence* $C^{(m)}(V^{(b)}) = \sum_r \pi_r C_r^{(m)}(V^{(b)})$, *where* $C_r^{(m)}$ *is the role-conditional directional moment. The role coefficients remain fixed across views and checkpoints.*

The identity follows by summing the view weights. Appendix A.8 shows how this eliminates the role-composition term in a general view perturbation.

For $B \geq 2$ valid paired views, we rank by sustained response:

$$S_i = \widehat{\mu}_i - \beta\widehat{\sigma}_i, \qquad \widehat{\mu}_i = \frac{1}{B}\sum_{b=1}^{B}\Delta H_i^{(b)}, \qquad \widehat{\sigma}_i^2 = \frac{1}{B-1}\sum_{b=1}^{B}(\Delta H_i^{(b)} - \widehat{\mu}_i)^2, \tag{9}$$

where $\beta \geq 0$ controls sensitivity to view variation. This rule has a finite-view robustness interpretation related to variance-based robust optimization (Namkoong & Duchi, 2017; Duchi & Namkoong, 2021).

Proposition 1 (Finite-view robustness) *For* $\rho \geq 0$ *and observed responses* $x_b = \Delta H_i^{(b)}$, *let* $\mathcal{Q}_\rho$ *contain probabilities* $q_b \geq 0$ *satisfying* $\sum_b q_b = 1$ *and* $B\sum_b (q_b - 1/B)^2 \leq \rho$. *Then*

$$\inf_{q\in\mathcal{Q}_\rho} \sum_b q_b x_b \geq \widehat{\mu}_i - \sqrt{\rho(B-1)/B}\,\widehat{\sigma}_i. \tag{10}$$

For $\rho = B\beta^2/(B-1)$, *the right-hand side equals* S_i.

Thus the score lower-bounds the response under bounded shifts toward less favorable observed views. Appendix A.9 proves the result and gives its equality condition. We select the top K_γ candidate scores, breaking ties by stable sample identifiers. Algorithm 1 specifies support and valid-view rules. Cached states make additional views a spectral-computation cost, and the resulting ranking serves multiple selection fractions and final-training seeds.

2.5 Why positive spectral response?

The ranking rule is complete once responses are defined, but it remains to justify why a decrease in spectral entropy should be favorable. Here, structure refers to anisotropy in the representations of the AST-derived observations, rather than to the AST spans themselves. We model the directional second moment of these weighted, normalized representations as a structure-related component mixed with uniform background variation.

Assumption 1 (Structure-preserving background reduction) *For a fixed example and observation protocol, let* $A \succeq 0$ *with* $\operatorname{tr} A = 1$ *in dimension* $d \geq 2$ *denote the structure-related directional second moment. At checkpoint* m, *the expected second moment is*

$$\mathcal{C}_m = (1-\epsilon_m)A + \epsilon_m I_d/d, \tag{11}$$

where ϵ_m *is the isotropic background weight. We assume reference adaptation reduces this weight, so* $0 \leq \epsilon_1 \leq \epsilon_0 \leq 1$.

Theorem 1 (Positive response under reduced isotropic background) *Under Assumption 1, the complete spectra satisfy* $H(\mathcal{C}_0) - H(\mathcal{C}_1) \geq 0$. *If* $\epsilon_1 < \epsilon_0$ *and* $A \neq I_d/d$, *the inequality is strict.*

Proof sketch. The spectrum at each checkpoint is a mixture of the spectrum of A and the uniform spectrum. Reducing ϵ_m makes this mixture less uniform, so entropy concavity gives the result. Appendix A.6 provides the complete proof, a quantitative KL-divergence bound, and the extension to orthogonal coordinate changes; Appendix A.7 treats spectral truncation.

Theorem 1 gives the theoretical direction for the score: when adaptation reduces isotropic background without changing the structure-related component, the response $H^{(0)} - H^{(1)}$ is nonnegative. This explains why Veri-eRank favors positive entropy reductions that persist across structural views.

3 EXPERIMENTS

We evaluate whether Veri-eRank improves RTL data selection, which design choices matter, how performance varies with the selected fraction, and the overhead required to construct a ranking.

3.1 EXPERIMENTAL SETUP

Data and models. We select from `expanded_rtlcoder_12k`, a simulation-validated RTL corpus released with VeriCoder (Wei et al., 2025). We evaluate Qwen2.5-Coder-7B Base (Hui et al., 2024) and Seed-Coder-8B-Base (ByteDance Seed et al., 2025), with a separate reference adaptation and ranking for each backbone.

Protocol and baselines. At a 20% selection fraction, all methods use identical three-epoch LoRA SFT from the original base checkpoint; Full SFT uses the full corpus. Veri-eRank uses one-epoch reference adaptation with $\beta = 0.5$. We compare EL2N (Paul et al., 2021), IFD (Li et al., 2024), SPICE (Chang et al., 2026), Embedding-KMeans (Arthur & Vassilvitskii, 2007), VerilogDB-style selection (Calzada et al., 2025), and VeriSelector (Shen et al., 2026). Implementation details appear in Appendix B.2.

Evaluation. We evaluate on RTLLM v2 (Lu et al., 2024; Liu et al., 2024a) (50 tasks), ResBench (Guo & Zhao, 2025) (56), and VerilogEval v2 (Pinckney et al., 2025) (156). Pass@1 (Chen et al., 2021) measures compilation and testbench success for one greedy candidate. Main comparisons, component ablations and selection-fraction sweeps use five fixed seeds.

3.2 MAIN RESULTS

Table 1: RTL-generation Pass@1 (%; five-seed mean ± SD). Among methods using 20% budgets, bold and underlined values mark the best and second-best selection results per backbone, and * marks significant Macro Avg. differences from Veri-eRank (two-sided paired t-test, $p < 0.05$).

(a) Qwen2.5-Coder-7B Base

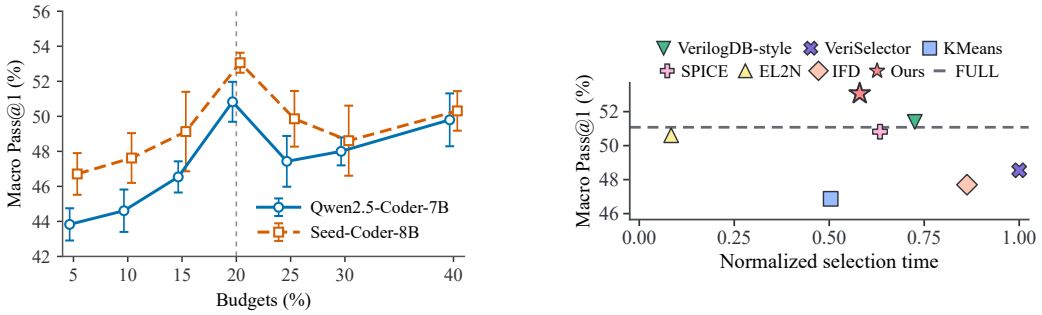
Method	RTLLM v2	ResBench	VerilogEval v2	Macro Avg.
Full SFT (100%)	42.40 ± 2.97	59.64 ± 4.11	47.31 ± 1.39	49.78 ± 2.39
SPICE	36.40 ± 1.67	55.36 ± 4.55	$\underline{44.49} \pm 0.97$	$45.41 \pm 1.17^*$
EL2N	38.80 ± 5.22	54.64 ± 3.48	42.44 ± 3.49	$45.29 \pm 2.31^*$
IFD	36.00 ± 3.74	56.43 ± 2.04	40.90 ± 1.66	$44.44 \pm 1.60^*$
VerilogDB-style	$\underline{39.60} \pm 2.19$	53.93 ± 2.65	42.82 ± 3.83	$45.45 \pm 2.18^*$
Embedding-KMeans	38.40 ± 3.85	$\underline{56.79} \pm 2.93$	42.31 ± 2.97	$\underline{45.83} \pm 2.92^*$
VeriSelector	34.00 ± 2.00	51.79 ± 3.34	37.82 ± 1.01	$41.20 \pm 1.70^*$
Veri-eRank	$\mathbf{44.40} \pm 2.19$	$\mathbf{61.43} \pm 0.98$	$\mathbf{46.67} \pm 1.72$	$\mathbf{50.83} \pm 1.14$

(b) Seed-Coder-8B Base

Method	RTLLM v2	ResBench	VerilogEval v2	Macro Avg.
Full SFT (100%)	42.00 ± 1.41	58.93 ± 2.53	52.31 ± 0.57	51.08 ± 0.84
SPICE	$\underline{45.60} \pm 1.67$	55.71 ± 3.43	51.15 ± 1.05	$50.82 \pm 1.27^*$
EL2N	44.00 ± 3.16	56.43 ± 0.98	$\underline{51.41} \pm 2.33$	$50.61 \pm 1.79^*$
IFD	36.80 ± 1.10	58.21 ± 5.14	48.08 ± 0.91	$47.70 \pm 1.71^*$
VerilogDB-style	44.40 ± 2.97	$\underline{59.29} \pm 4.26$	50.51 ± 2.81	$\underline{51.40} \pm 2.67$
Embedding-KMeans	39.20 ± 3.03	51.79 ± 2.53	49.62 ± 1.33	$46.87 \pm 1.87^*$
VeriSelector	40.40 ± 4.34	56.79 ± 4.62	48.46 ± 0.57	$48.55 \pm 2.23^*$
Veri-eRank	$\mathbf{46.00} \pm 2.00$	$\mathbf{60.36} \pm 2.33$	$\mathbf{52.82} \pm 0.97$	$\mathbf{53.06} \pm 0.57$

As shown in Table 1, Veri-eRank attains the highest mean Pass@1 among the 20% data-selection methods in all six backbone–benchmark combinations. Its five-seed macro-average is 50.83 ± 1.14 on Qwen and 53.06 ± 0.57 on Seed-Coder, exceeding SPICE by 5.42 and 2.24 percentage points (pp), respectively. These gains are positive in every matched run, with 95% confidence intervals above zero. Against the six selection baselines at the same 20% budget, macro-average gains are significant for all six comparisons on Qwen and five on Seed-Coder. The complete statistics are reported in Appendix B.3. With 2,457 selected examples, Veri-eRank increases macro-average Pass@1 over Full SFT by 1.05 pp on Qwen and 1.98 pp on Seed-Coder while using approximately 80% fewer final-training examples.

3.3 Performance across selection fractions and ranking overhead



(a) Selection fraction (b) Ranking overhead

Figure 3: Selection fraction and ranking overhead. **(a)** Five-seed mean macro-average Pass@1, with ± 1 SD. **(b)** Seed-Coder Pass@1 versus ranking overhead at a 20% selection fraction; the dashed line is Full SFT.

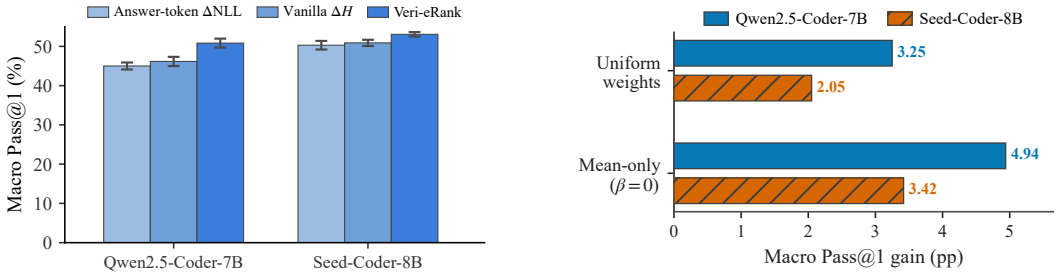
Selection fraction Figure 3(a) shows how macro-average accuracy changes as the selected fraction increases. From 5% to 20%, Qwen improves from 43.83 to 50.83 and Seed-Coder from 46.71 to 53.06, gains of 7.00 and 6.35 pp. Both backbones attain their highest mean macro-average Pass@1 at 20% within the evaluated 5–40% sweep. Relative to 40%, selecting 20% yields 1.03 pp higher mean macro-average Pass@1 on Qwen and 2.75 pp higher on Seed-Coder while using half as many examples. These results support concentrating the final-training budget on the leading portion of the ranking rather than simply expanding the selected subset. Appendix B.4 presents the corresponding benchmark-level trends and the complete numerical sweep from 5% to 40%.

Ranking overhead Figure 3(b) reports Seed-Coder ranking overhead and macro-average Pass@1. Veri-eRank reaches 53.06 macro-average Pass@1 with 76 minutes of ranking overhead, including reference adaptation, weighting, and spectral scoring. It improves on SPICE by 2.24 pp while reducing selection time from 83 to 76 minutes, and also exceeds IFD, VerilogDB-style, and VeriSelector in both accuracy and ranking efficiency. Complete costs, including the Qwen comparison, appear in Appendix B.5. Rankings can be reused across selection fractions; reported times cover subset construction only. Under the same final SFT protocol and 20% selection fraction, differences in training time across selection methods do not exceed two minutes within each backbone.

3.4 Why does Veri-eRank work?

We isolate three design choices in Veri-eRank: the adaptation signal used for scoring, the structural observations on which it is measured, and the stability-aware aggregation of responses across views. Full benchmark-level results appear in Appendix B.6.

Adaptation signal. Answer-token ΔNLL ranks examples by their mean code-token loss reduction under the shared model pair. Vanilla ΔH instead measures one uniformly weighted, all-code-token spectrum per checkpoint, without AST-based observations or resampling. In Figure 4(a), Veri-eRank improves macro-average Pass@1 over these controls by 5.83/2.77 and 4.67/2.17 percentage points (pp), respectively, on Qwen/Seed-Coder. These results indicate that the structural spectral response



(a) Scoring-rule comparison

(b) Core design ablations

Figure 4: Scoring and design controls at a 20% data budget. (a) Macro-average Pass@1 after SFT, reported as five-seed mean ± 1 SD. (b) Veri-eRank gains over each control in five-seed mean macro-average Pass@1 (pp).

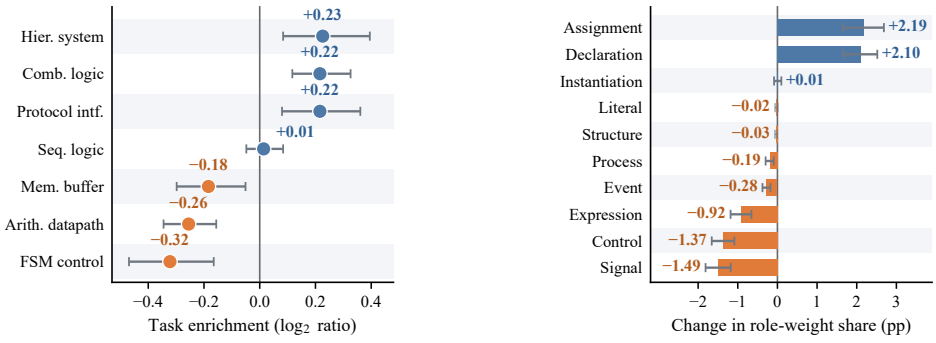
provides a more effective selection signal than direct loss reduction or unstructured spectral response.

Structural observations. Within the structural-view protocol, instruction-conditioned weights improve the macro average by 3.25/2.05 pp over uniform weights (Figure 4(b)). In a separate control, whole-span resampling improves it by 2.44/1.83 pp over token-level resampling (Table 10). The first comparison tests the instruction-dependent observation distribution; the second tests structural grouping of the observations. The complementary gains highlight the value of both instruction conditioning and span-level grouping in constructing structural observations.

Stability-aware aggregation. Response strength is the mean spectral-entropy reduction $\widehat{\mu}_i$ across paired structural views, while cross-view consistency corresponds to low dispersion $\widehat{\sigma}_i$ of those reductions. The score $S_i = \widehat{\mu}_i - 0.5\widehat{\sigma}_i$ balances the two. Compared with Mean-only, which uses $\widehat{\mu}_i$ on the same views, Veri-eRank improves macro-average Pass@1 by 4.94/3.42 pp (Figure 4(b)). It also improves over Stability-only, which ranks by $-\widehat{\sigma}_i$, by 3.07/3.55 pp (Table 11). These results support combining response strength and cross-view consistency in the ranking score.

3.5 What supervision does Veri-eRank prioritize?

On Qwen, we characterize the selected supervision by comparing the 2,457 selected examples with 1,000 equally sized random subsets matched on token length, AST span count, module count, and local parsing status. Figure 5 reports task-composition and structural-role shifts relative to this reference.



(a) Task-category enrichment

(b) Structural-role shift

Figure 5: Selected-supervision shifts on Qwen relative to matched random subsets. (a) Log_2 enrichment of training task categories. (b) Change in structural-role-weight share, in percentage points.

Task composition. Figure 5(a) describes which types of RTL tasks receive a larger share of the selected examples. Let $p_c^{(*)}$ be the selected fraction for category c and $p_c^{(\mathrm{ref})}$ its mean fraction across matched subsets. We use

$$E_c = \log_2 \frac{p_c^{(*)} + \epsilon}{p_c^{(\mathrm{ref})} + \epsilon}. \tag{12}$$

Positive enrichment identifies categories prioritized relative to the matched reference. Combinational logic, protocol/interface, and hierarchical-system examples reach 1.162, 1.162, and 1.169 times their reference shares, respectively. These are relative increases of 16.2%, 16.2%, and 16.9%, spanning local logic, module interfaces, and hierarchical organization. The resulting task mix reveals the selection's preferences beyond the matched size and parsing attributes.

Structural composition. Figure 5(b) examines the code roles represented within those examples. Let $q_r^{(*)}$ denote the average normalized structural-role-weight share of role r in the selected subset and $q_r^{(\mathrm{ref})}$ the corresponding average over matched subsets. We measure

$$\Delta q_r = q_r^{(*)} - q_r^{(\mathrm{ref})}. \tag{13}$$

Panel (b) reports $100\Delta q_r$ in percentage points. We average the within-example normalized structural-role weights from Section 2.2. The selected examples concentrate more scoring weight on declarations and assignments: their shares rise by 2.10 and 2.19 pp, respectively, and their combined share increases from 64.98% to 69.26%. Role preservation fixes each example's shares across views; selecting different examples changes their average across the training subset.

4 Related Work

Data selection for model adaptation. Prior methods use distribution or mixture matching (Xie et al., 2023b;a), training dynamics (Toneva et al., 2019; Swayamdipta et al., 2020), influence estimation (Koh & Liang, 2017; Pruthi et al., 2020), and gradient-based subset objectives (Mirzasoleiman et al., 2020; Killamsetty et al., 2021). For instruction tuning, LESS estimates task influence (Xia et al., 2024), while SPICE balances information gain and gradient conflict (Chang et al., 2026). Code-data selection includes clustering-based pruning (Tsai et al., 2024) and difficulty-aware sampling within clusters (Lv et al., 2025). These criteria do not explicitly measure adaptation-induced changes in specification-relevant RTL representations, whereas Veri-eRank captures such changes by comparing matched structural observations before and after reference adaptation.

Representation geometry and RTL data selection. Matrix entropy estimators (Sanchez Giraldo et al., 2015) and matrix-information objectives (Zhang et al., 2024) connect spectral properties to representation learning; Diff-eRank compares effective ranks across language-model states (Wei et al., 2024). RTL generation studies investigate domain-specific fine-tuning and testbench-based evaluation (Thakur et al., 2022; 2023), while MEV-LLM trains experts on complexity-partitioned data (Nadimi & Zheng, 2024). RTL curation uses tool-based checks (Calzada et al., 2025), simulation (Wei et al., 2025), and behavioral compatibility (Cai et al., 2026). Within a validated RTL pool, Veri-eRank ranks examples by stable spectral responses to shared reference adaptation, using matched, instruction-weighted structural observations.

5 Conclusion

Veri-eRank selects RTL supervision from stable spectral changes in specification-relevant structural representations. Its matched observations and stability-aware ranking are supported by a positive-response analysis and a finite-view robustness bound. With 20% of the candidate corpus, Veri-eRank achieves the highest mean Pass@1 among the compared data-selection baselines in all six backbone–benchmark settings and improves macro-average Pass@1 over Full SFT. Future work will test whether analogous structural observations support data selection in other structured-generation domains.

AI USE STATEMENT

We used AI tools to polish the writing, assist with mathematical proofs and support literature retrieval. The authors independently verified the mathematical arguments and references, reviewed all AI-assisted content, and take full responsibility for the manuscript.

ETHICS STATEMENT

This work studies data selection for RTL Verilog generation. All models, datasets, and benchmarks used in this study are obtained from their official open-source releases, with corresponding source links and bibliographic citations provided to document their provenance and acknowledge the original contributors. We evaluate the functional correctness of generated RTL through compilation and testbench simulation on the adopted benchmarks. The reported results characterize functional performance under these benchmark evaluation protocols.

REPRODUCIBILITY STATEMENT

Our implementation is available in an anonymous GitHub repository at `https://anonymous.4open.science/r/Veri-eRank/` and as a ZIP archive in the supplementary material. Both the repository and the supplementary code archive have been anonymized for double-blind review.

In addition, the appendix provides further details to facilitate reproducibility, including:

- Detailed implementation specifications, numerical settings, pseudocode and computational analysis for Veri-eRank (Appendices A.1–A.4 and A.10).
- Complete theoretical analyses and proofs supporting the proposed measurement and ranking method (Appendices A.5–A.9).
- Detailed data preparation, training and evaluation settings, computing environments, and baseline implementations (Appendices B.1 and B.2).
- Additional results and analyses, including significance analysis, budget sweeps, selection costs, ablations, selected-data analyses and capability evaluation (Appendices B.3–B.8).

REFERENCES

David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pp. 1027–1035, USA, 2007. Society for Industrial and Applied Mathematics. ISBN 9780898716245.

Andrei Z. Broder. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No.97TB100171)*, pp. 21–29. IEEE Comput. Soc, 1997. doi: 10.1109/sequen.1997.666900. URL `https://doi.org/10.1109/SEQUEN.1997.666900`.

ByteDance Seed, Yuyu Zhang, Jing Su, Yifan Sun, Chenguang Xi, Xia Xiao, Shen Zheng, Anxiang Zhang, Kaibo Liu, Daoguang Zan, Tao Sun, Jinhua Zhu, Shulin Xin, Dong Huang, Yetao Bai, Lixin Dong, Chao Li, Jianchong Chen, Hanzhi Zhou, Yifan Huang, Guanghan Ning, Xierui Song, Jiaze Chen, Siyao Liu, Kai Shen, Liang Xiang, and Yonghui Wu. Seed-Coder: Let the Code Model Curate Data for Itself. arXiv preprint arXiv:2506.03524, 2025. URL `https://arxiv.org/abs/2506.03524`.

Siyang Cai, Cangyuan Li, Wenjing Chang, Kun Wang, Haoyu Gao, Yinhe Han, and Ying Wang. RTLCurator: Label-Efficient Data Curation for RTL Generation. arXiv preprint arXiv:2607.29283, 2026. URL `https://arxiv.org/abs/2607.29283`.

Paul E. Calzada, Zahin Ibnat, Tanvir Rahman, Kamal Kandula, Danyu Lu, Sujan Kumar Saha, Farimah Farahmandi, and Mark Tehranipoor. VerilogDB: The Largest, Highest-Quality Dataset with a Preprocessing Framework for LLM-based RTL Generation. arXiv preprint arXiv:2507.13369, 2025. URL `https://arxiv.org/abs/2507.13369`.

Powei Chang, Jinpeng Zhang, Bowen Chen, Chenyu Wang, Chenlu Guo, Yixing Zhang, Yukang Gao, JianXiang Xiang, Yue Gao, Chaoqun Sun, Yiyi Chen, and Dongying kong. SPICE: Submodular penalized information–conflict selection for efficient large language model training. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=9rCRy58TPF.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. AlpaGasus: Training a Better Alpaca with Fewer Data. In *International Conference on Learning Representations*, volume 2024, pp. 34767–34797, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/9543942c237ded1b39b1fd37259ff88e-Abstract-Conference.html.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating Large Language Models Trained on Code. arXiv preprint arXiv:2107.03374, 2021. URL https://arxiv.org/abs/2107.03374.

John C. Duchi and Hongseok Namkoong. Learning models with uniform performance via distributionally robust optimization. *The Annals of Statistics*, 49(3):1378–1406, 2021. doi: 10.1214/20-aos2004. URL https://doi.org/10.1214/20-aos2004.

John C. Duchi, Peter W. Glynn, and Hongseok Namkoong. Statistics of robust optimization: A generalized empirical likelihood approach. *Mathematics of Operations Research*, 46(3):946–969, 2021. doi: 10.1287/moor.2020.1085. URL https://doi.org/10.1287/moor.2020.1085.

Quentin Garrido, Randall Balestriero, Laurent Najman, and Yann Lecun. RankMe: Assessing the Downstream Performance of Pretrained Self-Supervised Representations by Their Rank. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 10929–10974. PMLR, 2023. URL https://proceedings.mlr.press/v202/garrido23a.html.

Ce Guo and Tong Zhao. ResBench: A Resource-Aware Benchmark for LLM-Generated FPGA Designs. In *Proceedings of the 15th International Symposium on Highly Efficient Accelerators and Reconfigurable Technologies*, HEART 2025, pp. 25–34. ACM, 2025. doi: 10.1145/3728179.3728192. URL http://dx.doi.org/10.1145/3728179.3728192.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, Kai Dang, Yang Fan, Yichang Zhang, An Yang, Rui Men, Fei Huang, Bo Zheng, Yibo Miao, Shanghaoran Quan, Yunlong Feng, Xingzhang Ren, Xuancheng Ren, Jingren Zhou, and Junyang Lin. Qwen2.5-Coder Technical Report. arXiv preprint arXiv:2409.12186, 2024. URL https://arxiv.org/abs/2409.12186.

Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing - STOC '98*, pp. 604–613. ACM Press, 1998. doi: 10.1145/276698.276876. URL https://doi.org/10.1145/276698.276876.

Krishnateja Killamsetty, Durga S, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. GRAD-MATCH: Gradient Matching based Data Subset Selection for Efficient Deep Model Training. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 5464–5474, 2021. URL https://proceedings.mlr.press/v139/killamsetty21a.html.

Pang Wei Koh and Percy Liang. Understanding Black-box Predictions via Influence Functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 1885–1894, 2017. URL https://proceedings.mlr.press/v70/koh17a.html.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. From Quantity to Quality: Boosting LLM Performance with Self-Guided Data Selection for Instruction Tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7602–7635. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.naacl-long.421. URL https://aclanthology.org/2024.naacl-long.421/.

Shang Liu, Yao Lu, Wenji Fang, Mengming Li, and Zhiyao Xie. OpenLLM-RTL: Open Dataset and Benchmark for LLM-Aided Design RTL Generation. In *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*, ICCAD '24, pp. 1–9. ACM, 2024a. doi: 10.1145/3676536.3697118. URL http://dx.doi.org/10.1145/3676536.3697118.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. What Makes Good Data for Alignment? A Comprehensive Study of Automatic Data Selection in Instruction Tuning. In *International Conference on Learning Representations*, volume 2024, pp. 22353–22373, 2024b. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/6091f2bb355e960600f62566ac0e2862-Paper-Conference.pdf.

S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2): 129–137, March 1982. ISSN 1557-9654. doi: 10.1109/tit.1982.1056489. URL http://dx.doi.org/10.1109/TIT.1982.1056489.

Yao Lu, Shang Liu, Qijun Zhang, and Zhiyao Xie. RTLLM: An Open-Source Benchmark for Design RTL Generation with Large Language Model. In *2024 29th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 722–727. IEEE, 2024. doi: 10.1109/asp-dac58780.2024.10473904. URL http://dx.doi.org/10.1109/ASP-DAC58780.2024.10473904.

Weijie Lv, Xuan Xia, and Sheng-Jun Huang. Data-efficient LLM fine-tuning for code generation. *arXiv preprint arXiv:2504.12687*, 2025. URL https://arxiv.org/abs/2504.12687.

Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for Data-efficient Training of Machine Learning Models. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 6950–6960, 2020. URL https://proceedings.mlr.press/v119/mirzasoleiman20a.html.

Bardia Nadimi and Hao Zheng. A multi-expert large language model architecture for Verilog code generation. *arXiv preprint arXiv:2404.08029*, 2024. URL https://arxiv.org/abs/2404.08029.

Hongseok Namkoong and John Duchi. Variance-based regularization with convex objectives. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/5a142a55461d5fef016acfb927fee0bd-Paper.pdf.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. doi: 10.52202/068431-2011. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep Learning on a Data Diet: Finding Important Examples Early in Training. In *Advances in Neural Information Processing Systems*, volume 34, pp. 20596–20607. Curran Associates, Inc.,

2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/ac56f8fe9eea3e4a365f29f0f1957c55-Paper.pdf.

Nathaniel Pinckney, Christopher Batten, Mingjie Liu, Haoxing Ren, and Brucek Khailany. Revisiting verilogeval: A year of improvements in large-language models for hardware code generation. *ACM Trans. Des. Autom. Electron. Syst.*, 30(6), October 2025. ISSN 1084-4309. doi: 10.1145/3718088. URL https://doi.org/10.1145/3718088.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating Training Data Influence by Tracing Gradient Descent. In *Advances in Neural Information Processing Systems*, volume 33, pp. 19920–19930, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/e6385d39ec9394f2f3a354d9d2b88eec-Abstract.html.

Olivier Roy and Martin Vetterli. The Effective Rank: A Measure Of Effective Dimensionality. In *Proceedings of the 15th European Signal Processing Conference*, pp. 606–610, 2007. doi: 10.5281/ZENODO.40328. URL https://eurasip.org/Proceedings/Eusipco/Eusipco2007/Papers/a5p-h05.pdf.

Luis Gonzalo Sanchez Giraldo, Murali Rao, and Jose C. Principe. Measures of entropy from data using infinitely divisible kernels. *IEEE Transactions on Information Theory*, 61(1):535–548, 2015. doi: 10.1109/tit.2014.2370058. URL https://doi.org/10.1109/tit.2014.2370058.

Yiheng Shen, Wei Zheng, Xiao Wei, Hao Shen, Xiang Chen, and Guang Yang. Quality over quantity: Diversity-aware data selection for efficient verilog code generation, 2026. URL https://arxiv.org/abs/2609.22765.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. Dataset Cartography: Mapping and Diagnosing Datasets with Training Dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9275–9293, 2020. doi: 10.18653/v1/2020.emnlp-main.746. URL https://aclanthology.org/2020.emnlp-main.746/.

Shailja Thakur, Baleegh Ahmad, Zhenxing Fan, Hammond Pearce, Benjamin Tan, Ramesh Karri, Brendan Dolan-Gavitt, and Siddharth Garg. Benchmarking large language models for automated Verilog RTL code generation. *arXiv preprint arXiv:2212.11140*, 2022. URL https://arxiv.org/abs/2212.11140.

Shailja Thakur, Baleegh Ahmad, Hammond Pearce, Benjamin Tan, Brendan Dolan-Gavitt, Ramesh Karri, and Siddharth Garg. VeriGen: A large language model for Verilog code generation. *arXiv preprint arXiv:2308.00708*, 2023. URL https://arxiv.org/abs/2308.00708.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. An Empirical Study of Example Forgetting during Deep Neural Network Learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BJlxm30cKm.

Yun-Da Tsai, Mingjie Liu, and Haoxing Ren. Code less, align more: Efficient LLM fine-tuning for code generation with data pruning. *arXiv preprint arXiv:2407.05040*, 2024. URL https://arxiv.org/abs/2407.05040.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-Instruct: Aligning Language Models with Self-Generated Instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13484–13508. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.acl-long.754. URL https://aclanthology.org/2023.acl-long.754/.

Anjiang Wei, Huanmi Tan, Tarun Suresh, Daniel Mendoza, Thiago S. F. X. Teixeira, Ke Wang, Caroline Trippel, and Alex Aiken. VeriCoder: Enhancing LLM-Based RTL Code Generation through Functional Correctness Validation. arXiv preprint arXiv:2504.15659, 2025. URL https://arxiv.org/abs/2504.15659.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned Language Models are Zero-Shot Learners. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=gEZrGCozdqR.

Lai Wei, Zhiquan Tan, Chenghai Li, Jindong Wang, and Weiran Huang. Diff-eRank: A Novel Rank-Based Metric for Evaluating Large Language Models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 39501–39521. Curran Associates, Inc., 2024. doi: 10.52202/079017-1248. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/45cc967d7899616f51993b7b363d35b5-Paper-Conference.pdf.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting Influential Data for Targeted Instruction Tuning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54104–54132. PMLR, 2024. URL https://proceedings.mlr.press/v235/xia24c.html.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. DoReMi: Optimizing Data Mixtures Speeds Up Language Model Pretraining. In *Advances in Neural Information Processing Systems*, volume 36, pp. 69798–69818, 2023a. doi: 10.52202/075280-3059. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/dcba6be91359358c2355cd920da3fcbd-Abstract-Conference.html.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy S Liang. Data Selection for Language Models via Importance Resampling. In *Advances in Neural Information Processing Systems*, volume 36, pp. 34201–34227, 2023b. doi: 10.52202/075280-1482. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/6b9aa8f418bde2840d5f4ab7a02f663b-Abstract-Conference.html.

Yifan Zhang, Zhiquan Tan, Jingqin Yang, Weiran Huang, and Yang Yuan. Matrix Information Theory for Self-Supervised Learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 59897–59918. PMLR, 2024. URL https://proceedings.mlr.press/v235/zhang24bi.html.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: Less Is More for Alignment. In *Advances in Neural Information Processing Systems*, volume 36, pp. 55006–55021. Curran Associates, Inc., 2023. doi: 10.52202/075280-2400. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/ac662d74829e4407ce1d126477f4a03a-Paper-Conference.pdf.

A MEASUREMENT PROTOCOL AND THEORETICAL ANALYSIS

This appendix specifies the observation units, numerical processing, and resampling protocol used by Veri-eRank. It then analyzes the direction of the spectral response, structural composition control, and the robustness interpretation of the existing score.

A.1 FROM RTL SYNTAX TO TOKEN OBSERVATIONS

We parse the reference implementation y_i with Tree-sitter Verilog. Each named node provides a candidate source interval; comments and the source-file root are excluded. The deterministic hardware-semantic role abstraction assigns AST nodes to ten roles used for measurement and stratification (Table 2).

Table 2: Hardware-semantic roles used for span selection and stratification. Examples describe the scope of each role; isolated vocabulary tokens need not identify every AST role.

Role	Function	Representative constructs
`declaration`	Interfaces, objects, types	Port, net, variable, parameter, and type declarations
`process`	Procedural execution	`always`, `always_ff`, `always_comb`, `always_latch`, `initial`
`event`	Temporal triggers	Event controls, clock/reset edges, sensitivity lists
`control`	Branching and iteration	`if`, `case`, conditionals, and loops
`assignment`	Updates and drivers	Blocking, nonblocking, and continuous assignments
`expression`	Data transformation	Operators, calls, concatenations, expressions
`signal`	Named objects	Identifiers, variables, nets, signal references
`literal`	Constants	Numeric, bit-vector, and string literals
`instantiation`	Design hierarchy	Module instances and instantiation constructs
`structure`	Residual organization	Modules, generate regions, blocks, punctuation, unmatched nodes

Nested AST intervals would count some tokens repeatedly. We therefore greedily construct non-overlapping main spans. The priority order is `assignment`, `declaration`, `instantiation`, `event`, `control`, `process`, `expression`, `signal`, `literal`, and `structure`. Within one role, shorter spans are processed first, followed by earlier source positions. A candidate is retained only if its interval does not overlap an already retained span. An unmatched node falls back to `structure`.

Source intervals are aligned to tokenizer offsets in the assistant-code region. Let $T_i(s)$ denote the tokens uniquely owned by span s. Uncovered tokens and tokens with ambiguous boundary ownership are excluded from the geometric support. The three granularities have separate functions: roles define strata, spans define the resampling units, and hidden-state tokens provide the observations. In particular, a span is never pooled into a single embedding. After numerical filtering, selecting a span includes all of its remaining eligible tokens; it cannot restore excluded boundary or invalid tokens.

Worked example. Figure 6 traces the measured observation construction for the selected RTL example shown in Figure 1. In this example, overlapping conditional-statement candidates are rejected while local event and expression observations are retained. The resulting non-overlapping spans receive instruction-conditioned weights and are reused to construct paired views with preserved role mass.

Selected example: rtl_40f5141f | Qwen | rank 1,421 / 12,259

(a) Key AST candidate decisions

ID	AST type	Role	Lines	Decision	Overlaps
C4	always_keyword	process	7–7	Keep S11	-
C1	always_construct	process	7–13	Reject	S15,S17,S12,S11
C5	edge_identifier	event	7–7	Keep S12	-
C6	expression	expression	7–7	Keep S13	-
C2	conditional_stmt	control	8–12	Reject	S15,S17
C7	expression	expression	8–8	Keep S14	-
C3	conditional_stmt	control	10–12	Reject	S17

(b) Span weights and paired-view membership

17 spans / 26 eligible tokens

Span	Role	n_s	$a(s)$	W_s (%)	Draw 1	Draw 2	Draw 3
S1	structure	1	0.100	1.65	●	●	●
S2	signal	2	0.439	14.47	●	●	●
S3	declaration	1	0.449	7.40	●	●	●
S4	declaration	1	0.231	3.81	●	–	●
S5	declaration	1	0.354	5.83	–	●	●
S6	declaration	1	0.417	6.87	●	●	–
S7	declaration	1	1.000	16.47	●	–	–
S8	declaration	1	0.118	1.95	●	●	●
S9	declaration	1	0.100	1.65	–	●	●
S10	declaration	1	0.377	6.20	●	●	●
S11	process	1	0.318	5.24	●	●	●
S12	event	2	0.101	3.32	●	●	●
S13	expression	1	0.113	1.86	●	–	●
S14	expression	1	0.105	1.72	–	●	–
S15	assignment	6	0.148	14.67	–	–	●
S16	expression	1	0.101	1.66	●	●	●
S17	assignment	3	0.106	5.22	●	●	–

Filled dot: retained. Dash: not sampled. Three of 16 draws shown.

(c) Role-mass preservation

Assign.	Decl.	Event	Expr.	Process	Signal	Struct.
19.90%	50.18%	3.32%	5.25%	5.24%	14.47%	1.65%

Full support = draw 1 = draw 2 = draw 3 at the displayed precision.

Largest reported absolute residual: 4.6e-08.

Figure 6: Recorded candidate decisions, span weights, and paired-view membership for the selected RTL example; three of 16 draws are shown, and source locations use the original RTL line numbers.

A.2 Instruction-conditioned weighting

The base model evaluates y_i both in an instruction-aware context $c_i^+ = \mathrm{Prompt}(x_i, y_i)$ and in a code-only context $c_i^- = y_i$. The same reference code is teacher-forced in each context. For each evaluable code position t, logits at its preceding position predict the token at t. If $\mathcal{K}_{it}^c$ is the top-K_p

candidate set, its truncated probabilities and role probabilities are

$$q_{it}^c(v) = \frac{\exp \ell_{it}^c(v)}{\sum_{u \in \mathcal{K}_{it}^c} \exp \ell_{it}^c(u)}, \qquad v \in \mathcal{K}_{it}^c,$$
$$P_{it}^c(r) = \sum_{v \in \mathcal{K}_{it}^c} q_{it}^c(v) \mathbb{I}\{\rho_{\mathrm{lex}}(v) = r\}. \tag{14}$$

Here ℓ_{it}^c denotes the preceding-position logits, and ρ_{lex} is a deterministic lexical map. It recognizes Verilog keywords, identifier forms, literals, and operators, assigning unmatched candidates to `structure`. This map acts on predicted vocabulary tokens and differs from ρ_{AST}, which classifies observed syntax nodes. For example, an isolated candidate token need not reveal that an identifier belongs to a module instantiation. The entropy therefore measures uncertainty in this lexical projection of the role space, rather than uncertainty over a complete parse or over the full vocabulary.

For context c, let $J_i^c(s)$ be the positions aligned to s that have valid preceding logits. Because contextual tokenization and offsets can differ, these sets are defined separately for the two contexts. We use $\mathcal{E}_{it}^c = -\sum_r P_{it}^c(r) \log P_{it}^c(r)$ with $0 \log 0 = 0$, average over $J_i^c(s)$, and compute the positive entropy difference in Equation (2). The numerical constant ϵ_s safeguards evaluation at zero probability; it does not add probability mass to a new role.

Let $\mathcal{S}_{i,\mathrm{eval}}$ contain the spans evaluable in both contexts, and let $g_i^{\min}$ and $g_i^{\max}$ be their minimum and maximum gains. The complete convention for the weight in Equation (3) is

$$a_i(s) = \begin{cases} a_{\min}, & s \notin \mathcal{S}_{i,\mathrm{eval}}, \\ a_{\mathrm{eq}}, & s \in \mathcal{S}_{i,\mathrm{eval}}, \quad g_i^{\max} = g_i^{\min}, \\ a_{\min} + (a_{\max} - a_{\min}) \dfrac{g_i(s) - g_i^{\min}}{g_i^{\max} - g_i^{\min}}, & \text{otherwise.} \end{cases} \tag{15}$$

If no span is evaluable, all spans receive $a_{\min}$. The positive lower bound retains low-gain regions. A token inherits its owner's weight: $a_{it} = a_i(s)$ for $t \in T_i(s)$. Consequently, the total unnormalized mass of a span is proportional to its number of eligible tokens as well as its weight; the method does not impose equal mass per span. All weights are computed once with M_0 and remain fixed in the paired comparison.

A.3 Paired support, views, and numerical safeguards

Both models process the unchanged instruction-aware sequence c_i^+, and we extract hidden states from the final decoder layer. Initialize a common candidate support with uniquely owned AST tokens whose weights are positive and whose states are finite in both models. On the current support, normalize the weights and recompute each model's weighted center. Remove a token from *both* sides whenever its centered norm is zero or nonfinite on either side. Repeat until the support stabilizes or becomes unsupported. This procedure terminates because an unsuccessful iteration strictly removes tokens from a finite set.

The resulting weights w_{it}, model-specific centers, and unit directions $z_{it}^{(m)}$ follow Equation (4). The centers are computed on the complete shared support $\mathcal{T}_i$ and are held fixed across all views. Recentring every view would change the origin together with the sampled observations. Although hidden states are centered before normalization, $\sum_t w_{it} z_{it}^{(m)}$ need not be zero after each row is normalized. The feature-space matrix $Z^\top W Z$ is therefore a directional second moment, rather than necessarily the covariance of the normalized rows.

For each role r with positive full-support mass, form its effective span set $\mathcal{S}_{ir}$ from spans with eligible tokens. Each repetition draws $\max(1, \lfloor \alpha |\mathcal{S}_{ir}| \rfloor)$ whole spans without replacement within that role. Intersect their token sets with the shared support to obtain $V_{ir}^{(b)}$, and use Equation (8) to restore the full-support role mass π_{ir}. The same observations and weights are used for M_0 and M_1. These views are formed entirely in representation space by subsampling cached observations from the intact RTL sequence, requiring no additional Transformer forward passes.

For each model and view, symmetrize the weighted Gram matrix as $(G + G^\top)/2$ before eigendecomposition. Retain eigenvalues satisfying

$$\lambda_j > \epsilon_e, \qquad \lambda_j \geq \lambda_{\max}/\kappa, \tag{16}$$

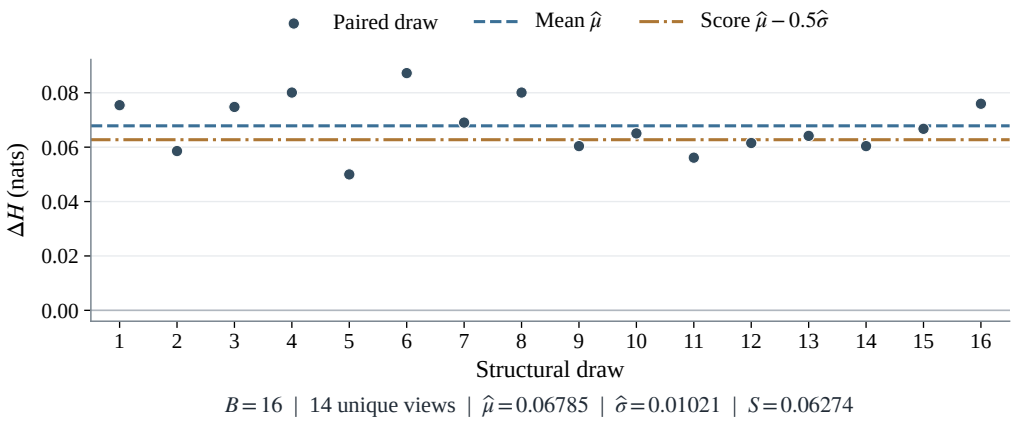


Figure 7: Paired spectral responses for the selected RTL example across 16 structural draws; the horizontal reference lines mark the mean response and the penalized ranking score.

then normalize the retained values to sum to one. The test is applied separately to the two spectra, which need not retain the same number of eigenvalues. Entropy and effective rank refer to these renormalized retained spectra. Empty or nonfinite spectra are invalid.

A sample must contain at least three valid spans, with at least one role containing two or more. We require exactly B valid paired repetitions and at least $U_{\min}$ distinct span-selection signatures among them. A signature is the ordered set of retained span identifiers. Random states are determined by the global seed, stable sample identifier, repetition index, and retry index. An invalid view is resampled; failure to complete a valid set yields an explicit unsupported status. Repeated signatures count as repeated draws in the mean and sample standard deviation. They are not deduplicated before computing Equation (9). The unperturbed full view is diagnostic only and is not an additional observation in that score.

Table 3: Frozen measurement configuration. The resampling seed is distinct from any training seed.

Parameter	Value
Reference adaptation	One epoch on the full candidate corpus
Representation layer	Final decoder layer
Probability candidates K_p	50
Span weights $(a_{\min}, a_{\max}, a_{\text{eq}})$	$(0.1, 1.0, 1.0)$
Within-role retention α	0.8
Paired repetitions B	16
Stability penalty β	0.5
Distinct signatures $U_{\min}$	3
Numerical safeguards ϵ_s, ϵ_e	$10^{-12}, 10^{-12}$
Relative spectral cutoff κ	10^6
Resampling seed	3407
Minimum structural support	Three spans; one role with at least two spans

Figure 7 reports the paired spectral responses of the selected RTL example shown in Figure 1. Its 16 draws contain 14 distinct span-selection signatures; repeated signatures retain their original multiplicity when computing $\widehat{\mu}_i$, $\widehat{\sigma}_i$, and $S_i = \widehat{\mu}_i - 0.5\widehat{\sigma}_i$. The dispersion describes variation across structural views of this example, rather than across final-training seeds.

Algorithm 1 Veri-eRank: paired representation measurement and budgeted selection

Require: Candidate corpus $\mathcal{D}$, base checkpoint M_0, budget γ; configuration in Table 3
Ensure: Subset $\mathcal{D}_\gamma$ and final model M_{final}

$M_1 \leftarrow \text{ReferenceSFT}(M_0, \mathcal{D}; 1 \text{ epoch})$
$\mathcal{I}_{\text{ok}} \leftarrow \varnothing$
for all $d_i = (x_i, y_i) \in \mathcal{D}$ **do**
    Parse y_i, assign roles, greedily select non-overlapping spans, and align token offsets
    Compute M_0 logits in c_i^+ and c_i^-; obtain span weights
    Extract M_0/M_1 states in c_i^+; jointly clean support and fix full-support centers
    if parsing, alignment, or shared structural support is insufficient **then**
        Mark i unsupported; **continue**
    end if
    Generate B paired role-stratified views; preserve each role's mass
    Compute filtered spectra and $\Delta H_i^{(b)}$ for each valid view, retrying invalid draws
    if fewer than B valid draws or fewer than $U_{\min}$ distinct signatures **then**
        Mark i unsupported; **continue**
    end if
    $S_i \leftarrow \text{Mean}_b\, \Delta H_i^{(b)} - \beta\, \text{Std}_b\, \Delta H_i^{(b)}$
    $\mathcal{I}_{\text{ok}} \leftarrow \mathcal{I}_{\text{ok}} \cup \{i\}$
end for
Assert $|\mathcal{I}_{\text{ok}}| \geq \lceil \gamma N \rceil$
$\mathcal{D}_\gamma \leftarrow$ top $\lceil \gamma N \rceil$ supported examples, with deterministic tie breaking
$M_{\text{final}} \leftarrow \text{FinalSFT}(\text{Fresh}(M_0), \mathcal{D}_\gamma)$
return $(\mathcal{D}_\gamma, M_{\text{final}})$

A.4 Selection algorithm

Algorithm 1 summarizes the complete protocol. Selection sorts supported examples by descending score and resolves ties using stable sample identifiers. If fewer than $\lceil \gamma N \rceil$ examples are supported, execution fails instead of reducing the requested budget. Final training reloads the original M_0 checkpoint with new adapter and optimizer states. Neither the reference adapter nor its optimizer state initializes the final model.

A.5 Properties of the spectral measurement

Positive semidefiniteness, unit trace, and the Gram dual. For a fixed model and view, write $A_V = W^{1/2} Z$. The token Gram matrix is $G = A_V A_V^\top$ and the feature second moment is $C = A_V^\top A_V$. Both are positive semidefinite; a singular value decomposition of A_V shows that their nonzero eigenvalues are its squared nonzero singular values. Moreover,

$$\operatorname{tr} G = \operatorname{tr} C = \sum_{t \in V} w_t \|z_t\|_2^2 = \sum_{t \in V} w_t = 1. \tag{17}$$

This identity holds before spectral filtering. For q retained positive eigenvalues, $0 \leq H \leq \log q$ and $1 \leq \exp(H) \leq q$. Consequently, $\Delta H = \log(\text{eRank}^{(0)} \,/\, \text{eRank}^{(1)})$. The measurement describes the normalized second-moment spectrum, rather than Euclidean volume or raw hidden-state norms.

Conditional invariances. Fix the source alignment, semantic weights, support, and sampled span sets. Under $h_t' = cQh_t + b$, where $c > 0$ and $Q^\top Q = I$, the weighted center becomes $\bar{h}' = cQ\bar{h} + b$ and each normalized direction becomes $z_t' = Qz_t$. Inner products, Gram eigenvalues, and entropies are unchanged, including when the transformation differs between checkpoints. Permuting observation rows together with their weights and view membership gives $G' = PGP^\top$, also preserving the spectrum. These statements concern the fixed measurement; they do not cover changes to tokenization, source order, semantic weights, or independent rescaling of individual rows before centering.

What pairing controls. Let $F_m(V)$ be the entropy at checkpoint m under a random admissible view, with all other settings fixed. The paired response satisfies

$$\text{Var}[F_0(V) - F_1(V)] = \text{Var}[F_0(V)] + \text{Var}[F_1(V)] - 2\,\text{Cov}[F_0(V), F_1(V)]. \tag{18}$$

Thus pairing reduces variance relative to independent views with the same marginals when the checkpoint entropies have positive covariance. Regardless of its sign, pairing fixes which token observations enter each comparison. The measured standard deviation describes sensitivity to these views, rather than variation over final-training seeds.

Local response along the realized adaptation path. Write the reference update as $\theta_1 = \theta_0 + \delta\theta$ and the entropy for a fixed example and view as $f_{i,V}(\theta)$. Suppose support is fixed, no eigenvalue crosses a filtering threshold, retained eigenvalues stay positive, and $f_{i,V}$ is twice continuously differentiable along the parameter segment. For some $\widetilde{\theta}$ on that segment, Taylor's theorem gives

$$\Delta H_i(V) = -\nabla f_{i,V}(\theta_0)^\top \delta\theta - \frac{1}{2}\delta\theta^\top \nabla^2 f_{i,V}(\widetilde{\theta})\delta\theta. \tag{19}$$

This characterizes the local response along the realized full-corpus adaptation path, including interactions between examples. Threshold crossings fall outside this local smoothness regime.

A.6 Why a positive spectral response can indicate structural consolidation

We give the complete proof of Theorem 1 under Assumption 1. The retained-structure mixture describes a distribution of the weighted directions *after* centering and row normalization. Let $\mathcal{P}_\star$ be a distribution on the unit sphere in a fixed d-dimensional space, $d \geq 2$, with directional second moment $A = \mathbb{E}_{\mathcal{P}_\star}[zz^\top]$. Thus $A \succeq 0$ and $\operatorname{tr} A = 1$. The uniform sphere distribution $\mathcal{P}_{\text{iso}}$ has second moment I_d/d. At checkpoint m, assume the directions follow the mixture $(1-\epsilon_m)\mathcal{P}_\star + \epsilon_m \mathcal{P}_{\text{iso}}$, up to an orthogonal coordinate change U_m. The population second moment is

$$\mathcal{C}_m = U_m \left[(1-\epsilon_m)A + \epsilon_m \frac{I_d}{d}\right] U_m^\top, \qquad U_m^\top U_m = I_d. \tag{20}$$

Here A represents preserved directional structure and ϵ_m the isotropic background component; we use this mixture as an explanatory model of structure-preserving adaptation.

Theorem 1: direction under reduced isotropic background. Let $H(\mathcal{C}_m)$ denote entropy of the complete unit-trace spectrum in Equation (20). If $0 \leq \epsilon_1 \leq \epsilon_0 \leq 1$, then $H(\mathcal{C}_0) - H(\mathcal{C}_1) \geq 0$. When $\epsilon_1 < \epsilon_0$, write $\eta = (\epsilon_0 - \epsilon_1)/(1-\epsilon_1)$ and let p_1 be the spectrum of $\mathcal{C}_1$, with $u = (1/d, \ldots, 1/d)$. The stronger bound is

$$H(\mathcal{C}_0) - H(\mathcal{C}_1) \geq \eta\,[\log d - H(p_1)] = \eta D_{\mathrm{KL}}(p_1 \| u). \tag{21}$$

It is strictly positive if $A \neq I_d/d$. The aligned discrepancy from the preserved structure also satisfies

$$\left\| U_m^\top \mathcal{C}_m U_m - A \right\|_F^2 = \epsilon_m^2 \left\| I_d/d - A \right\|_F^2. \tag{22}$$

Proof. Let $\lambda = (\lambda_1, \ldots, \lambda_d)$ be the spectrum of A. Orthogonal conjugation preserves eigenvalues, so $p_m = (1-\epsilon_m)\lambda + \epsilon_m u$. If $\epsilon_1 < \epsilon_0$, then $\epsilon_1 < 1$ and $0 < \eta \leq 1$. Substitution gives $p_0 = (1-\eta)p_1 + \eta u$. Entropy concavity yields

$$H(p_0) \geq (1-\eta)H(p_1) + \eta H(u) = (1-\eta)H(p_1) + \eta \log d, \tag{23}$$

which proves Equation (21). If $A \neq I_d/d$, then $p_1 \neq u$ because $\epsilon_1 < 1$; hence $D_{\mathrm{KL}}(p_1 \| u) > 0$. If $\epsilon_0 = \epsilon_1$, both spectra coincide, including the endpoint $\epsilon_0 = \epsilon_1 = 1$. If $A = I_d/d$, both spectra are uniform for every mixture weight. Finally, $U_m^\top \mathcal{C}_m U_m - A = \epsilon_m(I_d/d - A)$, giving Equation (22) by taking squared Frobenius norms. □

For a non-isotropic A, reducing the background weight therefore decreases both entropy and discrepancy from the preserved structure. This supplies a mechanism for orienting the score as $H^{(0)} - H^{(1)}$. More generally, $H(p_0) - H(p_1) = D_{\mathrm{KL}}(p_1 \| u) - D_{\mathrm{KL}}(p_0 \| u)$ for spectra in a common dimension: the response measures increased spectral departure from isotropy. The theorem distinguishes one structure-preserving cause of this increase from collapse that changes A. Accordingly, the theorem provides a theoretical basis for favoring positive spectral responses.

A.7 FINITE OBSERVATIONS AND SPECTRAL TRUNCATION

Theorem 1 concerns expected directional moments. The algorithm uses finite weighted token observations, whose second moments can be rank deficient when their population counterparts have full rank. The dimension d in the model is fixed across checkpoints and is not the number of retained eigenvalues or the number of AST roles. The separate effect of the implemented numerical truncation can be characterized exactly.

Consider an empirical unit-trace PSD spectrum p, represented in a common ambient feature dimension d. Let $\delta < 1$ be the total discarded mass, q the normalized retained spectrum, and v the normalized discarded spectrum when $\delta > 0$. With $h_2(\delta) = -\delta \log \delta - (1-\delta)\log(1-\delta)$,

$$H(p) = h_2(\delta) + (1-\delta)H(q) + \delta H(v). \tag{24}$$

To verify the identity, substitute $(1-\delta)q_j$ and δv_j for the retained and discarded eigenvalues in $-\sum_j p_j \log p_j$ and collect the two normalization terms. Since both conditional entropies lie in $[0, \log d]$,

$$|H(p) - H(q)| \le h_2(\delta) + \delta \log d =: E(\delta). \tag{25}$$

For $\delta = 0$, the discarded contribution and $E(0)$ are zero. Applying the triangle inequality to the two checkpoint spectra gives

$$|\Delta H_{\text{filtered}} - \Delta H_{\text{full}}| \le E(\delta_0) + E(\delta_1). \tag{26}$$

Thus a positive empirical full-spectrum response retains its sign if it exceeds this truncation margin. Equation 26 bounds the spectral-response error introduced by truncation.

A.8 STRUCTURAL COMPOSITION CONTROL

Proof of Lemma 1. For every represented role, the denominator in Equation (8) is positive. Summation gives

$$\sum_{t \in V_{ir}^{(b)}} w_{it}^{(b)} = \pi_{ir} \frac{\sum_{t \in V_{ir}^{(b)}} w_{it}}{\sum_{u \in V_{ir}^{(b)}} w_{iu}} = \pi_{ir}. \tag{27}$$

Summing over roles gives total mass one. To expose the effect on the measured geometry, fix one checkpoint and omit its index. Write C_r for its full-support conditional directional second moment within role r, and $C_r(V)$ for the corresponding moment over a sampled view, using normalized within-role weights. Then $C = \sum_r \pi_r C_r$. A general view with role masses $\widetilde{\pi}_r$ has $\widetilde{C}(V) = \sum_r \widetilde{\pi}_r C_r(V)$, so

$$\widetilde{C}(V) - C = \underbrace{\sum_r \pi_r [C_r(V) - C_r]}_{\text{within-role observation change}} + \underbrace{\sum_r (\widetilde{\pi}_r - \pi_r) C_r(V)}_{\text{structural composition change}}. \tag{28}$$

This identity follows by adding and subtracting $\sum_r \pi_r C_r(V)$. Because each conditional moment is PSD with unit trace, $\|C_r(V)\|_F \le 1$, and the norm of the second term is at most $\sum_r |\widetilde{\pi}_r - \pi_r|$. The proposed view weights set $\widetilde{\pi}_r = \pi_r$, eliminating this term exactly in both checkpoints. Variation can still arise from the retained within-role directions and their interaction in the joint spectrum; fixed role mass isolates that variation from changes in structural proportions.

A.9 A FINITE-VIEW ROBUSTNESS INTERPRETATION OF THE SCORE

Fix the $B \ge 2$ observed paired responses of one example, $x_b = \Delta H_i^{(b)}$, and write $\mu = B^{-1} \sum_b x_b$ and $s^2 = (B-1)^{-1} \sum_b (x_b - \mu)^2$. A hypothetical reweighting can place more emphasis on less favorable observed views. Using a chi-square neighborhood around the uniform weights (Duchi et al., 2021), define

$$\begin{aligned} \mathcal{Q}_\rho = \{ q \in \mathbb{R}^B : q_b \ge 0,\ \sum_b q_b = 1, \\ B \sum_b (q_b - 1/B)^2 \le \rho \}, \qquad \rho \ge 0. \end{aligned} \tag{29}$$

These weights are analytical stress-test weights on views of one example; they do not modify the dataset or add an optimization to the algorithm.

Proposition 1: a conservative reweighted response. For any fixed observed responses,

$$\inf_{q\in\mathcal{Q}_\rho}\sum_{b=1}^{B} q_b x_b \geq \mu - \sqrt{\rho\frac{B-1}{B}}\, s. \tag{30}$$

Consequently, with $\rho = B\beta^2/(B-1)$ and $\beta \geq 0$, the existing score $S_i = \mu - \beta s$ lower-bounds the worst average over this finite reweighting set.

Proof and equality conditions. For $q \in \mathcal{Q}_\rho$, its deviations from uniform sum to zero. Cauchy–Schwarz therefore gives

$$\begin{aligned}\sum_b q_b x_b &= \mu + \sum_b (q_b - 1/B)(x_b - \mu)\\ &\geq \mu - \left[\sum_b (q_b - 1/B)^2\right]^{1/2}\left[\sum_b (x_b-\mu)^2\right]^{1/2}\\ &\geq \mu - \sqrt{\rho/B}\sqrt{B-1}\, s,\end{aligned} \tag{31}$$

proving the bound. If $s = 0$, every reweighted average equals μ and the bound is exact. For $s > 0$ and $\rho = B\beta^2/(B-1)$, the equality candidate is

$$q_b^\star = \frac{1}{B} - \frac{\beta(x_b-\mu)}{(B-1)s}. \tag{32}$$

It sums to one and satisfies $B\sum_b (q_b^\star - 1/B)^2 = B\beta^2/(B-1) = \rho$. If every $q_b^\star \geq 0$, this candidate is feasible, attains equality in Equation (31), and yields the constrained worst average exactly. If a component is negative, the bound remains valid but is conservative. The case $\beta = 0$ gives the uniform distribution. □

This analysis explains the dispersion penalty: a more variable response can deteriorate further when unfavorable observed views receive more weight. It uses the sample-standard-deviation convention in Equation (9) and does not require independent views, a Gaussian model, or additional measurements. Repeated span signatures remain repeated entries, as in the implemented score. The result provides a deterministic bound over the observed views; it does not provide a confidence interval or a guarantee for unobserved views.

A.10 Computation and implementation

Scoring requires one full-corpus reference adaptation. Per example, M_0 evaluates the instruction-aware and code-only contexts, and M_1 evaluates the instruction-aware context. The M_0 instruction-aware forward pass can provide both logits for weighting and hidden states for geometry. All views reuse the extracted states.

For n_i eligible tokens and hidden width d, forming a full matrix of token inner products costs $O(n_i^2 d)$ time and $O(n_i^2)$ matrix storage per model. Each view can extract and reweight its submatrix. Dense eigendecomposition of a view with v_{ib} tokens costs $O(v_{ib}^3)$. Thus increasing B adds matrix computations rather than Transformer forward passes. The token-space Gram form is smaller than the feature-space form when $v_{ib} < d$; it does not eliminate the cubic cost of dense spectral decomposition.

Within a fixed backbone, tokenizer, prompt, layer, and reference-adaptation protocol, the resulting ranking can be reused across budgets and final-training seeds. For reference cost C_{ref}, scoring cost C_{score}, and R uses of the ranking, the shared overhead per use is $(C_{\mathrm{ref}} + C_{\mathrm{score}})/R$. Final training cost must still be added. Scores are recomputed for each backbone; reuse across different models is not assumed.

The mathematical specification fixes the measurement, while exact replay also requires the parser revision and node-to-role dispatch, tokenizer and prompt serialization, lexical candidate decoding, sample-ID seed derivation, retry limit, and numerical precision. These implementation artifacts, together with the reference checkpoint and training seeds, determine reproducibility beyond the parameter values in Table 3.

B Experimental Protocol and Additional Results

B.1 Data, training, and evaluation

The candidate dataset, `expanded_rtlcoder_12k`, is released with VeriCoder (Wei et al., 2025) and derives from the RTLCoder corpus. Simulation and feedback-based repair supply upstream functional supervision in addition to syntax checking. The baseline candidate pool contains 12,283 examples, of which 12,259 (99.80%) have valid Veri-eRank scores. The exact 20% budget is $K = \lceil 0.2 \times 12{,}283 \rceil = 2{,}457$. The distribution analysis uses the 12,259 supported examples.

Deduplication and near-duplicate screening. Before data selection and fine-tuning, we trim leading and trailing whitespace from the reference RTL and remove exact duplicates. Project cleaning records start from 12,351 examples: a single-file Icarus Verilog gate removes 42 that fail parsing or limited elaboration, while exact deduplication removes none. We then remove comments, lowercase the code, and construct token 3-grams. Using MinHash-LSH (Broder, 1997; Indyk & Motwani, 1998), we compare training examples with available benchmark reference code and remove 24 potential near-duplicate contaminants, leaving 12,285 examples. Length filtering removes two further examples, yielding the 12,283-example pool. Supplementary MinHash audits against the problem descriptions and testbench code of RTLLM v2 (Lu et al., 2024; Liu et al., 2024a), ResBench (Guo & Zhao, 2025), and VerilogEval v2 (Pinckney et al., 2025) detect no remaining overlap. These are static, lexical, and length checks.

Table 4: Training and evaluation settings.

Item	Setting
Final SFT epochs	3
LoRA rank	32
Optimizer	8-bit AdamW
Learning rate / weight decay	2×10^{-4} / 0.01
Effective batch size	16
Maximum generation length	1,024
GPU	NVIDIA RTX PRO 6000, 96 GB
Python / PyTorch	3.12.3 / 2.8.0
Transformers / vLLM	4.56.2 / 0.11.0

RTLLM v2 (Lu et al., 2024; Liu et al., 2024a) contains 50 designs spanning arithmetic, memory, control, and other modules.

ResBench (Guo & Zhao, 2025) contains 56 FPGA design tasks in 12 application categories.

VerilogEval v2 (Pinckney et al., 2025) contains 156 specification-to-RTL tasks derived from HDLBits.

For a task with n generated candidates and c passing candidates, the Pass@k estimator is $\mathrm{pass@}k = 1 - \binom{n-c}{k} / \binom{n}{k}$ (Chen et al., 2021). With one candidate and $k = 1$, each task has a binary outcome. For benchmark scores $p_{s,j}$ at training seed s, the macro score is $a_s = \frac{1}{3}\sum_{j=1}^{3} p_{s,j}$. We report the mean and sample standard deviation of a_s over five seeds. Micro Pass@1 pools all 262 tasks and weights benchmarks by their task counts.

B.2 Baseline implementation details

Selection protocol. We compare complementary data-selection criteria under a 20% data budget: prediction difficulty, representation coverage, gradient information, and hardware-specific quality and diversity. Each method selects training examples using its own criterion, and the resulting subsets are used for final SFT from the corresponding Base checkpoint. The scoring models and temporary adapters serve only to construct the subsets.

EL2N. Our generative adaptation of EL2N (Paul et al., 2021) measures prediction difficulty with the clean pretrained Base model. For the reference-RTL token positions T_i, it averages the distance between the next-token probability vector and the observed token's one-hot vector:

$$s_i^{\mathrm{EL2N}} = \frac{1}{|T_i|} \sum_{t \in T_i} \left\| p_{M_0}(\cdot \mid x_i, y_{i,<t}) - \mathrm{onehot}(y_{i,t}) \right\|_2 . \tag{33}$$

We score each example once before adaptation and select examples in decreasing score order. Averaging over answer tokens makes the score reflect prediction difficulty rather than total response length.

IFD. Instruction-following difficulty (Li et al., 2024) compares response likelihood with and without the instruction. Following its pre-experience procedure, we cluster prompt embeddings into 100 groups and sample up to ten examples per group from the middle question-perplexity range. One epoch of LoRA adaptation on this set produces the scoring model. We then compute $\mathrm{IFD}(x, y) = \mathrm{NLL}(y \mid x) / \mathrm{NLL}(y)$, using mean answer-token NLLs and holding the system and response formatting fixed. Examples with finite scores, positive denominator, and $\mathrm{IFD} \leq 1$ are ranked in decreasing IFD order. This prioritizes examples whose responses remain challenging when conditioned on their instructions.

Embedding-KMeans. Embedding-KMeans (Arthur & Vassilvitskii, 2007) represents each formatted instruction–RTL sequence by the mean of its non-padding final-layer hidden states from the corresponding Base model. We ℓ_2-normalize these vectors, run KMeans with 2,457 clusters, and select the real example nearest each centroid under Euclidean distance. The clustering uses k-means++ initialization, ten initializations, and at most 300 iterations. One representative per cluster provides coverage of the candidate pool in the backbone's embedding space.

SPICE. SPICE (Chang et al., 2026) selects examples by balancing gradient information gain against conflict with previously selected examples. Following the original paper's proxy-model protocol, we use Qwen2.5-Coder-0.5B for the Qwen experiments. For Seed-Coder, we use Seed-Coder-8B Base itself as the proxy because the family has no smaller counterpart; this follows the original paper's use of LLaMA2-7B as its own proxy. Per-example LoRA gradients support greedy selection within randomly partitioned pools of 120 candidates. The score combines a diagonal log-determinant information-gain approximation and a negative-cosine conflict penalty, with information scale $\alpha_{\mathrm{S}} = 1$ and conflict coefficient $\lambda_{\mathrm{S}} = 0.1$. We select 24 examples from each full pool and nine from the final pool, yielding 2,457 examples. The proxy is updated on selected examples after every ten pools and after the final partial group. All gradient computation and proxy updates are included in the reported selection time.

VerilogDB-style. Our fixed-budget adaptation of VerilogDB (Calzada et al., 2025) prioritizes reference-RTL quality using Icarus syntax and elaboration checks, Yosys synthesis checks, and top-module and port metadata completeness. These signals define ordered quality tiers: designs passing all checks receive the highest priority, followed by progressively less complete combinations. We fill the data budget in tier order. This baseline evaluates how hardware tool feedback guides training-data prioritization.

VeriSelector. VeriSelector (Shen et al., 2026) combines IFD-based ranking with structural diversity through KMeans. A prior model trained on examples sampled from 100 instruction clusters supplies IFD scores and text embeddings. The text embeddings are combined with separately standardized AST, control-flow-graph, and netlist features. Examples satisfying $\mathrm{IFD} < 1$ are grouped into 100 clusters, ranked by decreasing IFD within each cluster, and sampled for cluster c. This preserves diversity through the structural features while using instruction difficulty to order examples within each group.

B.3 Statistical Significance Tests

We compare Veri-eRank with each of the six data-selection baselines at the same 20% data budget, separately for each backbone. Two-sided paired t-tests use per-seed Macro Avg. Pass@1 scores from

the same five random seeds. Runs are paired by the same random seed, and the test is performed on the per-seed Macro Avg. Pass@1 scores. For each data-selection method, we report the mean paired gain of Veri-eRank, its 95% confidence interval (CI), the t statistic, and the p-value. The mean gain is defined as the Macro Avg. of Veri-eRank minus that of the corresponding method and we use $p < 0.05$ as the significance threshold.

Table 5: Statistical significance of the Macro Avg. Pass@1 differences between Veri-eRank and each data-selection baseline at the same 20% data budget. Mean gain and 95% CI are reported in percentage points (pp). Results are based on two-sided paired t-tests over the same five random seeds ($n = 5$, $\mathrm{df} = 4$). The p-values are reported.

Method	**Mean gain (pp)**	**95% CI (pp)**	$t(4)$	p
(a) Qwen2.5-Coder-7B Base				
SPICE	5.42	$[4.18,\ 6.66]$	12.14	0.0003
EL2N	5.54	$[2.82,\ 8.26]$	5.66	0.0048
IFD	6.39	$[4.04,\ 8.74]$	7.55	0.0016
VerilogDB-style	5.38	$[3.30,\ 7.46]$	7.17	0.0020
Embedding-KMeans	5.00	$[1.72,\ 8.29]$	4.23	0.0134
VeriSelector	9.63	$[6.73,\ 12.54]$	9.21	0.0008
(b) Seed-Coder-8B Base				
SPICE	2.24	$[0.63,\ 3.84]$	3.88	0.0179
EL2N	2.45	$[0.74,\ 4.15]$	3.99	0.0163
IFD	5.36	$[3.52,\ 7.21]$	8.07	0.0013
VerilogDB-style	1.66	$[-1.99,\ 5.32]$	1.26	0.2752
Embedding-KMeans	6.19	$[4.19,\ 8.19]$	8.60	0.0010
VeriSelector	4.51	$[1.27,\ 7.76]$	3.86	0.0181

As shown in Table 5, Veri-eRank achieves statistically significant improvements in Macro Avg. Pass@1 over all six data-selection baselines on Qwen2.5-Coder-7B and five of the six baselines on Seed-Coder-8B. The only comparison that does not reach the $p < 0.05$ threshold is VerilogDB-style on Seed-Coder-8B ($p = 0.2752$), while Veri-eRank still achieves a 1.66-pp higher mean Macro Avg. Pass@1 than VerilogDB-style on Seed-Coder-8B (53.06 vs. 51.40), as shown in Table 1.

B.4 Complete sample-budget sweep

Figure 8 and Table 6 report benchmark-level results for seven budgets from 5% to 40%, using the same five seeds at every budget. The 20% subset achieves the highest mean Pass@1 on all three benchmarks for both backbones. Relative to the neighboring 15% and 25% budgets, its macro-average gains are 4.29 and 3.40 pp on Qwen, and 3.93 and 3.20 pp on Seed-Coder, respectively.

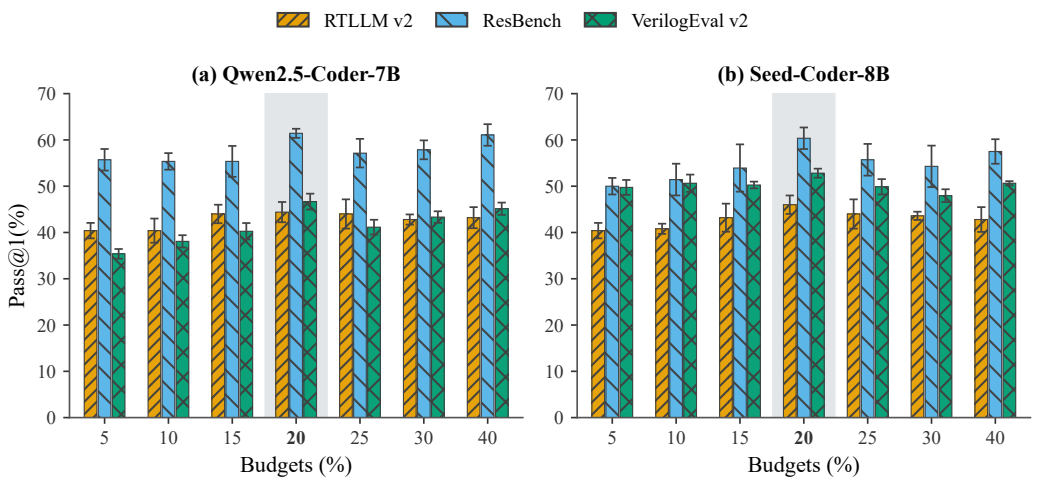


Figure 8: Benchmark-level Pass@1 across selection fractions. Bars show means over five final-training seeds, with error bars indicating ± 1 sample SD. Shading marks the default 20% fraction.

Table 6: Complete sample-budget sweep: Pass@1 (%). Macro Avg. is computed within each seed before aggregation across seeds. Bold marks the highest macro-average mean within each backbone, while shading identifies the default 20% budget.

(a) Qwen2.5-Coder-7B Base

Budget	Samples	RTLLM v2	ResBench	VerilogEval v2	Macro Avg.
5%	615	40.40 ± 1.67	55.71 ± 2.33	35.38 ± 1.05	43.83 ± 0.92
10%	1,229	40.40 ± 2.61	55.36 ± 1.79	38.08 ± 1.33	44.61 ± 1.21
15%	1,843	44.00 ± 2.00	55.36 ± 3.34	40.26 ± 1.78	46.54 ± 0.89
20%	2,457	44.40 ± 2.19	61.43 ± 0.98	46.67 ± 1.72	**50.83** ± 1.14
25%	3,071	44.00 ± 3.16	57.14 ± 3.09	41.15 ± 1.60	47.43 ± 1.45
30%	3,685	42.80 ± 1.10	57.86 ± 2.04	43.33 ± 1.25	48.00 ± 0.80
40%	4,914	43.20 ± 2.28	61.07 ± 2.33	45.13 ± 1.33	49.80 ± 1.51

(b) Seed-Coder-8B Base

Budget	Samples	RTLLM v2	ResBench	VerilogEval v2	Macro Avg.
5%	615	40.40 ± 1.67	50.00 ± 1.79	49.74 ± 1.61	46.71 ± 1.19
10%	1,229	40.80 ± 1.10	51.43 ± 3.43	50.64 ± 1.87	47.62 ± 1.42
15%	1,843	43.20 ± 3.03	53.93 ± 5.11	50.26 ± 0.73	49.13 ± 2.27
20%	2,457	46.00 ± 2.00	60.36 ± 2.33	52.82 ± 0.97	**53.06** ± 0.57
25%	3,071	44.00 ± 3.16	55.71 ± 3.43	49.87 ± 1.66	49.86 ± 1.59
30%	3,685	43.60 ± 0.89	54.29 ± 4.48	47.95 ± 1.39	48.61 ± 2.00
40%	4,914	42.80 ± 2.68	57.50 ± 2.65	50.64 ± 0.45	50.31 ± 1.13

B.5 Selection cost and performance

Figure 9 gives the Qwen cost–performance comparison, complementing the Seed-Coder comparison in Figure 3(b). Tables 7 and 8 report the complete numerical results for the two backbones. Performance values are the macro averages in Table 1. In both figures, the horizontal coordinate is $x = t/131$, where t is selection time in minutes; 131 minutes is the largest measured selection time across the two backbones.

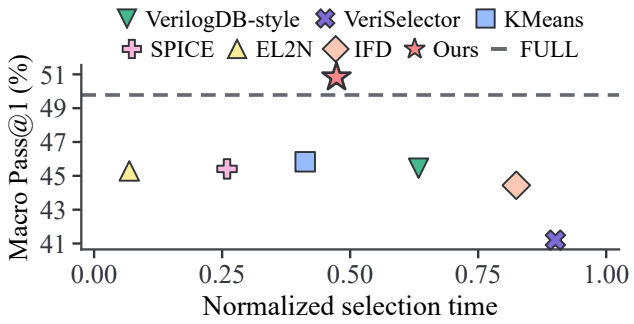


Figure 9: Selection cost and macro-average Pass@1 on Qwen2.5-Coder-7B Base under a 20% data budget, using the same time normalization as Figure 3(b). SPICE uses a 0.5B Qwen proxy following its original protocol. The dashed line gives Full SFT accuracy. KMeans denotes Embedding-KMeans; Ours denotes Veri-eRank. Table 7 gives the exact values.

On Qwen, Veri-eRank achieves 50.83 macro-average Pass@1 with 62 minutes of selection. It improves on the 0.5B-proxy SPICE configuration by 5.42 pp and exceeds Full SFT by 1.05 pp. It achieves higher accuracy in less selection time than IFD, VerilogDB-style, and VeriSelector. Relative to VerilogDB-style, it gains 5.38 pp while reducing selection time from 83 to 62 minutes (25.3%); relative to VeriSelector, it gains 9.63 pp while reducing selection time from 118 to 62 minutes (47.5%). Together with the Seed-Coder results, this places Veri-eRank on the accuracy–cost Pareto frontier of the compared methods under a 20% data budget for both backbones.

The proxy protocol follows Appendix B.2: SPICE uses a 0.5B Qwen proxy for Qwen and the 8B model itself for Seed-Coder. All reported Veri-eRank times include one-epoch reference adaptation and candidate scoring. The resulting ranking is reused when constructing additional subset budgets. Final fine-tuning and functional evaluation are excluded from selection time. A zero selection cost for Full SFT denotes the absence of subset construction; it does not include its full-data training cost.

Table 7: Selection cost and performance on Qwen2.5-Coder-7B Base. Selection methods are evaluated under a 20% data budget. Full SFT uses 100%. Time is in hours:minutes; performance is macro-average Pass@1 (%). SPICE uses a 0.5B Qwen proxy. For Veri-eRank, time includes reference adaptation and scoring. Final SFT and evaluation are excluded.

Method	Selection time (h:mm)	Macro Avg. (%)
Full SFT (100%)	0:00	49.78
Embedding-KMeans	0:54	45.83
EL2N	0:09	45.29
IFD	1:48	44.44
SPICE	0:34	45.41
VerilogDB-style	1:23	45.45
VeriSelector	1:58	41.20
Veri-eRank	1:02	**50.83**

Table 8: Selection cost and performance on Seed-Coder-8B Base. Selection methods are evaluated under a 20% data budget. Full SFT uses 100%. Time is in hours:minutes; performance is macro-average Pass@1 (%). SPICE uses Seed-Coder-8B Base itself as the proxy. For Veri-eRank, time includes reference adaptation and scoring. Final SFT and evaluation are excluded.

Method	Selection time (h:mm)	Macro Avg. (%)
Full SFT (100%)	0:00	51.08
Embedding-KMeans	1:06	46.87
EL2N	0:11	50.61
IFD	1:53	47.70
SPICE	1:23	50.82
VerilogDB-style	1:35	51.40
VeriSelector	2:11	48.55
Veri-eRank	1:16	**53.06**

B.6 Response signals, structural observations, and stability

We expand the three design choices examined in Section 3.4: which adaptation signal to use, which structural observations to measure, and how to aggregate responses across views. Every five-seed configuration selects 2,457 examples (20%) and uses the shared final-SFT protocol from a fresh base checkpoint. The reference model is used only for scoring, never to initialize final SFT.

Reporting convention. For benchmark percentages $P_{c,s,j}$, the run-level macro score is $A_{c,s} = \frac{1}{3}\sum_{j=1}^{3} P_{c,s,j}$. Tables report the mean and sample SD of $A_{c,s}$ across five final-training seeds; benchmark SDs are not averaged to obtain macro SD. Figure 4(a) uses mean $\pm$ SD, whereas panel (b) reports differences of means without inferential error bars. Paired intervals require the run-level differences, as in Appendix B.3; they cannot be recovered from marginal SDs alone.

B.6.1 Can alternative adaptation signals replace the structural score?

Two direct response signals. Vanilla Spectral ΔH uses all eligible assistant-code token observations, uniform weights, checkpoint-specific centering, and row-wise normalization. Both checkpoints see the same instruction–reference-code input. No AST support, resampling, role preservation, or dispersion penalty is used. Spectra follow the common absolute and condition-number cutoffs. With $G^{\text{code}}_{m,i}$ denoting this full-code Gram matrix,

$$S_i^{\text{vanilla}} = H(G^{\text{code}}_{0,i}) - H(G^{\text{code}}_{1,i}). \tag{34}$$

Answer-token ΔNLL uses the same fixed model pair but ranks the mean loss reduction on reference-code targets T_i^{ans}:

$$\begin{aligned} L_{m,i} &= -\frac{1}{|T_i^{\text{ans}}|}\sum_{t\in T_i^{\text{ans}}} \log p_{M_m}(y_{i,t} \mid x_i, y_{i,<t}), \\ S_i^{\Delta\text{NLL}} &= L_{0,i} - L_{1,i}. \end{aligned} \tag{35}$$

The instruction remains in the context, but prompt, format-marker, and padding targets are excluded from this average. Probabilities use the full vocabulary and correctly shifted causal logits, not truncated role probabilities or training-trajectory losses. Final SFT retains the common full-sequence loss.

Effective-rank control. We adapt the effective-rank difference of Diff-eRank (Wei et al., 2024) from model evaluation to fixed-budget sample ranking, using the same pretrained M_0 and reference-adapted M_1. Unlike Vanilla ΔH, this control uses the full formatted instruction–answer sequence, including template tokens. Positions with non-finite states or near-zero centered norms in either checkpoint are jointly excluded. On the remaining common support of N_i non-padding tokens, each checkpoint's final-layer states are mean-centered and row-wise ℓ_2-normalized to form $\mathbf{Z}_{m,i}$. We define $\Sigma_{m,i} = \mathbf{Z}_{m,i}^{\top}\mathbf{Z}_{m,i}/N_i$ and compute $H(\Sigma_{m,i})$ from eigenvalues above 10^{-12}, renormalized

to sum to one, without a condition-number cutoff. The ranking score is

$$S_i^{\Delta\mathrm{eRank}} = R_{0,i} - R_{1,i}, \qquad R_{m,i} = \mathrm{eRank}(\Sigma_{m,i}) = \exp H(\Sigma_{m,i}). \tag{36}$$

Because this control follows the Diff-eRank-style token scope and spectral processing, it tests the complete effective-rank-based alternative under the same selection budget.

Table 9: Alternative adaptation signals under a 20% budget. Values are five-seed mean ± SD of Pass@1 (%). Diff-eRank-based selection is the formula adaptation described in the text. Bold macro entries identify the full-method reference; they do not mark statistical significance or dominance on every benchmark.

(a) Qwen2.5-Coder-7B Base

Configuration	RTLLM v2	ResBench	VerilogEval v2	Macro Avg.
Answer-token ΔNLL	34.80 ± 1.79	53.93 ± 0.80	46.28 ± 1.23	45.00 ± 0.89
Vanilla Spectral ΔH	42.40 ± 2.19	57.50 ± 1.49	38.59 ± 2.19	46.16 ± 1.17
Diff-eRank-based selection	38.80 ± 4.15	60.00 ± 3.48	38.46 ± 3.04	45.75 ± 3.05
Veri-eRank	44.40 ± 2.19	61.43 ± 0.98	46.67 ± 1.72	$\mathbf{50.83 \pm 1.14}$

(b) Seed-Coder-8B Base

Configuration	RTLLM v2	ResBench	VerilogEval v2	Macro Avg.
Answer-token ΔNLL	46.00 ± 2.45	51.79 ± 3.34	53.08 ± 1.53	50.29 ± 1.10
Vanilla Spectral ΔH	41.20 ± 1.10	60.71 ± 1.79	50.77 ± 1.05	50.89 ± 0.79
Diff-eRank-based selection	42.80 ± 2.68	54.64 ± 3.24	48.85 ± 0.54	48.76 ± 1.20
Veri-eRank	46.00 ± 2.00	60.36 ± 2.33	52.82 ± 0.97	$\mathbf{53.06 \pm 0.57}$

The full method improves macro-average Pass@1 by 4.67/2.17 pp over Vanilla ΔH, 5.83/2.77 pp over ΔNLL, and 5.08/4.30 pp over the effective-rank control on Qwen/Seed-Coder, respectively.

B.6.2 What must be controlled in a structural observation?

This study separates relevance weighting from view construction. All hidden states come from the intact instruction–RTL sequence: subsampling changes which cached observations enter the geometry, not the input program.

Relevance and granularity. Uniform weights, originally *w/o Semantic Weight*, replace instruction-conditioned weights while retaining the structural-view protocol. Token-level resampling selects individual token observations rather than whole AST spans. These controls separately test the observation distribution and the perturbation granularity.

Composition and pairing. Without role preservation removes the fixed role-mass constraint across views. Independent checkpoint views, originally *w/o Paired View*, sample the two checkpoints independently. In the full protocol, preserving role masses removes the composition-change term described in Appendix A.8, while pairing keeps content and weights identical within each checkpoint comparison. The variance identity in Appendix A.5 further shows that pairing reduces response variance when checkpoint entropies are positively correlated.

The Qwen/Seed-Coder macro reductions are 3.25/2.05 pp for uniform weights, 2.44/1.83 pp for token-level resampling, 3.72/2.30 pp without role preservation, and 5.64/3.26 pp without pairing. On Qwen, the pairing comparison is especially pronounced on ResBench (61.43 versus 52.50). These results support defining both relevance and comparability in the observation protocol. They are local replacements relative to the full method, not additive component effects; for example, uniform weighting leaves Seed-Coder's ResBench mean unchanged.

Table 10: Instruction-conditioned and comparable structural observations. Each replacement is made relative to the full configuration. Results are five-seed mean ± SD of Pass@1 (%). Bold macro entries identify the reference. These are separate controlled replacements, not steps of an additive improvement chain.

(a) Qwen2.5-Coder-7B Base

Configuration	RTLLM v2	ResBench	VerilogEval v2	Macro Avg.
Uniform weights	42.00 ± 2.00	60.36 ± 4.45	40.38 ± 1.36	47.58 ± 1.98
Token-level resampling	44.00 ± 3.16	59.64 ± 2.40	41.54 ± 1.39	48.39 ± 1.65
Without role preservation	42.00 ± 2.45	58.57 ± 2.65	40.77 ± 0.97	47.11 ± 0.69
Independent checkpoint views	40.00 ± 2.83	52.50 ± 2.71	43.08 ± 2.54	45.19 ± 2.02
Veri-eRank	44.40 ± 2.19	61.43 ± 0.98	46.67 ± 1.72	$\mathbf{50.83 \pm 1.14}$

(b) Seed-Coder-8B Base

Configuration	RTLLM v2	ResBench	VerilogEval v2	Macro Avg.
Uniform weights	43.20 ± 3.03	60.36 ± 2.65	49.49 ± 2.10	51.01 ± 1.57
Token-level resampling	45.20 ± 2.28	57.86 ± 2.40	50.64 ± 0.91	51.23 ± 1.22
Without role preservation	42.40 ± 1.67	58.21 ± 0.98	51.67 ± 1.07	50.76 ± 1.03
Independent checkpoint views	41.60 ± 1.67	56.79 ± 0.80	51.03 ± 0.73	49.80 ± 0.73
Veri-eRank	46.00 ± 2.00	60.36 ± 2.33	52.82 ± 0.97	$\mathbf{53.06 \pm 0.57}$

B.6.3 Why aggregate response strength and view consistency?

The four rules below separate a complete observation, a multi-view mean, dispersion alone, and their combination:

$$\begin{aligned} S_i^{\text{single}} &= \Delta H_i^{\text{full}}, \quad S_i^{\text{mean}} = \widehat{\mu}_i, \\ S_i^{\text{stable}} &= -\widehat{\sigma}_i, \qquad\quad S_i^{\text{full}} = \widehat{\mu}_i - 0.5\widehat{\sigma}_i. \end{aligned} \tag{37}$$

Single full structural view is the original *w/o Resampling*; it retains AST-owned tokens and instruction-conditioned weights, unlike Vanilla all-code ΔH. Mean-only and Stability-only reuse the full method's paired span views and change only their aggregation. A single complete view is not the multi-view mean. Signed scores are retained; Stability-only does not additionally require a positive mean response.

Table 11: Single-view scoring and one-term rankings. Values are five-seed mean ± SD of Pass@1 (%). Mean-only uses $\beta = 0$ on the same paired views as the full score. Bold macro entries identify the full-method reference.

(a) Qwen2.5-Coder-7B Base

Configuration	RTLLM v2	ResBench	VerilogEval v2	Macro Avg.
Single full structural view	42.80 ± 2.28	58.21 ± 2.71	41.03 ± 0.64	47.35 ± 0.62
Mean-only ($\beta = 0$)	41.60 ± 2.97	56.07 ± 2.04	40.00 ± 2.15	45.89 ± 1.38
Stability-only ($-\widehat{\sigma}$)	38.00 ± 2.00	61.43 ± 2.40	43.85 ± 1.67	47.76 ± 1.58
Veri-eRank	44.40 ± 2.19	61.43 ± 0.98	46.67 ± 1.72	$\mathbf{50.83 \pm 1.14}$

(b) Seed-Coder-8B Base

Configuration	RTLLM v2	ResBench	VerilogEval v2	Macro Avg.
Single full structural view	44.80 ± 1.10	53.57 ± 2.53	51.92 ± 1.11	50.10 ± 0.66
Mean-only ($\beta = 0$)	44.00 ± 1.41	52.86 ± 2.71	52.05 ± 1.46	49.64 ± 0.98
Stability-only ($-\widehat{\sigma}$)	41.60 ± 2.97	58.21 ± 0.98	48.72 ± 2.40	49.51 ± 1.46
Veri-eRank	46.00 ± 2.00	60.36 ± 2.33	52.82 ± 0.97	$\mathbf{53.06 \pm 0.57}$

Table 11 shows that the full score exceeds the single-view control by 3.48/2.96 pp on Qwen/Seed-Coder, testing the multi-view mechanism as a whole. Mean-only reaches 45.89 ± 1.38 on Qwen and 49.64 ± 0.98 on Seed-Coder; adding the dispersion penalty increases the macro means by 4.94/3.42 pp. This comparison keeps the structural views fixed and isolates their aggregation. Conversely, Stability-only reaches 47.76/49.51, below the complete score by 3.07/3.55 pp. The complete score thus achieves higher macro means than either one-term ranking on both backbones.

Proposition 1 interprets the dispersion penalty as robustness to bounded reweighting of the observed structural views. This view-level dispersion is distinct from cross-seed training variability.

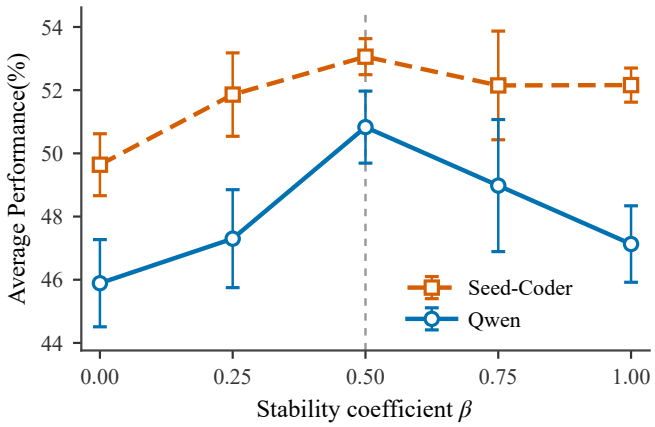


Figure 10: Stability coefficient. Macro-average Pass@1 is reported as mean $\pm$ SD over five seeds; the vertical line marks the default $\beta = 0.5$.

Table 12: Stability-coefficient sweep under a 20% data budget. Values are five-seed mean $\pm$ SD of Pass@1 (%); bold marks the highest macro average for each backbone.

(a) Qwen2.5-Coder-7B Base

β	RTLLM v2	ResBench	VerilogEval v2	Macro Avg.
0	41.60 ± 2.97	56.07 ± 2.04	40.00 ± 2.15	45.89 ± 1.38
0.25	42.40 ± 2.19	58.21 ± 4.11	41.28 ± 1.07	47.30 ± 1.55
0.50	44.40 ± 2.19	61.43 ± 0.98	46.67 ± 1.72	$\mathbf{50.83 \pm 1.14}$
0.75	46.00 ± 4.69	59.29 ± 2.93	41.67 ± 1.36	48.98 ± 2.09
1.00	42.40 ± 2.61	58.21 ± 2.99	40.77 ± 1.61	47.13 ± 1.21

(b) Seed-Coder-8B Base

β	RTLLM v2	ResBench	VerilogEval v2	Macro Avg.
0	44.00 ± 1.41	52.86 ± 2.71	52.05 ± 1.46	49.64 ± 0.98
0.25	45.60 ± 2.97	56.79 ± 2.33	53.21 ± 1.01	51.86 ± 1.32
0.50	46.00 ± 2.00	60.36 ± 2.33	52.82 ± 0.97	$\mathbf{53.06 \pm 0.57}$
0.75	44.00 ± 2.45	60.00 ± 2.71	52.44 ± 1.94	52.15 ± 1.72
1.00	43.60 ± 2.19	60.71 ± 1.79	52.18 ± 1.73	52.16 ± 0.54

Strength of the stability penalty. We subsequently vary β to assess sensitivity to the strength of the stability penalty, retaining $\beta = 0.5$ as the common default. All tested positive coefficients yield higher five-seed mean macro-average Pass@1 than $\beta = 0$ on both backbones. At the pre-specified default of $\beta = 0.5$, both backbones attain their highest mean macro-average Pass@1 within the tested range, improving by 4.94 pp on Qwen and 3.42 pp on Seed-Coder over the unpenalized score (Table 12), supporting incorporating cross-view consistency into the ranking score.

Taken together, these studies connect the response signal, structural observation protocol, and ranking rule to downstream selection performance.

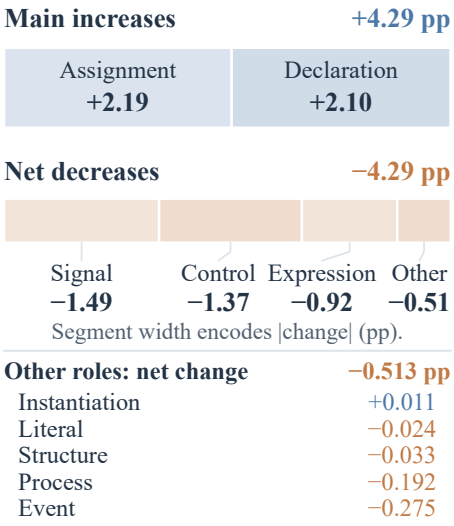
B.7 SELECTED-DISTRIBUTION ANALYSIS

Figure 5 in the main text presents task-category enrichment and role-weight shifts together. This appendix gives their aggregation definitions and the parsing diagnostic.

Task	Ref. → Sel. Share (%)	E_c [95% range]
Hier. system	8.56 → 10.01	**+0.226** [0.08, 0.40]
Comb. logic	16.11 → 18.72	**+0.216** [0.12, 0.33]
Protocol intf.	7.74 → 8.99	**+0.216** [0.08, 0.36]
Seq. logic	31.87 → 32.19	**+0.014** [−0.05, 0.08]
Mem. buffer	10.27 → 9.04	**−0.184** [−0.30, −0.05]
Arith. datapath	18.32 → 15.34	**−0.255** [−0.35, −0.16]
FSM control	7.13 → 5.70	**−0.322** [−0.47, −0.17]

$E_c = \log_2(\text{Selected/Reference})$

(a) Task-composition details



(b) Structural-role decomposition

Figure 11: Details of selected-supervision shifts on Qwen. (a) Task shares and enrichment with empirical 95% ranges from matched random subsets. (b) Decomposition of structural-role-weight share changes (pp).

The comparison uses one selected subset ($B_* = 1$) and $B_{\text{ref}} = 1{,}000$ matched subsets, each with $K = 2{,}457$ examples. Matching uses token length, AST span count, module count, and local AST parsing status. For category c and selection distribution s, define

$$p_c^{(s)} = \frac{1}{B_s} \sum_{b=1}^{B_s} \frac{N_c(D_b^{(s)})}{|D_b^{(s)}|}, \qquad \epsilon = \frac{0.5}{2{,}457}. \tag{38}$$

Here $N_c(D)$ counts examples assigned to category c; these fractions enter Equation (12). The empirical 95% intervals in Figure 5(a) use quantiles from the matched subsets. Six of the seven reported categories differ from their matched reference intervals; sequential logic remains approximately unchanged. Figure 11(a) reports the reference and selected task shares, together with enrichment values and their empirical 95% ranges.

Let $\Omega_r(D)$ sum the within-example normalized semantic weights assigned to role r. Each example contributes the sum of its normalized token weights within that role. The average share used in Equation (13) is

$$q_r^{(s)} = \frac{1}{B_s} \sum_{b=1}^{B_s} \frac{\Omega_r(D_b^{(s)})}{|D_b^{(s)}|}. \tag{39}$$

Because role weights are normalized, a larger share for one role reduces the share available to others. The declaration and assignment increases of 2.10 and 2.19 pp are rounded separately; their combined share rises from 64.98% to 69.26%. Figure 11(b) provides a grouped decomposition of the role-weight shifts and a breakdown of the roles aggregated as *Other*.

Parsing diagnostic. The mean fallback ratio is 3.75% for the selected examples and 4.09% for the matched reference. The generic `structure` role remains close to its reference share in Figure 5(b).

B.8 CAPABILITY-STRATIFIED EVALUATION

The Macro Avg. column in Table 1 weights the three benchmarks equally. Here, micro-averaged Pass@1 is the percentage of tasks passed within each capability category, with all 262 tasks pooled for the Overall row. This gives each task equal weight, directly measuring capability-specific success across the source benchmarks.

Table 13: Qwen capability-level comparison against SPICE. Each row reports the percentage of tasks passed in the indicated group (micro-averaged Pass@1). Categories with at least 40 tasks are shown; the Overall row pools all 262 tasks, including those from smaller categories. Gain is the Veri-eRank–SPICE difference in percentage points. Results use the same single final-training seed for both methods, one of the five seeds used in Table 1, rather than averages across seeds.

Capability	Tasks	Veri-eRank	SPICE	Gain
Overall	262	48.85%	45.42%	+3.44
Combinational logic	108	71.30%	64.81%	+6.48
Arithmetic datapath	43	41.86%	41.86%	+0.00
Sequential logic	52	42.31%	40.38%	+1.92
FSM/control	46	15.22%	13.04%	+2.17

As shown in Table 13, across all 262 held-out tasks, Veri-eRank solves 128 tasks compared with 119 for SPICE, increasing micro-averaged Pass@1 from 45.42% to 48.85% (+3.44 percentage points). This corresponds to a net gain of nine solved tasks.

For an evaluation category $\mathcal{T}_c$, we define the gain as $G_c = \mathrm{Pass@1}_{\text{Veri-eRank}}(\mathcal{T}_c) - \mathrm{Pass@1}_{\text{SPICE}}(\mathcal{T}_c)$, expressed in percentage points. The four displayed categories cover 249 of the 262 tasks. Veri-eRank matches or exceeds SPICE in all four, with gains of 6.48 pp in combinational logic, 1.92 pp in sequential logic, and 2.17 pp in FSM/control, while maintaining arithmetic-datapath performance.