

Responsibility-Aware Scheduling for Selective Rationalization

Dengchao Zhang^{1,2*}, Yunxiao Zhao^{1,2*†}, Zhichao Yan^{1,2}, Yue Fan³,
Shaoru Guo^{1,2}, and Ru Li^{1,2†}

1. School of Computer and Information Technology, Shanxi University, Taiyuan, China
2. Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, Shanxi University, Taiyuan, China
3. School of Computer Science and Technology, North University of China, Taiyuan, China
13546258147@163.com, yunxiaomr@163.com, yanzhichao@sxu.edu.cn
yuefan24@163.com, {gsr, liru}@sxu.edu.cn

Abstract

Select-then-predict rationalizers expose compact rationale-conditioned predictions, but their generator and predictor may interlock: low-quality rationales mislead the predictor, while delayed predictor adaptation or shortcut learning sends noisy feedback to the generator. Existing studies reduce this interaction by separating generator search from predictor training, but repeated predictor training increases training cost. To address this problem, we propose ReSR (Responsibility-Aware Scheduling for Selective Rationalization), a training-control framework that keeps the standard select-then-predict path while adaptively scheduling generator and predictor updates. ReSR uses online, probe, and reference diagnostics with rationale-structure signals to determine whether suboptimal performance arises from low-quality rationales or delayed predictor adaptation. A virtual-runtime scheduler then converts this attribution into generator and predictor debts and allocates the next optimization slice. Experiments on Toy and HateXplain indicate that ReSR improves the accuracy-rationale quality-compactness-cost balance: ReSR achieves the best Toy highlight F1 and performance comparable to the state-of-the-art approach on HateXplain highlight F1, while attaining stronger classification performance and lower cost. Our code is available at [ReSR](#).

1 Introduction

Selective rationalization aims to make model predictions interpretable by selecting short text spans that support the predicted label (Liu et al., 2025a; Zhao et al., 2024; Lei et al., 2016). A common formulation is the Rationalizing Neural Predictions (RNP) select-then-predict framework: a generator selects a rationale and a predictor makes the decision from the selected tokens (Yu et al., 2019;

*Equal contribution.

†Corresponding author.

	Cif-F1	HI-F1	R/S	Eff.	
Toy					Normalized score
FR	98.1	58.9	6.8	100.0	
MGR	54.7	15.8	1.7	96.7	
MCD	100.0	92.4	0.0	99.8	
G-RAT	49.1	0.0	7.7	93.1	
GenSPP	0.0	100.0	100.0	0.0	
HateXplain					
FR	86.5	31.6	1.1	99.8	
MGR	90.4	0.0	1.2	99.2	
MCD	15.7	36.4	3.8	100.0	
G-RAT	100.0	66.2	0.8	98.6	
GenSPP	0.0	100.0	100.0	0.0	

Figure 1: Baseline trade-off motivating ReSR. Each cell shows a dataset-wise min-max normalized score for prior baselines only. Darker is better. Cif-F1 and HI-F1 are normalized directly, while R/S and efficiency are reversed so that higher scores indicate more compact rationales and lower training cost. Raw metrics are in Table 3 and Table 5.

Liu et al., 2022). This decomposition makes prediction rationale-conditioned and sparsity explicit, but induces fragile dynamics: the generator relies on predictor feedback, while the predictor trains on the generator’s current masked-input distribution. However, this dependence may lead to interlocking (Yu et al., 2021). Rapidly changing masks induce a nonstationary masked-input distribution, whereas early adaptation to spurious correlations (Liu et al., 2025c) may reinforce masks that preserve predictive accuracy without improving rationale. Existing fixed-schedule approaches address such symptoms with shared encoders (Liu et al., 2022), multiple generators (Liu et al., 2023a), causal or complement-based constraints (Liu et al., 2023b), and guidance modules (Hu and Yu, 2024),

but still train the generator and predictor under a largely fixed update schedule.

Predictive sufficiency is not equivalent to rationale quality. A short rationale may support the correct label while omitting human-plausible rationales or relying on spurious correlations. Prior work has studied this gap through distribution mismatch between rationale-only and full-input behavior (Liu et al., 2024; Huang et al., 2021), diminishing returns in maximum mutual information (MMI) objectives once a partial rationale is selected (Liu et al., 2025b). These findings motivate evaluation beyond task accuracy, including rationale quality, compactness, and training cost, with aggregate scores used only as secondary summaries.

Recently, GenSPP (Ruggeri and Signorelli, 2025) offers a direct way to reduce generator-predictor interlocking by searching over candidate generators and evaluating them with separately trained predictors. This fully disjoint design limits the reinforcement of early mistakes, but population-level search requires repeated predictor training over many candidates and generations. Figure 1 summarizes a challenging trade-off: the strongest normalized scores fall on different methods rather than a single balanced baseline. Fixed-schedule baselines such as FR (Liu et al., 2022), MGR (Liu et al., 2023a), MCD (Liu et al., 2023b), and G-RAT (Hu and Yu, 2024) are efficient but often weak in rationale quality. GenSPP (Ruggeri and Signorelli, 2025) can produce strong or very compact rationales, but at substantially higher training cost and with weaker classification on HateXplain. We ask whether the efficient RNP inference path can be kept while better balancing these objectives.

To address this problem, we propose **ReSR** (Responsibility-Aware Scheduling for Selective Rationalization), a training-control framework that treats selective rationalization as responsibility attribution followed by update allocation. Rather than replacing gradient learning with global search, ReSR estimates whether suboptimal validation performance is more likely attributable to low-quality rationales or delayed predictor adaptation. Online, probe, and reference diagnostics are combined with rationale-structure signals to support this responsibility attribution. Inspired by operating system scheduling (Ramamritham and Stankovic, 1994), a virtual-runtime scheduler (VRSR) maps the resulting attribution to generator and predictor debts and allocates the next optimization slice. In addition, confidence-aware token guidance supports

generator updates during training and is removed at inference time, leaving the standard generator-rationale-predictor pathway unchanged.

Experiments on Toy (Ruggeri and Signorelli, 2025) and HateXplain (Mathew et al., 2021) indicate that ReSR improves the overall accuracy-rationale quality-compactness-cost balance rather than dominating every raw metric. On Toy, ReSR achieves the strongest highlight F1 with moderate rationale size. On HateXplain, GenSPP remains strongest in highlight F1 and compactness, whereas ReSR gives the best classification, higher rationale quality and greater compactness than fixed-schedule baselines, and much lower cost than GenSPP. Ablations identify adaptive scheduling and full-input reference anchoring as the main contributors, with token guidance and probe diagnostics providing complementary stabilization. Our contributions can be summarized as follows:

- We propose ReSR, a responsibility-aware training-control framework that allocates generator and predictor updates during selective rationalization.
- We instantiate ReSR with VRSR, which converts online, probe, reference, and rationale-structure diagnostics into update debts.
- Experiments show stronger accuracy-rationale quality-compactness balance on Toy and HateXplain with far lower cost than GenSPP.

2 Related Work

Selective Rationalization Selective rationalization explains a prediction by selecting a compact subset of the input as the rationale (Liu et al., 2026; Zhao et al., 2026; Yue et al., 2025), and has attracted growing attention across applications such as question answering (Xu et al., 2025; Fan et al., 2025; Jiang et al., 2026), document optimization for Retrieval-Augmented Generation (RAG) (Yan et al., 2025; Wei et al., 2026; Yan et al., 2026) and logic reasoning (Fan et al., 2026), recommendation systems (Deng et al., 2023), social network analysis (Yue et al., 2025), and image understanding (Yuan et al., 2022). Unlike post-hoc generated explanation (Zhao et al., 2025) or evaluation (Tan et al., 2021), the select-then-predict framework is the standard formulation: a generator selects a latent mask and a predictor makes the final decision from the masked input (Lei et al., 2016). This architecture makes predictions rationale-conditioned

and supports sparsity and continuity regularization, but tightly couples the generator and predictor: the generator learns through predictor feedback, while the predictor adapts to the generator’s current rationales. This coupling can cause degenerate masks, shortcut rationales, and training imbalance. Human-rationale benchmarks such as ERASER (DeYoung et al., 2020) and HateXplain (Mathew et al., 2021) further show that predictive sufficiency is not enough. Rationales should also match the spans humans regard as explanatory.

Fixed-schedule Rationalizers Most recent work keeps the RNP backbone and changes the architecture, objective, or regularization. FR folds generator and predictor encoders for information sharing (Liu et al., 2022); MGR uses multiple generators for diversity (Liu et al., 2023a) and MCD adds conditional-dependence constraints to reduce spurious rationales (Liu et al., 2023b). Complementary or invariant views also reduce shortcut reliance (Yu et al., 2019; Chang et al., 2020), while G-RAT guides hard-rationale learning with a soft rationale pathway (Hu and Yu, 2024). These methods address weak information flow, limited exploration, and spurious token selection, but still use largely fixed objectives or update rhythms. They do not decide whether the current error comes from poor generated rationales, predictor lag, or label information outside the selected rationale.

Guidance-schedule Rationalizers Several studies argue that task loss alone is insufficient for rationale quality. (Huang et al., 2021) propose preserving full-input feature and output behavior, rather than merely supporting the correct label, motivating rationale–full-input representation comparisons. (Hu and Yu, 2024) show that full-input or soft-selection guidance improves rationale learning, albeit within a fixed training loop. (Zhao et al., 2024) propose AGR, a causal-agent-guided framework that models external signals for calibrating rationale, highlighting auxiliary guidance beyond rationalization. Breaking free from MMI shows that once partial rationales suffice for classification, MMI objectives may yield diminishing returns and miss human-relevant rationales (Liu et al., 2025b). Together, these findings motivate evaluating rationales for predictive utility, compactness, full-input alignment, and human consistency.

Search-schedule Rationalizers Recently, GenSPP studies the interlocking problem directly (Rug-

geri and Signorelli, 2025). GenSPP breaks the coupled gradient loop by separating generator search from predictor training: candidate generators are evaluated with separately trained predictors, and genetic search explores the generator space. This fully disjoint design reduces the risk that generator and predictor reinforce early mistakes, making GenSPP a strong decoupled-rationalization baseline. The computational cost is high, since candidate evaluation repeats predictor training across individuals and generations.

Positioning of ReSR. ReSR treats the training loop as the object of control. Unlike previous fixed-schedule and guidance-schedule approaches, which improve the generator, predictor, or objective under fixed update schedules, ReSR uses responsibility diagnostics to choose the next update side. Unlike search-schedule GenSPP, which decouples through search and repeated predictor training, ReSR keeps the standard RNP inference pathway and performs lightweight training-time decoupling through scheduling. Online, probe, and reference diagnostics with rationale-structure signals estimate whether failure comes from poor rationale generation, predictor lag, or an unstable mask. VRSR then allocates the next optimization slice. Confidence-aware token guidance is training-only. Thus, ReSR lies between fixed-schedule select-then-predict rationalizers and fully disjoint search-based decoupling.

3 ReSR: Our Proposed Method

Let G_{θ_G} and P_{θ_P} denote the rationale generator and predictor parameterized by θ_G and θ_P , respectively. Given a sample $x = (x_1, \dots, x_n)$ with n tokens, the generator produces token-level scores $\pi = G_{\theta_G}(x)$, which are then binarized into a rationale mask,

$$m = \text{HARDMASK}(\pi). \quad (1)$$

The rationale is then extracted by applying the binary mask to the input, retaining selected tokens while masking out the remaining ones:

$$r = x \odot m. \quad (2)$$

Moreover, the predictor makes the final prediction solely based on the selected rationale, thereby explicitly conditioning the prediction on the rationale identified by the generator:

$$\hat{y} = P_{\theta_P}(r). \quad (3)$$

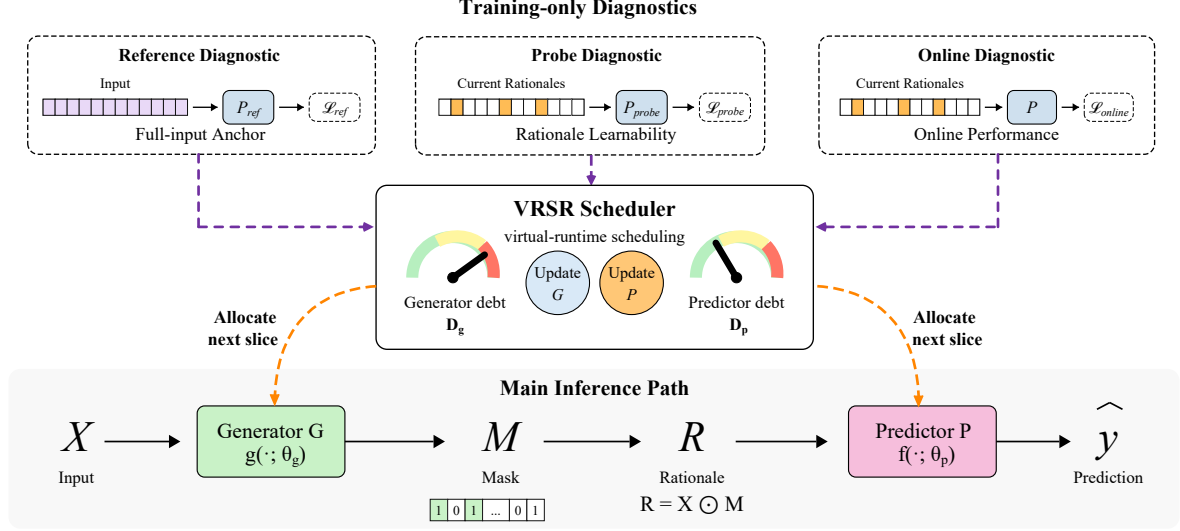


Figure 2: Overview of ReSR. The inference path remains the standard generator-rationale-predictor pipeline. During training, diagnostic branches estimate generator-predictor imbalance and feed VRSR, which allocates the next optimization slice to the generator or predictor. Solid black arrows denote the inference path, purple dashed arrows denote training-only diagnostics, and orange dashed arrows denote scheduler-controlled update allocation. The generator-side token guidance used during training is omitted for clarity.

Inference uses only this select-then-predict path. The reference predictor, probe predictor, VRSR scheduler, and token guidance are training-only.

The core difficulty is responsibility ambiguity: low-quality rationales may mislead the predictor, while predictor lag can make useful rationales look ineffective. ReSR keeps RNP training but controls the update loop. After each slice, ReSR estimates whether the current failure is more likely due to generator-side rationale quality or predictor lag, then assigns the next slice to the corresponding side. Figure 2 presents an overview of ReSR. The Main Inference Path remains unchanged, while the Training-only Diagnostics block contains the reference, probe, and online diagnostics. The VRSR Scheduler integrates these signals into the generator and predictor debts.

3.1 Training-only Diagnostics

Reference diagnostic and online diagnostic.

ReSR uses three losses that task loss alone cannot fully separate. The reference diagnostic trains a reference predictor P_{ref} on full inputs and gives L_{ref} , a full-input anchor for comparison. The probe diagnostic trains a fresh probe predictor P_{probe} on rationales from the frozen current generator and gives L_{probe} , a learnability estimate for the current rationale distribution. The online diagnostic gives L_{online} , the current predictor loss on the same masked-input distribution.

Operating-system term	ReSR use
Runnable process	Update client
Runtime	Slice count
Priority/weight	Debt weight
Virtual runtime	Normalized progress
Starvation prevention	Age guard
Scheduling decision	Next slice

Table 1: Operating-system-inspired view of ReSR. VRSR borrows the resource-allocation principle of operating-system schedulers.

The gaps determine pressure. A high L_{probe} relative to L_{ref} means even a fresh predictor cannot learn from the current rationales, so generator pressure rises. A high L_{online} with a low L_{probe} means the rationale is learnable but the online predictor lags, so predictor pressure rises. Rationale rate, fragmentation, and mask drift add generator-side signals for collapse, over-selection, and unstable masks.

Probe diagnostic. The probe is diagnostic only, which reuses the embedding layer and reinitializes the predictor encoder and classifier before each diagnostic run, and trains for a small fixed budget on deterministic masks from the frozen generator. The probe uses no gold rationale annotations. Only the probe validation loss is retained. The probe is discarded and never used for checkpoint outputs or test-time inference.

3.2 VRSR Scheduler

VRSR maps diagnostics to update allocation by maintaining two virtual runtimes, vr_g and vr_p , for generator and predictor slices, respectively. Table 1 summarizes the scheduling analogy. At each validation epoch, generator debt is

$$D_g = \alpha \max(0, L_{\text{probe}} - (L_{\text{ref}} + \epsilon)) + \beta |R - r^*| + \gamma F + \delta \Delta + \eta \text{age}_g, \quad (4)$$

where R denotes the rationale rate, r^* the target rate, F the proportion of adjacent valid-token pairs whose generated binary mask states differ, Δ the validation-mask drift, and age_g the time since the last generator update. A larger F indicates more frequent transitions between selected and unselected tokens and therefore a more fragmented rationale. F is computed without gold rationale annotations. The first term is the probe-reference gap, and the others encode budget mismatch, structure, drift, and starvation. Here, $\epsilon \geq 0$ denotes a tolerance margin for the probe-reference gap. Formally, predictor debt can be represented as

$$D_p = \mu \max(0, L_{\text{online}} - L_{\text{probe}}) + \nu \text{age}_p. \quad (5)$$

All coefficients indicate nonnegative weights for the corresponding terms. This debt grows when the probe can learn the rationale but the online predictor still performs poorly. In implementation, both loss gaps pass through thresholds and exponential moving averages to reduce validation noise. Table 13 summarizes the diagnostic-pressure rules in Appendix C. Furthermore, VRSR converts debts into weights:

$$w_g = 1 + \max(0, D_g), \quad (6)$$

$$w_p = 1 + \max(0, D_p), \quad (7)$$

and computes the corresponding virtual-runtime scores as

$$s_g = \frac{vr_g}{w_g}, \quad (8)$$

$$s_p = \frac{vr_p}{w_p}. \quad (9)$$

The next slice goes to the side with the smaller score. After the slice, that side’s virtual runtime advances by the inverse of the corresponding weight, so high debt increases future access to slices. Warmup, collapse recovery, inheritance, and age guards prevent early instability and starvation.

Algorithm 1 ReSR Training

Require: Training set $\mathcal{D}_{tr}$, validation set $\mathcal{D}_{val}$, generator G_{θ_G} , predictor P_{θ_P} , scheduler state Ω

Ensure: Best checkpoint by rationale-aware validation score

- 1: Train reference predictor P_{ref} on full inputs
- 2: Initialize VRSR state Ω
- 3: **for** epoch = 1 to T **do**
- 4: phase $\leftarrow$ VRSR-DECIDE(Ω)
- 5: **if** phase = generator **then**
- 6: Freeze θ_P ; update θ_G with task, rationale, and token-guidance losses
- 7: **else**
- 8: Freeze θ_G ; update θ_P on HardMask($G_{\theta_G}(x)$)
- 9: **end if**
- 10: Compute $L_{\text{online}}, L_{\text{probe}}, L_{\text{ref}}, R, F, \Delta$
- 11: Save checkpoint if validation score improves
- 12: $\Omega \leftarrow$ VRSR-UPDATE($\Omega, L_{\text{online}}, L_{\text{probe}}, L_{\text{ref}}, R, F, \Delta$)
- 13: **end for**
- 14: **return** best checkpoint

3.3 Training-only Supporting Signal

Scheduling decides when to update the generator, not which tokens to prefer. During generator slices, a full-input teacher supplies token-importance scores α_{tok} when the teacher prediction is confident. The token-alignment loss L_{TA} encourages probability on high-confidence rationale tokens. This signal is absent at inference.

3.4 Training Procedure

Algorithm 1 summarizes training. Each epoch is an optimization slice: a generator slice freezes the predictor and updates the generator with task, rationale, and token-guidance losses; a predictor slice freezes the generator and updates the predictor on hard masks. Validation then computes the diagnostics and updates VRSR.

3.5 Complexity Analysis

Table 2 compares ReSR with MGR and GenSPP. Inference uses only the generator and online predictor, so ReSR matches a standard select-then-predict rationalizer: $\Theta(B_{\text{test}} F(n))$. The reference predictor, probe predictor, VRSR scheduler, and token-guidance alignment are training-only. They add constant-factor training cost but no population-generation search loop. With fixed probe/reference budgets, ReSR has the same asymptotic order in sequence length and training batches as ordinary gradient-based rationalization, whereas GenSPP multiplies cost by generations and population size. Appendix B.2 provides the derivation.

Method	Training Complexity	Inference Complexity	Main Source of Extra Cost
MGR	$\Theta(EBgF(n))$	$\Theta(B_{\text{test}}F(n))$	Multiple generators per training batch
GenSPP	$\Theta(GI(E_{\text{sp}}B_{\text{tr}} + B_{\text{val}})F(n))$	$\Theta(B_{\text{test}}F(n))$	Population-generation genetic search
ReSR	$\Theta((EB + EK_pE_p + K_rE_r + EV)F(n))$	$\Theta(B_{\text{test}}F(n))$	Training diagnostics and token guidance

Table 2: Complexity comparison. $F(n) = \Theta(bn(dh + h^2))$ denotes one mini-batch model pass for batch size b and sequence length n . E is the number of epochs, B the number of batches, g the number of MGR generators, and G, I, E_{sp} are the GenSPP generations, population size, and refinement epochs. K_r and K_p are the mini-batches used by reference training and each probe diagnostic. E_r and E_p are their budgets, and V is the number of validation batches per outer epoch.

4 Experiments

We evaluate ReSR from three angles: main benchmark performance on shared splits, training efficiency, and component ablations.

4.1 Datasets

We experiment on two complementary datasets. **Toy** is the controlled synthetic benchmark from GenSPP (Ruggeri and Signorelli, 2025), where labels depend on short ground-truth highlight patterns. The dataset tests whether a model recovers the intended rationale, not only the label. **HateXplain** (Mathew et al., 2021) is a human-annotated hate-speech benchmark with token-level rationales. HateXplain contains noisier social-media text and rationales that are often fragmented. All methods use the same train, validation, and test splits.

4.2 Baselines

We group representative baselines by how they handle generator-predictor training. The baseline group contains RNP-style select-then-predict methods: **FR** (Liu et al., 2022), **MGR** (Liu et al., 2023a), **MCD** (Liu et al., 2023b), and **G-RAT** (Hu and Yu, 2024). They cover encoder sharing, multiple generators, causal conditional-dependence constraints, and soft-rationale guidance while retaining a fixed update schedule. The search-based decoupling baseline is **GenSPP** (Ruggeri and Signorelli, 2025), which evaluates candidate generators with separately trained predictors. GenSPP directly tests whether ReSR can recover part of this decoupling benefit while retaining ordinary RNP training.

4.3 Evaluation Metrics

Following GenSPP and prior rationalization benchmarks, we report both task and rationale metrics. **Clf-F1** is macro F1 over task labels. **HI-F1** is token-level F1 against gold rationales. **Rationale rate** R is the mean fraction of valid non-padding tokens selected, and **rationale size** S is the mean number of selected valid tokens per example. Higher is better

for Clf-F1 and HI-F1. lower is better for R and S . Section 5 also reports the Balanced Rationalization Score (BRS), a secondary normalized summary of accuracy-rationale quality-compactness balance.

Efficiency is measured by total wall-clock time and by each method’s natural training unit: epochs for ReSR and the fixed-schedule baselines, and generations for GenSPP. All methods use the GenSPP codebase splits, features, and backbone family. The main benchmark follows the shared five-seed protocol. We additionally evaluate ReSR and the fixed-schedule baselines under an annotation-free checkpoint-selection control using minimum validation loss and the same ten prespecified seeds. Efficiency uses a matched representative single-GPU run. Full training and checkpoint-selection details are in Appendix A.1.

5 Results and Analysis

5.1 Main Results

Table 3 shows that no baseline dominates all metrics. FR, MGR, MCD, and G-RAT are efficient fixed-schedule baselines, but their HI-F1 is much lower than ReSR and GenSPP. GenSPP gives the most compact rationales and the best HateXplain HI-F1, but GenSPP is much slower and has weaker HateXplain classification. ReSR occupies the middle region: high classification, strong rationale quality, moderate compactness, and far lower cost than the search-based decoupling baseline. We treat BRS as a secondary summary; raw Clf-F1, HI-F1, R , S , and efficiency remain the main basis for comparison.

Results on Toy. ReSR obtains the best rationale quality on Toy, reaching 85.47 HI-F1 with near-perfect classification (99.95 Clf-F1). ReSR outperforms all fixed-schedule baselines and GenSPP in HI-F1, although GenSPP selects fewer tokens. This matches ReSR’s intended role: when rationales are sparse and stable, controlling the generator–predictor update order helps recover annotated ra-

Model	Toy					HateXplain				
	Clf-F1 $\uparrow$	HI-F1 $\uparrow$	$R \downarrow$	$S \downarrow$	BRS $\uparrow$	Clf-F1 $\uparrow$	HI-F1 $\uparrow$	$R \downarrow$	$S \downarrow$	BRS $\uparrow$
FR	99.91 $\pm$ 0.07	58.88 $\pm$ 6.54	14.71 $\pm$ 0.48	2.94 $\pm$ 0.10	46.39	72.17 $\pm$ 0.62	26.65 $\pm$ 9.20	25.80 $\pm$ 1.03	3.46 $\pm$ 0.03	40.62
MGR	99.68 $\pm$ 0.40	48.70 $\pm$ 9.15	14.99 $\pm$ 0.34	3.00 $\pm$ 0.07	20.49	72.32 $\pm$ 0.61	18.97 $\pm$ 11.96	25.69 $\pm$ 1.28	3.47 $\pm$ 0.12	25.91
MCD	99.92 $\pm$ 0.05	66.80 $\pm$ 3.85	15.08 $\pm$ 0.35	3.02 $\pm$ 0.07	55.35	69.38 $\pm$ 0.64	27.81 $\pm$ 0.91	24.58 $\pm$ 1.15	3.49 $\pm$ 0.13	23.43
G-RAT	99.65 $\pm$ 0.36	44.96 $\pm$ 22.63	14.65 $\pm$ 0.34	2.93 $\pm$ 0.07	15.47	72.70 $\pm$ 0.96	35.06 $\pm$ 3.68	25.99 $\pm$ 1.65	3.45 $\pm$ 0.10	61.69
GenSPP	99.39 $\pm$ 0.50	68.59 $\pm$ 3.91	9.38 $\pm$ 0.63	1.88 $\pm$ 0.13	49.17	68.76 $\pm$ 0.88	43.26 $\pm$ 1.78	7.37 $\pm$ 1.13	0.94 $\pm$ 0.14	70.00
ReSR	99.95 $\pm$ 0.08	85.47 $\pm$ 9.04	14.77 $\pm$ 0.38	2.95 $\pm$ 0.08	81.16	72.92 $\pm$ 1.11	42.30 $\pm$ 2.71	16.99 $\pm$ 2.38	2.28 $\pm$ 0.34	87.60

Table 3: Main results on Toy and HateXplain. ReSR denotes our final responsibility-aware scheduling configuration with confidence-aware token guidance. Higher is better for Clf-F1, HI-F1, and BRS, while lower is better for rationale rate R and rationale size S . BRS is the Balanced Rationalization Score with primary-objective weighting defined in Section 5. This table follows the original five-seed checkpoint-selection protocols (Ruggeri and Signorelli, 2025), and the matched annotation-free control is reported in Appendix A.4.

tionales rather than merely shrinking masks.

Results on HateXplain. ReSR reaches 42.30 HI-F1 and the best classification performance (72.92 Clf-F1). ReSR’s HI-F1 is below GenSPP (43.26), but above FR, MGR, MCD, and G-RAT; ReSR also selects fewer tokens than these fixed-schedule baselines ($R = 16.99$, $S = 2.28$). Thus, GenSPP remains strongest in HateXplain HI-F1 and compactness, while ReSR gives the best classification, higher rationale quality than fixed-schedule baselines, and much lower cost than GenSPP.

Annotation-free checkpoint selection. For the primary fair comparison with the fixed-schedule baselines, we select checkpoints by minimum validation loss without using validation rationale annotations and evaluate ReSR and the corresponding baselines over the same ten seeds. ReSR improves HI-F1 over MCD on Toy by 10.69 points (95% CI [6.70, 14.85], $p = 0.0020$). On HateXplain, the mean improvement over G-RAT is 1.34 points, but the confidence interval includes zero (95% CI [−0.99, 4.10], $p = 0.3809$), so we do not interpret this individual difference as statistically significant. Appendix A.4 reports the complete results.

Balanced Rationalization Score. Following aggregate benchmark reporting such as MTEB (Muennighoff et al., 2023), we report BRS as a secondary score for the accuracy-rationale quality-compactness trade-off. Clf-F1 and HI-F1 reflect the main selective-rationalization objectives, while R and S measure compactness (Lei et al., 2016; DeYoung et al., 2020; Mathew et al., 2021). We min-max normalize each metric per dataset, reversing R and S so that higher is better, and use weights (0.30, 0.50, 0.10, 0.10) for Clf-F1, HI-F1, R , and S . Thus, 80% of BRS measures prediction and human-rationale alignment, while 20% measures

compactness. Under this weighting, ReSR has the highest BRS on both datasets. Appendix C.1 gives the full formula and alternative weights. GenSPP benefits when compactness is weighted more, whereas ReSR is strongest when prediction and rationale quality are primary. We use BRS only to summarize trade-offs, not to replace metric-wise comparison. Means and standard deviations are over five seeds; small differences, especially on Toy, should be read cautiously.

5.2 Component Ablations

Table 4 removes the VRSR scheduler, reference branch, probe, and training-only support signal. Removing VRSR causes the clearest failure: Toy HI-F1 drops from 85.47 $\pm$ 9.04 to 27.47 $\pm$ 27.92. The large variance across seeds reflects two observed outcomes: some runs converge normally, whereas others exhibit collapse in both the generator and predictor. Thus, this ablation indicates training instability after removing the full scheduler rather than the isolated contribution of each internal control rule. Appendix C.3 further compares the full VRSR scheduler with FixedAlt (a fixed 1:1 alternating schedule) and LossGap (a schedule based on the larger validation loss gap) under a common protocol using ten seeds. Removing the probe lowers HI-F1 on both datasets, supporting the probe’s role as a learnability signal for separating predictor lag from poor rationales. Removing the reference branch has a small Toy effect but reduces HateXplain HI-F1 from 42.30 to 39.44, suggesting that full-input anchoring matters more on noisy real data. Removing token guidance leaves Toy nearly unchanged within seed variance, but lowers HateXplain HI-F1 to 38.79 and increases R and S , indicating that the support signal mainly improves rationale quality and compactness on HateXplain.

Variant	Toy				HateXplain			
	Clf-F1↑	HI-F1↑	$R \downarrow$	$S \downarrow$	Clf-F1↑	HI-F1↑	$R \downarrow$	$S \downarrow$
ReSR	99.95 ± 0.08	85.47 ± 9.04	14.77 ± 0.38	2.95 ± 0.08	72.92 ± 1.11	42.30 ± 2.71	16.99 ± 2.38	2.28 ± 0.34
w/o VRSR	54.12 ± 37.76	27.47 ± 27.92	13.34 ± 8.80	2.67 ± 1.76	72.40 ± 0.43	41.49 ± 4.17	17.71 ± 3.14	2.40 ± 0.44
w/o Reference	99.72 ± 0.51	84.76 ± 7.93	14.88 ± 0.23	2.98 ± 0.05	72.70 ± 0.95	39.44 ± 6.44	17.93 ± 3.58	2.42 ± 0.48
w/o Probe	99.53 ± 0.74	81.58 ± 10.77	14.82 ± 0.18	2.96 ± 0.04	71.75 ± 1.06	40.38 ± 4.58	18.42 ± 2.80	2.50 ± 0.42
w/o Support	99.85 ± 0.08	84.48 ± 9.01	14.76 ± 0.39	2.95 ± 0.08	71.60 ± 1.76	38.79 ± 2.45	20.61 ± 1.11	2.80 ± 0.10

Table 4: Main ablation results for ReSR on Toy and HateXplain. We report the same five fixed random seeds as in the main results. The “w/o Support” row removes the final training-only token guidance signal.

Model	Units	Toy			HateXplain	
		Time (h)↓	Min./unit↓		Time (h)↓	Min./unit↓
FR	102 epochs	0.291	0.171		63 epochs	0.260
MGR	119 epochs	0.379	0.191		63 epochs	0.287
MCD	87 epochs	0.297	0.205		47 epochs	0.250
G-RAT	179 epochs	0.478	0.160		79 epochs	0.312
GenSPP	100 generations	2.991	1.794		100 generations	4.631
ReSR	342 epochs	0.694	0.122		124 epochs	0.202

Table 5: Efficiency comparison under a matched representative-run setup. Units are epochs for ReSR and the fixed-schedule baselines, and generations for GenSPP. Wall-clock time is measured with `/usr/bin/time -v`.

Human Annotation

Aspect: HateXplain offensive-language rationale

Label: Offensive **Pred:** – **HI-F1:** –

Text: or the construction company hired some **sleep**er cell asshole

FR

Aspect: HateXplain offensive-language rationale

Label: Offensive **Pred:** Offensive **HI-F1:** 0.00

Text: or the construction company **hired** some sleeper cell **asshole**

MGR

Aspect: HateXplain offensive-language rationale

Label: Offensive **Pred:** Offensive **HI-F1:** 0.00

Text: **or the** construction company hired some sleeper cell **asshole**

MCD

Aspect: HateXplain offensive-language rationale

Label: Offensive **Pred:** Offensive **HI-F1:** 0.00

Text: or **the** construction company hired some sleeper cell **asshole**

G-RAT

Aspect: HateXplain offensive-language rationale

Label: Offensive **Pred:** Offensive **HI-F1:** 0.00

Text: or the construction company **hired** some sleeper cell **asshole**

GenSPP

Aspect: HateXplain offensive-language rationale

Label: Offensive **Pred:** Offensive **HI-F1:** 0.00

Text: or the construction company hired some sleeper cell **asshole**

ReSR

Aspect: HateXplain offensive-language rationale

Label: Offensive **Pred:** Offensive **HI-F1:** **0.80**

Text: or the construction company hired some **sleep**er cell **asshole**

Table 6: Qualitative HateXplain case study in an aspect-label-prediction-text visualization format. Green marks the human rationale, red marks baseline selections, and blue marks the ReSR selection. All methods predict the correct label, so the difference lies in the selected rationale: ReSR recovers the annotated span “sleeper cell” with one adjacent token, while the baselines select peripheral context or the trailing insult. GenSPP outputs retain one extra raw token in this sample; after alignment, four of five runs select only “asshole” and one run selects no token, so we report the common non-empty selection.

Appendix C.2 summarizes the failure mode tested by each ablation.

5.3 Efficiency Comparison

Table 5 presents a matched representative-run comparison. ReSR may train for more epochs than the lightest fixed-schedule baselines, but its per-epoch cost remains small: 0.122 minutes on Toy and 0.202 on HateXplain. By contrast, even a sin-

gle $n=1$ GenSPP search takes 2.991 hours on Toy and 4.631 hours on HateXplain. This matches Table 2: ReSR pays diagnostic overhead but avoids GenSPP’s population-generation search.

5.4 Qualitative Case Study

Table 6 shows a HateXplain example where all methods predict correctly, isolating rationale selection. ReSR recovers “sleeper cell” with one adja-

cent token, whereas the other methods select unannotated context, often including the sentence-final offensive token. This case illustrates the distinction between predictive sufficiency and rationale alignment: correct predictions may rely on different textual rationales.

5.5 Discussion

The results point to training control as ReSR’s main contribution: scheduling and probe/reference diagnostics control collapse and responsibility attribution, while token guidance improves the balance between rationale quality and compactness on HateXplain without changing inference. The remaining H1-F1 gap to GenSPP suggests that scheduling can be combined with stronger rationale supervision or richer generators for fragmented human rationales.

6 Conclusion

In this paper, we propose ReSR, a responsibility-aware scheduling framework for selective rationalization. The ReSR keeps the standard RNP inference pathway, but treats training as responsibility attribution followed by update allocation. Online, probe, and reference diagnostics with rationale-structure signals are converted by VRSSR into generator or predictor update slices, while confidence-aware token guidance supports generator updates only during training. Experiments on Toy and HateXplain show that ReSR improves the accuracy-rationale quality-compactness-cost profile without changing inference. Ablations identify VRSSR scheduling and full-input reference anchoring as the main contributors, with probe diagnostics serving as a complementary learnability signal. Together, these results position training control between fixed-schedule select-then-predict baselines and fully disjoint search-based decoupling.

Limitations

ReSR improves RNP training dynamics, but does not fully remove generator-predictor coupling as fully disjoint search-based methods do. The deployed model still follows the standard generator-rationale-predictor pipeline. Thus, ReSR should be viewed as a lightweight training-time decoupling strategy rather than a fully separated optimization procedure. Future work may explore fully decoupled or hybrid optimization frameworks. In addition, rationalization focuses on the interaction

between explanations (i.e., rationales in this work) and predictions, thereby promoting their cooperation. However, fostering efficient and effective prediction-explanation cooperation in large language models for rationalization applications remains an open challenge.

Ethics Statement

This paper does not involve the presentation of a new dataset and the utilization of demographic or identity characteristics information.

Acknowledgements

We would like to thank all the anonymous reviewers for their valuable and constructive feedback. This work was supported by the National Natural Science Foundation of China (Nos. 62376144 and 62506218), the China Scholarship Council, the Natural Language Processing Innovation Team (Sanjin Talents) Project of Shanxi Province, and the Science and Technology Cooperation and Exchange Special Project of Shanxi Province (No. 202204041101016).

References

- Shiyu Chang, Yang Zhang, Mo Yu, and Tommi Jaakkola. 2020. Invariant rationalization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1448–1458. PMLR.
- Zhiying Deng, Jianjun Li, Zhiqiang Guo, and Guohui Li. 2023. [Multi-aspect interest neighbor-augmented network for next-basket recommendation](#). *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.
- Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. [ERASER: A benchmark to evaluate rationalized NLP models](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.
- Yue Fan, Hu Zhang, Ru Li, Yujie Wang, Guangjun Zhang, Hongye Tan, and Jiye Liang. 2025. Weakly-supervised explainable question answering via question aware contrastive learning and adaptive gate mechanism. *Information Sciences*, 697:121763.
- Yue Fan, Hu Zhang, Yunxiao Zhao, Guangjun Zhang, Hao Zhan, Ru Li, Hongye Tan, and Yuanlong Wang. 2026. Leibniz: Theory-of-mind driven neuro-symbolic logical reasoning via multi-agent collaboration. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 20186–20210,

- San Diego, California, United States. Association for Computational Linguistics.
- Shuaibo Hu and Kui Yu. 2024. [Learning robust rationales for model explainability: A guidance-based approach](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):18243–18251.
- Yongfeng Huang, Yujun Chen, Yulun Du, and Zhilin Yang. 2021. Distribution matching for rationalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13090–13097.
- Xu Jiang, Yurong Cheng, Boyang Li, Ye Yuan, and Guoren Wang. 2026. [S-explainer: A self-explaining nlp framework based on cooperative rationalization and multi-hop evidence reasoning](#). *Knowledge-Based Systems*, 336:114754.
- Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. [Rationalizing neural predictions](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas. Association for Computational Linguistics.
- Wei Liu, Zhiying Deng, Lang Gao, Yuankai Zhang, Haozhao Wang, Jun Wang, Rui Zhang, Heng Huang, and Ruixuan Li. 2026. Escaping graphical causal priors: Multi-generator cooperation for rationalization. *Artificial Intelligence*, 359:104601.
- Wei Liu, Zhiying Deng, Zhongyu Niu, Jun Wang, Haozhao Wang, and Ruixuan Li. 2025a. Exploring practical gaps in using cross entropy to implement maximum mutual information criterion for rationalization. *Transactions of the Association for Computational Linguistics*, 13:577–594.
- Wei Liu, Zhiying Deng, Zhongyu Niu, Jun Wang, Haozhao Wang, Zhigang Zeng, and Ruixuan Li. 2025b. Breaking free from mmi: A new frontier in rationalization by probing input utilization. In *International Conference on Learning Representations*, volume 2025, pages 55351–55374.
- Wei Liu, Zhongyu Niu, Lang Gao, Zhiying Deng, Jun Wang, Haozhao Wang, and Ruixuan Li. 2025c. [Adversarial cooperative rationalization: The risk of spurious correlations in even clean datasets](#). In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 39126–39146. PMLR.
- Wei Liu, Haozhao Wang, Jun Wang, Zhiying Deng, Yuankai Zhang, Cheng Wang, and Ruixuan Li. 2024. Enhancing the rationale-input alignment for self-explaining rationalization. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 2218–2230. IEEE.
- Wei Liu, Haozhao Wang, Jun Wang, Ruixuan Li, Xinyang Li, YuanKai Zhang, and Yang Qiu. 2023a. [MGR: Multi-generator based rationalization](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12771–12787, Toronto, Canada. Association for Computational Linguistics.
- Wei Liu, Haozhao Wang, Jun Wang, Ruixuan Li, Chao Yue, and YuanKai Zhang. 2022. FR: Folded rationalization with a unified encoder. In *Advances in Neural Information Processing Systems*, volume 35, pages 6954–6966.
- Wei Liu, Jun Wang, Haozhao Wang, Ruixuan Li, Zhiying Deng, Yuankai Zhang, and Yang Qiu. 2023b. D-separation for causal self-explanation. *Advances in Neural Information Processing Systems*, 36:43620–43633.
- Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2021. [Hatexplain: A benchmark dataset for explainable hate speech detection](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(17):14867–14875.
- Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2023. [MTEB: Massive text embedding benchmark](#). In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia. Association for Computational Linguistics.
- Krithi Ramamritham and John A. Stankovic. 1994. [Scheduling algorithms and operating systems support for real-time systems](#). *Proceedings of the IEEE*, 82(1):55–67.
- Federico Ruggeri and Gaetano Signorelli. 2025. [Interlocking-free selective rationalization through genetic-based learning](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1175–1191, Vienna, Austria. Association for Computational Linguistics.
- Hongye Tan, Xiaoyue Wang, Yu Ji, Ru Li, Xiaoli Li, Zhiwei Hu, Yunxiao Zhao, and Xiaoqi Han. 2021. [GCRC: A new challenging MRC dataset from Gaokao Chinese for explainable evaluation](#). In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1319–1330, Online. Association for Computational Linguistics.
- Yiyang Wei, Tingyu Song, Siyue Zhang, and Yilun Zhao. 2026. A survey of reasoning-intensive retrieval: Progress and challenges. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 42101–42122.
- Hao Xu, Yunxiao Zhao, Jiayang Zhang, Zhiqiang Wang, and Ru Li. 2025. [LOG: A local-to-global optimization approach for retrieval-based explainable multi-hop question answering](#). In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9085–9095, Abu Dhabi, UAE. Association for Computational Linguistics.

Zhichao Yan, Jiapu Wang, Jiaoyan Chen, Xiaoli Li, Jiye Liang, Ru Li, and Jeff Z. Pan. 2025. [Atomic fact decomposition helps attributed question answering](#). *IEEE Transactions on Knowledge and Data Engineering*, 37(12):6959–6972.

Zhichao Yan, Yunxiao Zhao, Jiapu Wang, Jiaoyan Chen, Xiaoli Li, Ru Li, and Jeff Z. Pan. 2026. Beyond factual accuracy: Evaluating global reasoning integrity in rag systems with logicscore. *arXiv preprint arXiv:2601.15050*.

Mo Yu, Shiyu Chang, Yang Zhang, and Tommi Jaakkola. 2019. [Rethinking cooperative rationalization: Introspective extraction and complement control](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4094–4103, Hong Kong, China. Association for Computational Linguistics.

Mo Yu, Yang Zhang, Shiyu Chang, and Tommi S. Jaakkola. 2021. Understanding interlocking dynamics of cooperative rationalization. In *Advances in Neural Information Processing Systems*, volume 34, pages 12822–12835. Curran Associates, Inc.

Hao Yuan, Lei Cai, Xia Hu, Jie Wang, and Shuiwang Ji. 2022. [Interpreting image classifiers by generating discrete masks](#). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(4):2019–2030.

Linan Yue, Qi Liu, YiChao Du, Li Wang, Yanqing An, and Enhong Chen. 2025. [Towards reliable and faithful explanations: A disentanglement-augmented approach for selective rationalization](#). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(11):9906–9921.

Yunxiao Zhao, Zhiqiang Wang, Xiaoli Li, Jiye Liang, and Ru Li. 2024. [AGR: Reinforced causal agent-guided self-explaining rationalization](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 510–518, Bangkok, Thailand. Association for Computational Linguistics.

Yunxiao Zhao, Zhiqiang Wang, Xingtong Yu, Xiaoli Li, Jiye Liang, and Ru Li. 2026. [Learnable game-theoretic policy optimization for data-centric self-explanation rationalization](#). *IEEE Transactions on Knowledge and Data Engineering*, 38(2):1159–1173.

Yunxiao Zhao, Hao Xu, Zhiqiang Wang, Xiaoli Li, Jiye Liang, and Ru Li. 2025. [Explaining black-box language models with knowledge probing systems: A post-hoc explanation perspective](#). In *International Conference on Database Systems for Advanced Applications*, pages 220–235. Springer.

A Reproducibility Details

This appendix collects the details needed to reproduce ReSR and to interpret the supplementary

analyses behind the main claims. Following common practice in rationalization papers, we place implementation and selection details first, then provide complexity and efficiency details, robustness checks, and additional scheduler diagnostics.

A.1 Experimental Protocol

All neural baselines and ReSR use the same train, validation, and test splits, input features, and backbone family as the GenSPP codebase. Toy uses one-hot token features, and HateXplain uses Twitter GloVe embeddings. The main benchmark and component ablations use the same five random seeds. The supplementary scheduler comparison among Full VRSR, FixedAlt, and LossGap and the annotation-free checkpoint-selection control each use the same ten prespecified random seeds and retain all runs. Online experiment logging is disabled throughout.

The main benchmark with five seeds follows each method’s original protocol for checkpoint selection. ReSR uses the validation score that incorporates rationale quality, as described in Appendix A.3. For the primary comparison without validation rationale annotations, ReSR and the corresponding baselines with fixed update schedules instead use minimum validation loss and the same ten prespecified random seeds. GenSPP retains the selection protocol used in its public implementation. This protocol evaluates candidates at the population level without using human rationale annotations, but it differs from the matched criterion based on minimum validation loss. The final ReSR configuration includes VRSR, online/probe/reference diagnostics, and generator-side confidence-aware token guidance. The reference and probe predictors use only task labels and input tokens, not gold rationale annotations. All diagnostics and support signals are training-only and removed at inference time.

Efficiency is measured in a matched single-GPU setup on the same server (p100-4) and Conda environment (gen_spp_env). Performance metrics in the main tables use five seeds, the supplementary scheduler comparison and annotation-free checkpoint-selection control use ten seeds, and efficiency uses one representative run. GenSPP is run with $n=1$. This isolates cost: ReSR adds diagnostic and scheduling overhead to gradient training, while GenSPP pays for repeated population-based search.

A.2 ReSR Configuration Details

Table 7 summarizes the final ReSR configurations. The table combines basic rationalization settings, core VRSR settings, diagnostic budgets, and checkpoint selection settings. The core debt coefficients, virtual runtime rule, warmup, and inheritance settings are shared across datasets. Dataset-specific differences concern the basic rationalization settings, diagnostic budgets, collapse safeguards, and checkpoint selection thresholds. The diagnostic budgets only control computation, while the selection thresholds do not enter the VRSR debt calculation.

Param.	Toy	HateX.
Target rate r^*	0.15	0.22
Sparsity coef.	1.5	1.0
Continuity coef.	2.0	off
Margin ϵ	0.0	0.0
Gen. weights	(1, 1, 1, 1, 0.1)	(1, 1, 1, 1, 0.1)
Pred. weights	(1, 0.1)	(1, 0.1)
VR step q	1.0	1.0
Warmup	1	1
Inheritance	(0.25, 3, 2)	(0.25, 3, 2)
Collapse int.	[0.05, 0.98]	[0.02, 0.98]
Recovery	5	2
Probe batches	50	10
Probe epochs	3	1
Ref. batches	100	50
Ref. epochs	3	3
Selection	(4, 1, 0.97, 5)	(1, 0.25, 0.68, 2)

Table 7: Final ReSR hyperparameters. Generator weights are $(\alpha, \beta, \gamma, \delta, \eta)$, predictor weights are (μ, ν) , inheritance is (gap, patience, slices), and selection is $(\lambda_R, \lambda_F, \tau, \lambda_{\text{clf}})$.

A.3 Model Selection

ReSR selects checkpoints using a rationale-aware validation score rather than classification quality alone. In practice, this choice is important: the best classifier is not always the best rationalizer. Our validation rule favors stronger highlight quality while penalizing overly large or fragmented rationales and severe classification collapse:

$$\begin{aligned} \text{score}_{\text{val}} = & \text{HIF1} - \lambda_R |R - r^*| \\ & - \lambda_F F - \lambda_{\text{clf}} \max(0, \tau - \text{ClfF1}). \end{aligned} \quad (10)$$

For the selection parameters $(\lambda_R, \lambda_F, \tau, \lambda_{\text{clf}})$, we use

$$\begin{aligned} \text{Toy:} & \quad (4, 1, 0.97, 5), \\ \text{HateXplain:} & \quad (1, 0.25, 0.68, 2). \end{aligned}$$

This rule prioritizes highlight quality, keeps the mask close to the target budget, and rejects checkpoints with severe classification collapse.

A.4 Checkpoint Selection without Rationale Annotations

To remove the potential effect of rationale annotations in the validation set, we select checkpoints for ReSR and the strongest baseline with a fixed update schedule on each dataset according to minimum validation loss. The comparison uses the same ten prespecified random seeds. Table 8 reports the complete results.

We estimate 95% confidence intervals from 100,000 paired bootstrap resamples and compute p values using exact two-sided sign-flip tests. On Toy, ReSR improves HI-F1 over MCD by 10.69 points (95% CI [6.70, 14.85], $p = 0.0020$). On HateXplain, the mean improvement over G-RAT is 1.34 points, but this difference is not statistically significant (95% CI [-0.99, 4.10], $p = 0.3809$). GenSPP is not included in this comparison because its public implementation uses validation task loss, sparsity, and confidence to evaluate candidate generators at the population level rather than selecting checkpoints across epochs according to minimum validation loss. Neither the minimum validation loss protocol nor the GenSPP protocol uses human rationale annotations, but their selection rules are not identical.

A.5 Scheduler Control Rules

In the implementation, one optimization slice corresponds to one training epoch. The scheduler is updated once per validation epoch using the epoch-aggregated online loss, highlight rate, fragmentation, and drift. The probe predictor is retrained from scratch at every validation epoch under the current generator snapshot, using the probe budget shown in Table 7. The reference predictor is trained once on full inputs before the main ReSR schedule begins.

Warmup forces predictor updates for the first slice on both datasets. After warmup, VRSR compares the generator and predictor virtual-runtime scores, unless one of two override rules is triggered. First, if the validation rationale rate falls outside the dataset-specific collapse interval in Table 7, ReSR treats this as mask collapse and forces generator updates for the configured number of recovery slices. Second, if either bottleneck gap exceeds the inheritance threshold $g = 0.25$ for $p = 3$ consecutive validations, ReSR temporarily forces the affected side for $s = 2$ slices. These rules implement the warmup, recovery, and priority-inheritance behav-

Dataset	Method	Clf-F1 $\uparrow$	HI-F1 $\uparrow$	$R \downarrow$	$S \downarrow$
Toy	MCD	99.91 $\pm$ 0.08	64.75 $\pm$ 3.60	14.98 $\pm$ 0.41	3.00 $\pm$ 0.08
Toy	ReSR	99.98 $\pm$ 0.02	75.44 $\pm$ 8.53	14.61 $\pm$ 0.67	2.92 $\pm$ 0.13
HateXplain	G-RAT	72.55 $\pm$ 0.62	36.86 $\pm$ 1.82	25.60 $\pm$ 0.75	3.42 $\pm$ 0.08
HateXplain	ReSR	73.21 $\pm$ 0.94	38.19 $\pm$ 2.92	17.56 $\pm$ 3.90	2.36 $\pm$ 0.61

Table 8: Results under minimum-validation-loss checkpoint selection without using validation rationale annotations. Means and standard deviations are computed over the same ten prespecified random seeds.

ior described in the main text.

B Complexity and Efficiency Details

The efficiency claim in the main text depends on two points: the training-only diagnostics do not use additional rationale labels, and their cost is fixed rather than multiplied by population search. We therefore describe diagnostic fairness before deriving the complexity terms.

B.1 Diagnostic Fairness and Cost

The reference predictor and probe predictor use only the task labels and input tokens available to all methods; they do not use gold rationale annotations. They are training-time diagnostics and are discarded at inference. The reference predictor is trained once on full inputs, while the probe is retrained from scratch with a fixed small budget under the current frozen generator. This means the probe does train additional predictor instances, but the probe budget is fixed and is explicitly included in the wall-clock efficiency comparison. In contrast, GenSPP repeatedly evaluates many candidate generators through population-level search, which introduces generation and population multipliers absent from ReSR.

B.2 Complexity Derivation

We derive the complexity terms used in Table 2. Let n be the sequence length, b the batch size, d the embedding dimension, and h the recurrent hidden size. A single generator–predictor forward/backward pass is dominated by recurrent encoding and costs

$$F(n) = \Theta(bn(dh + h^2)).$$

MGR. MGR trains g generators over E epochs and B batches per epoch. Since each batch evaluates multiple generator branches, the dominant training cost is

$$\Theta(EBgF(n)).$$

At inference time, the selected rationale pathway requires only a constant number of generator/predictor passes, giving $\Theta(B_{\text{test}}F(n))$.

GenSPP. GenSPP wraps rationale optimization inside a genetic search. For G generations and population size I , each individual is refined for E_{sp} epochs over B_{tr} training batches and evaluated over B_{val} validation batches. The resulting training cost is

$$\Theta(GI(E_{\text{sp}}B_{\text{tr}} + B_{\text{val}})F(n)).$$

The final inference pathway still uses a learned rationale predictor, so inference remains $\Theta(B_{\text{test}}F(n))$.

ReSR. ReSR performs one generator or predictor update slice per epoch, giving a main gradient-training cost of $\Theta(EBF(n))$. The reference predictor is trained once with K_r batches for E_r inner epochs, which costs $\Theta(K_rE_rF(n))$. The probe predictor is freshly trained during validation; with K_p probe batches and E_p probe epochs per outer epoch, this costs $\Theta(EK_pE_pF(n))$. Validation diagnostics, including online/reference losses and rationale-structure checks, use a constant number of passes over V validation batches per epoch, giving $\Theta(EVF(n))$. Teacher token guidance is applied only during generator updates and introduces another constant number of full-input passes, so this cost is absorbed into the main $\Theta(EBF(n))$ term. Combining these terms gives

$$\Theta((EB + EK_pE_p + K_rE_r + EV)F(n)).$$

Thus, ReSR has the same inference complexity as standard select-then-predict rationalization. ReSR’s training complexity has additional diagnostic terms, but ReSR does not contain the population-size and generation-count multipliers that dominate GenSPP.

B.3 Efficiency Benchmark Protocol

For the efficiency comparison, we benchmark the fixed-schedule baselines, GenSPP, and ReSR under

Source	When paid	Cost term	Role
Reference training	Before scheduling starts	$K_r E_r F(n)$	Full-input loss anchor
Probe diagnostics	After validation slices	$E K_p E_p F(n)$	Rationale learnability estimate
Token guidance	Generator slices	Extra teacher pass	High-confidence rationale-token signal
VRSR decision	Validation epoch	$O(1)$	Next generator/predictor slice

Table 9: Training-time overhead decomposition for ReSR. Probe diagnostics and reference/probe validation passes dominate the extra cost; scheduler decisions are constant time and are included in Table 5.

the same hardware environment. ReSR and the fixed-schedule baselines are measured with wall-clock time, selected epoch count, and hours per epoch under the same representative run. GenSPP is search-based, so we report generations and hours per generation. The purpose of this experiment is computational cost rather than seed variance; the main accuracy and rationale metrics are reported separately over the selected five seeds.

C Robustness of Main Claims

This section collects supplementary analyses that address two possible concerns about the main conclusions: how the secondary BRS ranking changes under different weight choices, and whether scheduler-side diagnostics are interpretable beyond final test metrics.

C.1 BRS Sensitivity Analysis

Table 10 checks whether the aggregate BRS conclusion is sensitive to the weighting scheme. We recompute BRS from the raw mean values in Table 3. For method i , metric m , and dataset d , let $a_m^{(d)} = \min_j x_{j,m}^{(d)}$ and $b_m^{(d)} = \max_j x_{j,m}^{(d)}$. The normalized value is

$$\tilde{x}_{i,m}^{(d)} = \begin{cases} 100 \frac{x_{i,m}^{(d)} - a_m^{(d)}}{b_m^{(d)} - a_m^{(d)}}, & m \in \{\text{Clf}, \text{HI}\}, \\ 100 \frac{b_m^{(d)} - x_{i,m}^{(d)}}{b_m^{(d)} - a_m^{(d)}}, & m \in \{R, S\}. \end{cases}$$

The main-text BRS is

$$\begin{aligned} \text{BRS}_i^{(d)} &= 0.30\tilde{x}_{i,\text{Clf}}^{(d)} + 0.50\tilde{x}_{i,\text{HI}}^{(d)} \\ &\quad + 0.10\tilde{x}_{i,R}^{(d)} + 0.10\tilde{x}_{i,S}^{(d)}. \end{aligned}$$

We then vary the weights over classification, rationale quality, rationale rate, and rationale size. The primary-objective setting is the main BRS reported in Table 3; this setting assigns most weight to Clf-F1 and HI-F1 and treats R and S as compactness constraints. The equal-weight setting

treats all four metrics identically, the rationale-quality-heavy setting emphasizes token-level rationale overlap, and the compactness-heavy setting gives larger weight to rationale rate and size. GenSPP is strongest under equal and compactness-heavy weighting because GenSPP selects much shorter rationales, while ReSR is strongest when prediction and rationale quality are treated as primary objectives. This does not make BRS a replacement for the raw metrics; instead, the sensitivity analysis confirms why we report BRS only as a secondary summary alongside the raw metrics and the efficiency comparison.

C.2 Ablation Roles

Table 11 lists the diagnostic role removed by each main ablation and the failure mode the ablation is intended to test.

C.3 Simple Scheduler Controls and Stability Across Seeds

The w/o VRSR ablation evaluates training stability after removing the full scheduling mechanism rather than the contribution of every internal rule separately. In this variant, the generator and predictor debt coefficients are set to zero. Warmup, inheritance, and collapse recovery are disabled, so the scheduler reduces to 1:1 alternation. On Toy, the HI-F1 result of 27.47 ± 27.92 reflects two observed outcomes. Some runs converge normally, whereas others undergo collapse in both the generator and predictor. The large variance therefore indicates instability across seeds without scheduling control. It does not mean that every run collapses.

To examine simpler alternatives, FixedAlt explicitly alternates generator and predictor updates at a 1:1 ratio, while LossGap updates the side with the larger validation loss gap. Full VRSR, FixedAlt, and LossGap use the same ten prespecified random seeds, and all runs are retained. The w/o VRSR row in Table 4 follows the five-seed main ablation protocol, so its mean is not expected to match the

Dataset	Weights	FR	MGR	MCD	G-RAT	GenSPP	ReSR
Toy	Primary-objective (.30, .50, .10, .10)	46.39	20.49	55.35	15.47	49.17	81.16
	Equal (.25, .25, .25, .25)	35.18	16.09	37.14	15.47	64.58	52.89
	Rationale-quality-heavy (.20, .60, .10, .10)	40.54	16.23	51.28	10.83	55.00	81.16
	Compactness-heavy (.20, .35, .25, .20)	33.62	14.33	37.80	12.75	65.42	57.59
HateXplain	Primary-objective (.30, .50, .10, .10)	40.62	25.91	23.43	61.69	70.00	87.60
	Equal (.25, .25, .25, .25)	28.95	21.99	14.72	40.63	75.00	72.96
	Rationale-quality-heavy (.20, .60, .10, .10)	35.58	17.35	25.57	58.84	80.00	87.21
	Compactness-heavy (.20, .35, .25, .20)	27.95	17.68	17.61	42.44	80.00	75.19

Table 10: BRS sensitivity analysis under secondary weighting schemes over (Clf-F1, HI-F1, rationale rate, rationale size). The primary-objective row is the main BRS reported in Table 3; the other rows show how the aggregate changes under equal weighting or when rationale quality or compactness receives larger weight.

Ablation	Removed component	What the drop indicates
w/o VRSR	Adaptive scheduler	Update allocation is needed to avoid generator–predictor imbalance.
w/o Probe	Learnability probe	Probe loss helps separate poor rationales from predictor lag.
w/o Reference	Full-input reference	Full-input anchoring discourages shortcut rationales.
w/o Support	Token guidance	Token-level support improves rationale quality and compactness on real data.

Table 11: Diagnostic roles of ReSR ablations. Each row links a removed component to its tested failure mode.

ten-seed FixedAlt result.

On Toy, Full VRSR obtains 81.52 ± 8.76 HI-F1, compared with 48.95 ± 29.48 for FixedAlt and 45.71 ± 29.53 for LossGap. Its improvement over FixedAlt is 32.57 points, with a 95% confidence interval of $[11.81, 55.83]$ and $p = 0.0020$. These results suggest that neither fixed alternation nor a single validation loss gap reproduces the performance and stability of Full VRSR on Toy. On HateXplain, Full VRSR has the highest mean HI-F1 and lower variance than both simple controls, while the three strategies have similar Clf-F1.

C.4 Diagnostic Pressure Rules

Table 13 expands the compact mapping from observed validation signals to generator or predictor pressure used by VRSR.

D Scheduler-Dynamics Diagnostics

Since ReSR is a scheduler-based method, final test metrics are not the only useful basis for interpretation. The implementation logs the generator debt D_g , predictor debt D_p , selected next phase, rationale rate, fragmentation, mask drift, and probe/reference gaps at validation time. These signals make the scheduler behavior inspectable rather than opaque. In particular, D_g should increase when rationales are unstable or difficult for the probe to learn; D_p should increase when the online predictor lags behind a probe trained on the same

masked-input distribution. We therefore recommend plotting update phase ratios, D_g/D_p curves, rationale rate, and highlight F1 when analyzing new datasets or scheduler settings. These diagnostics are also useful for detecting failure cases such as mask collapse, predictor starvation, or shortcut rationales. Table 14 summarizes the intended interpretation of each logged signal, including which failure mode the signal checks and what behavior we expect when the scheduler works as intended.

This dynamic view is important for interpreting probe ablation. The probe does not have to maximize final highlight F1 in every table row to be useful. The probe’s intended role is to expose whether a poor online loss comes from predictor lag or a rationale that is not learnable even by a fresh predictor. Therefore, online-probe gaps, phase-allocation changes, and collapse-recovery behavior are more direct indicators of the probe’s diagnostic value than a single final F1 column. In the Table 4, removing the probe slightly lowers mean HI-F1 and increases variance, suggesting that the probe acts mainly as a complementary learnability diagnostic rather than a standalone prediction module.

E Use of AI Assistants

We use ChatGPT to improve the presentations of this paper¹.

¹<https://chatgpt.com/>

Dataset	Scheduler	Clf-F1 $\uparrow$	HI-F1 $\uparrow$	$R \downarrow$	$S \downarrow$
Toy	Full VRSR	99.97 $\pm$ 0.03	81.52 $\pm$ 8.76	14.64 $\pm$ 0.45	2.93 $\pm$ 0.09
Toy	FixedAlt	76.68 $\pm$ 34.97	48.95 $\pm$ 29.48	14.08 $\pm$ 6.29	2.82 $\pm$ 1.26
Toy	LossGap	73.54 $\pm$ 31.62	45.71 $\pm$ 29.53	24.42 $\pm$ 25.39	4.88 $\pm$ 5.08
HateXplain	Full VRSR	72.30 $\pm$ 1.01	41.92 $\pm$ 2.17	16.17 $\pm$ 2.02	2.16 $\pm$ 0.35
HateXplain	FixedAlt	72.23 $\pm$ 0.67	40.08 $\pm$ 4.48	18.06 $\pm$ 2.77	2.43 $\pm$ 0.39
HateXplain	LossGap	72.65 $\pm$ 1.06	39.55 $\pm$ 3.10	19.67 $\pm$ 1.74	2.70 $\pm$ 0.25

Table 12: Comparison of Full VRSR with two simpler scheduler controls over the same ten prespecified random seeds. FixedAlt performs explicit 1:1 alternation, while LossGap updates the side with the larger validation loss gap.

Diagnostic source	Observed pattern	Interpretation	Pressure
Online vs. probe	$L_{\text{online}} > L_{\text{probe}}$	Predictor lags behind a learnable rationale.	Predictor
Probe vs. reference	$L_{\text{probe}} > L_{\text{ref}} + \epsilon$	Current rationale lacks learnable signal.	Generator
Mask budget	Large $ R - r^* $	Selection rate misses the target budget.	Generator
Mask structure	High F or Δ	Rationale is fragmented or unstable.	Generator
Fairness guard	High age_g or age_p	One side has waited too long.	Older side

Table 13: Diagnostic pressure rules used by ReSR. Each source contributes pressure before VRSR allocates the next optimization slice.

Logged signal	Check	Expected behavior
Online-probe gap	Predictor lag	Larger gap raises predictor debt.
Probe-reference gap	Rationale learnability	Larger gap raises generator debt.
Phase ratio	Update allocation	Phase ratio follows debt curves.
Rate and collapse	Mask budget control	Collapse triggers generator recovery.
Across-seed variance	Training stability	Diagnostics reduce unstable runs.

Table 14: Scheduler-dynamics checks logged during validation. They indicate whether VRSR separates predictor lag from poor rationale quality before allocating the next optimization slice.