SPIRE: LET THE STUDENT CHOOSE ITS OWN SUPERVISION FOR IMPLICIT REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Implicit chain-of-thought reasoning compresses explicit reasoning traces into a short sequence of latent states. Recent methods enrich latent supervision with teacher key–value (KV) states, but identifying informative teacher states does not determine which states best supervise each student step. We investigate this allocation problem and find that teacher-state utility depends on the receiving student step, while preserving the teacher's reasoning order improves performance. We introduce SPIRE (Student-Selected Supervision for Implicit Reasoning), a framework in which the student chooses its own supervision. Given a compact pool of teacher states, SPIRE uses the student's current representations to jointly select an ordered, non-reusing supervision path, assigning a distinct teacher target to each active student step. We solve this selection exactly with dynamic programming and characterize its stability under score noise. SPIRE introduces no additional trainable routing parameters and leaves the latent inference procedure unchanged. Experiments on three arithmetic reasoning benchmarks and three model backbones demonstrate performance gains over implicit CoT baselines.

1 INTRODUCTION

Explicit chain-of-thought (CoT) reasoning provides a rich intermediate trajectory in natural language. Continuous latent reasoning pursues a complementary goal: expressing the same computation in a compact sequence of hidden states, without decoding every intermediate step into text (Hao et al., 2024; Shen et al., 2025; Wu et al., 2025). Learning such compact trajectories calls for supervision that transfers the structure of explicit reasoning into latent computation.

Recent methods have progressively enriched supervision for continuous latent reasoning. Coconut introduces a curriculum that replaces explicit rationale tokens with continuous thoughts (Hao et al., 2024); CODI aligns a designated implicit representation with its explicit self-teacher counterpart (Shen et al., 2025); and SIM-CoT reconstructs textual reasoning steps from latent states through a training-only decoder (Wei et al., 2026). KaVa brings supervision closer to the teacher's internal computation by distilling informative states from its key–value (KV) cache (Kuzina et al., 2026). Together, these advances make increasingly rich teacher signals available to the latent student. This progression primarily concerns what supervision to provide. An equally important question is how to allocate that supervision across the student's reasoning steps. For teacher-KV supervision, teacher-side importance identifies informative states, but does not account for which student step will receive them. A state that is informative for the teacher may not best supervise a given student step.

We take a student-centered view: *the student should choose its own supervision*. A long teacher trajectory offers many possible supervision targets, whereas the student reasons through only a small number of latent steps. The allocation should therefore reflect what each student step needs at its current stage of reasoning. Allowing the student to choose its targets turns teacher supervision into a path tailored to the student's own trajectory. Figure 1 illustrates this view: the teacher provides rich reasoning signals, and the student chooses the path through them.

Our preliminary study motivates three principles for student-selected supervision (Section 3). First, selection should reflect the receiving student step: the same teacher state can have different utility for different steps. Second, choices should be coordinated across student steps: independent selection often assigns the same teacher state to multiple steps, motivating a trajectory-level view of supervision. Third, the selected states should preserve the teacher's reasoning order: presenting the same states in

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

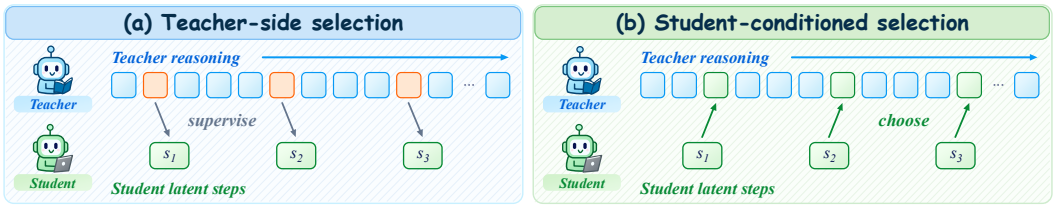


Figure 1: The teacher provides the signal; the student chooses its own supervision. **(a)** Teacher-side selection determines which states supervise the latent student. **(b)** Student-conditioned selection tailors these targets to the receiving student steps while preserving teacher order. Highlighted blocks are selected teacher states; downward arrows denote supervision and upward arrows denote the student's choice. This schematic illustrates training-time selection.

their original order improves accuracy over importance-score ordering. Together, these observations motivate a student-specific supervision path that follows the teacher's reasoning progression.

We introduce *SPIRE* (**S**tudent-Selected Su**p**ervision for **I**mplicit **Re**asoning), which formulates supervision allocation as a joint path-selection problem between a long teacher trajectory and a short student trajectory. Each student step selects a teacher state, and the selections together form an ordered supervision path. A teacher-side proposal first forms a compact candidate pool, which the student scores according to its current reasoning states. Dynamic programming finds the path with the highest total matching score while preserving teacher order and avoiding repeated selections. The selected keys and values provide step-specific training targets. The selector reuses existing model representations and introduces no additional trainable parameters. Teacher-state selection is used only to construct training targets; at inference, the student follows its original latent reasoning procedure.

Experiments on three arithmetic reasoning benchmarks and three model backbones show that SPIRE improves performance over implicit CoT baselines. Ablation studies demonstrate the benefits of student-conditioned selection and non-reusing supervision paths, while efficiency measurements show performance gains at comparable latent inference latency. Together, these results highlight supervision allocation as an effective dimension of implicit reasoning: the student benefits not only from richer teacher signals, but also from choosing how those signals guide its reasoning.

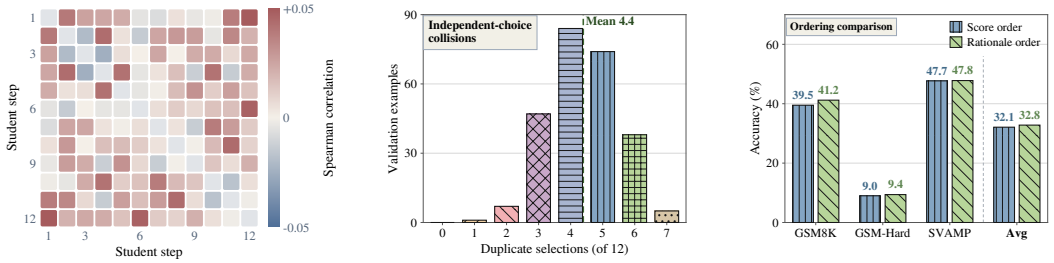
Our contributions are:

- We study supervision allocation in implicit reasoning. Preliminary studies motivate student-specific selection, cross-step coordination, and teacher-order preservation.
- We introduce SPIRE, which lets the student jointly select an ordered, non-reusing path of teacher states as supervision. We solve the selection objective exactly with dynamic programming and characterize the stability of the selected path under score perturbations.
- Experiments across three arithmetic reasoning benchmarks and three model backbones show gains over implicit CoT baselines. Ablations support student-conditioned selection and non-reusing paths, while efficiency measurements show comparable latent inference latency.

2 Related Work

Continuous latent reasoning and supervision. Continuous latent reasoning replaces textual steps with hidden states: Coconut (Hao et al., 2024) feeds them back recursively, while PCCoT (Wu et al., 2025) updates latent positions in parallel through Jacobi iteration. CODI (Shen et al., 2025) aligns a designated implicit representation with an explicit self-teacher, while LoRi (Solgi et al., 2026) aligns trajectories through low-rank factors. SIM-CoT (Wei et al., 2026) reconstructs textual reasoning steps through training-only decoders, and MUX (Suleymanzade et al., 2026) distills reasoning spans into multiplexed latent tokens. KaVa (Kuzina et al., 2026) provides dense supervision from teacher KV states selected using teacher-side importance and redundancy signals. SPIRE uses the student's current representations to jointly allocate teacher states across latent steps.

Structured selection. Classical sequence alignment uses dynamic programming to optimize matches jointly, as in Needleman–Wunsch (Needleman & Wunsch, 1970) and band-constrained



(a) **Student-step preferences.** Agreement between steps' teacher-state utility rankings.

(b) **Independent choices.** Duplicates equal 12 choices minus the number of distinct teacher states.

(c) **Ordering comparison.** Test accuracy with 12 latent steps using the same selected teacher states; Δ is rationale minus score order (percentage points).

Figure 2: Teacher-state allocation diagnostics. (a–b) A shared frozen intervention audit; correlations are averaged across examples, with gray self-correlations. (c) Ordering the same teacher states.

dynamic time warping (DTW) (Sakoe & Chiba, 1978). SPIRE jointly selects an ordered subsequence of distinct teacher states for supervision, unlike DTW's repeated matches.

3 Preliminary Study

An explicit rationale produces many internal teacher states, whereas a latent student has few reasoning steps. Using these states as supervision creates an allocation problem: which teacher state should guide each student step? We ask what properties useful supervision should satisfy.

3.1 Teacher-State Allocation

Problem formulation. Let the teacher and student trajectories be

$$\mathcal{T} = (\tau_1, \ldots, \tau_N), \qquad \mathcal{S} = (s_1, \ldots, s_K), \qquad K \ll N, \tag{1}$$

where teacher states follow reasoning-token order and the student has only K latent reasoning steps. Given these two trajectories, our goal is to identify the teacher states that provide the most useful supervision for the student's latent reasoning steps. We represent such an allocation by

$$\mathbf{a} = (a_1, \ldots, a_K) \in \{1, \ldots, N\}^K, \qquad a_i = j \iff \tau_j \text{ supervises } s_i. \tag{2}$$

We ask three questions: **RQ1:** Should teacher-state selection depend on the receiving student step? **RQ2:** What happens when student steps select teacher states independently? **RQ3:** Does preserving teacher reasoning order improve supervision?

Experiment setup. All three research questions use Qwen2.5-0.5B-Instruct with 12 latent reasoning steps. For RQ1 and RQ2, we analyze a pretrained PCCoT (Wu et al., 2025) model at the checkpoint with the best validation performance. We freeze its parameters and introduce individual teacher KV states into different student steps on 256 validation examples, measuring the resulting changes in answer loss. For RQ3, we select 12 teacher states using KaVa's scoring criterion (Kuzina et al., 2026) and compare two ways of assigning them to the 12 student steps: descending importance-score order and original rationale order. We evaluate the models on GSM8K (Cobbe et al., 2021), GSM-Hard (Gao et al., 2023), and SVAMP (Patel et al., 2021). Section 5.1 describes the data and evaluation. Appendix C.1 defines the utility statistics. Figure 2 summarizes the results.

3.2 RQ1: Should Teacher-State Selection Depend on the Receiving Student Step?

We test whether a teacher state is equally useful to different student steps. For each validation example, the frozen model produces an explicit reasoning trace and a 12-step latent trajectory. We

sample 12 token positions at regular intervals along the explicit trace and collect their keys and values, obtaining teacher states from different stages of reasoning.

To measure the effect of teacher state j on student step i, we temporarily mix 75% of the student's original KV with 25% of the teacher KV:

$$(K_i^S, V_i^S) \leftarrow (1-\epsilon)(K_i^S, V_i^S) + \epsilon(K_j^T, V_j^T), \qquad \epsilon = 0.25.$$

We apply this mixture at every layer and KV head, then recompute the answer-token cross-entropy. Each intervention starts from the original student trajectory, so changes do not accumulate across tests. We measure the utility of teacher state j for student step i by the reduction in answer loss:

$$U_{ij} = \mathrm{CE}_{\mathrm{base}} - \mathrm{CE}_{i \leftarrow j}. \tag{3}$$

Here $\mathrm{CE}_{\mathrm{base}}$ is the loss before intervention and $\mathrm{CE}_{i \leftarrow j}$ is the loss after mixing teacher state j into student step i. A positive U_{ij} indicates that the intervention improves answer prediction. Testing all 12 teacher states at all 12 student steps yields a 12×12 utility matrix for each example.

Each utility row ranks teacher states by their benefit to one student step. If a shared teacher ranking were sufficient, these rankings would agree across steps, and a teacher state preferred by other steps would also be useful to the receiving step. Instead, the rankings show little agreement: their mean pairwise Spearman correlation is 0.013 (Figure 2a). Selecting a teacher state according to its average utility at the other 11 steps also yields 0.01334 higher answer cross-entropy than selecting the receiving step's own best state. These findings motivate selection tailored to the receiving student step: *the student chooses its own supervision*. Details appear in Appendix C.1.

3.3 RQ2: What Happens When Student Steps Select Teacher States Independently?

RQ1 suggests that teacher-state selection should reflect the receiving student step. A direct approach is to let each step independently select the teacher state with the highest measured utility:

$$a_i^{\mathrm{local}} = \arg\max_j U_{ij}. \tag{4}$$

Applying this rule to the same utility matrices yields only 7.6 distinct teacher states across 12 student steps on average, corresponding to 4.4 duplicate selections (Figure 2b). Different steps can prefer the same teacher state, concentrating supervision on a small subset. This motivates coordinating choices across the student trajectory; Section 5.4 evaluates the effect of permitting reuse.

3.4 RQ3: Does Preserving Teacher Reasoning Order Improve Supervision?

After examining which teacher states to select, we ask whether their presentation order affects supervision. We compare two ways of arranging the same selected teacher states: **score order**, which sorts them by descending teacher-side importance, and **rationale order**, which follows their original positions in the teacher's reasoning trace. Both conditions use the same model, training data, selected states, latent budget, losses, and training schedule; only the order changes.

Rationale order improves accuracy by 1.7 percentage points on GSM8K, 0.4 on GSM-Hard, and 0.1 on SVAMP, yielding a 0.7-point average gain (Figure 2c). With the selected states held fixed, these results support preserving the teacher's reasoning order when allocating supervision to student steps. A supplementary 24-step comparison shows larger gains in the same direction: rationale order improves all three benchmarks and raises average accuracy by +2.6 percentage points (Appendix C.2).

Design implications. These studies motivate three principles for allocating teacher supervision: (1) select teacher states according to their usefulness to the receiving student step; (2) consider the choices across student steps together; and (3) preserve the original order of the selected teacher states. Section 4 instantiates these principles in SPIRE.

4 Methodology

4.1 Overview

SPIRE formulates teacher-state allocation as a *joint path-selection problem* (Figure 3). The same backbone runs in two modes: an explicit-rationale pass produces teacher keys and values, and a latent

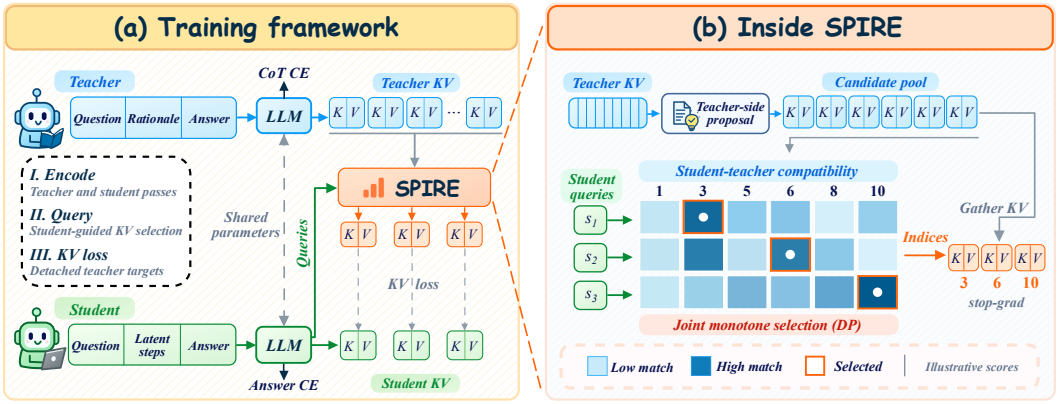


Figure 3: SPIRE training framework and internal selection mechanism. (a) Explicit and latent passes share a trainable backbone; selected teacher KV states provide detached targets for the corresponding student slots. (b) Inside SPIRE, each grid row represents a student step and each column a teacher candidate, labelled by its original position. Darker cells indicate higher query–key compatibility. Orange borders mark one jointly selected candidate per step, at strictly increasing positions. These indices retrieve original teacher KV targets. The three-slot grid is schematic, not measured routing data. Routing inputs are detached; the base hidden-state loss is omitted. Inference is unchanged.

pass produces the student's reasoning states. A teacher-side proposal first forms a compact pool of candidate states. Each student step then scores these candidates using its current representation. SPIRE considers the scores across steps jointly and selects a path that preserves teacher order and assigns a distinct teacher state to each active student step. The selected keys and values serve as step-specific training targets. Latent inference remains unchanged.

4.2 Candidate Construction and Student Scoring

Candidate construction. A long teacher rationale provides many potential supervision states. We first use teacher-side importance and redundancy scores to form a compact candidate pool, reducing the cost of subsequent joint selection while leaving the assignment of supervision to the student. Following previous work (Kuzina et al., 2026), we use a teacher-side R-KV compression signal: importance I_j measures the normalized similarity between answer queries and rationale keys, while redundancy R_j measures similarity to other rationale keys. Token j receives proposal score

$$u_j = \lambda I_j - (1-\lambda) R_j. \tag{5}$$

Here λ balances importance and redundancy. We keep the M valid positions with the highest proposal scores, or all valid positions if fewer than M are available, and arrange them in their original rationale order. Their token indices form the candidate set Ω.

Student scoring. To choose its supervision, each student step needs to assess how well a candidate teacher state matches its current reasoning state. Attention already provides representations for such matching: queries express what a position attends to, while keys represent the information available at other positions. We reuse the student's attention query and each candidate's teacher key to compute a cosine compatibility score, introducing no additional trainable scoring module. Both retain rotary positional encoding (RoPE; Su et al., 2021), preserving positional information. Joint selection (Section 4.3) enforces teacher chronology.

At layer ℓ, let $Q_i^{S,\ell,q}$ denote the attention query of student step i at query head q, and $K_j^{T,\ell,h}$ the key of teacher candidate j at KV head h, both after RoPE. Under grouped-query attention, multiple query heads share one KV head. Writing $\mathcal{G}(h)$ for this group of query heads, we average their cosine similarities to obtain the compatibility score:

$$C_{ij}^{\ell,h} = \mathrm{sg}\left[\frac{1}{|\mathcal{G}(h)|}\sum_{q\in\mathcal{G}(h)}\frac{(Q_i^{S,\ell,q})^\top K_j^{T,\ell,h}}{\|Q_i^{S,\ell,q}\|_2\,\|K_j^{T,\ell,h}\|_2}\right]. \tag{6}$$

A higher score indicates stronger query–key compatibility between student step i and teacher candidate j. Collecting these scores forms a matrix whose rows correspond to student steps and whose columns correspond to candidate teacher states. Unlike the teacher-side proposal score u_j, this score depends on the receiving student step. Here sg detaches student queries and teacher keys, blocking gradients through path selection. Training losses update the model through the selected targets.

4.3 Joint Path Selection

For one layer and KV head, write the compatibility scores as C_{ij}. Choosing each row independently can repeat teacher states or violate their reasoning order. SPIRE therefore selects the targets jointly, requiring teacher positions to advance with student steps (Figure 3b).

Let a_i denote the teacher position assigned to student step i. With K student steps and candidate set Ω, we select the path with the highest total compatibility score subject to this constraint:

$$\mathbf{a}^\star = \arg \max_{a_1 < \cdots < a_{K'},\, a_i \in \Omega} \sum_{i=1}^{K'} C_{i,a_i}, \qquad K' = \min(K, |\Omega|). \tag{7}$$

The constraint $a_1 < \cdots < a_{K'}$ preserves rationale order and prevents reuse by assigning successive steps to later teacher states. With fewer than K valid candidates, only the first K' student steps receive KV supervision. Selection is independent for each layer and KV head.

A dynamic program finds the optimal path; Section 4.5 gives the recurrence and exactness guarantee. The selected indices retrieve the original teacher keys and values, which serve as supervision targets for the corresponding student steps:

$$\widetilde{K}_i^{T,\ell,h} = K_{a_i^\star}^{T,\ell,h}, \qquad \widetilde{V}_i^{T,\ell,h} = V_{a_i^\star}^{T,\ell,h}. \tag{8}$$

4.4 Training Objective and Inference

Base training objectives. Following the shared-backbone distillation framework of PCCoT (Wu et al., 2025) and the teacher-KV supervision approach of KaVa (Kuzina et al., 2026), we train explicit and latent reasoning passes jointly. The explicit pass learns to generate the rationale and answer through cross-entropy $\mathcal{L}_{\mathrm{CoT}}$, while the latent pass learns to predict the answer from its latent reasoning states through cross-entropy $\mathcal{L}_{\mathrm{ans}}$. We retain PCCoT's hidden-state distillation objective, $\mathcal{L}_{\mathrm{hid}}$, which aligns the two passes' pre-answer hidden states across layers using standardized Smooth-L_1 loss.

Path-based KV supervision. Once the path is selected, each active student step is trained to match the keys and values of its assigned teacher state. We use mean squared error:

$$\mathcal{L}_{\mathrm{KV}} = \operatorname*{mean}_{\ell,h,i} \frac{\|K_i^{S,\ell,h} - \widetilde{K}_i^{T,\ell,h}\|_2^2 + \|V_i^{S,\ell,h} - \widetilde{V}_i^{T,\ell,h}\|_2^2}{2d_{\mathrm{head}}}, \tag{9}$$

where d_{head} is the KV head dimension and the mean includes only active slots across layers and KV heads. The total objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{CoT}} + \mathcal{L}_{\mathrm{ans}} + \beta_{\mathrm{hid}}\mathcal{L}_{\mathrm{hid}} + \beta_{\mathrm{KV}}\mathcal{L}_{\mathrm{KV}}. \tag{10}$$

Here β_{hid} and β_{KV} weight hidden-state and KV distillation, respectively; we use $\beta_{\mathrm{KV}} = 1$ by default. The teacher hidden states and selected KV states are treated as fixed targets within each update, so the distillation losses backpropagate through the student representations, not through the teacher targets. The explicit cross-entropy loss separately updates the shared backbone. The routing decision introduces no auxiliary loss or trainable module.

Inference. Inference uses PCCoT's latent reasoning and answer decoding (Wu et al., 2025), omitting the training-only explicit teacher pass, candidate construction, and path selection.

4.5 Exact Selection and Stability

Exact selection. We solve Equation 7 by extending the best compatible partial path at each student step. Let $m = |\Omega|$ and index the candidates by $1, \ldots, m$ in rationale order. For $K' = \min(K, m) \geq 1$,

let $D_i(j)$ be the highest total matching score for the first i student steps when step i selects candidate j. The preceding step must select an earlier candidate, giving the recurrence

$$D_1(j) = C_{1j}, \qquad D_i(j) = C_{ij} + \max_{t<j} D_{i-1}(t). \tag{11}$$

The optimal total score is $\max_j D_{K'}(j)$. A running maximum of the previous row evaluates all earlier candidates in one scan; stored predecessor indices then recover the selected path.

Proposition 1 (Exact joint selection)**.** *If at least one finite-score path is feasible, the recurrence in Equation 11 and traceback return a path maximizing the total matching score in Equation 7. Computing its value takes $O(K'm)$ time and $O(m)$ working memory; traceback stores $O(K'm)$ backpointers.*

Appendix A.1 proves the result and describes feasibility masks and deterministic tie handling.

Selection stability. The score margin is the difference between the best path's total score and that of its closest competitor. A larger margin means that a larger change in relative scores is needed to change the selected path. For a fixed candidate pool, Appendix A.2 formalizes this relationship under an independent score-noise model: the probability bound for selecting a different path decreases exponentially with the squared margin relative to the noise scale.

5 Experiments

5.1 Experimental Setup

Models and data. Following previous work (Wu et al., 2025; Kuzina et al., 2026), we study three model backbones: Qwen2.5-0.5B-Instruct and Qwen2.5-3B-Instruct (Qwen et al., 2025), and Llama-3.2-1B-Instruct (Meta, 2024). The common training dataset is GSM8K-Aug (Deng et al., 2024), with 385,620 examples. We evaluate on three arithmetic reasoning benchmarks: GSM8K (Cobbe et al., 2021), GSM-Hard (Gao et al., 2023), and SVAMP (Patel et al., 2021), with 1,319, 1,319, and 1,000 test examples, respectively. The code will be open-sourced once the paper is accepted.

Baselines. We compare SPIRE with implicit CoT methods: Coconut (Hao et al., 2024), CODI (Shen et al., 2025), PCCoT (Wu et al., 2025), KaVa (Kuzina et al., 2026), SIM-CoT (Wei et al., 2026), MUX (Suleymanzade et al., 2026), and LoRi (Solgi et al., 2026). We also include No-CoT, which predicts the answer directly, and Full-CoT, which generates an explicit reasoning trace before the answer, as references for reasoning without and with textual intermediate steps.

Training and evaluation. We conduct training on four NVIDIA H800 GPUs. For all three backbones, the common training protocol uses global batch 128 and 30,130 optimizer steps. We use 12 latent reasoning steps for SPIRE and the implicit baselines. Each baseline retains its own latent-state update mechanism. SPIRE uses three Jacobi iterations, adopting the iteration count from previous work (Wu et al., 2025). We retain the teacher-side proposal weight $\lambda = 0.1$ and hidden-state distillation weight $\beta_{\text{hid}} = 10$ used in previous work (Kuzina et al., 2026), and set SPIRE's candidate-pool budget to $M = 18$. We evaluate the checkpoint with the best validation accuracy. Appendix B provides the complete training settings and evaluation protocols.

5.2 Main Results

Overall performance. SPIRE leads the compared implicit CoT methods on every benchmark and backbone (Table 1), exceeding the strongest baseline's average accuracy by 1.3, 4.6, and 3.0 points on Qwen2.5-0.5B, Llama-3.2-1B, and Qwen2.5-3B, respectively.

Supervision along the student trajectory. CODI (Shen et al., 2025) supervises an answer-boundary representation; SPIRE supplies teacher-KV targets to individual latent steps. Against KaVa's teacher-KV supervision, student-selected allocation gains 1.3, 5.2, and 3.2 average accuracy points across the three backbones, improving every benchmark.

Table 1: Accuracy (%) across three backbones. Avg denotes the unweighted mean across tasks. **Bold** and underlined values indicate the best and second-best implicit results. Full-CoT is an explicit-reasoning reference.

	Qwen2.5-0.5B-Instruct				Llama-3.2-1B-Instruct				Qwen2.5-3B-Instruct			
Method	GSM8K	Hard	SVAMP	Avg	GSM8K	Hard	SVAMP	Avg	GSM8K	Hard	SVAMP	Avg
No-CoT	31.8	7.4	38.0	25.7	18.7	4.3	26.3	16.4	38.1	9.9	54.2	34.1
Full-CoT (explicit)	50.1	13.3	60.2	41.2	56.6	12.9	61.5	43.7	71.6	35.9	81.7	63.1
Coconut	31.9	6.7	41.9	26.9	36.8	8.5	50.6	32.0	38.1	11.1	56.2	35.1
CODI	12.7	2.9	30.9	15.5	15.2	3.6	32.7	17.2	15.3	3.9	52.6	23.9
PCCoT	33.1	7.8	44.9	28.6	37.5	8.9	51.7	32.7	36.2	10.8	58.4	35.2
KaVa	39.5	9.0	47.7	32.1	47.9	11.1	55.2	38.1	43.2	12.5	60.2	38.6
SIM-CoT	10.9	2.7	27.9	13.8	44.2	11.5	56.0	37.2	39.1	11.4	60.7	37.1
MUX	40.2	8.6	46.9	31.9	48.7	10.7	56.8	38.7	44.0	12.1	59.4	38.5
LoRi	38.8	9.4	45.4	31.2	47.2	11.6	57.0	38.6	42.5	13.0	61.0	38.8
SPIRE	**42.4**	**10.0**	**47.8**	**33.4**	**58.6**	**13.8**	**57.4**	**43.3**	**48.4**	**13.1**	**64.0**	**41.8**

5.3 Inference Efficiency

SPIRE offers a favorable accuracy–latency trade-off (Figure 4). On Qwen2.5-0.5B, it improves average accuracy over KaVa from 32.1% to 33.4% at essentially the same generation latency, 0.410 versus 0.412 seconds per example. Full-CoT achieves higher accuracy (41.2%) but requires 1.774 seconds per example, approximately 4.3 times SPIRE's generation time. Selection is training-only, so SPIRE improves accuracy without adding inference operations. All methods use the same 128 validation requests, with 16 warm-up requests and three timed repetitions. CUDA-synchronized generation time excludes model loading and preprocessing. Appendix D.3 details the timing protocol.

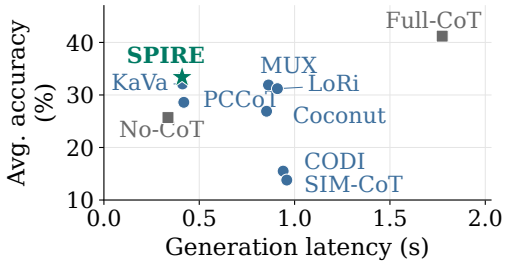


Figure 4: Accuracy–latency trade-off on Qwen2.5-0.5B (single H800, batch-one decoding). The star marks SPIRE; squares mark No-CoT and Full-CoT.

5.4 Ablation Study

Table 2 compares four selection variants on Qwen2.5-0.5B with the same training data, 12 latent steps, and 30,130-step training budget.

Table 2: Ablation study on Qwen2.5-0.5B (accuracy, %).

Variant	GSM8K	Hard	SVAMP	Avg
SPIRE	**42.4**	**10.0**	**47.8**	**33.4**
Teacher-only	41.2	9.4	47.8	32.8
Cross-Q	40.9	9.4	46.8	32.4
Reuse	23.2	4.9	33.0	20.4
Learned-Q	39.5	8.8	47.3	31.9

Teacher-only assigns the teacher's top 12 states in rationale order, achieving 32.8% average accuracy. With 12 candidates for 12 steps, the strict path is fixed. SPIRE expands the pool to 18 candidates and lets the student select its supervision path, reaching 33.4%.

Cross-Q substitutes another example's student queries without changing the teacher candidates or path constraints, testing whether the queries must match the example receiving supervision. Reuse permits repeated states without changing matching scores or teacher order, testing the distinct-target constraint. Their average accuracies are 32.4% and 20.4%, respectively, versus SPIRE's 33.4%. Learned-Q adds 2.75 million query-projection parameters to detached student states while retaining the Top18-to-12 path and teacher-KV targets; it averages 31.9%. SPIRE's reused queries perform better without additional routing parameters. Appendix D.1 details the ablation protocols. Appendix D.2 examines sensitivity to the candidate-pool size and KV supervision weight.

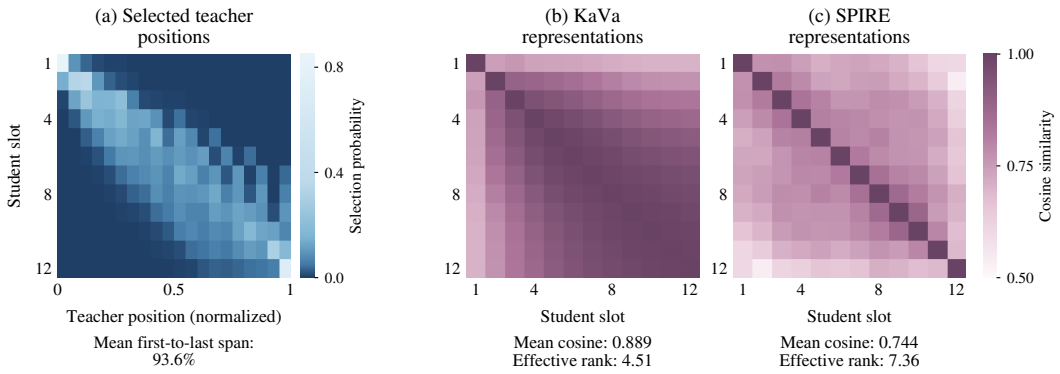


Figure 5: Selected teacher positions and student representations on Qwen2.5-0.5B. (a) Selected teacher-position distributions for each student step; positions are shown as percentages along the teacher trace. (b–c) Pairwise cosine similarity between the final student steps over all 256 validation examples, with a shared color scale; self-comparisons are masked in gray. Annotations report mean off-diagonal cosine and effective rank computed from the latent representations.

5.5 Analysis of Supervision Paths

We analyze SPIRE's selected teacher states, their allocation across steps, and student representations using trained Qwen2.5-0.5B models on 256 validation examples (Figure 5).

Selected teacher states. Figure 5a shows where each student step selects teacher states, with color indicating selection frequency. Earlier steps select earlier teacher positions, while later steps select later positions. On average, the first and last selected states span 93.6% of the teacher trajectory, providing supervision across a broad range of reasoning positions.

Coordination across steps. To examine whether preserving order alone produces diverse targets, we recompute order-preserving paths while allowing repeated selections, keeping the trained model, matching scores, and candidate pool unchanged. Repetition occurs in 99.5% of paths, with only 3.89 distinct teacher states per path on average. Thus, several student steps concentrate on the same targets. In the end-to-end ablation (Section 5.4), reuse reduces average accuracy from 33.4% to 20.4%. The path-overlap measurement holds the trained model fixed; the Reuse row instead measures the accuracy of a model trained with reusable targets.

Student representations. Figure 5b–c compares SPIRE and KaVa through pairwise cosine similarities among their 12 latent steps, averaged over the same validation examples. Gray diagonal cells mark self-comparisons; darker red indicates greater similarity between different steps on a shared color scale. SPIRE has lower mean inter-step similarity than KaVa (0.744 versus 0.889), indicating more distinct representations. Its higher effective rank (7.36 versus 4.51) also shows that these representations span more latent directions. Together, the position and representation measurements characterize both sides of the allocation: selected teacher states span the reasoning trace, while the student's latent steps show lower redundancy than KaVa's. These descriptive diagnostics complement the trained ablations above, which test how query alignment and target reuse affect accuracy. Appendix D.4 details how selection overlap, teacher positions, and representation metrics are measured.

6 Conclusion

We presented SPIRE, a framework that lets the student choose its own supervision for implicit reasoning. It jointly selects distinct teacher states in reasoning order to supervise each student step, reusing model representations without adding trainable routing parameters or changing latent inference. Experiments across three arithmetic reasoning benchmarks and three model backbones demonstrate performance gains over implicit CoT baselines, while ablations support student-specific selection and non-reusing paths. These findings highlight the value of letting students choose how teacher signals guide their reasoning.

Reproducibility Statement

The appendix describes the implementation of SPIRE and the experimental setup, including model configurations, training settings, checkpoint selection, and evaluation protocols. It also provides additional details on the preliminary studies, ablations, and analyses to facilitate reproduction of our results.

AI Use Statement

AI-assisted tools were used to polish the language, improve clarity, and organize the manuscript. The authors reviewed and revised the suggestions to ensure that the final text accurately reflects the methods, experiments, and findings, and take responsibility for the manuscript's content.

References

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. URL https://arxiv.org/abs/2110.14168.

Yuntian Deng, Yejin Choi, and Stuart Shieber. From explicit CoT to implicit CoT: Learning to internalize CoT step by step. *arXiv preprint arXiv:2405.14838*, 2024. URL https://arxiv.org/abs/2405.14838.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 10764–10799, 2023. URL https://proceedings.mlr.press/v202/gao23f.html.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

Anna Kuzina, Maciej Pioro, Paul N. Whatmough, and Babak Ehteshami Bejnordi. KaVa: Latent reasoning via compressed KV-cache distillation. In *International Conference on Learning Representations*, 2026.

Meta. Llama 3.2 model card, 2024. URL https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct.

Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3): 443–453, 1970. doi: 10.1016/0022-2836(70)90057-4.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2080–2094, 2021. URL https://aclanthology.org/2021.naacl-main.168/.

Qwen, An Yang, Baosong Yang, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2025. URL https://arxiv.org/abs/2412.15115.

Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978. doi: 10.1109/TASSP.1978.1163055.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. CODI: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025.

Ryan Solgi, Jiayi Tian, and Zheng Zhang. LoRi: Low-rank distillation for implicit reasoning. *arXiv preprint arXiv:2606.05315*, 2026.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021. URL https://arxiv.org/abs/2104.09864.

Ayhan Suleymanzade, Halil Alperen Gozeten, Michael Bronstein, Ismail Ilkan Ceylan, and Jinwoo Kim. MUX: Continuous reasoning via multiplexed tokens, 2026. Forthcoming.

Xilin Wei, Xiaoran Liu, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Jiaqi Wang, Xipeng Qiu, and Dahua Lin. SIM-CoT: Supervised implicit chain-of-thought. In *International Conference on Learning Representations*, 2026.

Haoyi Wu, Zhihao Teng, and Kewei Tu. Parallel continuous chain-of-thought with jacobi iteration. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025.

A Theory and Proofs

A.1 Exact Joint Selection

For m candidates indexed in rationale order, the strict path class is

$$\mathcal{A}^{<}_{K',m} = \{(a_1, \ldots, a_{K'}) \in \{1, \ldots, m\}^{K'} : a_1 < \cdots < a_{K'}\}. \tag{12}$$

Feasibility masks restrict this class to paths whose selected entries are all valid. We prove Proposition 1 for this feasible subset.

For a valid state (i, j), let $D_i(j)$ be the maximum cumulative score among all strict length-i paths ending at candidate j. The base case is $D_1(j) = C_{1j}$. Any strict path ending at (i, j) must have its preceding candidate at some $t < j$; conversely, appending j to any valid optimum ending at such a t yields a valid strict path. Optimal substructure therefore gives Equation 11. Induction on i proves that every $D_i(j)$ has the stated meaning, and $\max_j D_{K'}(j)$ is consequently the global optimum over the feasible subset of $\mathcal{A}^{<}_{K',m}$.

A left-to-right prefix maximum evaluates all $\max_{t<j} D_{i-1}(t)$ terms in linear time per row, and two score rows suffice to compute the optimal value. The production implementation retains one parent index per state for a deterministic one-pass traceback, hence $O(K'm)$ path-recovery memory. Candidate masks assign $-\infty$ to invalid states, with $\max \varnothing = -\infty$. The feasibility assumption ensures that the final row contains at least one finite state. Variable cardinality replaces K by $K' = \min(K, |\Omega|)$ and drops inactive tail rows. Fixed deterministic rules for prefix and terminal ties make the returned optimum reproducible. With no valid candidates, there is no active supervision path and no dynamic program is needed.

Size of the feasible path space. The strict class contains $\binom{m}{K'}$ paths. At $m = M = 18, K' = 12$, it contains 18,564 paths, compared with $\binom{29}{12} = 51{,}895{,}935$ weakly monotone paths with replacement and $18^{12} = 1{,}156{,}831{,}381{,}426{,}176$ independent assignments.

A.2 Stability under Score Perturbations

We ask when perturbing the matching scores changes the selected path. The reference matrix $\bar{C}$ defines the path to be retained; its margin measures how far its total score exceeds the best alternative. This is stability of the matching-score optimum under a specified noise model. For the analysis below, probabilities and expectations are conditional on the fixed candidate pool and feasibility masks.

Theorem 2 (Conditional path recovery)**.** *Condition on a fixed pool of m valid candidates and fixed feasibility masks, with at least one feasible path and $1 \le K' \le m$. Suppose $C_{ij} = \bar{C}_{ij} + \varepsilon_{ij}$, where $\bar{C}$ is a fixed reference score matrix. Under the conditional distribution, the score noises are independent, mean-zero, and σ-sub-Gaussian with $\sigma > 0$: for every real t, $\mathbb{E}[\exp(t\varepsilon_{ij})] \le \exp(t^2\sigma^2/2)$. Let*

$\bar{\mathbf{a}}$ *be the unique feasible maximizer under* $\bar{C}$*, and let* $\hat{\mathbf{a}}$ *be the maximizer under* C *returned by the deterministic tie rule. If every feasible alternative to* $\bar{\mathbf{a}}$ *has reference cumulative-score gap at least* $\Delta > 0$*, then*

$$\Pr(\hat{\mathbf{a}} \neq \bar{\mathbf{a}}) \leq \min\left\{1, \left[\binom{m}{K'} - 1\right] \exp\left(-\frac{\Delta^2}{4K'\sigma^2}\right)\right\}. \tag{13}$$

Proof. Condition on the candidate pool and masks as in the theorem. For an alternative $\mathbf{a} \neq \bar{\mathbf{a}}$, let

$$\Delta_{\mathbf{a}} = \sum_i \bar{C}_{i,\bar{a}_i} - \sum_i \bar{C}_{i,a_i} \geq \Delta. \tag{14}$$

The alternative matches or exceeds the reference-optimal path's perturbed score only if

$$\sum_i (\varepsilon_{i,a_i} - \varepsilon_{i,\bar{a}_i}) \geq \Delta_{\mathbf{a}}. \tag{15}$$

Terms shared by the two paths cancel. The remaining sum contains at most $2K'$ independent σ-sub-Gaussian variables with coefficients in $\{-1, 1\}$, so it is sub-Gaussian with variance proxy at most $2K'\sigma^2$. Its upper tail is therefore bounded by $\exp[-\Delta_{\mathbf{a}}^2/(4K'\sigma^2)]$, which is at most the exponential term in Equation 13. There are at most $\binom{m}{K'} - 1$ feasible alternatives, since masks can only restrict the strict path class. A union bound over these alternatives, capped at one, completes the proof. If there is only one feasible path, selection is unchanged for every perturbation. □

A.3 Candidate Restriction

Candidate construction reduces the search space. The following observation identifies when this restriction leaves the optimal matching score unchanged.

Proposition 3 (Candidate-pool containment)**.** *Fix a score matrix over* N *teacher states, a path length* K'*, and feasibility constraints. Let* $\mathcal{A}^{<}_{K',N}$ *denote the feasible full-teacher strict path class and let* $\mathcal{A}^{<}_{K',\Omega}$ *additionally restrict every selected state to* Ω*, retaining the same scores, path length, and constraints. If a full-space optimum* $\mathbf{a}^\star$ *is contained in* Ω*, then restricted and full optimization have the same optimal score. If the full optimum is unique, both return the same path.*

Proof. Because $\mathcal{A}^{<}_{K',\Omega} \subseteq \mathcal{A}^{<}_{K',N}$, the restricted score cannot exceed the full optimum. Containment makes $\mathbf{a}^\star$ feasible in the restricted class, so the restricted score cannot be smaller either. The scores are equal; uniqueness then forces path equality. □

Containment is a sufficient condition for zero search regret.

B Implementation and Experimental Details

Implementation details. The teacher-side proposal scores content before rotary position encoding; student scoring uses position-encoded queries and keys (Equations 5–6). The implementation scales normalized compatibility by $\sqrt{d_{\text{head}}}$, leaving the maximizing path unchanged while scaling its score and margin.

Training settings. Following previous work (Wu et al., 2025; Kuzina et al., 2026), SPIRE jointly trains explicit and latent reasoning passes with intermediate supervision. The common protocol across the three backbones uses 385,620 GSM8K-Aug training examples, global batch 128, and 30,130 optimizer steps. Training uses BF16, cosine learning-rate decay, a 0.03 warmup ratio, gradient clipping at 2, and LoRA rank/alpha/dropout 128/32/0.1 on query, key, value, attention output, and feed-forward down/up/gate projections. Table 3 lists the model-specific settings from the study protocol.

The implicit baselines use 12 latent steps and a shared evaluation protocol.

Evaluation protocol. We select checkpoints by validation exact match and use greedy decoding with batch size one on all 1,319 GSM8K, 1,319 GSM-Hard, and 1,000 SVAMP examples. GSM8K uses exact match of the answer segment; GSM-Hard and SVAMP use the last parsed number. Nonnumeric answers count as incorrect.

Table 3: Model-specific settings for the common training protocol.

Backbone	Learning rate	Weight decay	Sequence length
Qwen2.5-0.5B	5×10^{-4}	0.01	512
Llama-3.2-1B	8×10^{-4}	0.1	1024
Qwen2.5-3B	1×10^{-4}	0.01	512

C Preliminary Study Details

C.1 Teacher-State Preferences Across Student Steps

Each row of the utility matrix U from Section 3.2 describes how different teacher states affect one student step; each column describes how the same teacher state affects different steps. We use this matrix to measure step-dependent preferences and overlap among selections.

Step-dependent utility. Some teacher states may be useful across many steps, while some student steps may be more sensitive to intervention. To separate these average effects from pair-specific preferences, we subtract the row and column means and restore the overall mean:

$$R_{ij} = U_{ij} - \bar{U}_{i\cdot} - \bar{U}_{\cdot j} + \bar{U}, \qquad I(U) = \frac{\sum_{ij} R_{ij}^2}{\sum_{ij} (U_{ij} - \bar{U})^2}.$$

Here $I(U)$ is the interaction fraction; it is zero when total variation is zero. We then test how well preferences transfer between steps. For each step i, we select the teacher state with the highest average utility over the other steps,

$$j_{-i} = \arg\max_j \frac{1}{K-1} \sum_{r \neq i} U_{rj}, \qquad \mathrm{Regret}(U) = \frac{1}{K} \sum_i \left(\max_j U_{ij} - U_{i,j_{-i}} \right).$$

This regret measures the answer-loss cost of using other steps' preferences instead of the receiving step's own best choice.

Agreement and overlap. We compute Spearman correlation between pairs of student-step rankings and average the correlations within each example. Tied utilities receive average ranks. If all teacher states have the same utility for a step, its ranking provides no variation, so pairs involving that step are omitted. Examples with no remaining pairs are excluded from the correlation average. To measure overlap, we select the highest-utility state in each row and count the distinct selected indices. Other statistics are computed for every example and then averaged.

C.2 Teacher-State Ordering with 24 Latent Steps

We additionally compare score order and rationale order with 24 latent steps. Both conditions use the same selected teacher states and training settings, changing only their presentation order. Table 4 reports the results.

Table 4: Teacher-state ordering with 24 latent steps on Qwen2.5-0.5B (accuracy, %).

Order	GSM8K	GSM-Hard	SVAMP	Avg
Score	42.9	9.5	47.5	33.3
Rationale	47.5	11.1	49.1	35.9

Rationale order improves all three benchmarks, with a 2.6-point average gain, complementing the 12-step comparison in the main text.

D SUPPLEMENTARY EXPERIMENTS

D.1 ABLATION DETAILS

The variants in Section 5.4 share the training data, 12 latent steps, and optimizer-step budget. Their selection mechanisms differ as follows.

Teacher-only. The teacher-side proposal selects its top 12 states, which are assigned directly to student steps in rationale order. SPIRE instead selects 12 targets from a pool of 18 candidates, so this comparison changes both the candidate pool and the allocation rule.

Cross-Q. We cyclically permute student queries across examples within each local microbatch, so each example receives another example's queries. Its teacher candidates, teacher order, and remaining selection procedure stay unchanged.

Reuse. We replace the strictly increasing path constraint with a nondecreasing constraint. Multiple student steps can then select the same teacher state, while matching scores, candidate pools, and teacher order remain unchanged.

Learned-Q. Separate bias-free linear projections produce routing queries from the student's layer-local hidden states at the final latent iteration. Each layer and KV head has its own projection, adding 2.75 million parameters. The inputs are detached, so routing gradients update the projections without flowing into the student representations through this branch. The forward pass still selects a hard strict path from the Top18 candidate pool and uses the selected teacher keys and values as targets. During backpropagation, a straight-through estimator uses the soft path marginals computed by forward–backward dynamic programming to train the projections.

D.2 PARAMETER SENSITIVITY

We vary the candidate-pool size M and the teacher-KV loss weight β_{KV} on Qwen2.5-0.5B, changing one parameter at a time. The candidate-pool experiment fixes $\beta_{\mathrm{KV}} = 1$, while the loss-weight experiment fixes $M = 18$. Both retain 12 latent steps, three Jacobi iterations, and the remaining training and evaluation settings. Figure 6 reports average accuracy across GSM8K, GSM-Hard, and SVAMP.

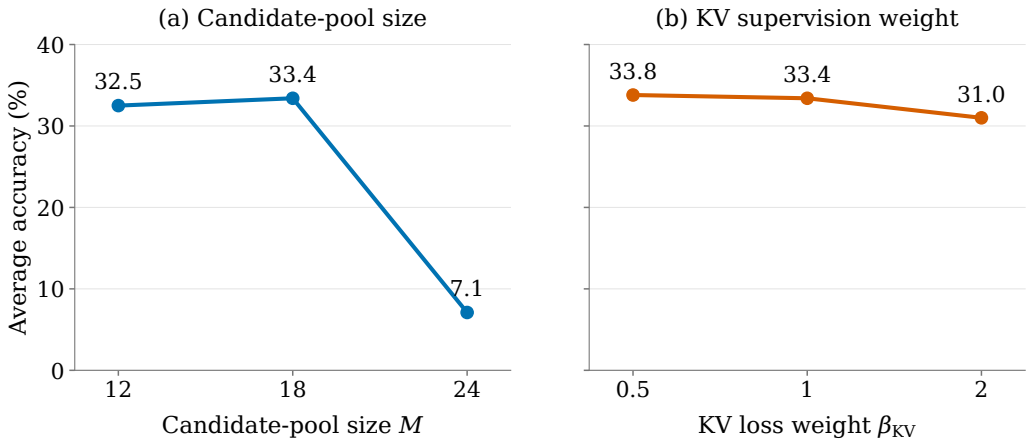


Figure 6: Parameter sensitivity on Qwen2.5-0.5B. (a) Candidate-pool size with $\beta_{\mathrm{KV}} = 1$. (b) KV supervision weight with $M = 18$. The student uses 12 latent steps throughout; accuracy is averaged across the three test benchmarks.

Candidate-pool size. Average accuracy rises from 32.8% at $M = 12$ to 33.4% at $M = 18$, then falls to 7.1% at $M = 24$. When $M = K = 12$, the strict path is unique: the teacher-selected candidates are assigned in rationale order, and student scores cannot change the assignment. A pool of 18 candidates leaves room for student-dependent selection. Expanding it to 24 sharply reduces performance in this experiment, showing that a larger candidate pool is not necessarily better.

KV supervision weight. Weights of 0.5 and 1 yield similar average accuracies, 33.8% and 33.4%, respectively; increasing the weight to 2 reduces accuracy to 31.0%. The tested settings therefore favor moderate KV supervision over a stronger distillation weight.

D.3 Inference Efficiency Details

We measure batch-one generation latency for all methods on a single NVIDIA H800 using PyTorch 2.7.0, CUDA 12.6, and Transformers 4.46.3. Each method runs in a fresh process on the same 128 validation requests, after 16 warm-up requests, with three repetitions. Timing excludes model loading and preprocessing and uses CUDA synchronization around generation. Mean latency is computed over the resulting 384 timed requests per method.

D.4 Supervision Paths and Representations

Selection overlap. To measure how student steps concentrate on the same teacher states, we reuse the trained SPIRE model's matching scores and candidate pools. We replace the strictly increasing path constraint with a nondecreasing constraint, preserving teacher order while allowing repeated selections.

For each example, layer, and KV head, a path is counted as repeated if any teacher state is selected more than once. Its distinct-state count is the number of different teacher indices it contains. We report the fraction of repeated paths and the mean distinct-state count across these groups. Each example contributes the same 24 layers and two KV heads, so this aggregation weights examples equally. The results are reported in Section 5.5.

Teacher coverage. The average span of 93.6% reported in Section 5.5 describes how far the selected supervision extends along the teacher trace. Position and span statistics are computed for validation examples that yield a supervision path. We normalize each selected state's position by its zero-based rank among the N valid teacher states, divided by $N - 1$. A path's span is the difference between its last and first selected positions; we average spans across layers and KV heads within each example. Figure 7 plots the fraction of examples whose mean span is at or below each horizontal-axis value. The median span is 92.3%, and 73.5% of examples have spans of at least 90%. Thus, broad early-to-late coverage occurs across examples, rather than being driven by a few long-span paths. Span measures the extent of the selected supervision, not the density of states selected within it.

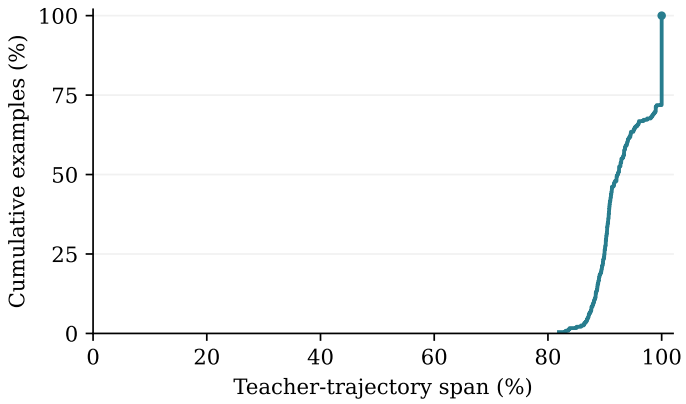


Figure 7: Distribution of teacher-trajectory spans. Each example contributes its mean span across layers and KV heads.

Representation geometry. We compare SPIRE and KaVa on the same 256 validation examples, capturing each model's 12 final-layer latent states after three Jacobi iterations, before projection and answer decoding. Parameters are frozen during capture, while the original training-mode computation, including dropout, is retained.

Mean inter-step cosine averages similarity between different steps, excluding self-comparisons. Lower cosine indicates less similar representation directions. For the uncentered state matrix $H \in \mathbb{R}^{12\times d}$, effective rank is

$$r_{\text{eff}}(H) = \exp\left(-\sum_i p_i \log p_i\right), \qquad p_i = \frac{\sigma_i}{\sum_j \sigma_j},$$

where σ_i are the singular values of H. It approaches one when a single direction dominates and reaches 12 when all 12 singular values are equal. We compute both metrics per example and then average across examples. SPIRE's lower cosine and higher effective rank therefore indicate both less similar steps and a more evenly distributed representation spectrum.