FORKLEFT: ENTROPY-FIRST ROLLOUTS FOR PREFIX-ALIGNED AUTOREGRESSIVE-TO-DIFFUSION DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Autoregressive Next-Token Prediction (NTP) has enabled strong reasoning capabilities in language models, while Diffusion Language Models (DLMs) offer flexible token orders and parallel generation. We ask whether DLMs can acquire NTP-style reasoning through distillation without giving up their native generation process. Direct distillation, however, faces a fundamental mismatch: an autoregressive teacher predicts from a left prefix, whereas a DLM can condition on tokens on both sides. We introduce FORKLEFT, a distillation framework that resolves this mismatch by separating the student's rollout from teacher supervision. During training, the student first performs entropy-first rollouts that commit uncertain positions and expose potential forks. We then fix the resulting student prefix and distill an NTP teacher under the same context, with answer correctness determining the supervision source. At inference, the student returns to its native confidence-first parallel decoding. With Qwen3-30B-A3B-Base, FORKLEFT improves Efficient-DLM-4B on all ten benchmarks, raising MATH500 from 72.60% to 79.60% and consistently outperforming three alternative designs. The gains scale with teacher strength and generalize to SDAR-4B with only 500 updates. At matched scale, the distilled 4B and 8B students exceed the published SDAR-Chat and OPDLM models on seven benchmarks, showing that DLMs can learn NTP-style reasoning without sacrificing native parallel generation. Code and datasets will be released upon acceptance.

1 INTRODUCTION

Autoregressive (AR) next-token prediction has produced a mature family of strong reasoning models. Diffusion language models (DLMs) offer what AR models lack: they resolve masked positions iteratively, write several tokens in parallel, and choose their own generation order (Nie et al., 2025; Ye et al., 2025). Distilling an AR teacher into a DLM student could transfer the reasoning capabilities of mature AR models to this more flexible generation paradigm, but the teacher predicts each token from a left prefix, whereas the student predicts it from whatever it has already written on either side.

Existing DLM post-training does not address this mismatch. Random masking trains on states that iterative decoding never visits (He et al., 2025), and confidence can become a shortcut rather than a measure of reasoning (Kim & No, 2026; Ni et al., 2026). Outcome-reward methods such as GRPO (Shao et al., 2024) credit many parallel decisions with one terminal scalar, so they cannot tell which token went wrong. On-policy distillation (OPD) (Su et al., 2026) supplies a denser signal, a teacher distribution at every position of the student's response, but must still decide which trajectory to teach on and under what information to compare a causal teacher with a bidirectional student. Borrowing both decisions from the student's inference procedure fails. The inference sampler commits its most confident positions first, so it resolves the hardest decision in a response, an uncertain fork, last, when its own tokens already surround it; Ni et al. (2026) observe this deferral of high-entropy tokens. Trained on such trajectories, the student never faces the fork on its own, and the teacher never sees a state at which to correct it. Comparing the two models in the student's bidirectional view fails for a different reason: to predict `C` in positions `A`–`E`, the teacher sees `AB` while the student may also see its own `D` and `E`, so a disagreement mixes what each model knows with what each model sees (Figure 1).

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.

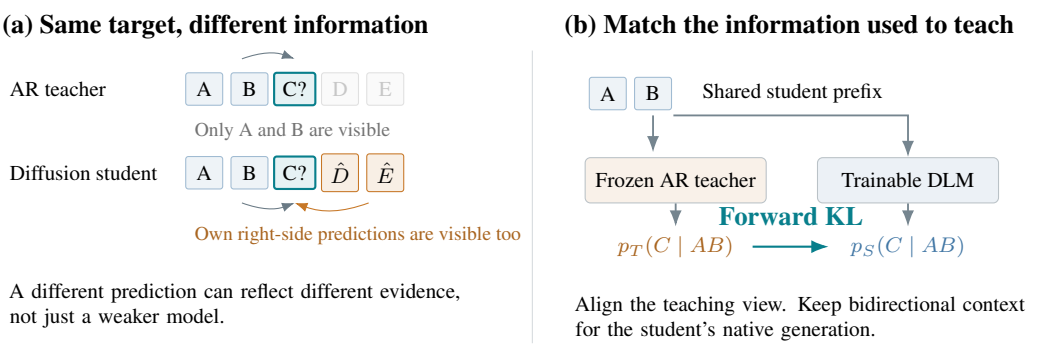


Figure 1: **A common target does not imply a common prediction problem.** (a) To predict C, the teacher sees AB; the student can also see its own $\hat{D}, \hat{E}$. Gray tokens are unavailable to the teacher; letters are schematic. (b) We teach with the same completed student prefix AB, hiding C and its suffix. Native generation remains bidirectional.

We propose FORKLEFT, which makes both decisions for learning rather than for decoding: *expose the uncertain fork, then teach on the left*. During training, the student commits its highest-entropy predictions first, so each fork is decided before later context can prop it up, and the teacher never rewrites the rollout, so an early mistake stays visible in the response. After generation, answer correctness decides how each question is supervised, and the teacher and the student predict each selected token from the same student-written left prefix. These tokens are drawn uniformly along the response rather than at the uncertain positions, so a confidently wrong token can also be corrected. At inference, the student returns to its native confidence-first parallel sampler.

We test the two decisions by crossing rollout order (high- or low-entropy first) with supervision view (shared prefix or bidirectional). Of the four configurations, entropy-first rollouts with prefix-aligned teaching score highest on all ten benchmarks. The full method improves Efficient-DLM-4B on every task, adds to SFT on the same questions, gains more from stronger teachers for both student sizes, generalizes to SDAR-4B, and at matched scale exceeds the published SDAR-Chat and OPDLM models on seven benchmarks.

We make three contributions:

1. **We identify two coupled failures in AR-to-DLM transfer.** Confidence-first rollouts hide uncertain forks, and bidirectional OPD compares teacher and student under different information. Reversing both yields a mean gain of $+4.93$ pp, whereas reversing only the supervision view yields $+2.31$ pp and only the rollout order $+0.14$ pp (Section 4.3).
2. **We introduce ForkLeft.** Its entropy-first student rollouts, correctness routing, prefix-aligned teacher distributions, and confidence ranking deliberately reverse generation priority between training and inference without replacing student tokens with teacher tokens.
3. **We establish broad effectiveness and an empirical teacher-scaling trend.** Across ten benchmarks, two student sizes, four teachers, and a second student family, ForkLeft yields consistent gains; its mean gain rises monotonically with teacher strength; and the distilled 4B and 8B students achieve state-of-the-art accuracy among published diffusion LMs of matched scale on seven benchmarks.

2 RELATED WORK

Post-training diffusion language models. MDPO trains on progressively generated states, block-wise SFT adapts supervision to block decoding, and reward optimization supplies outcome feedback (He et al., 2025; Sun et al., 2025). JustGRPO constrains RL training to left-to-right order and keeps parallel decoding at inference, avoiding a confidence-driven flexibility trap (Ni et al., 2026); the Confidence Shortcut identifies a related failure (Kim & No, 2026), and Kim et al. (2025) show that masked diffusion models train under harder orderings than the adaptive order they decode with.

ForkLeft uses this training–inference asymmetry to transfer token-level AR knowledge: uncertain decisions come first in training and last in inference.

On-policy distillation. OPD queries a teacher on student-generated contexts. OPDLM applies it to AR-to-DLM conversion, supervising the student's own trajectories with the frozen AR teacher's token-level distributions (Su et al., 2026); diffusion self-distillation uses later denoising states (Dat et al., 2026) or self-generated answer suffixes (Luo et al., 2026). ForkLeft designs both the trajectory and supervision view, comparing causal teacher and bidirectional student only on a shared student-generated prefix.

Learning from uncertain tokens. High-entropy tokens can concentrate learning signal (Wang et al., 2025; Xu et al., 2026). ForkLeft does not restrict loss to them: entropy chooses when a token is committed, while uniform sampling chooses where supervision lands. Uncertainty changes the trajectory rather than merely reweighting its loss.

3 METHOD

ForkLeft separates *writing a response* from *learning from it*. The student writes the training response with its native bidirectional model, committing uncertain positions first; the teacher then supervises selected positions under the student's own left prefix; a confidence regularizer shapes the priority the student will use after training; and inference returns to the native confidence-first sampler. We follow one update through these four steps in Figure 2: panel A is Section 3.1, panels B and C are Section 3.2, and panel D is Section 3.3 and Section 3.4. Entropy chooses which positions the rollout commits; uniform sampling chooses which positions receive a loss.

Consider five response positions, `A` through `E`, which denote schematic student outputs rather than reference answers. A native state might be `AB[M][M]E`: the student has written `A`, `B`, and `E`, and may use `E` to generate `C`. Once the response is complete, we mask `C` and its suffix and condition the teacher only on the completed prefix `AB` to predict `C`. This gives the teacher the same target under a well-defined left-to-right context while leaving the student's native bidirectional generation unchanged.

3.1 EXPOSE THE FORK: ENTROPY-FIRST STUDENT ROLLOUTS

We show one rollout pass in Figure 2A. Let P be a tokenized question, θ the student parameters, and ϕ the frozen teacher parameters. In a generation segment, the student starts with masked response positions. Before pass r, let $z^{(r)}$ be the current state, $\mathcal{M}_r$ its unresolved positions, and R the pass budget. One native forward pass produces a token distribution $p^{\text{den}}_{\theta,i}(\cdot \mid P, z^{(r)})$ at every unresolved position. We prioritize distributions with greater entropy H:

$$\mathcal{C}_r = \operatorname{TopK}_{i \in \mathcal{M}_r}\Big(\underbrace{H\big(p^{\text{den}}_{\theta,i}(\cdot \mid P, z^{(r)})\big)}_{\text{uncertainty at position } i};\ \underbrace{k_r}_{\text{commits this pass}} \Big), \qquad k_r = \left\lceil \frac{|\mathcal{M}_r|}{R - r + 1} \right\rceil . \tag{1}$$

Here $\mathcal{C}_r$ contains the k_r positions to commit. Each receives its argmax token from that same forward pass. The next pass recomputes predictions after all these commitments become visible. Crucially, the teacher neither chooses nor replaces these tokens. If an uncertain student prediction creates a wrong branch, later student tokens are generated on that branch; the completed response therefore records the consequence that on-policy supervision must repair. This order is the first axis of the comparison in Section 4.3: with the same prefix-aligned loss, committing high-entropy positions first adds 2.62 pp on average over committing low-entropy positions first.

In our example, C commits before D when it is more uncertain; if both commit in one pass, neither informs the other within that pass. We write the completed response as $c = (c_1, \ldots, c_L)$.

3.2 VERIFY, ROUTE, AND TEACH ON THE SHARED PREFIX

We route each completed response as in Figure 2B and teach in the view of Figure 2C. To teach C, both models receive AB and neither receives C, D, or E. A and B come from the completed response

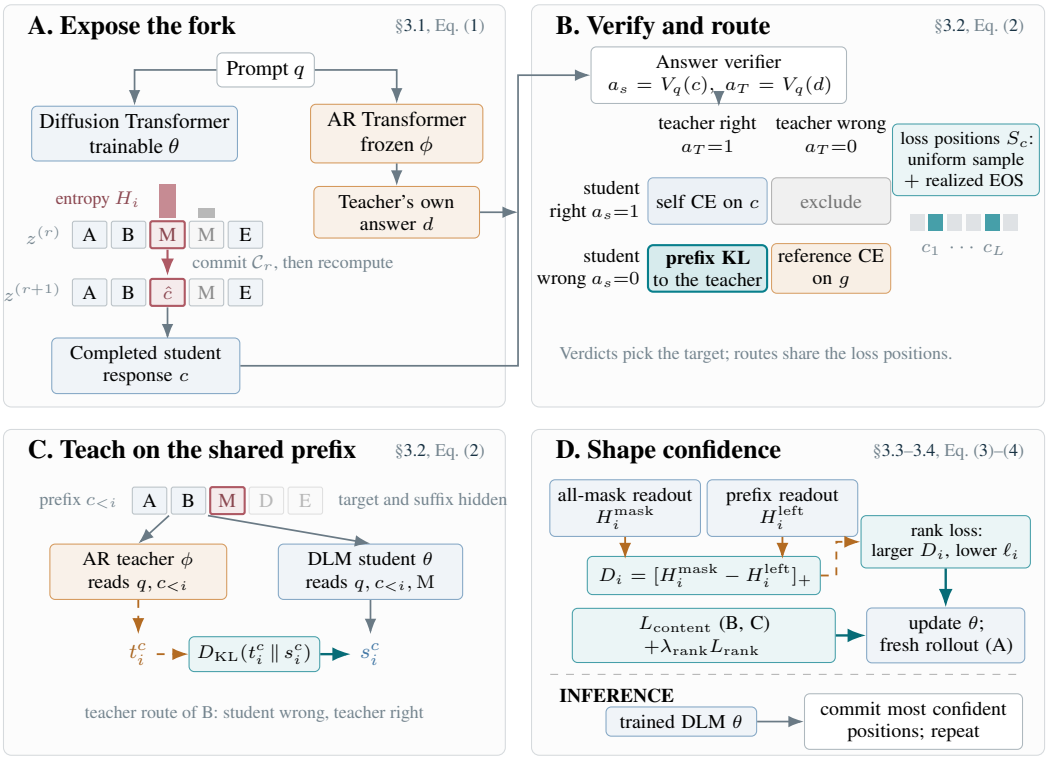


Figure 2: **ForkLeft exposes the fork, aligns the view, and keeps native inference.** One update on the running example A–E: an entropy-first rollout, with the frozen teacher answering only for verification (A); two verdicts select one of four routes, with loss positions sampled uniformly (B); both models predict the target from the same student prefix (C); prefix sensitivity ranks confidences and decoding stays confidence-first (D). Dashed orange arrows carry fixed targets; teal arrows carry gradients.

and need not both have been visible when C was committed, so prefix supervision does not impose left-to-right generation.

For any continuation y, write the teacher distribution as $t_i^y = p_\phi(\cdot \mid P, y_{<i})$ and the student's prefix readout as $s_i^y = p_{\theta,i}^{\ell}(\cdot \mid P, y_{<i})$; we compare normalized distributions over the same token events (the output-head adapter is in the appendix). We form S_y from uniformly sampled eligible positions plus the realized end-of-sequence (EOS) index, if present. Entropy does not select loss positions, so a confident error can still receive supervision.

A final-answer verifier marks the student response and a separate greedy teacher response d as correct or not, $a_s, a_T \in \{0, 1\}$, and the pair of verdicts selects one of four routes (Figure 2B). When both answers are correct, the student already has a working solution, so we reinforce its own tokens instead of pulling them toward the teacher. When only the teacher is correct, we match the teacher's prefix distributions; this is the one route that uses the teacher (Section 4.4 isolates it). When both are wrong, neither model offers a reliable target, so we supervise an available reference g under its own prefixes. When only the student is correct, we drop the question, because imitating a wrong teacher would unlearn a correct answer. Both asymmetric routes are common: on GSM8K the evaluated response pairs fall on them 206 and 105 times (Appendix F). On the teacher route we query the teacher on the student's AB prefix, which may differ from the prefix in the teacher's own answer. These choices give

$$L_{\mathrm{content}}(q) = \begin{cases} \mathrm{mean}_{i \in S_c}[-\log s_i^c(c_i)], & (a_s, a_T) = (1,1)\text{: both correct,} \\ \mathrm{mean}_{i \in S_c}[D_{\mathrm{KL}}(t_i^c \| s_i^c)], & (a_s, a_T) = (0,1)\text{: teacher only,} \\ \mathrm{mean}_{i \in S_g}[-\log s_i^g(g_i)], & (a_s, a_T) = (0,0)\text{: both wrong, } g \text{ available,} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Each mean divides by the selected-set size; an empty set contributes zero. Reference targets use reference prefixes. Exclusion removes both losses. The prefix view in the second route is the second axis of Section 4.3: under the same entropy-first rollouts, replacing the student's bidirectional readout by the prefix readout raises the mean gain from 0.14 to 4.93 pp.

3.3 Shape confidence with a prefix-sensitivity signal

We add a second, training-only objective (Figure 2D) that regularizes how confidently the student predicts selected positions. For each $i \in S_c$, compare an all-mask readout with the completed-prefix readout. Their positive entropy contrast is $D_i = \mathrm{stopgrad}([H_i^{\mathrm{mask}} - H_i^{\mathrm{left}}]_+)$. This quantity measures sensitivity to the two input constructions. It is a training proxy, rather than a token-correctness label.

Let $\ell_i = \max_u \log s_i^c(u)$ be the live prefix confidence and $\mathcal{P}_q = \{(i,j) : i,j \in S_c,\ D_i > D_j + \mu\}$ the qualifying pairs. With positive margins μ and δ, we use

$$L_{\mathrm{rank}}(q) = \frac{1}{|\mathcal{P}_q|} \sum_{(i,j)\in\mathcal{P}_q} \underbrace{[\ell_i - \ell_j + \delta]_+}_{\text{penalize } i \text{ more confident than } j}, \qquad L_{\mathrm{rank}}(q) = 0 \text{ if } \mathcal{P}_q = \varnothing. \tag{3}$$

For example, if $D_C > D_D + \mu$, this loss favors $\ell_C \le \ell_D - \delta$: a larger entropy contrast receives lower prefix confidence. The scores and pairs are detached; gradients pass through ℓ_i. Ranking uses the student response on every retained route, including reference supervision. The intent runs in three steps: the term is a training loss only; it lowers prefix confidence where revealing the prefix changes the prediction most; and because inference commits by confidence, the trained student should then defer those positions until more context is available. The last step is learned through shared parameters rather than enforced, which is why we call the term a surrogate.

3.4 Training objective and native inference

For a question batch $\mathcal{B}$, let $\mathcal{B}^+$ exclude the student-correct, teacher-wrong cases. We optimize

$$L(\theta) = \frac{1}{|\mathcal{B}|} \sum_{q\in\mathcal{B}^+} \left[L_{\mathrm{content}}(q) + \lambda_{\mathrm{rank}} L_{\mathrm{rank}}(q)\right]. \tag{4}$$

The original batch size remains the denominator, so excluded questions contribute zero, and λ_{rank} balances the two terms. Responses, verdicts, selected positions, and teacher targets stay fixed during an update, and each update collects fresh responses (Algorithm 1).

At inference, the student ranks unresolved positions by maximum token probability, commits the selected batch, and recomputes on its native bidirectional state; this confidence-first rule differs from minimum-entropy ranking, and only the student runs. The reversed order is deliberate: entropy-first training exposes an uncertain fork before its continuation can conceal it, whereas confidence-first inference lets the same position use as much reliable context as the sampler can provide.

4 Experiments

We ask four questions in turn. Does the complete method improve a student that keeps its native diffusion inference (Section 4.2)? Do entropy-first rollouts and prefix-aligned teaching each matter, and do they work best together (Section 4.3)? Do teacher distributions add information beyond SFT on the same questions (Section 4.4)? And do the gains grow with stronger teachers (Section 4.5) and generalize to a second student family (Section 4.6), and how do the distilled students compare with published diffusion models of the same size (Section 4.7)?

4.1 Evaluation setup

We trained Efficient-DLM-4B (Fu et al., 2026) on math-domain questions for 2000 updates with Qwen3-30B-A3B-Base (Yang et al., 2025), using a 1024-token response budget, 32-token blocks, and at most 32 passes per block. Our primary baseline is the same student checkpoint before distillation, which we call the *base*. We also evaluate EDLM-8B, distill SDAR-4B with the same teacher, and

compare with the published SDAR-Chat and OPDLM models at 4B and 8B. Qwen3-30B-A3B has 30B total and about 3B active parameters per token; we abbreviate it to Qwen3-30B.

We evaluate ten benchmarks with two protocols. GSM8K, MATH500, AIME24, AIME25, LiveMathBench-Hard (LMB-H), and MT-AIME2024 use generated answers from each diffusion family's native sampler with a 2048-token budget (4096 for MT-AIME). GPQA-Diamond, MLogiQA, CEval, and MMLU use masked-answer-slot likelihoods over the offered answer letters. Both protocols use the OPDLM prompt template and answer scorer. We compute accuracy and changes from correct counts before rounding, in percentage points (pp; details in Appendix C).

4.2 ForkLeft improves the student across all ten benchmarks

The first test is whether teaching under shared prefixes improves a student that keeps its native evaluation procedure. It does: EDLM-4B improves on all ten benchmarks (Table 1). MATH500 rises from 72.60% to 79.60%, GSM8K from 82.71% to 85.22%, and MT-AIME from 8.55% to 15.82%; the student need not use a causal sampler at test time.

Table 1: **ForkLeft improves EDLM-4B on all ten benchmarks.** The same student before and after 2000 updates with Qwen3-30B-A3B-Base; accuracies in percent, gains from unrounded counts; AR reference models in Table 5.

Benchmark	Items	Base	ForkLeft	Gain (pp)	Benchmark	Items	Base	ForkLeft	Gain (pp)
GSM8K	1319	82.71	**85.22**	+2.50	GPQA-D	198	30.81	**32.32**	+1.52
MATH500	500	72.60	**79.60**	+7.00	MLogiQA	800	45.63	**53.13**	+7.50
AIME24	30	10.00	**16.67**	+6.67	CEval	1346	70.73	**73.70**	+2.97
AIME25	30	6.67	**13.33**	+6.67	MMLU	14 042	71.04	**76.05**	+5.01
LMB-H	45	8.89	**11.11**	+2.22	MT-AIME	1650	8.55	**15.82**	+7.27

The gains also hold under multiple-choice likelihood scoring, which measures answer preference rather than generation: MLogiQA rises from 45.63% to 53.13% and MMLU from 71.04% to 76.05%.

The gains develop throughout training (Figure 3): every benchmark is nondecreasing across the five evaluated checkpoints, by 500 updates MATH500 reaches 76.80% and MLogiQA 50.25%, and eight benchmarks improve further between 1000 and 2000 updates while AIME25 and LMB-H remain level (counts in Table 3).

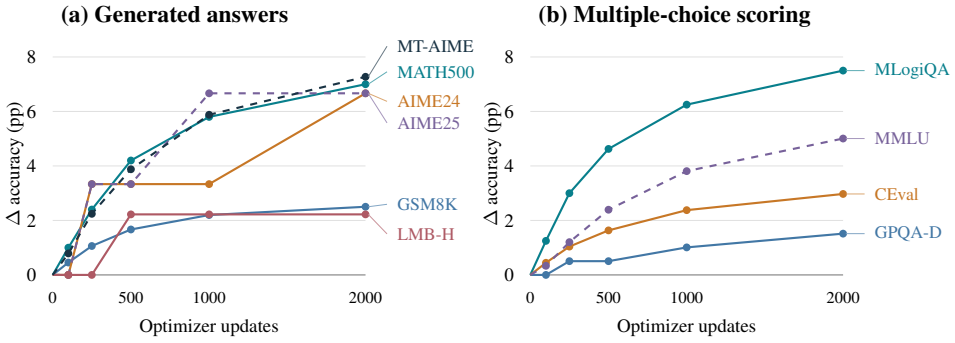


Figure 3: **The student continues to improve as training proceeds.** Accuracy gains over the EDLM-4B base at 100, 250, 500, 1000, and 2000 updates, grouped by evaluation protocol on shared scales. Every trajectory is nondecreasing; AIME25 and LMB-H plateau from 1000 to 2000 updates.

4.3 Prefix alignment and entropy-first rollouts work together

The main results establish that the full method helps. We next test its central design: should we change the order of student commitments, the teacher–student information view, or both? We compare two rollout orders (high or low entropy first) and two student views during OPD (prefix-only

or bidirectional). All four configurations generate training responses with the native bidirectional student; a prefix-only loss changes teaching, not the rollout.

The proposed combination wins every benchmark (Figure 4). Under the prefix view, high-entropy-first training strictly improves all ten tasks over low-entropy-first training; under high-entropy order, prefix-aligned OPD strictly improves all ten tasks over bidirectional OPD (per-benchmark counts in Table 4).

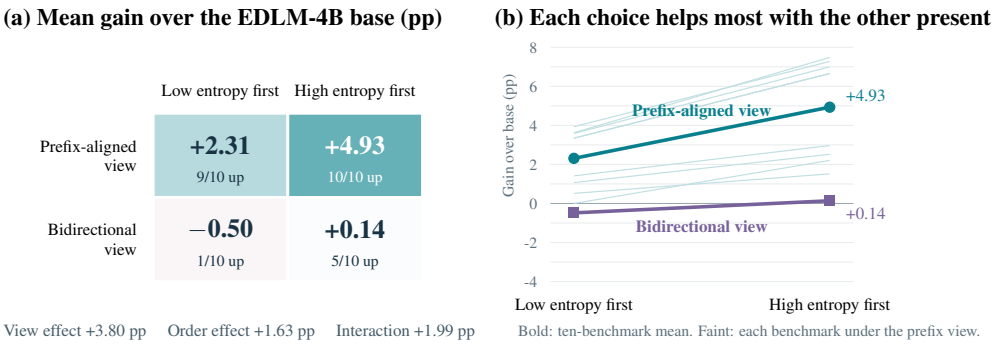


Figure 4: **Entropy-first rollouts and prefix alignment reinforce each other.** (a) Mean gain over the EDLM-4B base for the four order–view configurations, with the number of benchmarks that improve. (b) The same means as an interaction plot; faint lines are the prefix-view benchmarks.

The two changes reinforce each other. Averaged over the other factor, the prefix view adds 3.80 pp and entropy-first order 1.63 pp, but the order effect is 2.62 pp under the prefix view and only 0.64 pp under the bidirectional view, a descriptive interaction of 1.99 pp.

4.4 Teacher distributions add to SFT on the same questions

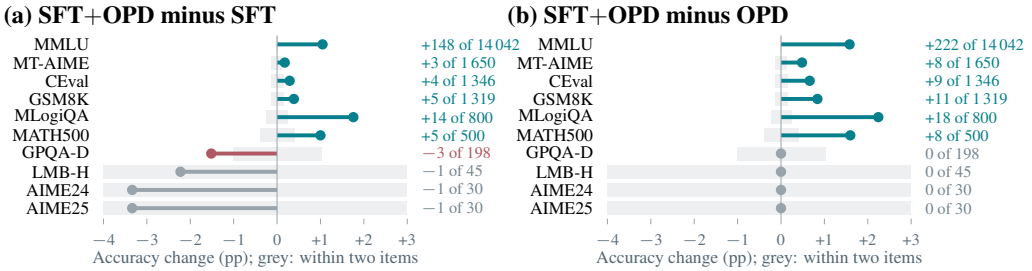


Figure 5: **Teacher distributions add to hard targets on the same questions.** All three objectives train on the teacher-correct, student-wrong questions of the main run; rows are ordered by benchmark size, right labels give the net change in correct answers, and grey marks changes of at most two items. (a) Adding OPD to SFT raises six benchmarks; GPQA-D is the only larger decrease. (b) Adding SFT to OPD lowers none.

Part of a distillation gain could come from practicing more correct answers rather than from the teacher's distribution. The teacher can add information only where it knows something the student does not, namely on questions it answers correctly and the student does not; in the main run the routing places about 30% of the training questions on this route. We train SFT, OPD, and SFT+OPD on exactly these questions, so the three objectives differ only in the supervision they apply. All three improve the base on nine or ten benchmarks, and SFT and OPD are close on most of them.

We draw two points from Figure 5. First, the objectives are complementary: adding OPD to SFT improves six benchmarks, by 148 net correct answers on MMLU, 14 on MLogiQA, 5 each on GSM8K and MATH500, 4 on CEval, and 3 on MT-AIME, and adding SFT to OPD lowers none.

AIME24, AIME25, and LMB-H move by a single item each, and GPQA-D, three items below SFT, is the only larger decrease. Second, the two signals differ in kind: SFT places its target on one token, whereas OPD supplies the teacher's distribution over alternatives after the student's own prefix. For a shared logit vector the gradients are $p - e_y$ and $p - t$, and any mixture of the two objectives targets the interpolated distribution (Section A.6), so the combination adds the teacher's ranking of alternatives to the hard target rather than repeating it.

4.5 Gains rise with teacher strength for both student sizes

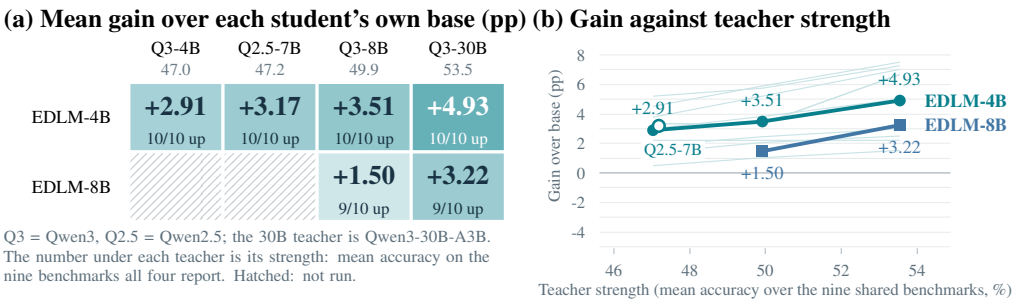


Figure 6: **Gains rise with teacher strength for both student sizes.** (a) Mean gain over each student's own base for every trained pair, with the number of benchmarks that improve. (b) The same means against teacher strength; faint lines are the EDLM-4B benchmarks, the hollow marker is the Qwen2.5-7B teacher, and no curve is fitted.

If ForkLeft transfers teacher information, a stronger teacher should produce a larger gain. We test Qwen3 teachers at 4B, 8B, and 30B-A3B and Qwen2.5-7B-Base for EDLM-4B, and Qwen3 at 8B and 30B-A3B for EDLM-8B, with gains relative to each student's own base; because the 30B teacher activates only about 3B parameters per token, we order teachers by strength, their mean accuracy on the nine benchmarks all four report.

The trend holds for both students (Figure 6). For EDLM-4B the ten-task mean gain rises from 2.91 pp with Qwen3-4B through 3.17 with Qwen2.5-7B and 3.51 with Qwen3-8B to 4.93 pp with Qwen3-30B, and the 30B teacher beats the 8B teacher on nine benchmarks and ties LMB-H. The 8B student, despite its stronger base, follows the same trend: its mean gain rises from 1.50 to 3.22 pp, and the larger teacher wins all ten benchmark comparisons against the 8B teacher; relative to the base, nine benchmarks improve and LMB-H changes from $8/45$ to $7/45$.

The rise is an empirical scaling trend: within the tested range, every increase in teacher strength increased the distilled gain for both student sizes, and the trend had not flattened at the strongest teacher. The points are actual checkpoints, including a mixture-of-experts teacher, so we fit no law (counts in Table 6).

4.6 The recipe generalizes to a second student family

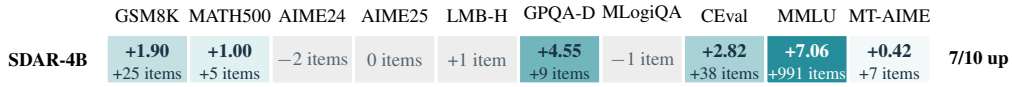


Figure 7: **ForkLeft generalizes to SDAR-4B.** Accuracy change per benchmark after 500 updates with Qwen3-30B-A3B-Base (pp and net correct answers); grey: within two items.

We ask next whether the recipe helps a diffusion family it was not developed on, and distill the same Qwen3-30B teacher into SDAR-4B (Cheng et al., 2026), a block-diffusion model converted from Qwen3-4B, for 500 updates. Seven benchmarks improve, including MMLU by 7.06 pp and GPQA-D by 4.55 pp, and the other three move by at most two items (Figure 7, Table 7).

4.7 Comparison with published diffusion models at matched scale

Table 2: **Distilled students exceed published diffusion models of the same size on seven benchmarks.** Accuracy (%); SDAR-Chat and OPDLM scores from Su et al. (2026) under our prompt template and scorer.

Model	MATH500	AIME24	AIME25	MLogiQA	CEval	MMLU	MT-AIME
SDAR-Chat-4B	72.80	10.00	7.50	46.50	62.90	74.90	3.00
OPDLM-4B	72.80	14.40	12.60	46.50	66.90	65.50	5.30
EDLM-4B + ForkLeft	**79.60**	**16.67**	**13.33**	**53.13**	**73.70**	**76.05**	**15.82**
SDAR-Chat-8B	78.60	10.00	10.00	46.30	70.20	78.60	4.00
OPDLM-8B	71.20	14.70	12.40	42.00	73.30	70.90	7.90
EDLM-8B + ForkLeft	**81.20**	**20.00**	**16.67**	**55.38**	**78.08**	**79.11**	**18.00**

We compare the distilled students with published diffusion models of the same size, SDAR-Chat (Cheng et al., 2026) and OPDLM (Su et al., 2026), using the scores that Su et al. (2026) report under our prompt template and answer scorer. We show in Table 2 the seven benchmarks on which both students score highest; they are the same at both scales: MATH500, AIME24, AIME25, MLogiQA, CEval, MMLU, and MT-AIME, and the margin over the better published model is largest on MT-AIME (10.5 and 10.1 pp at 4B and 8B) and smallest on MMLU (1.2 and 0.5 pp); the 4B student also exceeds both published 8B models on six of these seven benchmarks, all but MMLU. On the remaining three benchmarks, SDAR-Chat and OPDLM retain complementary strengths on GSM8K, GPQA-D, and LMB-H, likely reflecting differences in backbone specialization, instruction tuning, and decoding configuration (analysis in Appendix E). The advantage on the other seven tasks is learned rather than inherited from the starting checkpoints (base rows in Table 8): after distillation, each student matches or exceeds each published system on at least eight of the ten benchmarks. SDAR-Chat adds instruction tuning after conversion and OPDLM trains on under 0.1B tokens, so these are comparisons of published systems, not matched-budget controls.

5 Discussion and limitations

A teacher's generated answer is not the student's ceiling. EDLM-4B exceeds its teacher on MT-AIME ($261/1650$ versus $256/1650$), and EDLM-8B does so on MMLU (79.11% versus 77.79%). The student keeps its pretrained abilities and combines supervision sources, and teacher probabilities carry more than a greedy answer; we do not isolate these explanations, and on AIME25 the base already exceeds the teacher.

Scope. Our ablations isolate rollout order and supervision view, and evaluate routing, loss sampling, and confidence ranking only within the complete method. A correct teacher answer does not guarantee correct distributions on every student prefix, and training needs verified answers, teacher queries, and a shared tokenizer (Appendix C, Section F.3). The comparison in Section 4.7 uses reported scores, so it inherits the decoding and training differences listed in Appendix E.

6 Conclusion

ForkLeft commits the student's uncertain predictions early, teaches through shared left prefixes without rewriting the trajectory, and hands the result back to the native parallel sampler. Because the teacher, the verifier, and the confidence probes act only during training, the distilled student keeps the inference cost and parallelism of the original DLM. It improves EDLM-4B on all ten benchmarks, wins the order–view comparison on every one, adds to SFT on the same questions, gains more from stronger teachers, generalizes to a second student family, and exceeds published diffusion models of matched scale on seven benchmarks. The recipe also suggests a principle for flexible-order generators: the state that is best for learning is not the state that is safest for decoding. A student can face its forks during training, be corrected there under the information its teacher had, and still decode in the order it prefers. *Expose the fork for learning, align the teacher's view, and preserve confidence-first parallel decoding.*

AI USE STATEMENT

We used AI assistants, including OpenAI Codex, to draft and revise text, to survey related literature, and to prepare code and figures. The authors developed the conceptual framing and method, designed and executed all experiments, and verified all numerical results. The authors reviewed all AI-assisted material and remain responsible for the methods, experiments, claims, references, and final manuscript.

REFERENCES

Shuang Cheng, Yihan Bian, Dawei Liu, Yuhua Jiang, Yihao Liu, Linfeng Zhang, Qian Yao, Zhongbo Tian, Wenhai Wang, Qipeng Guo, Kai Chen, Biqing Qi, and Bowen Zhou. SDAR: A synergistic diffusion-AutoRegression paradigm for scalable sequence generation. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 22058–22075, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.1110. URL https://aclanthology.org/2026.findings-acl.1110/.

Phuong Tuan Dat, Qi Li, and Xinchao Wang. dOPSD: On-policy self-distillation for diffusion language models, 2026. URL https://arxiv.org/abs/2607.04428.

Yijia Fan, Jing Yang, Mingyu Liu, Kaitong Cai, Jian Wang, Keze Wang, and Jusheng Zhang. Reinforcement learning for diffusion LLMs via energy-based Gibbs alignment. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 45938–45948, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.2131. URL https://aclanthology.org/2026.acl-long.2131/.

Yonggan Fu, Lexington Whalen, Zhifan Ye, Xin Dong, Shizhe Diao, Jingyu Liu, Chengyue Wu, Hao Zhang, Enze Xie, Song Han, Maksim Khadkevich, Jan Kautz, Yingyan Celine Lin, and Pavlo Molchanov. Efficient-DLM: From autoregressive to diffusion language models, and beyond in speed, 2026. URL https://arxiv.org/abs/2512.14067.

Kairong Han, Nuanqiao Shan, Ziyu Zhao, Zijing Hu, Xinpeng Dong, Junjian Ye, Lujia Pan, Fei Wu, and Kun Kuang. C^2DLM: Causal concept-guided diffusion large language models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 821–838, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.40. URL https://aclanthology.org/2026.findings-acl.40/.

Haoyu He, Katrin Renz, Yong Cao, and Andreas Geiger. MDPO: Overcoming the training-inference divide of masked diffusion language models, 2025. URL https://arxiv.org/abs/2508.13148.

Dueun Kim and Albert No. The confidence shortcut: A reasoning failure mode of masked diffusion models, 2026. URL https://arxiv.org/abs/2605.29123.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions, 2025. URL https://arxiv.org/abs/2502.06768.

Nianyi Lin, Jiajie Zhang, Lei Hou, and Juanzi Li. Boundary-guided policy optimization for memory-efficient RL of diffusion large language models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7539–7550, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.343. URL https://aclanthology.org/2026.acl-long.343/.

Yifu Luo, Zeyu Chen, Haoyu Wang, Xinhao Hu, Yuxuan Zhang, Zhizhou Sha, and Shiwei Liu. Learning from the self-future: On-policy self-distillation for dLLMs, 2026. URL https://arxiv.org/abs/2606.18195.

Zanlin Ni, Shenzhi Wang, Yang Yue, Tianyu Yu, Weilin Zhao, Yeguo Hua, Tianyi Chen, Jun Song, Cheng Yu, Bo Zheng, and Gao Huang. The flexibility trap: Rethinking the value of arbitrary order in diffusion language models. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026. URL https://arxiv.org/abs/2601.15165.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan LI. Large language diffusion models. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 50608–50646. Curran Associates, Inc., 2025. doi: 10.52202/085713-1689. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/48b383b24230e0e6e649d9c98dae4d8c-Paper-Conference.pdf.

Jingyang Ou, Jiaqi Han, Minkai Xu, Shaoxuan Xu, Jianwen Xie, Stefano Ermon, Yi Wu, and Chongxuan Li. Principled RL for diffusion LLMs emerges from a sequence-level perspective. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2512.03759.

Leyi Pan, Shuchang Tao, Yunpeng Zhai, Zheyu Fu, Liancheng Fang, Minghua He, Lingzhe Zhang, Zhaoyang Liu, Bolin Ding, Aiwei Liu, and Lijie Wen. d-TreeRPO: Towards more reliable policy optimization for diffusion language models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 21801–21822, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.994. URL https://aclanthology.org/2026.acl-long.994/.

Kevin Rojas, Jiahe Lin, Kashif Rasul, Anderson Schneider, Yuriy Nevmyvaka, Molei Tao, and Wei Deng. Improving reasoning for diffusion language models via group diffusion policy optimization. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=JaqvespRBP.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Xingyu Su, Jacob Helwig, Shubham Parashar, Atharv Chagi, Lakshmi Jotsna, Degui Zhi, James Caverlee, Dileep Kalathil, and Shuiwang Ji. Data-efficient autoregressive-to-diffusion language models via on-policy distillation, 2026. URL https://arxiv.org/abs/2606.06712.

Bowen Sun, Yujun Cai, Ming-Hsuan Yang, and Yiwei Wang. Blockwise SFT for diffusion language models: Reconciling bidirectional attention and autoregressive decoding, 2025. URL https://arxiv.org/abs/2508.19529.

Xiaohang Tang, Rares Dolga, Sangwoong Yoon, and Ilija Bogunovic. wd1: Weighted policy optimization for reasoning in diffusion language models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=L2rfd2Czbj.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 115452–115486. Curran Associates, Inc., 2025. doi: 10.52202/085713-3850. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/a797c2d2e0c1fdabf4d1ab8cd0b465c6-Paper-Conference.pdf.

Hanchen Xia, Baoyou Chen, Yutang Ge, Guojiang Zhao, and Siyu Zhu. T*: Progressive block scaling for masked diffusion language models through trajectory aware reinforcement learning. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 808–816, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-391-3. doi: 10.18653/v1/2026.acl-short.67. URL https://aclanthology.org/2026.acl-short.67/.

Shaoan Xie, Lingjing Kong, Xiangchen Song, Xinshuai Dong, Guangyi Chen, Eric P. Xing, and Kun Zhang. Advancing reasoning in diffusion language models with denoising process rewards. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 42703–42720, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1978. URL https://aclanthology.org/2026.acl-long.1978/.

Yuanda Xu, Hejian Sang, Zhengze Zhou, Ran He, Zhipeng Wang, and Alborz Geramifard. TIP: Token importance in on-policy distillation, 2026. URL https://arxiv.org/abs/2604.14084.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7B: Diffusion large language models, 2025. URL https://arxiv.org/abs/2508.15487.

Huaisheng Zhu, MingYu Liu, Junze Liu, Zhen Ge, Tian Wang, Jiri Gesi, Dakuo Wang, Weiqi Zhang, Houyu Zhang, Yufan Guo, Xian Li, Bing Yin, and Sujay Sanghavi. On-policy self-distillation for efficient diffusion language models with early-stage calibration. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 26954–26965, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.1344. URL https://aclanthology.org/2026.findings-acl.1344/.

APPENDIX CONTENTS

A Formal definitions and update semantics **14**
A.1 Native and prefix readouts . . . 14
A.2 Rollout schedule and supervision routes . . . 14
A.3 Eligible positions, EOS, and normalization . . . 15
A.4 Ranking readouts, geometry, and gradients . . . 15
A.5 The sampling distribution and the update gradient . . . 15
A.6 Hard and distributional targets . . . 16
A.7 Native inference and the scope of the rationale . . . 16

B Training algorithm and update semantics **17**

C Implementation and evaluation details **17**
C.1 Training and evaluation protocol . . . 17
C.2 Run provenance and reproducibility . . . 17

D Complete numerical results **18**
D.1 Training progress . . . 18
D.2 Rollout order, teaching view, and supervision objectives . . . 18
D.3 Teacher and autoregressive reference models . . . 18
D.4 Teacher configurations for both student sizes . . . 19
D.5 Generalization to a second student family . . . 19

E Comparison with published diffusion models **20**

F Additional evidence and interpretation **21**
F.1 Useful right context is compatible with prefix teaching . . . 21
F.2 Correctness routing exploits response-level differences . . . 21
F.3 Resolution, dependence, and uncertainty . . . 21

G Additional related work **22**

A FORMAL DEFINITIONS AND UPDATE SEMANTICS

We specify the distributions, input views, and sampling rules behind the main-text update. These definitions preserve the generation and supervision procedures; they introduce no additional training stages.

A.1 NATIVE AND PREFIX READOUTS

Let q be a question and $P = \mathrm{tok}(q)$ its prompt. The student parameters are θ, and the frozen teacher parameters are ϕ. For token-aligned pairs, all distributions below are normalized on a common valid vocabulary $\mathcal{V}$ that excludes padding and mask outputs. Native denoising predicts an unresolved position from the configured masked state:

$$p^{\mathrm{den}}_{\theta,i}(\cdot \mid P, z^{(r)}), \qquad i \in \mathcal{M}_r, \qquad H(p) = -\sum_{u \in \mathcal{V}} p(u) \log p(u). \tag{5}$$

The state $z^{(r)}$ may contain student commitments on either side of i. The student's native attention pattern determines which positions it can read within that state.

For supervision, we use a complete prefix of continuation y:

$$\begin{aligned} p^{\ell}_{\theta,i}(\cdot \mid P, y_{<i}) &= \mathrm{Read}_\theta(P \parallel y_{<i}; i), \\ p^{\ell}_{\phi,i}(\cdot \mid P, y_{<i}) &= p_\phi(\cdot \mid P \parallel y_{<i}). \end{aligned} \tag{6}$$

A same-position masked-token head appends a fixed mask query and reads that position. A shifted next-token head reads the preceding logits corresponding to the target. Both receive the same textual prefix, with the target and suffix hidden. Matching text does not replace this output-index adapter. Changing suffix identities cannot change this prefix input, but a truncated prefix and a fixed mask canvas can have different sequence geometry.

The abbreviations s^y_i and t^y_i in the main text denote these student and teacher distributions. A direct token KL requires matching token events as well as matching text; every teacher–student pair in this paper shares a tokenizer.

A.2 ROLLOUT SCHEDULE AND SUPERVISION ROUTES

For each active generation segment, passes satisfy $1 \le r \le R$. We apply (1) while $\mathcal{M}_r$ is nonempty. A single forward pass supplies every selected argmax token. We write all selected tokens before the next forward pass, so a commitment cannot affect another prediction within the same pass. If R equals the number of initially masked positions, the schedule commits one token per pass; smaller pass budgets permit larger batches. Entropy priority is recomputed at each pass and need not decrease monotonically over the trajectory.

Let c be the completed student response, $d_\phi(q)$ the teacher's greedy response under fixed decoding settings, and $V_q(y) \in \{0, 1\}$ the final-answer verifier. We write the route explicitly as a function of both question and student response:

$$b(q, c) = \begin{cases} \mathrm{CE}, & V_q(c) = 1,\ V_q(d_\phi(q)) = 1, \\ \mathrm{LEFTOPD}, & V_q(c) = 0,\ V_q(d_\phi(q)) = 1, \\ \mathrm{GOLD}, & V_q(c) = 0,\ V_q(d_\phi(q)) = 0, \\ \mathrm{EXCLUDE}, & V_q(c) = 1,\ V_q(d_\phi(q)) = 0. \end{cases} \tag{7}$$

CE and LEFTOPD use student prefixes and selected student positions. GOLD uses the reference's own prefixes, tokens, and selected positions. Without a reference, GOLD contributes zero content loss but retains the student's ranking term. EXCLUDE removes both losses.

Teacher responses and verdicts can be cached for an unchanged prompt and decoding configuration. LEFTOPD distributions must nevertheless be queried on the current student prefixes. A successful teacher response selects a supervision source; it does not verify every teacher prediction on an erroneous student prefix.

A.3 Eligible positions, EOS, and normalization

Let L_y be the realized nonpadding response length. We preserve the specified candidate set $E_y = \{1, \dots, L_y - 1\}$, with $E_y = \varnothing$ when $L_y \le 1$. For $0 \le \rho \le 1$, define

$$\begin{aligned} &n_y = \min(\lceil \rho L_y \rceil, |E_y|), \quad U_y \sim \mathrm{Unif}\{U \subseteq E_y : |U| = n_y\}, \\ &S_y = U_y \cup \{e_y\}, && \text{if a realized termination EOS exists at } e_y, \\ &S_y = U_y, && \text{otherwise.} \end{aligned} \tag{8}$$

The union counts EOS once. A truncated response without EOS receives no fabricated termination target; its final index remains outside the specified candidate set. Padding is never a target. The exact response truncation and EOS extraction convention belongs to the evaluation and training configuration, rather than being inferred from these equations.

For any selected set S, write its mean as

$$A_S(f) = \begin{cases} |S|^{-1} \sum_{i \in S} f_i, & |S| > 0, \\ 0, & |S| = 0. \end{cases} \tag{9}$$

This is the operator $\mathrm{mean}_{i \in S}$ in (2). Forced EOS inclusion gives termination explicit weight, so the selected mean is not an unbiased uniform mean over all response tokens. KL first sums over the vocabulary, then averages over selected positions. The complete objective finally divides by the original question-batch size, as in (4). It does not divide by unsampled sequence positions or by the number of retained questions. This convention keeps excluded questions at zero weight without rescaling the remaining questions. If every question is excluded, we perform no update.

A.4 Ranking readouts, geometry, and gradients

At a selected student position i, let $p^{\mathrm{mask}}_{\theta,i}$ be the readout from the prompt and the configured fully masked continuation canvas. The probe hides the whole continuation, including positions outside an active rollout block. The prefix probe instead uses (6) on the completed student prefix. We define

$$\begin{aligned} H^{\mathrm{mask}}_i &= H\big(p^{\mathrm{mask}}_{\theta,i}\big), & H^{\mathrm{left}}_i &= H(s^c_i), \\ D_i &= \mathrm{stopgrad}\big([H^{\mathrm{mask}}_i - H^{\mathrm{left}}_i]_+\big), & \ell_i &= \max_{u \in \mathcal{V}} \log s^c_i(u). \end{aligned} \tag{10}$$

Both probes hide the target and use no reference future tokens. Because the configured all-mask canvas and truncated prefix may differ in geometry, D_i measures their entropy contrast rather than an intervention that changes only the revealed text. The positive part retains entropy reductions and discards increases.

We form $\mathcal{P}_q = \{(i,j) : i, j \in S_c,\ D_i > D_j + \mu\}$ using fixed scores during each update. For any qualifying pair, its hinge in (3) vanishes precisely when $\ell_i \le \ell_j - \delta$. Thus the ranking preference assigns lower prefix confidence to positions with greater positive entropy contrast. Only the live confidences receive ranking gradients; the scores, selected positions, and pair membership are detached. At a tie, we may use the gradient of an active maximizing token. Empty pair sets contribute zero.

Ranking always uses S_c and student prefixes, including on the GOLD route; reference content loss uses its separate set S_g. Content and ranking share student parameters and can favor different confidence levels. This regularizer defines a preference in the prefix view, while native-state behavior depends on the resulting shared parameters.

A.5 The sampling distribution and the update gradient

Let $\mathcal{D}_{\mathrm{train}}$ denote the question distribution and θ_k the student snapshot used to collect a batch. Under rollout order π, write $c \sim d^{\pi}_{\theta_k}(\cdot \mid q)$ for the conditional law of a completed response. With deterministic argmax decoding and fixed tie handling, this law may be a point mass. Let $\nu_\rho(\cdot \mid c)$ denote the uniform-plus-EOS selection rule in (8). The teacher-branch contribution is

$$\begin{aligned} \mathcal{L}_{\mathrm{OPD}}(\theta; \theta_k, \pi) = \mathbb{E}_{q \sim \mathcal{D}_{\mathrm{train}}} \mathbb{E}_{c \sim d^{\pi}_{\theta_k}(\cdot|q)} \mathbb{E}_{S \sim \nu_\rho(\cdot|c)} \\ \big[\mathbf{1}\{b(q,c) = \mathrm{LEFTOPD}\} A_S((D_{\mathrm{KL}}(t^c_i \| s^c_i))_{i \in S})\big]. \end{aligned} \tag{11}$$

Here t_i^c is frozen and s_i^c depends on the updated parameters θ. We do not condition the expectation on retained questions: the branch indicator and the original-batch mean preserve excluded questions' zero contribution. The CE and GOLD terms use the same question and response distribution, with their branch indicators and, for GOLD, independently selected reference positions. Their explicit finite-batch form is (4).

During optimization, we hold θ_k, sampled responses, verdicts, selected indices, teacher targets, and ranking targets fixed. The resulting gradient passes through live student readouts. It contains no policy-gradient term through generated tokens, commit order, or routing. We derive the local hard- and distributional-target gradients in Section A.6.

A.6 Hard and distributional targets

Let z be a logit vector, $p = \mathrm{softmax}(z)$ the student distribution, e_y the one-hot vector for a hard target, and t a fixed, normalized teacher distribution. At temperature one,

$$\nabla_z[-\log p_y] = p - e_y, \qquad \nabla_z D_{\mathrm{KL}}(t\|p) = p - t. \tag{12}$$

To obtain (12), write the KL as $\sum_u t_u \log t_u - \sum_u t_u \log p_u$ and differentiate $\log p_u = z_u - \log \sum_v \exp z_v$. The teacher-entropy term is constant with respect to student logits. The resulting difference lies in the target vector: a hard label places unit mass on one token, whereas the teacher distributes mass over alternatives. This calculation explains why the objectives can supply different updates; it does not imply orthogonal gradients or identify which answers each objective corrects.

The two targets can also be mixed. For $\alpha \in [0, 1]$,

$$\nabla_z\big[\alpha\, D_{\mathrm{KL}}(t\|p) + (1-\alpha)(-\log p_y)\big] = p - \big[\alpha\, t + (1-\alpha)\, e_y\big], \tag{13}$$

so any mixture pulls p toward the interpolated target $\alpha t + (1-\alpha)e_y$: $\alpha = 0$ recovers SFT, $\alpha = 1$ recovers OPD, and intermediate values add the teacher's ranking of alternatives to the hard target with weight α. The identity holds for one logit vector, that is, when both terms are evaluated on the same prefix. When the objectives use different prefixes, each has its own logit vector; the shared-subset comparison in Section 4.4 sums SFT and OPD losses computed on their own prefixes, so it realizes this mixture only where the prefixes coincide. We therefore use the identity to explain why the signals differ, not to assign the combined arm a single α. The shared-subset experiment therefore isolates supervision on a fixed set of questions. Because the three objectives construct targets under their own prefixes, it is a matched-example analysis rather than a token-normalized efficiency comparison.

A.7 Native inference and the scope of the rationale

At inference, the student uses its native masked states and confidence score. With its configured batch budget k_r, it selects

$$\mathcal{C}_r^{\mathrm{test}} = \mathrm{TopK}_{i \in \mathcal{M}_r}\left(\max_{u \in \mathcal{V}} p_{\theta,i}^{\mathrm{den}}(u \mid P, z^{(r)}); k_r\right). \tag{14}$$

Maximum token probability and minimum entropy are different ranking functions. Neither the frozen teacher nor the verifier or prefix probes participates in this inference procedure.

The equations above define the algorithm and its update. The shared-prefix rationale has a narrower mathematical support: for one fixed prefix, let t be a teacher distribution, let z index right contexts, and let $w_z > 0$ be context weights. If the student distributions s_z are unconstrained, the isolated objective $\sum_z w_z D_{\mathrm{KL}}(t\|s_z)$ has its unique minimum at $s_z = t$ for every supported context. Each KL is nonnegative and equals zero only when its two normalized distributions coincide. This conclusion remains true when t has zero-probability tokens, because normalization leaves no extra mass outside its support at equality. Thus a fixed prefix target rewards the same prediction across all these contexts in this isolated loss. Shared-prefix teaching avoids imposing that comparison on richer native inputs. This observation motivates the choice of view; it does not prove that the full regularized model discards useful right context. Likewise, entropy-first ordering and confidence ranking are design choices whose utility we assess empirically, rather than consequences of this KL fact.

B TRAINING ALGORITHM AND UPDATE SEMANTICS

We specify update order, loss normalization, and branch exclusions in Algorithm 1. Sampling and teacher predictions are fixed during an update. The frozen teacher cache is populated before routing.

Algorithm 1 FORKLEFT training and inference

Require: Student θ, frozen teacher ϕ, batch $\mathcal{B}$, verifier, reference continuations where available, teacher cache

Initialize retained-question loss list $\mathcal{A} \leftarrow \varnothing$
for each question $q \in \mathcal{B}$ **do**
  Generate student continuation c by the batched schedule in (1)
  if the teacher cache has no entry for the prompt and decoding configuration **then**
    Generate greedy teacher response d; cache d and $V_q(d)$
  end if
  Verify c and select $b(q, c)$ by (7)
  if $b(q, c) = \text{EXCLUDE}$ **then**
    continue ▷ no content or ranking loss
  end if
  Draw S_c by (8)
  Initialize $L_{\text{content}} \leftarrow 0$
  if $b(q, c) = \text{CE}$ or $b(q, c) = \text{LEFTOPD}$ **then**
    Average the selected CE or LEFTOPD loss over S_c
    Store the result in L_{content}
  else if $b(q, c) = \text{GOLD}$ and reference g is available **then**
    Draw S_g by (8); average reference CE on $P, g_{<i}$
    Store the result in L_{content}
  end if
  Compute detached D_i and qualifying pairs on S_c
  Compute live prefix-confidence ranking L_{rank} by (3)
  Append $L_{\text{content}} + \lambda_{\text{rank}} L_{\text{rank}}$ to $\mathcal{A}$
end for
if $\mathcal{A} \neq \varnothing$ **then**
  Update θ using $(\sum_{L \in \mathcal{A}} L)/|\mathcal{B}|$; discard rollouts
end if
Inference: native student sampling, most confident positions first; no teacher, verifier, or prefix-sensitivity probe

C IMPLEMENTATION AND EVALUATION DETAILS

C.1 TRAINING AND EVALUATION PROTOCOL

We used math-domain questions, a response budget of 1024 tokens, 32-token blocks, and at most 32 passes per block. The main EDLM-4B run uses 2000 updates; the SDAR extension uses 500 updates. Generated-answer benchmarks use each student's native sampler. AR reference models use greedy decoding. The generation budget is 2048 tokens, or 4096 for MT-AIME. Multiple-choice evaluation scores the offered answer letters at the masked answer slot. We use the OPDLM prompt template and answer scorer in both protocols.

C.2 RUN PROVENANCE AND REPRODUCIBILITY

Each result in this paper is produced by a completed training or evaluation run associated with the stated student checkpoint, teacher checkpoint, update budget, and decoding protocol. The primary EDLM-4B experiment uses 2000 updates, the transfer experiment uses 500 updates, and the checkpoint curve evaluates the same run at fixed intermediate updates. The order–view ablation shares its student initialization, teacher, training questions, and evaluation pipeline across all four configurations. The shared-subset experiment fixes the selected questions and changes only the supervision objective under study.

D COMPLETE NUMERICAL RESULTS

Every entry below is a correct-answer count with denominator n. Accuracy is $100c/n$; changes use the counts before rounding to two decimals. All main-text plots are computed from these exact counts. Cross-task averages, where used, weight the ten benchmark accuracies equally, not individual questions. No pooled total is treated as an accuracy estimate on independent samples.

D.1 TRAINING PROGRESS

We preserve the five checkpoint counts in Table 3.

Table 3: Correct counts for the EDLM-4B base and every measured training checkpoint.

Benchmark	n	Base	100	250	500	1000	2000
GSM8K	1319	1091	1097	1105	1113	1120	1124
MATH500	500	363	368	375	384	392	398
AIME24	30	3	3	4	4	4	5
AIME25	30	2	2	3	3	4	4
LMB-H	45	4	4	4	5	5	5
GPQA-D	198	61	61	62	62	63	64
MLogiQA	800	365	375	389	402	415	425
CEval	1346	952	958	966	974	984	992
MMLU	14 042	9976	10 023	10 145	10 312	10 511	10 679
MT-AIME	1650	141	154	178	205	238	261

D.2 ROLLOUT ORDER, TEACHING VIEW, AND SUPERVISION OBJECTIVES

We report the four order–view configurations and the three shared-subset objectives together in Table 4. The left block tests how we construct and teach a student response. The right block holds the teacher-correct, student-wrong questions fixed and changes the objective. The two comparisons answer different questions and should not be pooled.

Table 4: Counts underlying the order–view and shared-subset comparisons. B/L: bidirectional/left view; low/high: training entropy priority. The three rightmost columns use the same selected subset.

Benchmark	n	B-low	B-high	L-low	L-high	OPD	SFT	Both
GSM8K	1319	1087	1094	1105	1124	1112	1118	1123
MATH500	500	361	364	381	398	389	392	397
AIME24	30	2	3	4	5	4	5	4
AIME25	30	2	2	3	4	3	4	3
LMB-H	45	4	4	4	5	4	5	4
GPQA-D	198	60	61	62	64	63	66	63
MLogiQA	800	367	369	394	425	405	409	423
CEval	1346	947	950	971	992	982	987	991
MMLU	14 042	9960	10 041	10 296	10 679	10 436	10 510	10 658
MT-AIME	1650	138	144	206	261	251	256	259

D.3 TEACHER AND AUTOREGRESSIVE REFERENCE MODELS

We retain the AR checkpoints as contextual reference scores in Table 5. They are not matched post-training controls for the diffusion student. The primary method comparison remains the same student before and after distillation. Missing cells indicate unreported measurements, not zero accuracy.

Table 5: Reference checkpoint counts. Q3-30B, Q3-8B and Q2.5-7B are Base checkpoints; Q3-8B-I denotes the separately reported Qwen3-8B, and Math7B-I denotes Qwen2.5-Math-7B-Instruct. EDLM-8B is the base. These are reference scores, not matched post-training baselines.

Benchmark	n	Q3-30B	Q3-8B	Q3-4B	Q2.5-7B	EDLM-8B	Q3-8B-I	Math7B-I
GSM8K	1319	1192	1123	1160	1085	1118	1194	1178
MATH500	500	399	391	357	350	391	374	420
AIME24	30	11	2	4	5	4	3	3
AIME25	30	0	6	6	1	3	4	3
LMB-H	45	8	9	3	8	8	5	8
GPQA-D	198	71	64	74	64	67	66	62
MLogiQA	800	449	406	387	401	405	385	287
CEval	1346	1177	1101	957	1087	1010	1041	687
MMLU	14 042	10 923	10 438	9429	10 017	10 830	10 209	7427
MT-AIME	1650	256	176	–	–	224	–	–

D.4 Teacher configurations for both student sizes

We report all teacher configurations in Table 6. The Qwen3 rows supply the main teacher-scaling comparison. Qwen2.5-7B supplies an additional cross-generation teacher comparison, with a mean gain of 3.17 pp over EDLM-4B. We keep that configuration off the Qwen3 line in Figure 6 because teacher family changes together with size. Teacher strength on that figure's axis is each teacher's mean accuracy over the nine benchmarks that all four teachers report (MT-AIME is unreported for Qwen3-4B and Qwen2.5-7B): 47.0, 47.2, 49.9, and 53.5 for Qwen3-4B, Qwen2.5-7B-Base, Qwen3-8B-Base, and Qwen3-30B-A3B-Base, from the counts in Table 5.

Table 6: Teacher–student extension counts. Columns give student/teacher sizes. Teachers 4, 8, and 30 denote Qwen3-4B, Qwen3-8B-Base, and Qwen3-30B-A3B-Base; 7 denotes Qwen2.5-7B-Base.

Benchmark	n	4/4	4/7	4/8	4/30	8/8	8/30
GSM8K	1319	1109	1113	1118	1124	1140	1149
MATH500	500	382	385	389	398	398	406
AIME24	30	4	4	4	5	5	6
AIME25	30	3	3	3	4	4	5
LMB-H	45	5	5	5	5	6	7
GPQA-D	198	62	62	63	64	68	70
MLogiQA	800	401	407	412	425	430	443
CEval	1346	977	981	985	992	1038	1051
MMLU	14 042	10 397	10 434	10 513	10 679	10 957	11 108
MT-AIME	1650	227	233	236	261	275	297

D.5 Generalization to a second student family

We report the complete SDAR comparison after 500 updates in Table 7. The seven positive changes include 991 additional MMLU answers and 38 additional CEval answers. We retain the two declines and the unchanged task alongside those gains. The counts specify the size of each observed change without assigning it to noise.

Table 7: SDAR-4B with Qwen3-30B-A3B-Base: base and 500-update correct counts.

Benchmark	n	Base	500 updates	Δ correct
GSM8K	1319	1095	1120	+25
MATH500	500	359	364	+5
AIME24	30	4	2	-2
AIME25	30	2	2	+0
LMB-H	45	3	4	+1
GPQA-D	198	62	71	+9
MLogiQA	800	369	368	-1
CEval	1346	947	985	+38
MMLU	14 042	10 041	11 032	+991
MT-AIME	1650	144	151	+7

E Comparison with published diffusion models

We complete Table 2 in Table 8 with all ten benchmarks and with each student's base. The SDAR-Chat and OPDLM scores are those reported by Su et al. (2026): Table 1 of that paper for GSM8K, MATH500, AIME24, AIME25, LMB-H, GPQA-D, CEval, and MMLU, and Table 3 for MLogiQA and MT-AIME. That paper evaluates every benchmark under greedy static decoding with block size 4 and uses the prompt template and answer scorer that we adopt in Section 4.1; our students instead decode with each family's native sampler at a 2048-token budget (4096 for MT-AIME), so the prompts and scoring match while the decoding rule differs. That paper labels the SDAR rows SDAR-4B and SDAR-8B; their MMLU, GSM8K, MATH500, and GPQA-D scores equal the SDAR-Chat entries in Table 1 of Cheng et al. (2026), so we label them SDAR-Chat. They are instruction-tuned checkpoints and exceed the SDAR-4B base that Section 4.6 distills (Table 7; 89.9% against 83.02% on GSM8K).

The systems also differ in training. SDAR-Chat converts a Qwen3 base model to block diffusion and then applies instruction tuning; OPDLM converts with on-policy distillation on about 0.076B and 0.066B tokens at 4B and 8B, respectively, toward data efficiency; ForkLeft trains the converted Efficient-DLM checkpoints on math-domain questions for 2000 updates. We therefore read Table 8 as a comparison of published systems at matched parameter count, not as a controlled comparison of objectives.

On the remaining three benchmarks, SDAR-Chat and OPDLM retain complementary strengths on GSM8K, GPQA-D, and LMB-H. The records support each of the three differences named in Section 4.7. Starting checkpoint: before distillation, EDLM-4B and EDLM-8B score 82.71% and 84.76% on GSM8K, below both published systems at each scale, and EDLM-8B scores 33.84% on GPQA-D against 40.2% for SDAR-Chat-8B. Instruction tuning: our measurement of the SDAR-4B base before instruction tuning is 83.02% on GSM8K (Table 7), below the distilled EDLM-4B (85.22%), whereas SDAR-Chat-4B reports 89.9%, so that gap arises after conversion. Decoding: the published scores use greedy static decoding with block size 4, and our students use their native samplers, as described above.

Measured in test items, most of these gaps are small. GPQA-D trails SDAR-Chat-4B by about one of 198 items and OPDLM-8B by about two, and LMB-H trails OPDLM-8B by two of 45 items ($7/45$ against $9/45$) and ties OPDLM-4B at $5/45$. The clear gaps are GSM8K against SDAR-Chat, about 62 and 55 of 1319 items at 4B and 8B, GSM8K against OPDLM-4B, about 31 items, and GPQA-D against SDAR-Chat-8B, about ten items. These three are also the benchmarks on which ForkLeft adds least over its own base: the three smallest gains at 4B and three of the four smallest at 8B (Table 1). On the seven benchmarks where the students lead, the margin over the better published model is largest on MT-AIME (10.5 and 10.1 pp at 4B and 8B) and smallest on MMLU (1.2 and 0.5 pp).

The lead is not inherited from the starting checkpoints. Before distillation, EDLM-4B matches or exceeds each published system on only four benchmarks; EDLM-8B does so on six against SDAR-Chat and five against OPDLM. After distillation, each student matches or exceeds each published system on at least eight of the ten benchmarks.

Table 8: All ten benchmarks for the published models and the distilled students. SDAR-Chat and OPDLM scores as reported by Su et al. (2026); EDLM is each student's base and +ForkLeft the distilled student; bold marks the best of the three published or distilled systems per benchmark and scale.

	4B				8B			
Benchmark	SDAR-Chat	OPDLM	EDLM	+ForkLeft	SDAR-Chat	OPDLM	EDLM	+ForkLeft
GSM8K	**89.90**	87.60	82.71	85.22	**91.30**	87.10	84.76	87.11
MATH500	72.80	72.80	72.60	**79.60**	78.60	71.20	78.20	**81.20**
AIME24	10.00	14.40	10.00	**16.67**	10.00	14.70	13.33	**20.00**
AIME25	7.50	12.60	6.67	**13.33**	10.00	12.40	10.00	**16.67**
LMB-H	6.90	11.10	8.89	**11.11**	8.90	**20.00**	17.78	15.56
GPQA-D	**33.00**	29.10	30.81	32.32	**40.20**	36.10	33.84	35.35
MLogiQA	46.50	46.50	45.63	**53.13**	46.30	42.00	50.63	**55.38**
CEval	62.90	66.90	70.73	**73.70**	70.20	73.30	75.04	**78.08**
MMLU	74.90	65.50	71.04	**76.05**	78.60	70.90	77.13	**79.11**
MT-AIME	3.00	5.30	8.55	**15.82**	4.00	7.90	13.58	**18.00**

F Additional evidence and interpretation

F.1 Useful right context is compatible with prefix teaching

We compare target-masked bidirectional and prefix-only predictions in an existing static probe. Among $19\,115$ disagreements, the bidirectional view alone matches the reference token at $15\,998$ positions, the left view alone at 519, and neither at 2598. Their shares are 83.69%, 2.72%, and 13.59%. The left-only-correct cases account for $519/120\,453 \approx 0.43\%$ of all probed positions. These are static token predictions, not observed decoding transitions or answer-level errors. They motivate preserving useful native context while defining the teacher comparison under shared information.

Mean entropy is 0.6134 nats in the left view and 0.0438 in the bidirectional view. Entropy alone does not determine the gradient in (12): confident errors can still receive substantial updates. Uniform loss sampling covers these positions independently of rollout entropy.

F.2 Correctness routing exploits response-level differences

On GSM8K, the evaluated teacher/student responses are both correct on 986 questions, teacher-only correct on 206, student-only correct on 105, and both wrong on 22. On MATH500 the corresponding counts are 288, 111, 75, and 26. The teacher's net advantage is therefore 101 GSM8K answers and 36 MATH500 answers, smaller than the teacher-only-correct pool in each case. These counts explain the motivation for routing: an aggregate teacher advantage does not make the teacher's evaluated answer preferable for every question.

The response union is $1297/1319$ on GSM8K and $474/500$ on MATH500. It measures coverage under an answer verifier, not an achieved distilled score. These held-out verdicts describe evaluation responses; benchmark inference receives no training labels.

F.3 Resolution, dependence, and uncertainty

MT-AIME contains 55 translations of the same 30 questions, so its 1650 instances are not independent math problems.

One item changes accuracy by $100/n$ pp: 3.33 on a 30-item AIME set, 2.22 on LMB-H, and approximately 0.51 on GPQA-D. Net correct counts help interpret those increments but are not confidence intervals. Marginal before/after totals do not identify which questions were recovered or lost, and cannot supply the paired-discordance counts for a significance test.

G ADDITIONAL RELATED WORK

Policy optimization for diffusion reasoning. Recent work adapts reinforcement learning to the non-autoregressive structure of diffusion language models. wd1 reweights policy updates to improve reasoning, ESPO develops a sequence-level view of the diffusion policy, and GDPO extends group-based policy optimization to diffusion generation (Tang et al., 2026; Ou et al., 2026; Rojas et al., 2026). These methods must assign a sequence-level outcome to many token decisions made across denoising steps and possible generation orders. They primarily improve how reward is estimated or optimized; their supervision still originates from outcome quality rather than the full token distribution of a stronger causal model. This distinction is important for AR-to-DLM transfer, where the teacher can provide dense information about plausible alternatives even when two responses receive the same final reward.

Trajectory structure and process-level credit. A second line of work exposes more structure inside the denoising trajectory. d-TreeRPO organizes rollouts as a tree and propagates advantages bottom-up, providing finer credit than a single terminal reward (Pan et al., 2026). DGA instead casts alignment as energy-based Gibbs distribution matching, avoiding direct dependence on a tractable sequence likelihood (Fan et al., 2026). BGPO derives a memory-efficient lower-bound objective that permits more accurate likelihood approximation with larger Monte Carlo sample sets (Lin et al., 2026), whereas DPR estimates rewards for intermediate denoising intervals so that optimization can act on the reasoning process rather than only its final answer (Xie et al., 2026). These approaches refine credit within diffusion generation, but do not by themselves resolve which context a bidirectional student and a causal teacher should share when their token distributions are aligned.

Self-distillation, causal structure, and block curricula. Other methods change the source or timing of supervision. COPSD transfers information from later, better-calibrated decoding states to earlier states, targeting inaccurate early predictions through on-policy self-distillation (Zhu et al., 2026). Related self-distillation methods use later denoising states or self-generated future continuations as targets (Dat et al., 2026; Luo et al., 2026). C^2DLM introduces teacher-derived causal concept structure into the student's attention pattern, encouraging reasoning to follow relations between semantic concepts rather than unconstrained token interaction (Han et al., 2026). $T^\star$ addresses another aspect of the trajectory: it progressively increases the diffusion block size while re-optimizing the policy, allowing the model to acquire greater parallelism without abruptly changing its decoding regime (Xia et al., 2026). JustGRPO likewise demonstrates that the order used to learn need not be the order used to decode, using a constrained training order while retaining flexible parallel inference (Ni et al., 2026).

Autoregressive-to-diffusion transfer. Efficient-DLM studies efficient conversion from autoregressive checkpoints to diffusion language models while preserving the speed benefits of parallel generation (Fu et al., 2026). OPDLM further introduces on-policy distillation for this setting: the diffusion student generates trajectories, and a frozen autoregressive teacher supplies token-level distributions on student-induced contexts (Su et al., 2026). This establishes a substantially denser learning signal than outcome-only optimization, but it leaves two diffusion-specific choices open: which student states should be exposed during training, and under what information the causal teacher and bidirectional student should be aligned. ForkLeft addresses these choices explicitly. It uses entropy-first student rollouts to expose uncertain branches, retains the student's decisions rather than replacing them with teacher tokens, and then performs distillation under a shared left-prefix view. Native confidence-first parallel decoding is restored at inference, separating the state most useful for learning from the order most stable for generation.