TRAINING PARALLEL SPECULATIVE DRAFT MODELS BY DIRECTLY MINIMIZING EXPECTED DECODING ROUNDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Speculative decoding accelerates large language model inference by using a low-cost draft model to propose tokens that the full-size target model verifies in parallel. Parallel and semi-autoregressive (semi-AR) drafters improve drafting efficiency by proposing an entire block in a single forward passes, but training them raises a new difficulty: the draft distribution for a given position depends on where the decoding round starts, and where rounds start depends on how many tokens earlier rounds accepted. Existing training objectives typically rely on block-local surrogates that ignore this cross-round coupling, and therefore do not directly optimize the global decoding efficiency. In this work, we develop a theoretical framework for training and evaluating these drafters by representing speculative decoding as a Markov reward process. This formulation yields the Expected Decoding Rounds (EDR) objective, which weights local rejection costs by state occupancies and exactly equals the expected number of decoding rounds. Unlike prior surrogate objectives, EDR introduces no auxiliary hyperparameters. We then derive an exact temporal-difference gradient that supports unbiased stochastic optimization from target-model rollouts. The same framework also yields an exact offline evaluator for round counts, enabling paired drafter comparisons on shared target rollouts without running speculative decoding. Finetuning two state-of-the-art drafters, DSpark and DFly, with EDR consistently improves mean accepted length and outperforms existing training objectives across nine benchmarks spanning math reasoning, code generation, and chat.

1 INTRODUCTION

Autoregressive (AR) generation is a central bottleneck in large language model (LLM) inference. Since each token can be produced only after its preceding tokens are available, the number of function evaluations (NFEs) required to generate a text sequence grows linearly with its length.

Speculative decoding (Leviathan et al., 2023; Chen et al., 2023) reduces this sequential cost by using a lightweight *draft model* to propose multiple candidate tokens, which the full-size *target model* verifies in parallel in a single forward pass. With rejection-sampling verification, the generated sequence follows the target model's distribution exactly. Therefore, speculative decoding accelerates inference without sacrificing quality, and the goal is to reduce expensive target model verifications—equivalently, decoding rounds—while keeping drafting itself inexpensive.

Early work employed AR drafters (Li et al., 2024b) that predict draft tokens sequentially. To reduce the sequential work of drafting, parallel drafters (Cai et al., 2024; Chen et al., 2026) have emerged to propose multiple draft tokens in one forward pass, typically built on masked diffusion language models (Sahoo et al., 2024; Shi et al., 2024). Their semi-AR variants (Cheng et al., 2026; Liu et al., 2026b) add limited sequential dependence within each draft block. Parallel drafting supports longer draft blocks and deeper drafter networks, with potential to significantly accelerate inference.

Challenges in parallel drafter training: cross-round dependence. For a fixed architecture, a drafter's quality affects inference speed primarily through the number of decoding rounds, so the natural training objective is to minimize the expected number of rounds.

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.

For an AR drafter, the proposal at position t is conditioned on the full realized prefix $x_{<t}$ regardless of where the current round started, so its acceptance probability depends only on this prefix. As a result, the standard local training objective for AR drafters—matching the target model's next-token distribution at every prefix—is directly aligned with this goal (Yin et al., 2024). In contrast, for a parallel or semi-AR drafter, the proposal can instead depend on an earlier prefix $x_{\leq n}$ with $n < t$ (in a round starting at n), with access to only some or none of the intermediate tokens. Consequently, the same output position can have different draft distributions depending on where the round starts.

This additional dependence breaks the direct link between local distribution matching and global decoding efficiency. The draft for position t from a round starting at n matters only if the decoding trajectory first reaches a round starting at n and then accepts all preceding draft tokens in that round up to position $t-1$. Both events depend on earlier acceptance outcomes induced by the drafter itself. Thus, changing the drafter affects both acceptance within a round and which draft distributions are encountered in subsequent rounds, making the training objective inherently coupled across rounds.

However, existing objectives for parallel and semi-AR drafters do not account for these cross-round dependencies and instead rely on local surrogates. Examples include the expected number of tokens accepted within a single round (Li et al., 2026b) and draft-target distributional discrepancies under hand-crafted weights (Chen et al., 2026; Wu et al., 2026). This motivates the central question:

Can we train parallel and semi-AR drafters by directly minimizing the expected number of decoding rounds?

Our contributions. We answer this question affirmatively. Our approach relies on two key observations. First, conditioned on the output sequence, speculative decoding with a parallel or semi-AR drafter can be formulated as a Markov reward process (MRP) (Howard, 1971): each state records where the current round started and which position is being verified, and each rejection costs one round. Second, since rejection-sampling verification ensures that the output follows the same target distribution for every drafter, averaging over target rollouts recovers the expected number of rounds of actual decoding. Together, these insights allow us to capture the dependencies across decoding rounds explicitly and characterize the expected number of decoding rounds exactly, yielding a unified framework for training and evaluating drafters.

Building on this framework, we introduce the *Expected Decoding Rounds* (EDR) objective, whose minimization is equivalent to minimizing the expected number of decoding rounds. We then derive a temporal-difference gradient estimator, enabling drafter training from target-model rollouts. EDR applies across drafter architectures and introduces no hyperparameters. We further prove that drafters maximizing the expected accepted length for each round can be strictly suboptimal for EDR, theoretically justifying the advantages of our global objective. The framework also yields an exact offline evaluator of expected round counts, enabling comparisons of drafters on the same target-model rollouts without running online speculative decoding.

Empirically, we finetune two state-of-the-art (SOTA) drafters, DSpark (Cheng et al., 2026) and DFly (Liu et al., 2026b), with EDR while keeping their architectures or inference procedures unchanged. Across nine benchmarks spanning mathematical reasoning, code generation, and chat, a single epoch of EDR finetuning consistently improves mean accepted length over the original drafters and existing training objectives.

Our main contributions are summarized as follows:

- **A Markov reward process formulation.** We provide a theoretical framework that exactly characterizes expected speculative decoding rounds for parallel and semi-AR drafters.
- **An exact and trainable global objective.** We introduce EDR, establish its equivalence to minimizing the expected number of decoding rounds, derive a tractable gradient for drafter training, and prove its advantage over local surrogates.
- **An exact offline evaluator.** We provide an offline evaluator of the expected number of decoding rounds, enabling paired evaluation of drafters on the same target-model rollouts.
- **Consistent empirical gains.** EDR finetuning improves two SOTA drafters across nine benchmarks, outperforming existing training objectives.

2 PRELIMINARIES

Let $\mathcal{V}$ be a vocabulary of ordinary tokens and $\overline{\mathcal{V}} := \mathcal{V} \cup \{\texttt{<EOS>}\}$, where <EOS> denotes the end-of-sequence token. For two distributions p, q, define the total variation (TV) distance $\mathsf{TV}(p, q) := \sum_{v \in \overline{\mathcal{V}}} [p(v) - q(v)]_+$, where $[z]_+ := \max(0, z)$. For any sequence $x = (x_1, x_2, \ldots)$, we denote $x_{\le a} = (x_1, x_2, \ldots, x_a)$, $x_{<a} := x_{1:a-1}$, and $x_{a:b} := (x_a, x_{a+1}, \ldots, x_b)$ for $1 \le a \le b$.

Target distribution. Let p denote the distribution of the target model. It generates tokens autoregressively as $x_t \sim p_t(\cdot \mid x_{<t})$ for $t \ge 1$ and stops upon sampling <EOS>. Denote by $L := \inf\{\ell \ge 0 : x_{\ell+1} = \texttt{<EOS>}\}$ the sequence length excluding <EOS>. We assume L has a universal constant upper bound for $x \sim p$. A complete sequence $x = (x_{1:L}, \texttt{<EOS>})$ has probability $\prod_{t=1}^{L} p_t(x_t \mid x_{<t}) p_{L+1}(\texttt{<EOS>} \mid x_{1:L})$, and we call x *target-supported* if $p(x) > 0$.

Speculative decoding. Speculative decoding generates a token sequence x in rounds. Consider a round that starts from a committed prefix $x_{1:n}$. A drafter q first proposes a block of B *draft tokens* $\widetilde{x}_{n+1:n+B}$ with $\widetilde{x}_t \sim q_t$. For simplicity, we write q_t for the draft distribution at position t in the current round, suppressing its dependence on the round start n, committed prefix $x_{1:n}$, and preceding within-block proposals $\widetilde{x}_{n+1:t-1}$. We will make this explicit in the next paragraph.

The target model then verifies the draft block from left to right in a single forward pass. When position t is verified, all earlier drafts in the block have been accepted, so the current prefix is $x_{<t} = (x_{1:n}, \widetilde{x}_{n+1:t-1})$. Write $p_t = p_t(\cdot \mid x_{<t})$. The draft $\widetilde{x}_t$ is accepted with probability $\min\{1, p_t(\widetilde{x}_t)/q_t(\widetilde{x}_t)\}$, in which case it is committed as $x_t = \widetilde{x}_t$. Otherwise, it is replaced with a *correction token* $x_t \sim r_t$ drawn from the residual distribution $r_t(v) := [p_t(v) - q_t(v)]_+ / \mathsf{TV}(p_t, q_t)$, $v \in \overline{\mathcal{V}}$. The remaining drafts in the block are discarded, and the next round begins from the updated prefix $x_{1:t}$. If all B draft tokens are accepted, the same forward pass also yields one additional *bonus token* $x_{n+B+1} \sim p_{n+B+1}(\cdot \mid x_{\le n+B})$, and the next round starts after it. Generation stops when <EOS> is committed. This rejection-sampling verification ensures that the output sequence satisfies $x \sim p$ for any drafter q (Leviathan et al., 2023).

Draft distribution. Drafters differ in what q_t conditions on. An AR drafter proposes each token from the full prefix, whereas parallel and semi-AR drafters propose the whole block from the prefix committed at the start of the round. We now refine the shorthand q_t to make this explicit. For a round starting from a committed prefix $x_{1:n}$, let $q_{n,t}(\cdot \mid x_{1:n}; \widetilde{x}_{n+1:t-1})$ denote the draft distribution for position $t = n+1, \ldots, n+B$. The subscript t identifies the position being predicted, and n records the position where the round starts. The semicolon separates the committed prefix $x_{1:n}$ from the preceding within-block proposals $\widetilde{x}_{n+1:t-1}$, which the drafter may use in whole or in part.

This notation covers AR, parallel, and semi-AR drafters. An AR drafter is the special case $q_{n,t}(\cdot \mid x_{1:n}; \widetilde{x}_{n+1:t-1}) = q_t(\cdot \mid x_{1:n}, \widetilde{x}_{n+1:t-1})$, which conditions on the concatenated prefix and has no further dependence on the round start n. A parallel drafter samples the block independently given the committed prefix $x_{1:n}$, i.e., $q_{n,t}(\cdot \mid x_{1:n}; \widetilde{x}_{n+1:t-1}) = q_{n,t}(\cdot \mid x_{1:n})$. A semi-AR drafter allows partial dependence on preceding within-block drafts while maintaining parallel prediction across some positions. For parallel and semi-AR drafters, the same current prefix can therefore induce different draft distributions depending on the round start n. This dependence is the main difference from AR drafters and the central challenge this work addresses. Pseudocode for speculative decoding with a general drafter is given in Algorithm 2 in Appendix A.

Decoding rounds and mean accepted length. Each decoding round requires one forward pass of the target model. Let $\mathcal{T}^q$ denote the number of decoding rounds needed to produce a complete output under drafter q, including the one that produces <EOS>. A common metric is the mean accepted length (MAL), the ratio of the expected output length to the expected number of rounds,

$$\mathsf{MAL}(q) := \frac{\mathbb{E}[L+1]}{\mathbb{E}[\mathcal{T}^q]}, \tag{1}$$

where $L+1$ is the length of the complete output including all committed tokens, including correction tokens, bonus tokens, and the terminal <EOS> token. As the output sequence follows the target distribution, $\mathbb{E}[L]$ is independent of the drafter q for a fixed target model and evaluation corpus. Consequently, maximizing $\mathsf{MAL}(q)$ is equivalent to minimizing $\mathbb{E}[\mathcal{T}^q]$.

3 MAIN RESULTS

This section presents our main results. Section 3.1 presents a Markov reward process (MRP) formulation for speculative decoding. Section 3.2 introduces the EDR objective and Section 3.3 presents a temporal-difference form of its gradient. Section 3.4 gives an offline estimator for the MAL, and suboptimality of block-local objectives is discussed in Section 3.5.

3.1 SPECULATIVE DECODING AS A MARKOV REWARD PROCESS

Recall that $\mathcal{T}^q$ denotes the number of decoding rounds in a speculative decoding run with drafter q, and let $X = (X_1, \ldots, X_L, \texttt{<EOS>})$ be the output sequence of the same run. Since speculative decoding ensures $X \sim p$ for any drafter q, the overall expected number of decoding rounds satisfies

$$\mathbb{E}[\mathcal{T}^q] = \mathbb{E}_{X\sim p}[\tau^q(X)], \qquad \text{with } \tau^q(x) := \mathbb{E}[\mathcal{T}^q \mid X = x], \tag{2}$$

where $\tau^q(x)$ is the expected number of rounds conditioned on producing the sequence x. Therefore, it suffices to characterize $\tau^q(x)$ for any target-supported sequence x.

Markov reward process. In light of this, we fix a target-supported sequence $x = (x_1, \ldots, x_L, \texttt{<EOS>})$. Conditioned on the output x of a speculative decoding run, the remaining randomness is which positions are filled by accepted drafts and which by the target model after a rejection, and this determines the number of decoding rounds $\mathcal{T}^q(x)$.

This can be described as a Markov chain on states (n, t) for $0 \le n < t \le L + 1$. (see Appendix B.1 for detailed derivation). State (n, t) represents that the current round started at n (i.e., $x_{1:n}$ was committed) and is verifying the draft for position t. Decoding starts at $(0, 1)$ and terminates when <EOS> is produced at position $L + 1$.

Note that at state (n, t) with $t \le n+B$, the draft for position t is drawn from $q_{n,t}(\cdot \mid x_{1:n}; x_{n+1:t-1})$. By Bayes' rule, given that position t outputs x_t, the conditional probablity that it was produced by an accepted draft is

$$A^q_{n,t}(x_{\le t}) := \min\left\{1, \frac{q_{n,t}(x_t \mid x_{1:n}; x_{n+1:t-1})}{p_t(x_t \mid x_{<t})}\right\}. \tag{3}$$

Denote by $R^q_{n,t} := 1 - A^q_{n,t}$ the rejection probability. Acceptance continues the current round, whereas rejection starts a new round after position t, provided that $x_t \neq$ <EOS>. Therefore, the Markov chain has two transitions:

$$(n, t) \longrightarrow \begin{cases} (n, t + 1), & \text{with probability } A^q_{n,t}(x_{\le t}), \\ (t, t + 1), & \text{with probability } R^q_{n,t}(x_{\le t}). \end{cases} \tag{4}$$

We set $A^q_{n,t} = 0$ and $R^q_{n,t} = 1$ for $t > n + B$, so the same transition rule covers the case when all tokens in a draft block are accepted and a new round starts after that.

Let $\omega^q_{n,t}(x)$ denote the *occupancy* of state (n, t): the probability that this state is visited, conditioned on the output sequence being x. It is not hard to see that $\omega^q_{n,t}(x)$ depends on x only through $x_{<t}$ and hence we write $\omega^q_{n,t}(x_{<t}) = \omega^q_{n,t}(x)$. Starting from $\omega^q_{0,1}(x) = 1$, the occupancies satisfy the forward recursion:

$$\omega^q_{n,t+1}(x_{\le t}) = \omega^q_{n,t}(x_{<t})A^q_{n,t}(x_{\le t}), \quad n < t, \tag{5a}$$

$$\omega^q_{t,t+1}(x_{\le t}) = \sum_{n=0}^{t-1} \omega^q_{n,t}(x_{<t})R^q_{n,t}(x_{\le t}), \quad 1 \le t \le L. \tag{5b}$$

Since every round begins at a diagonal state $(t, t + 1)$, the number of rounds equals the number of diagonal states visited, yielding the following exact expression (see Appendix B.2 for proof).

Proposition 1. *For any target-supported sequence $x = (x_{1:L}, \texttt{<EOS>})$ and any drafter q, the expected number of decoding rounds to produce x is*

$$\tau^q(x) = \sum_{t=0}^{L} \omega^q_{t,t+1}(x_{\le t}) = 1 + \sum_{t=1}^{L}\sum_{n=0}^{t-1} \omega^q_{n,t}(x_{<t})R^q_{n,t}(x_{\le t}). \tag{6}$$

The second identity has a direct MRP interpretation: rejection starts a new round and incurs unit cost, while acceptance incurs zero cost. Hence, the rejection probability $R^q_{n,t}$ is the expected cost of visiting state (n, t), and the occupancy-weighted sum of costs gives the round count.

3.2 THE EDR OBJECTIVE

By (2), averaging (6) over $x \sim p$ provides an exact expression for the overall expected number of decoding rounds $\mathbb{E}[\mathcal{T}^q]$. However, each rejection probability $R^q_{n,t}(x_{\leq t})$ uses the target only through the single realized token x_t, even though the full distribution $p_t(\cdot \mid x_{<t})$ is available. Hence, we average over this token under the target distribution p_t, conditioned on x_t being a non-EOS token.

Definition 1. Define the *local rejection cost* of state (n, t), for $0 \leq n < t \leq L$, as

$$c^q_{n,t}(x_{<t}) := \mathop{\mathbb{E}}_{X_t \sim p'_t(\cdot|x_{<t})} [R^q_{n,t}(x_{<t}, X_t)] \tag{7}$$

where $p'_t(v \mid x_{<t}) := p_t(v \mid x_{<t})/(1 - p_t(\texttt{<EOS>} \mid x_{<t}))$ for $v \neq \texttt{<EOS>}$ is the target distribution normalized over non-EOS tokens. For $t \leq n + B$, its value is given by

$$c^q_{n,t}(x_{<t}) = \frac{1}{1 - p_t(\texttt{<EOS>} \mid x_{<t})} \sum_{v \in \mathcal{V}} [p_t(v \mid x_{<t}) - q_{n,t}(v \mid x_{1:n}; x_{n+1:t-1})]_+, \tag{8}$$

and for $t > n + B$, it is equal to 1. Note that $c^q_{n,t}$ is well-defined because every position $t \leq L$ outputs a non-EOS token so $p_t(\texttt{<EOS>} \mid x_{<t}) < 1$.

In words, the cost $c^q_{n,t}(x_{<t})$ is the rejection probability associated with state (n, t), given that the committed output token x_t is non-EOS. It measures the local distributional discrepancy between the drafter $q_{n,t}$ and target p_t for position t, given by the TV distance between the draft and target distributions for position t, excluding EOS and normalized by the target's non-EOS mass. Excluding EOS reflects the fact that when the target model produces EOS, it never starts another decoding round and incurs no penalty for draft rejection.

We now introduce the *Expected Decoding Rounds (EDR)* objective, which exactly equals the overall expected number of decoding rounds, as formalized in Theorem 1 below. The proof can be found in Appendix B.3.

Theorem 1. *For any drafter q, the EDR objective equals to expected number of decoding rounds:*

$$\mathcal{L}_{\mathrm{EDR}}(q) := 1 + \mathbb{E}_{X \sim p}\left[\sum_{t=1}^{L} \sum_{n=0}^{t-1} \omega^q_{n,t}(X_{<t}) c^q_{n,t}(X_{<t})\right] = \mathbb{E}_{X \sim p}[\tau^q(X)] = \mathbb{E}[\mathcal{T}^q]. \tag{9}$$

The EDR objective is a sum of local rejection costs $c^q_{n,t}$, measuring the local distributional discrepancy between the drafter $q_{n,t}$ and target p_t, each weighted by the corresponding occupancy $\omega^q_{n,t}$. The quantity inside the expectation, which we call the *per-sequence EDR loss*,

$$\widehat{\mathcal{L}}_{\mathrm{EDR}}(q; x) := 1 + \sum_{t=1}^{L} \sum_{n=0}^{t-1} \omega^q_{n,t}(x_{<t})\, c^q_{n,t}(x_{<t}). \tag{10}$$

can be computed exactly given a target-sampled sequence x via the forward occupancy recursion (5) and the local rejection cost (7). Therefore, averaging over a batch of target rollouts provides an empirical EDR loss for drafters that directly targets minimizing the expected number of rounds.

We make several important remarks.

- *Replacement of $R^q_{n,t}$ with $c^q_{n,t}$* The EDR loss (10) replaces $R^q_{n,t}(x)$ in (6) with its conditional expectation $c^q_{n,t}(x)$. For an individual rollout, it generally differs from $\tau^q(x)$, but they agree after averaging over $x \sim p$. In return, the EDR loss uses the next-token distribution rather than one sampled token, so its gradient aggregates information across the vocabulary.
- *Exact occupancy weighting.* Prior surrogate objectives (Chen et al., 2026; Wu et al., 2026) do not account for EOS termination in their local discrepancy terms, and more importantly,

Algorithm 1 One EDR training step ($\mathrm{sg}(\cdot)$ denotes the stop-gradient operator)

Require: drafter q_θ, target model p, block size B

1: Sample independent target rollouts for training dataset prompts
2: $x^{(i)} = (x^{(i)}_{1:L_i}, \texttt{<EOS>}) \sim p \mid i\text{-th prompt} \quad i = 1, 2, \ldots, m$
3: Compute $p_t(\cdot \mid x^{(i)}_{<t})$ for all i and t in one batched target forward pass
4: **for** $i = 1, \ldots, M$ (in parallel) **do**
5: Compute $A^{(i)}_{n,t}$ and $c^{(i)}_{n,t}$ from q_θ at all states (n, t)
6: Compute $\omega^{(i)}_{n,t}$ by (5) and $V^{(i)}_{n,t}$ by (12), without gradients
7: **end for**
8: $\ell(\theta) \leftarrow M^{-1} \sum_{i=1}^{M} \sum_{t=1}^{L_i} \sum_{n=0}^{t-1} \mathrm{sg}(\omega^{(i)}_{n,t}) \left[c^{(i)}_{n,t} + A^{(i)}_{n,t}\, \mathrm{sg}(V^{(i)}_{n,t+1} - V^{(i)}_{t,t+1})\right]$
9: Update θ using $\nabla_\theta \ell(\theta)$

weight them with hand-crafted weights independent of the drafter q. These weights can be interpreted as fixed surrogates for the drafter-dependent occupancies $\omega^q_{n,t}$. In contrast, EDR combines EOS-aware local costs with the actual occupancies, yielding an exact objective for the expected number of decoding rounds.

- *Parallel vs. AR drafters.* For an AR drafter, $q_{n,t}(\cdot \mid x_{1:n}; x_{n+1:t-1}) = q_t(\cdot \mid x_{<t})$ for all $n < t$. Without a block cap ($B = \infty$), EDR reduces to

$$\mathcal{L}_{\mathrm{EDR}}(q) = 1 + \mathbb{E}_{X \sim p}\left[\sum_{t=1}^{L} c_t(X_{<t})\right] \quad \text{with} \quad c_t(x_{<t}) := \frac{\sum_{v \in \mathcal{V}} [p_t(v \mid x_{<t}) - q_t(v \mid x_{<t})]_+}{1 - p_t(\texttt{<EOS>} \mid x_{<t})}. \tag{11}$$

EDR hence decouples into local prefix costs, each minimized by matching q_t to p_t. This recovers the result of Yin et al. (2024, Theorem 1) established for AR drafters and extends it to sequences that terminate with EOS. And a finite block size adds only a mild dependence on n through the boundary states. Therefore, Theorem 1 highlights the key difference between parallel and AR drafters, and EDR captures the cross-round dependencies exactly.

3.3 TD-FORM GRADIENT OF EDR

The per-sequence EDR loss in (10) depends on the drafter q through both the local costs $c^q_{n,t}$ and the occupancies $\omega^q_{n,t}$. Differentiating it with respect to q directly is expensive because it must backpropagate through the entire forward recursion for occupancy $\omega^q_{n,t}$ (5). Fortunately, the MRP formulation gives the gradient in a temporal-difference (TD) (Sutton, 1988) form that uses only local derivatives.

For a fixed target-supported x, consider the MRP with transitions (4) and costs $c^q_{n,t}(x)$ from (7). Define the value function $V^q_{n,t}(x)$, the expected cumulative cost starting from state (n, t). The value function satisfies the Bellman equation (Bellman, 1966)

$$V^q_{n,t}(x) = c^q_{n,t}(x_{<t}) + A^q_{n,t}(x_{\le t}) V^q_{n,t+1}(x) + R^q_{n,t}(x_{\le t}) V^q_{t,t+1}(x), \qquad 0 \le n < t \le L, \tag{12}$$

with the terminal condition $V^q_{n,L+1}(x) = 0$ for all $0 \le n \le L$. Unlike the cost and occupancy, the value function $V^q_{n,t}(x)$ depends on the whole sequence x. The value function of the initial state gives the per-sequence EDR loss:

$$V^q_{0,1}(x) = \widehat{\mathcal{L}}_{\mathrm{EDR}}(q; x) - 1. \tag{13}$$

We now present the gradient of the per-sequence EDR loss in the following theorem. The proof of (13) and Theorem 2 are given in Appendix B.4.

Theorem 2 (EDR gradient)**.** *We parametrize the drafter $q = q_\theta$ by neural network parameters θ. For any target-supported sequence x, the gradient of the per-sequence EDR loss* (10) *is given by*

$$\nabla_\theta \widehat{\mathcal{L}}_{\mathrm{EDR}}(q_\theta; x) = \sum_{t=1}^{L} \sum_{n=0}^{t-1} \omega^q_{n,t}(x_{<t}) \left[\nabla_\theta c^q_{n,t}(x_{<t}) + \nabla_\theta A^q_{n,t}(x_{\le t}) \left(V^q_{n,t+1}(x) - V^q_{t,t+1}(x)\right)\right]. \tag{14}$$

As a result, one has

$$\mathbb{E}_{X\sim p}\left[\nabla_\theta \widehat{\mathcal{L}}_{\mathrm{EDR}}(q_\theta; X)\right] = \nabla_\theta \mathcal{L}_{\mathrm{EDR}}(q_\theta). \tag{15}$$

The gradient has a standard one-step TD form, separated into two contributions: the immediate effect on the local rejection cost, and the downstream effect of changing the acceptance probability. The latter is weighted by the value gap, which measures the difference between the cumulative costs of the two possible subsequent states. The occupancies account for how often each state is reached.

Implementation. We compute the occupancies $\omega^q_{n,t}$ by the forward recursion (5) and the value functions $V^q_{n,t}$ by the backward Bellman recursion (12). Since they enter the gradient expression (14) only as weights, we hold them fixed during differentiation. Gradients are required only through the local quantities $c^q_{n,t}$ and $A^q_{n,t}$.

By (15), averaging (14) over a batch of i.i.d. target rollouts gives an unbiased stochastic gradient of EDR, which directly optimizes the expected number of decoding rounds. One EDR training step is summarized in Algorithm 1.

3.4 Exact offline evaluation

In addition to drafter training, the diagonal occupancy representation in Proposition 1 gives an exact evaluator for any drafter q. For token sequence $x = (x_1, \ldots, x_L, \texttt{<EOS>})$, define

$$\tau^q_{\mathrm{off}}(x) := \sum_{t=0}^{L} \omega^q_{t,t+1}(x). \tag{16}$$

This quantify averages out all the randomness underlying speculative decoding under a fixed $X = x$ and does not require running it online. Therefore, for a dataset of rollouts drawn from the target model $\mathcal{D} = \{x^{(i)} = (x^{(i)}_{1:L_i}, \texttt{<EOS>})\}_{i=1}^N$, we can estimate the expected number of decoding rounds and the MAL as

$$\widehat{\mathcal{T}}_{\mathcal{D}}(q) := \frac{1}{N}\sum_{i=1}^{N} \tau^q_{\mathrm{off}}(x^{(i)}), \quad \text{and} \quad \widehat{\mathsf{MAL}}_{\mathcal{D}}(q) := \frac{N^{-1}\sum_{i=1}^N (L_i+1)}{\widehat{\mathcal{T}}_{\mathcal{D}}(q)}. \tag{17}$$

Using shared rollouts enables paired comparisons without introducing independent rollout variation across drafters.

3.5 Suboptimality of block-local surrogates

Finally, we show that maximizing accepted length independently within each block can be strictly suboptimal for the global MAL. Since maximizing MAL is equivalent to minimizing the expected number of decoding rounds (see Section 2), which EDR equals exactly (Theorem 1), this theoretically justifies optimizing the EDR objective rather than block-local surrogates.

We consider semi-AR drafters that condition on earlier drafts in the block only through the immediately preceding token, i.e., $q_{n,t}(\cdot \mid x_{1:n}; x_{n+1:t-1}) = q_{n,t}(\cdot \mid x_{1:n}; x_{t-1})$, referred to as *first-order drafters*. This class includes existing semi-AR drafters such as DSpark (Cheng et al., 2026) and DFly (Liu et al., 2026b).

The following theorem shows that even when the expected accepted length is optimized in every round, the resulting MAL can be a constant factor below the best achievable within the same drafter class. The construction and proof are deferred to Appendix B.5.

Theorem 3. *For any block size $B \geq 3$, there exists a target distribution p such that, for any first-order drafter $\hat{q}$ that maximizes the expected accepted length in every round, there exists another first-order drafter q^* satisfying*

$$\frac{\mathsf{MAL}(q^*)}{\mathsf{MAL}(\hat{q})} > 1 + \frac{1}{3B+9}.$$

The gap arises from the cross-round interaction ignored by block-local objectives. The draft distribution in the current round determines not only how many tokens are accepted in that round, but also where the next round starts. Consequently, a drafter that is locally optimal for the current round can induce a less favorable distribution of future round starts and result in a lower global MAL.

Table 1: **One-epoch finetuning results.** We report mean accepted length (MAL; higher is better) after one epoch of draft-model finetuning. Target and draft sampling configurations in evaluation are the same as in training. Block size $B = 7$ for all drafters. **Bold** marks the best finetuned result.

Target	Drafter ($B = 7$)	Math			Code			Chat		
		GSM8K	MATH	AIME25	MBPP	HumanEval	LCB	MT-Bench	Alpaca	Arena-Hard
Qwen3-4B ($T = 0.7$, top-$p = 0.8$, top-$k = 20$)	DSpark	6.23	6.12	5.46	5.28	5.52	5.46	3.70	3.62	3.59
	E2E	**6.32**	**6.20**	5.57	**5.34**	5.58	5.54	3.75	3.66	3.67
	EDR (Ours)	**6.32**	**6.20**	**5.59**	**5.34**	**5.59**	**5.56**	**3.78**	**3.70**	**3.70**
Qwen3-8B ($T = 1.0$, no top-p/top-k)	DFly	6.30	5.94	5.15	5.26	5.58	5.23	3.57	3.45	3.18
	E2E	6.34	5.98	5.18	**5.32**	5.62	5.26	3.61	3.49	3.21
	EDR (Ours)	**6.35**	**6.00**	**5.22**	**5.32**	**5.63**	**5.29**	**3.65**	**3.52**	**3.25**

4 EXPERIMENTS

We evaluate whether directly optimizing EDR improves self-speculative decoding over existing block-local training objectives. We consider two SOTA self-speculative drafters, DSpark (Cheng et al., 2026) and DFly (Liu et al., 2026b), paired with Qwen3 (Yang et al., 2025) target models of different scales.

4.1 EXPERIMENTAL SETUP

Models and sampling. We evaluate two target–drafter pairs: Qwen3-4B with DSpark and Qwen3-8B with DFly. We set block size $B = 7$ for both drafters. For Qwen3-4B, target trajectories are sampled with temperature 0.7, top-$p = 0.8$, and top-$k = 20$, while the DSpark proposal distribution uses temperature 0.7 without top-p or top-k truncation. For Qwen3-8B, both the target and DFly drafter use temperature 1.0 without top-p or top-k truncation. Thinking mode are disable in all settings.

Training objectives. Starting from the corresponding well-trained released draft-model checkpoint, we finetune only the drafter parameters. We compare our exact EDR objective against the end-to-end (E2E) block-local objective (Li et al., 2026b), using the same model architecture, training data, optimizer configuration, and number of update steps for both objectives. For all methods, we use at most 512 training anchors per trajectory. For EDR, we use importance sampling to select training anchors. For details, see Appendix C.2.

Training data. We train on Open-PerfectBlend (Xu et al., 2024). Rather than using reference answers, we generate a fresh trajectory for each prompt from the target model using the same target sampling configuration as evaluation. Each drafter is finetuned for one epoch with batch size 96.

Evaluation. We use the same evaluation prompts as DSpark (Cheng et al., 2026) on nine benchmarks spanning math (Cobbe et al., 2021; Lightman et al., 2024; Zhang & Math-AI, 2025), code (Austin et al., 2021; Chen et al., 2021; Jain et al., 2025), and chat (Zheng et al., 2023; Taori et al., 2023; Li et al., 2024a). We generate at least $1{,}500$ target trajectories per benchmark (multiple trajectories for each prompt) under the same sampling configurations as in the training. We compute MAL using the offline estimator introduced in Section 3.4. All methods for the same target model and sampling configurations are evaluated on exactly the same target trajectories. This provides paired comparisons across training objectives and removes additional variance from independently sampled evaluation outputs.

All training and evaluation are performed on a single NVIDIA H200 GPU. For more experimental details, see Appendix D.

4.2 EXPERIMENTAL RESULTS

Table 1 shows that EDR consistently improves speculative decoding across both DSpark and DFly without modifying the drafter architecture. Compared with E2E finetuning, EDR achieves higher

or equal MAL across all benchmarks, with larger gains on AIME25, LiveCodeBench, and the chat benchmarks. These results suggest that directly optimizing the global expected number of decoding rounds provides additional gains beyond block-local end-to-end surrogates, and that the improvement transfers across different self-speculative drafter architectures and target-model scales.

5 Related Work

Drafter architectures for speculative decoding. Speculative decoding uses a draft model to propose multiple tokens, which the target model verifies in parallel. The latency per generated token depends on the accepted length per round as well as the costs of drafting and verification. Architectural work therefore seeks to increase accepted length while keeping drafting inexpensive. Medusa (Cai et al., 2024) attaches multiple prediction heads to target-model features, while Hydra (Ankner et al., 2024) introduces sequential dependencies between these heads to improve draft accuracy. EAGLE-3 (Li et al., 2026c) combines features from multiple target layers. To reduce sequential drafting overhead, parallel and diffusion-based drafters generate multiple proposals in a single forward pass. DFlash (Chen et al., 2026) uses a lightweight block-diffusion drafter conditioned on target-model context features. DiffuSpec (Li et al., 2026a) repurposes a pretrained diffusion language model as a parallel drafter and uses causal-consistency path search to construct proposals for autoregressive verification. To improve within-block consistency without returning to fully autoregressive drafting, DSpark (Cheng et al., 2026) and DFly (Liu et al., 2026b) combine parallel backbones with lightweight sequential heads that condition on preceding draft tokens.

Training objectives for speculative drafters. Drafter training commonly uses cross-entropy or KL-divergence objectives to align draft predictions with the target distribution (Li et al., 2026c; Chen et al., 2026). However, distributional matching under these losses does not necessarily maximize token acceptance when drafter capacity is limited. LK losses (Samarin et al., 2026) address this distinction through acceptance-oriented objectives. For multi-token drafting, the first rejection invalidates all subsequent proposals in the block, making the contributions of different positions unequal. DFlash (Chen et al., 2026) accounts for this asymmetry using fixed exponentially decaying position weights. D-PACE (Wu et al., 2026) replaces this fixed schedule with adaptive weights derived from a differentiable accepted-length surrogate based on draft confidences. The end-to-end (E2E) multi-step TV loss (Li et al., 2026b) moves beyond independent token-level matching by incorporating the multiplicative structure of acceptance across positions within a block. Recent work addresses the mismatch between drafter training and inference by incorporating verification feedback. Draft-OPD (Lei et al., 2026) replays draft proposals from inference-time anchors and applies acceptance-dependent distillation losses. Verification-Aware Training (Gu et al., 2026) uses simulated verification outcomes to supervise an auxiliary head and adapt position weights. Although these methods incorporate inference-time behavior into training, their objectives remain surrogates for MAL. Our work establishes an exact connection between drafter training and inference-time MAL: target-model rollouts alone suffice to construct an exact expected-round objective and an unbiased gradient estimator. This directly addresses the objective mismatch.

6 Conclusion

We conclude with several directions for future work. First, our current framework focuses on a fixed draft block size. However, under high-concurrency serving, verifying long draft blocks can become inefficient, especially when many later proposals are likely to be rejected. Adaptive block sizing can reduce unnecessary verification by shortening the draft block when confidence is low, thereby further reducing inference latency (Hu et al., 2026; Huang et al., 2024). An important direction is to extend EDR to adaptive block sizes and to study whether the optimal block size can be selected dynamically for each decoding round. Second, EDR requires computing occupancy weights ω and value functions V for multiple anchor positions. This is more computationally expensive than block-local objectives. It would be interesting to explore more efficient ways to utilize the occupancy and value computations. Finally, as shown in Table 1, MAL varies substantially across datasets, suggesting that some target distributions are intrinsically more difficult to draft than others. Our current work focuses on optimizing the drafter given a target distribution, but does not provide a complete theoretical characterization of this difficulty. A promising direction is to identify properties of the target distribution that determine the best achievable speculative decoding efficiency.

AI USE STATEMENT

In this work, we used generative AI tools for [formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, design or provide feedback on research methodology or experiments, implement methods, interpret results]. We have not used generative AI tools for [generate synthetic data sets, help develop theoretical models or conceptual frameworks, propose or refine hypotheses, assist with translation, clean and reformat dataset], and [support qualitative and thematic data analysis] are not applicable to this work. Additionally, we used generative AI tools for [create or edit software code]. We have reviewed all AI-assisted work. [We checked all LLM polished proofs following the proof sketch written by human are mathematically correct. We checked an LLM proposed example used in the Theorem 3 proof is correct. We tested the LLM generated code]. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

See the ICLR 2027 AI Policy for Authors for more details. This statement should not be more than 1 page.

REPRODUCIBILITY STATEMENT

All theoretical claims and theorems are proved in the paper or in the appendix.

REFERENCES

Zachary Ankner, Rishab Parthasarathy, Aniruddha Nrusimha, Christopher Rinard, Jonathan Ragan-Kelley, and William Brandon. Hydra: Sequentially-dependent draft heads for medusa decoding. *arXiv preprint arXiv:2402.05109*, 2024.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Richard Bellman. Dynamic programming. *science*, 153(3731):34–37, 1966.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D Lee, Deming Chen, and Tri Dao. Medusa: Simple llm inference acceleration framework with multiple decoding heads. *arXiv preprint arXiv:2401.10774*, 2024.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

Jian Chen, Yesheng Liang, and Zhijian Liu. Dflash: Block diffusion for flash speculative decoding. *arXiv preprint arXiv:2602.06036*, 2026.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Xin Cheng, Xingkai Yu, Chenze Shao, Jiashi Li, Yunfan Xiong, Yi Qian, Jiaqi Zhu, Shirong Ma, Xiaokang Zhang, Jiasheng Ye, et al. Dspark: Confidence-scheduled speculative decoding with semi-autoregressive generation. *arXiv preprint arXiv:2607.05147*, 2026.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Cunxiao Du, Jing Jiang, Xu Yuanchen, Jiawei Wu, Sicheng Yu, Yongqi Li, Shenggui Li, Kai Xu, Liqiang Nie, Zhaopeng Tu, et al. Glide with a cape: A low-hassle method to accelerate speculative decoding. *arXiv preprint arXiv:2402.02082*, 2024.

Geonmo Gu, Byeongho Heo, HeeJae Jun, Yoohoon Kang, Sangmin Lee, Sangdoo Yun, and Dongyoon Han. Verification-aware training for speculative decoding. *arXiv preprint arXiv:2608.30135*, 2026.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pp. 65–70, 1979.

R.A. Howard. *Dynamic Probabilistic Systems: Semi-Markov and decision processes*. Decision and Control Series. Wiley, 1971. ISBN 9780471416661. URL https://books.google.com/books?id=vuZQAAAAMAAJ.

Xinyi Hu, Yuhao Shen, Baolin Zhang, Hengxin Zhang, Jun Dai, Shuang Ge, Lei Chen, Yue Li, and Mingcheng Wan. Echo: Elastic speculative decoding with sparse gating for high-concurrency scenarios. *arXiv preprint arXiv:2604.09603*, 2026.

Kaixuan Huang, Xudong Guo, and Mengdi Wang. Specdec++: Boosting speculative decoding via adaptive candidate lengths. *arXiv preprint arXiv:2405.19715*, 2024.

Naman Jain, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, volume 2025, pp. 58791–58831, 2025.

Haodi Lei, Yafu Li, Haoran Zhang, Shunkai Zhang, Qianjia Cheng, Xiaoye Qu, Ganqu Cui, Bowen Zhou, Ning Ding, Yun Luo, et al. Draft-opd: On-policy distillation for speculative draft models. *arXiv preprint arXiv:2605.29343*, 2026.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International conference on machine learning*, pp. 19274–19286. PMLR, 2023.

Guanghao Li, Zhihui Fu, Min Fang, Qibin Zhao, Ming Tang, Chun Yuan, and Jun Wang. Diffuspec: Unlocking diffusion language models for speculative decoding. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 20896–20910, 2026a.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*, 2024a.

Yucheng Li, Huiqiang Jiang, Yang Xu, Jianxin Yang, Yi Zhang, Yizhong Cao, Yuhao Shen, Fan Zhou, Rui Men, Jianwei Zhang, et al. Breaking entropy bounds: Accelerating rl training via mtp with rejection sampling. *arXiv preprint arXiv:2606.12370*, 2026b.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle: Speculative sampling requires rethinking feature uncertainty. *arXiv preprint arXiv:2401.15077*, 2024b.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle: Speculative sampling requires rethinking feature uncertainty. *arXiv preprint arXiv:2401.15077*, 2024c.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle-2: Faster inference of language models with dynamic draft trees. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 7421–7432, 2024d.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle-3: Scaling up inference acceleration of large language models via training-time test. *Advances in Neural Information Processing Systems*, 38:136737–136756, 2026c.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Fangcheng Liu, Yehui Tang, Zhenhua Liu, Yunsheng Ni, Kai Han, and Yunhe Wang. Kangaroo: Lossless self-speculative decoding via double early exiting. *arXiv preprint arXiv:2404.18911*, 2024.

Fuliang Liu, Xue Li, Ketai Zhao, Yinxi Gao, Ziyan Zhou, Zhonghui Zhang, Zhibin Wang, Wanchun Dou, Sheng Zhong, and Chen Tian. Dart: Diffusion-inspired speculative decoding for fast llm inference. *arXiv preprint arXiv:2601.19278*, 2026a.

Hong Liu, Rui Cen, Junhan Shi, Guangshuo Qin, Jiebin Zhang, Tianyu Liu, Runzhi Fan, Guoliang Zhao, Ruobing Xie, Kai Zhang, et al. Angelspec: Towards real-world high performance inference with speculative decoding. *arXiv preprint arXiv:2607.25852*, 2026b.

Subham Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

Alexander Samarin, Sergei Krutikov, Anton Shevtsov, Sergei Skvortsov, Filipp Fisin, and Alexander Golubev. Lk losses: Direct acceptance rate optimization for speculative decoding. *arXiv preprint arXiv:2602.23881*, 2026.

Franklin E Satterthwaite. An approximate distribution of estimates of variance components. *Biometrics bulletin*, 2(6):110–114, 1946.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37: 103131–103167, 2024.

Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3 (1):9–44, 1988.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`, 2023.

Bernard L Welch. The generalization of 'student's'problem when several different population varlances are involved. *Biometrika*, 34(1-2):28–35, 1947.

Tianyu Wu, Yu Yao, Zhenting Qi, Han Zheng, Zhuohan Wang, Haoran Ma, Lawrence Liao, Himabindu Lakkaraju, Ju Li, and Yilun Du. D-pace: Dynamic position-aware cross-entropy for parallel speculative drafting. *arXiv preprint arXiv:2605.18810*, 2026.

Tengyu Xu, Eryk Helenowski, Karthik Abinav Sankararaman, Di Jin, Kaiyan Peng, Eric Han, Shaoliang Nie, Chen Zhu, Hejia Zhang, Wenxuan Zhou, et al. The perfect blend: Redefining rlhf with mixture of judges. *arXiv preprint arXiv:2409.20370*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Ming Yin, Minshuo Chen, Kaixuan Huang, and Mengdi Wang. A theoretical perspective for speculative decoding algorithm. *Advances in Neural Information Processing Systems*, 37:128082–128117, 2024.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2025, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

A SPECULATIVE DECODING ALGORITHM

Algorithm 2 summarizes the speculative decoding procedure we consider.

Algorithm 2 Self-speculative decoding with a semi-autoregressive drafter

Require: Block size B, target distributions p_t, draft distributions $q_{n,t}$

$n \leftarrow 0, \mathcal{T} \leftarrow 0$ ▷ n: number of committed ordinary tokens
while true do
  Draw a draft block $\widetilde{x}_{n+1:n+B} \sim q_{n,n+1}(\cdot \mid x_{1:n}) \cdots q_{n,n+B}(\cdot \mid x_{1:n}; \widetilde{x}_{n+1:n+B-1})$
  Evaluate $p_t(\cdot \mid x_{1:n}, \widetilde{x}_{n+1:t-1})$ for $t = n+1, \ldots, n+B+1$ in one target-model pass
  $\mathcal{T} \leftarrow \mathcal{T} + 1$
  rejected $\leftarrow$ false
  for $t = n+1, \ldots, n+B$ **do**
    $p \leftarrow p_t(\cdot \mid x_{1:n}, \widetilde{x}_{n+1:t-1})$
    $q \leftarrow q_{n,t}(\cdot \mid x_{1:n}; \widetilde{x}_{n+1:t-1})$
    Draw $U_t \sim \mathrm{Unif}[0,1]$
    if $U_t \le \min\{1, p(\widetilde{x}_t)/q(\widetilde{x}_t)\}$ **then**
      $x_t \leftarrow \widetilde{x}_t$
    else
      Draw $x_t \sim [p-q]_+ / \mathsf{TV}(p,q)$
      rejected $\leftarrow$ true
    end if
    if $x_t =$ <EOS> **then**
      return $(x_{1:t-1}, \texttt{<EOS>})$ and $\mathcal{T}$
    end if
    if rejected **then**
      $n \leftarrow t$
      break
    end if
  end for
  if not rejected **then**
    Draw $\widetilde{x}_{n+B+1} \sim p_{n+B+1}(\cdot \mid x_{1:n+B})$ ▷ bonus token
    if $\widetilde{x}_{n+B+1} =$ <EOS> **then**
      return $(x_{1:n+B}, \texttt{<EOS>})$ and $\mathcal{T}$
    else
      $x_{n+B+1} \leftarrow \widetilde{x}_{n+B+1}$
      $n \leftarrow n+B+1$
    end if
  end if
end while

B PROOFS

B.1 MARKOV PROCESS FORMULATION IN SECTION 3.1

In this section, we give a detailed derivation of the rejection-sampling transition dynamics conditioned on a fixed target-supported output sequence $x = (x_{1:L}, \texttt{<EOS>})$. Specifically, we define the Markov chain states, derive the conditional transition probabilities, and prove that the occupancy $\omega^q_{n,t}(x)$ depends on x only through $x_{<t}$. Throughout, all probabilities refer to speculative decoding with a fixed drafter q.

States and decoding histories. For each position t reached by decoding, let N_t denote the round-start index of the round that produces X_t. Thus, immediately before processing position t, the state is

$$S_t = (N_t, t), \qquad S_1 = (0, 1).$$

If $N_t = n$, then the current round started from the committed prefix $X_{1:n}$, and all preceding drafts in this round have been accepted. Consequently, their values equal $X_{n+1:t-1}$. For $t \le n+B$, position t is processed by draft verification; for $t = n+B+1$, it is processed by drawing a target bonus token. States with $t > n+B+1$ are unreachable.

Let

$$\mathcal{H}_t := \sigma(X_{<t}, N_1, \ldots, N_t)$$

denote the history of committed tokens and round-start indices before processing position t. We first analyze the next output token conditional on this history, without conditioning on the complete output sequence.

The local rejection-sampling law. Consider a history with $X_{<t} = x_{<t}$ and $N_t = n$ at a draft position $t \le n + B$. For brevity, write

$$p_t(v) := p_t(v \mid x_{<t}), \qquad q_t(v) := q_{n,t}(v \mid x_{1:n}; x_{n+1:t-1}), \qquad v \in \overline{\mathcal{V}}.$$

Let acc_t and rej_t denote the acceptance and rejection events at this position. The verification rule gives

$$\Pr(X_t = v, \mathrm{acc}_t \mid \mathcal{H}_t) = \min\{p_t(v), q_t(v)\},$$
$$\Pr(X_t = v, \mathrm{rej}_t \mid \mathcal{H}_t) = [p_t(v) - q_t(v)]_+.$$

The first identity follows by multiplying the proposal probability by the verifier's acceptance probability. For the second identity, rejection has total probability $\mathrm{TV}(p_t, q_t)$, and the correction token is drawn from the residual distribution $[p_t - q_t]_+ / \mathrm{TV}(p_t, q_t)$. When $\mathrm{TV}(p_t, q_t) = 0$, rejection has probability zero and no residual draw is needed.

Adding the two identities yields

$$\Pr(X_t = v \mid \mathcal{H}_t) = \min\{p_t(v), q_t(v)\} + [p_t(v) - q_t(v)]_+ = p_t(v \mid X_{<t}). \tag{18}$$

At a bonus-token position, the same identity holds because the token is drawn directly from the target distribution. Therefore, conditional on any decoding history, the next committed token has the target conditional distribution, regardless of the current round-start index.

Transition kernels conditioning on the complete output. Equation (18) implies that, on any history with $X_{<t} = x_{<t}$,

$$\Pr(X = x \mid \mathcal{H}_t) = p(x_{t:L+1} \mid x_{<t}) := \prod_{s=t}^{L+1} p_s(x_s \mid x_{<s}). \tag{19}$$

To see this rigorously, use backward induction along the fixed sequence x. At $t = L + 1$, the required remaining output is just <EOS>, so the identity follows directly from (18). Suppose it holds at $t + 1$ for every possible decoding history. After committing $X_t = x_t$, every possible resulting history has the same remaining-output probability $p(x_{t+1:L+1} \mid x_{\le t})$. Multiplying this probability by $\Pr(X_t = x_t \mid \mathcal{H}_t) = p_t(x_t \mid x_{<t})$ proves the identity at t.

In particular, after committing the same non-EOS token x_t, the future suffix has the same probability whether the token was accepted or produced by residual sampling. Thus, for $t \le L$ at a draft position, Bayes' rule gives

$$\begin{aligned}
\Pr(\mathrm{acc}_t \mid \mathcal{H}_t, X = x) &= \frac{\min\{p_t(x_t), q_t(x_t)\}\, p(x_{t+1:L+1} \mid x_{\le t})}{p_t(x_t)\, p(x_{t+1:L+1} \mid x_{\le t})} \\
&= \frac{\min\{p_t(x_t), q_t(x_t)\}}{p_t(x_t)} \\
&= \min\left\{1, \frac{q_{n,t}(x_t \mid x_{1:n}; x_{n+1:t-1})}{p_t(x_t \mid x_{<t})}\right\} \\
&= A^q_{n,t}(x_{\le t}).
\end{aligned}$$

All target-probability factors being cancelled are positive because x is target-supported. Similarly,

$$\Pr(\mathrm{rej}_t \mid \mathcal{H}_t, X = x) = 1 - A^q_{n,t}(x_{\le t}) = R^q_{n,t}(x_{\le t}).$$

Notice that the ratio here is $q_t(x_t)/p_t(x_t)$, rather than the verifier's ratio $p_t(\widetilde{X}_t)/q_t(\widetilde{X}_t)$: we are conditioning on the committed output token, not on the proposed token.

If the token is accepted, the current round continues. If it is rejected, the next round starts after position t. At a bonus-token position $t = n + B + 1$, the next round also starts after position t, provided that the bonus token is not EOS. Accordingly, setting

$$A^q_{n,t} = 0, \qquad R^q_{n,t} = 1, \qquad t > n + B,$$

incorporates the bonus-token case into the same transition rule. At such a position, $R^q_{n,t} = 1$ represents a forced round transition, not an actual draft rejection.

It follows that, under $\Pr(\cdot \mid X = x)$, the transitions for $t \le L$ are

$$(n,t) \longrightarrow \begin{cases} (n,t+1), & \text{with probability } A^q_{n,t}(x_{\le t}), \\ (t,t+1), & \text{with probability } R^q_{n,t}(x_{\le t}). \end{cases}$$

With x fixed, these probabilities depend on the past history only through the current state (n,t). This establishes the Markov property.

The conditional chain has states $0 \le n < t \le L+1$ and is stopped at position $L+1$. All states $(n, L+1)$ are terminal for the round-count process: the final EOS is produced in the current round and does not start another round, regardless of whether it is accepted, sampled as a correction, or sampled as a bonus token.

Occupancy depends only on the committed prefix. Since the position index increases by one at every transition, state (n,t) is visited exactly when $N_t = n$. Hence,

$$\omega^q_{n,t}(x) = \Pr(N_t = n \mid X = x).$$

By (19), conditional on the committed prefix $x_{<t}$, the probability of the remaining output $x_{t:L+1}$ is the same for every possible round-start history. Therefore,

$$\begin{aligned} \omega^q_{n,t}(x) &= \frac{\Pr(N_t = n, X_{<t} = x_{<t})\, p(x_{t:L+1} \mid x_{<t})}{\Pr(X_{<t} = x_{<t})\, p(x_{t:L+1} \mid x_{<t})} \\ &= \Pr(N_t = n \mid X_{<t} = x_{<t}) \\ &=: \omega^q_{n,t}(x_{<t}). \end{aligned}$$

This proves that the occupancy depends only on $x_{<t}$, even though it was originally defined by conditioning on the complete output sequence.

Starting from $\omega^q_{0,1} = 1$, the conditional Markov transitions give the forward recursions

$$\begin{aligned} \omega^q_{n,t+1}(x_{\le t}) &= \omega^q_{n,t}(x_{<t}) A^q_{n,t}(x_{\le t}), && 0 \le n < t \le L, \\ \omega^q_{t,t+1}(x_{\le t}) &= \sum_{n=0}^{t-1} \omega^q_{n,t}(x_{<t}) R^q_{n,t}(x_{\le t}), && 1 \le t \le L. \end{aligned}$$

The first recursion accounts for continuing the current round. The second sums over all states from which a new round can start after position t. These are precisely the occupancy recursions used in Section 3.1.

B.2 Proof of Proposition 1

Proof. The random variable $\mathcal{T}^q$ counts the number of decoding rounds. Conditioned on a fixed output sequence x, a round starts after position t exactly when state $(t, t+1)$ is visited. This event has probability $\omega^q_{t,t+1}(x_{\le t})$. Therefore, by linearity of expectation

$$\tau^q(x) = \mathbb{E}[\mathcal{T}^q \mid X = x] = \sum_{t=0}^{L} \omega^q_{t,t+1}(x_{\le t}).$$

Substituting the occupancy recursion recursion (5b) and using $\omega_{0,1} = 1$ gives

$$\tau^q(x) = \sum_{t=0}^{L} \omega^q_{t,t+1}(x_{\le t}) = 1 + \sum_{t=1}^{L} \sum_{n=0}^{t-1} \omega^q_{n,t}(x_{<t}) R^q_{n,t}(x_{\le t}).$$

□

B.3 Proof of Theorem 1

Proof. To handle the randomness of the sequence length L, We rewrite Proposition 1 and take the expectation over $X \sim p$,

$$\mathbb{E}[\tau^q(X)] = 1 + \sum_{t \ge 1} \sum_{n=0}^{t-1} \mathbb{E}\left[\mathbf{1}_{\{t \le L\}} \omega^q_{n,t}(X_{<t}) R^q_{n,t}(X_{\le t})\right]. \tag{20}$$

Before X_t is revealed, $\omega^q_{n,t}(X_{<t})$ is determined by $X_{<t}$, and $t \le L$ is exactly the event $X_t \neq$ <EOS>. Hence, conditioning on $X_{<t}$ and using the definition of $c^q_{n,t}$,

$$\begin{aligned}
&\mathbb{E}\big[\mathbf{1}_{\{t\le L\}}\omega^q_{n,t}(X_{<t})R^q_{n,t}(X_{\le t}) \mid X_{<t}\big] \\
&\quad = \omega^q_{n,t}(X_{<t})\Pr(X_t \neq \texttt{<EOS>} \mid X_{<t})c^q_{n,t}(X_{<t}) \\
&\quad = \mathbb{E}\big[\mathbf{1}_{\{t\le L\}}\omega^q_{n,t}(X_{<t})c^q_{n,t}(X_{<t}) \mid X_{<t}\big]. \qquad (21)
\end{aligned}$$

Taking expectations and summing over n and t proves (9). □

B.4 Proof of Theorem 2

Proof. We first prove (13), which relates the initial-state value to the per-sequence EDR loss.

$$\begin{aligned}
V^q_{0,1}(x) &= \sum_{t=1}^{L}\left[\sum_{n=0}^{t-1}\omega^q_{n,t}(x_{<t})V^q_{n,t}(x) - \sum_{n=0}^{t}\omega^q_{n,t+1}(x_{\le t})V^q_{n,t+1}(x)\right] \\
&= \sum_{t=1}^{L}\sum_{n=0}^{t-1}\omega^q_{n,t}(x_{<t})\left[V^q_{n,t}(x) - A^q_{n,t}(x_{\le t})V^q_{n,t+1}(x) - R^q_{n,t}(x_{\le t})V^q_{t,t+1}(x)\right] \\
&= \sum_{t=1}^{L}\sum_{n=0}^{t-1}\omega^q_{n,t}(x_{<t})c^q_{n,t}(x_{<t}) \\
&= \widehat{\mathcal{L}}_{\mathrm{EDR}}(q_\theta; x) - 1.
\end{aligned}$$

Now we prove to Theorem 2. Differentiating the Bellman equation (12) at one state gives

$$\begin{aligned}
\nabla_\theta V^q_{n,t} = \underbrace{\nabla_\theta c^q_{n,t} + \nabla_\theta A^q_{n,t}\left(V^q_{n,t+1} - V^q_{t,t+1}\right)}_{\text{local TD gradient}} \\
+ A^q_{n,t}\nabla_\theta V^q_{n,t+1} + R^q_{n,t}\nabla_\theta V^q_{t,t+1}, \qquad (22)
\end{aligned}$$

where we used $R^q_{n,t} = 1 - A^q_{n,t}$. Unrolling the last two terms from the initial state weights each local TD gradient by the probability of visiting that state, namely $\omega^q_{n,t}$. Apply (22) recursively from $(0,1)$. Every time a state (n,t) is reached, its local term is accumulated once, while the two recursive terms propagate with transition probabilities $A^q_{n,t}$ and $R^q_{n,t}$. The total coefficient on the local term at (n,t) is therefore exactly its state occupancy $\omega^q_{n,t}$. Using (13) yields (14). Rigorously,

$$\begin{aligned}
\nabla_\theta\widehat{\mathcal{L}}_{\mathrm{EDR}}(q_\theta; x) &= \nabla_\theta V^q_{0,1}(x) \\
&= \sum_{t=1}^{L}\left[\sum_{n=0}^{t-1}\omega^q_{n,t}(x_{<t})\nabla_\theta V^q_{n,t}(x) - \sum_{n=0}^{t}\omega^q_{n,t+1}(x_{\le t})\nabla_\theta V^q_{n,t+1}(x)\right] \\
&= \sum_{t=1}^{L}\sum_{n=0}^{t-1}\omega^q_{n,t}(x_{<t})\Big[\nabla_\theta V^q_{n,t}(x) - A^q_{n,t}(x_{\le t})\nabla_\theta V^q_{n,t+1}(x) - R^q_{n,t}(x_{\le t})\nabla_\theta V^q_{t,t+1}(x)\Big] \\
&= \sum_{t=1}^{L}\sum_{n=0}^{t-1}\omega^q_{n,t}(x_{<t})\left[\nabla_\theta c^q_{n,t}(x_{<t}) + \nabla_\theta A^q_{n,t}(x_{\le t})\left(V^q_{n,t+1}(x) - V^q_{t,t+1}(x)\right)\right].
\end{aligned}$$

□

B.5 Proof of Theorem 3

Proof. Let $B \ge 3$ and let the vocabulary be $\mathcal{V} = \{0, 1, a\}$. We independently generate m subsequences $W_1, \ldots, W_m$, where

$$W_i = \begin{cases} 0a^{B-2}0a, & \text{with probability } 2/3, \\ 1a^{B-2}1, & \text{with probability } 1/3, \end{cases}$$

Here a^{B-2} denotes $B-2$ repetition of token a. We refer to these as type-0 and type-1 subsequences, respectively. Let the target distribution be $X = W_1 W_2 \cdots W_m$<EOS>, so that $\mathbb{E}[L] = \left(B + \frac{2}{3}\right) m$. The integer m will be chosen at the end of the proof.

We consider first-order drafters of the form

$$q_{n,t}\left(\cdot \mid x_{1:n}; x_{n+1:t-1}\right) = q_{n,t}\left(\cdot \mid x_{1:n}; x_{t-1}\right).$$

The block-local optimum. Consider a round that starts exactly at the beginning of a subsequence, following a committed prefix $x_{1:n}$. The first $B-1$ draft positions can match the target distribution exactly:

$$\widehat{q}_{n,n+1} = \frac{2}{3}\delta_0 + \frac{1}{3}\delta_1, \qquad \widehat{q}_{n,n+j} = \delta_a, \quad 2 \le j \le B-1.$$

Here and below, these equalities are understood on target-supported within-block prefixes.

At the final draft position $n+B$, the target repeats the first bit of the subsequence. Since $B \ge 3$, the immediately preceding token is a for both types. A first-order drafter must therefore use the same distribution at this position for both types. To maximize the expected accepted length, $\widehat{q}$ should maximize the probablity of accepting the last token in the round, i.e,

$$\widehat{q}_{n,n+B} = \delta_0.$$

If $W_i = 0a^{B-2}0a$, all B draft tokens are accepted, and the target bonus token commits the final a. If $W_i = 1a^{B-2}1$, the proposal is rejected at the final draft position, and residual sampling commits the final 1. In either case, the round commits exactly the entire subsequence, and the next round starts at the beginning of W_{i+1}. After the m subsequences have been generated, one additional round generates <EOS>. Therefore, every block-local optimal drafter $\widehat{q}$ satisfies

$$\mathbb{E}[\mathcal{T}^{\widehat{q}}] = m + 1.$$

A continuation-aware drafter. We next construct another first-order drafter q^* that intentionally sacrifices accepted length in some rounds to obtain more favorable future round starts.

Whenever a round starts at the beginning of a subsequence, let q^* match the target distribution in the first $B-1$ positions:

$$q^*_{n,n+1} = \frac{2}{3}\delta_0 + \frac{1}{3}\delta_1, \qquad q^*_{n,n+j} = \delta_a, \quad 2 \le j \le B-1,$$

but choose

$$q^*_{n,n+B} = \delta_1.$$

If $W_i = 0a^{B-2}0a$, the final proposal is rejected. The round commits the corrected 0 and advances by B tokens, leaving the final a for the next round. If $W_i = 1a^{B-2}1$, all B draft tokens are accepted. The bonus token then commits the first token of the next subsequence, or <EOS> if $i = m$.

Whenever a round starts inside a subsequence, let q^* match the target conditionals at every draft position. This is possible within the first-order drafter class: the bit defining the current subsequence is already contained in the committed prefix. Any fresh bit from the next subsequence appears at draft position $j \ge 2$, and its repeated occurrence is at position $j + B - 1 \ge B + 1$, outside the draft block. Before that repeated occurrence, the intervening tokens are all a. Thus no draft position needs access to an earlier within-block bit beyond the immediately preceding token. Every such round therefore advances by $B+1$ tokens, unless generation terminates earlier at <EOS>.

To count the decoding rounds, extend $W_1, W_2, \ldots$ to an infinite independent sequence for this calculation. Group consecutive subsequences between successive rounds that start at the beginning of a subsequence.

If a group starts with a type-0 subsequence, its first round leaves the final a uncommitted. Each subsequent round either leaves the final a of another type-0 subsequence or commits an entire type-1 subsequence and returns to a subsequence boundary. The group therefore ends at the first type-1 subsequence after its initial type-0 subsequence. Its expected number of subsequences is $1 + 3 = 4$, and it uses exactly one round per subsequence.

If a group starts with a type-1 subsequence, every round advances by $B+1$ tokens until a subsequence boundary is reached again. For any initial segment of this group, its total length is $B+1$ times its number of subsequences, minus its number of type-1 subsequences. Consequently, the first return to a subsequence boundary occurs upon completing the $(B+1)$-st type-1 subsequence, counting the initial one. The expected number of subsequences in this group is $1+3B$. If the group contains ℓ subsequences, its total length is $(\ell-1)(B+1)$, so it uses exactly $\ell-1$ rounds and saves one round.

Let ℓ_j denote the number of subsequences in group j. The groups are independent and identically distributed, with

$$\mathbb{E}[\ell_j] = \frac{2}{3}\cdot 4 + \frac{1}{3}(1+3B) = B+3.$$

Let

$$N := \min\left\{ k \geq 1 : \sum_{j=1}^{k} \ell_j \geq m \right\},$$

and let H be the number of these N groups that start with a type-1 subsequence. Since $\{N \geq j\}$ depends only on the preceding groups and $N \leq m$, summing expectations gives

$$m \leq \mathbb{E}\left[\sum_{j=1}^{N} \ell_j\right] = \sum_{j=1}^{m} \mathbb{E}\big[\mathbf{1}_{\{N\geq j\}}\ell_j\big] = (B+3)\mathbb{E}[N],$$

$$\mathbb{E}[H] = \sum_{j=1}^{m} \Pr(N \geq j,\ \text{group } j \text{ starts with type 1}) = \frac{1}{3}\mathbb{E}[N].$$

All groups except possibly the last are fully generated before termination, and each completed type-1 group saves one round. For the last group, suppose that r of its subsequences remain before <EOS>. These subsequences contain at most $r(B+1)$ tokens, and only the first round of the group can lose one token to rejection. Thus generating these subsequences together with <EOS> requires at most

$$\left\lceil \frac{r(B+1)+2}{B+1} \right\rceil = r+1$$

rounds. At most one of the H type-1 groups is the last group, so

$$\mathcal{T}^{q^*} \leq m+2-H.$$

Taking expectations yields

$$\begin{aligned}\mathbb{E}[\mathcal{T}^{q^*}] &\leq m+2-\mathbb{E}[H] = m+2-\frac{1}{3}\mathbb{E}[N] \\ &\leq \left(1-\frac{1}{3(B+3)}\right)m+2.\end{aligned}$$

Finally, choose any integer $m > (3B+9)(3B+11)$. Since both drafters preserve the same target distribution,

$$\begin{aligned}\frac{\mathsf{MAL}(q^*)}{\mathsf{MAL}(\widehat{q})} &= \frac{\mathbb{E}[\mathcal{T}^{\widehat{q}}]}{\mathbb{E}[\mathcal{T}^{q^*}]} \\ &\geq \frac{m+1}{\left(1-\frac{1}{3(B+3)}\right)m+2} \\ &> 1+\frac{1}{3B+9}.\end{aligned}$$

This proves the theorem. □

C Additional details for Expected Decoding Rounds

C.1 Draft EOS probability does not reduce EDR

For the EDR objective, a correctly drafted terminal <EOS> does not save a target-model verification round. Conditional on reaching the terminal position, accepting a drafted <EOS> terminates in the current verification pass. If the draft instead misses <EOS>, the correction branch can generate <EOS> in that same pass and terminates before another round starts. The same holds when the terminal token is produced as the target bonus token after block exhaustion. Accordingly, the EDR sum runs only over ordinary-token positions $t \le L$.

This is also visible in the dense local cost $c_{n,t}$. Its numerator is

$$\sum_{a \in \mathcal{V}} \left[p_t(a \mid x_{<t}) - q_{n,t}(a \mid x_{1:n}; x_{n+1:t-1}) \right]_+ ,$$

which excludes <EOS>. Thus matching target EOS mass is not directly rewarded by EDR.

C.2 Fixed-size anchor sampling

In this section, we explain how to use importance sampling to select a fixed number of anchors in EDR training. We apply anchor sampling separately to each target rollout in Algorithm 1. For a rollout $x = (x_{1:L}, \texttt{<EOS>})$, define

$$\mathcal{N}_+(x) := \left\{ 0 \le n < L : \omega^{q_\theta}_{n,n+1}(x_{1:n}) > 0 \right\}.$$

An anchor outside $\mathcal{N}_+(x)$ contributes zero to (14), since its occupancy $\omega^{q_\theta}_{n,t}(x_{<t})$ is zero for every $t > n$.

Let ρ be the maximum number of training anchors per rollout, set to 512 in our experiments. If $\rho \ge |\mathcal{N}_+(x)|$, select $\mathcal{I} = \mathcal{N}_+(x)$ and set $\pi_n(x) = 1$ for every $n \in \mathcal{N}_+(x)$. Otherwise, choose $\lambda(x) > 0$ such that

$$\sum_{n \in \mathcal{N}_+(x)} \min\left\{1, \lambda(x) \omega^{q_\theta}_{n,n+1}(x_{1:n})\right\} = \rho, \tag{23}$$

and set

$$\pi_n(x) := \min\left\{1, \lambda(x) \omega^{q_\theta}_{n,n+1}(x_{1:n})\right\}. \tag{24}$$

We draw a set $\mathcal{I} \subseteq \mathcal{N}_+(x)$ of exactly ρ anchors by random-start systematic sampling, with

$$\Pr(n \in \mathcal{I} \mid x) = \pi_n(x).$$

For rollout $x^{(i)}$, denote the sampled anchor set by $\mathcal{I}^{(i)}$ and write $\pi^{(i)}_n := \pi_n(x^{(i)})$. Using the same notation as Algorithm 1, replace its loss by

$$\ell(\theta) = \frac{1}{M} \sum_{i=1}^{M} \sum_{n \in \mathcal{I}^{(i)}} \sum_{t=n+1}^{L_i} \mathrm{sg}\left(\frac{\omega^{(i)}_{n,t}}{\pi^{(i)}_n} \right) \left[c^{(i)}_{n,t} + A^{(i)}_{n,t} \, \mathrm{sg}\Big(V^{(i)}_{n,t+1} - V^{(i)}_{t,t+1} \Big) \right]. \tag{25}$$

The occupancies and value functions are still computed by the full recursions in Algorithm 1, without gradients; sampling restricts only which anchor contributions are differentiated. The sampled anchor sets and inclusion probabilities are also held fixed during differentiation.

Conditioned on the target rollouts, the inclusion probabilities cancel the inverse-probability weights. Hence,

$$\begin{aligned}
&\mathbb{E}_{\{\mathcal{I}^{(i)}\}_{i=1}^M} \left[\nabla_\theta \ell(\theta) \,\middle|\, \{x^{(i)}\}_{i=1}^M \right] \\
&\quad = \frac{1}{M} \sum_{i=1}^{M} \sum_{n \in \mathcal{N}_+(x^{(i)})} \sum_{t=n+1}^{L_i} \omega^{(i)}_{n,t} \left[\nabla_\theta c^{(i)}_{n,t} + \nabla_\theta A^{(i)}_{n,t} \left(V^{(i)}_{n,t+1} - V^{(i)}_{t,t+1} \right) \right] \\
&\quad = \frac{1}{M} \sum_{i=1}^{M} \nabla_\theta \widehat{\mathcal{L}}_{\mathrm{EDR}}(q_\theta; x^{(i)}),
\end{aligned} \tag{26}$$

where the last equality follows from (14). Taking the expectation over the target rollouts and applying (15) gives

$$\mathbb{E}[\nabla_\theta \ell(\theta)] = \nabla_\theta \mathcal{L}_{\mathrm{EDR}}(q_\theta).$$

Thus anchor sampling preserves its expectation.

D Additional Details for Experiments

D.1 Experimental Hyperparameters

For all training settings and methods, we use the hyperparameters summarized in Table 2.

Table 2: Training hyperparameters used in all experiments.

Hyperparameter	**Value**
Learning rate	3×10^{-5}
Optimizer	AdamW
AdamW β_1	0.9
AdamW β_2	0.999
Learning-rate schedule	4% linear warmup, then constant
Gradient clipping	Global norm ≤ 1
Global batch size	96

D.2 Statistical Significance Analysis

We assess the statistical significance of the MAL differences between DFly drafters finetuned with E2E and EDR using Qwen3-8B as the target model. For each benchmark, we hold the evaluation prompts fixed and evaluate both drafters on the same target rollouts. We estimate the sampling variance from within-prompt paired differences in offline expected round counts, compute two-sided p-values using a Welch-Satterthwaite approximation (Satterthwaite, 1946; Welch, 1947), and apply Holm-Bonferroni method (Holm, 1979) across the nine benchmarks. Since both drafters share the same expected output length, testing equality of their expected round counts is equivalent to testing equality of their MAL.

Table 3 shows that EDR achieves higher MAL than E2E on all nine benchmarks. All improvements remain statistically significant after Holm correction, with $p_{\mathrm{Holm}} < 0.001$.

Table 3: **Fixed-prompt significance analysis for Qwen3-8B with DFly.** We report MAL for E2E and EDR finetuning, $\Delta\mathrm{MAL} = \mathrm{MAL}_{\mathrm{EDR}} - \mathrm{MAL}_{\mathrm{E2E}}$, and Holm-adjusted two-sided p-values. Differences are computed before rounding.

Dataset	E2E MAL	EDR MAL	ΔMAL	p_{Holm}
GSM8K	6.34125	**6.34649**	$+0.00524$	4.64×10^{-4}
MATH500	5.97944	**5.99506**	$+0.01562$	7.92×10^{-34}
AIME25	5.18413	**5.21711**	$+0.03298$	3.67×10^{-284}
MBPP	5.31545	**5.32376**	$+0.00831$	9.00×10^{-8}
HumanEval	5.62217	**5.63452**	$+0.01236$	5.79×10^{-21}
LiveCodeBench	5.25915	**5.28863**	$+0.02947$	2.72×10^{-25}
MT-Bench	3.61215	**3.64628**	$+0.03413$	2.26×10^{-149}
Alpaca	3.48503	**3.52207**	$+0.03704$	3.16×10^{-67}
Arena-Hard-v2	3.20785	**3.24771**	$+0.03986$	7.62×10^{-26}