SPECS: Faster Test-Time Scaling through Speculative Drafts and Dynamic Switching

Mert Cemri[1,*] **Nived Rajaraman**[1,4*] **Rishabh Tiwari**[1,*] **Xiaoxuan Liu**[1] **Kurt Keutzer**[1]
Ion Stoica[1] **Kannan Ramchandran**[1] **Ahmad Beirami**[2] **Ziteng Sun**[3]

[1]UC Berkeley [2]Google DeepMind [3]Google Research [4]Microsoft Research
[*]Equal contribution

{cemri, nived.rajaraman, rishabhtiwari}@berkeley.edu
ahmad.beirami@gmail.com, zitengsun@google.com

Abstract

Scaling test-time compute has driven the recent advances in the reasoning capabilities of large language models (LLMs). However, increased compute often comes at the expense of higher user-facing latency, directly impacting user experience. Current test-time scaling methods primarily optimize for accuracy based on total compute resources (FLOPs), often overlooking latency constraints. To address this gap, we propose SPECS, a latency-aware test-time scaling method. SPECS uses a smaller, faster model to generate multiple candidates for each step of reasoning in parallel, and evaluates these candidates using a larger model and a quality critic. We design a theoretically grounded soft verification strategy to select a high-quality candidate to continue the generation. SPECS also employs a dynamic switching mechanism to use speculative drafts only for easier steps to maintain reasoning accuracy. Empirical results on a diverse and challenging set of reasoning and alignment benchmarks show that SPECS matches or surpasses the accuracy of SOTA test-time scaling methods while reducing latency by up to $\sim$26%. Our theoretical analysis shows that as the amount of parallel compute scales, SPECS converges to an optimum of a KL-regularized reinforcement learning problem and is robust to misspecification in the quality critic.

1 Introduction

Modern LLMs excel at multi-step reasoning, and scaling test-time compute has played a major role in achieving these reasoning capabilities by letting these models tackle harder problems with extra "thinking" compute resources [1–8]. To date, test-time scaling methods have primarily optimized performance based on total compute, demonstrating improved downstream task performance with increased FLOPs or generated tokens [7, 9]. However, user experience often depends more directly on serving *latency*, especially in low-throughput scenarios like personalized interactions or applications serving few users [10, 11]. With this in mind, we study the critical research question:

How do we design test-time scaling methods which optimize for inference latency while maintaining accuracy?

The problem of improving the latency of autoregressive generation has received significant attention, and resulted in approaches such as speculative decoding[1] [12, 13]. However, the objective of test-time scaling is *not* to just decode losslessly from a larger model, but is to maximize reasoning accuracy.

[1]Speculative decoding enables lossless decoding from a large model by using a smaller drafter to propose candidate tokens. The candidate tokens from a small (draft) model are verified in parallel by a large (target) model, reducing decoding latency.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

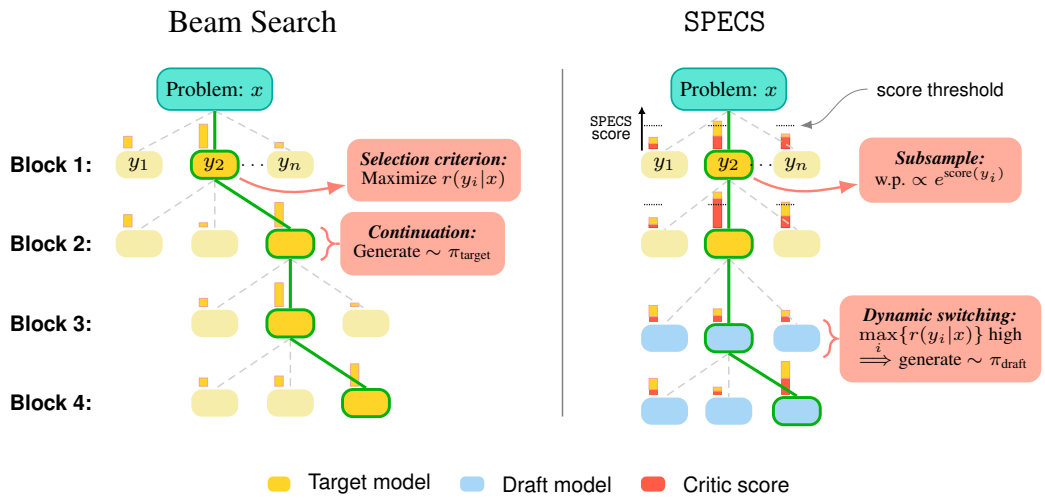


Figure 1: Visualization of Beam Search vs SPECS. In beam search the trajectories are generated by the target model (p) and scored using a critic ($r_{\texttt{critic}}$). In contrast, in SPECS the beams are dynamically switched to generation from the draft model, and scored by a combination of the target model and the critic, resulting in better latency-performance tradeoff. Draft proposal and selection are further controlled by the SUBSAMPLE subroutine (see Section 2 for details).

In this paper, we fix this mismatch by developing algorithms which directly try to optimize the *latency-accuracy tradeoff*. Our main contributions are summarized as follows.

1.1 Main Contributions

Contribution 1. *We propose a novel tree-search algorithm for test-time scaling,* SPECS*, which uses speculative drafting and dynamic switching to optimize the latency-accuracy tradeoff.*

SPECS (SPECulative drafting for faster test-time Scaling) belongs to the widely-adopted family of tree search approaches for test-time scaling [5, 6, 9, 14, 15]. In each iteration, multiple candidate reasoning steps (i.e., beams) are generated in parallel. The quality of each candidate is then judged by a *critic*, which could be either a separate process reward model (PRM) [16], or based on self-rewards such as model confidence [17, 18]. SPECS begins by generating from a large model, and dynamically switches to generating from a smaller draft model when the critic scores the trace highly. This builds on novel insights and empirical evidence we collect showing that the accuracy gap between the small and large models is negligible for such traces.

Contribution 2. *We develop a novel and theoretically grounded strategy to select promising candidate steps generated by the draft model, which integrates target model and critic scores.*

As a byproduct of such a scoring mechanism, we show that SPECS is theoretically sound. As beam-width (i.e., parallel compute) increases, the algorithm is shown to converge gracefully to the optimal solution of a KL-regularized reward maximization problem, up to an error term depending on the misspecification of the critic reward. Furthermore, our theory shows that SPECS is robust to misalignment in the critic's scores. We test robustness to additive noise in the critic's scores empirically (Section B.11).

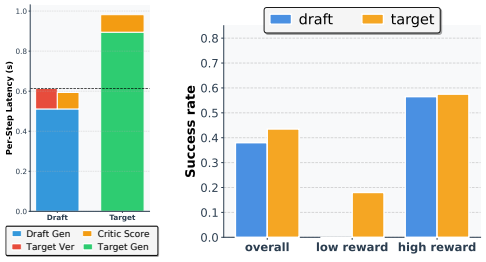
Contribution 3. *We validate* SPECS *across a wide range of challenging reasoning and alignment benchmarks to demonstrate its improved latency-accuracy tradeoff, demonstrating up to a 26% latency reduction while achieving on-par or higher accuracy compared to baselines.*

We evaluate SPECS on several math reasoning (MATH-500, AMC23, OlympiadBench), general purpose reasoning (GPQA), and alignment (HH-RLHF) benchmarks, comparing it against several

strong baselines including beam search, Best-of-N, RSD [19], and concurrent work which proposes GSI [20]. We evaluate it using the the (Qwen-1.5B-Instruct & Qwen-7B-Instruct) and (Llama3.2-1B-Instruct & Llama3.1-8B-Instruct) families of draft-target model pairs. On the reasoning benchmarks, we use Skywork-o1-Open-PRM-Qwen-2.5-7B as the critic. We also conduct detailed ablation studies to test the functionality and robustness of the individual components of SPECS in improving the latency-accuracy curve over baselines (Section 4.3).

Paper organization. We present a description of SPECS in Section 2, as well as several observations that motivate its design, which may also be of independent interest. The theoretical guarantees for SPECS are presented in Section 3. Empirical evaluations and findings are presented in Section 4. The paper concludes with a summary and discussions on future work. We refer readers to Section A for a detailed comparison to related work.

2 Speculative Drafting with Soft Verification



(a) (b)

Figure 2: **(a)** Latency of generation from the target model (Qwen2.5-7B-Instruct) vs. generation from the draft model (Qwen2.5-1.5B-Instruct) with scoring: scoring by the target model and critic can be parallelized. **(b)** We generate 8 steps from the target model and complete the remaining steps either using the draft or the target model. The prefixes generated by the target model are bucketed into high (critic score is ≥ 0.5) and low critic score. Using the draft model to complete high-critic-score traces achieves a similar solve rate as if they were completed using the target model. Section B.3 repeats (a) and (b) for the Qwen and Llama pairs on three datasets and gives the full setup.

In this section, we motivate design of SPECS by discussing some observations that lead to its development, which may be of independent interest. A natural baseline for test-time scaling is *beam search*, where multiple candidate steps (i.e., beams) are generated from a large model in parallel, and generation is continued from promising candidates (as scored by a critic). The most significant latency cost for beam search with the large model is autoregressive generation to obtain the beams.

This latency cost can be reduced by instead speculatively generate beams using a smaller and faster draft model (as in speculative decoding [12]), which are then validated using the large target model, and scores the resulting responses using the critic. The latency gain comes from the fact that autoregressive sampling in modern LLMs is bottlenecked by memory bandwidth, so validating a step using the target model (i.e., computing the logits of each token) can be parallelized and is much faster than generating a reasoning steps from the same model, despite costing the same number of FLOPs. Our first observation is that validating a reasoning step using the target model and scoring it using the critic can be executed in parallel. Thus, in the test-time scaling setting, the gains of speculative execution can be even more dramatic. We carry out an experiment in Figure 2a to understand the typical per-step latency improvements with modern reasoning and reward models. Repeating this measurement for both model pairs, we find that drafting lowers the mean step latency from 1.55 to 1.25 s (Qwen) and from 1.18 to 0.84 s (Llama) (Section B.3).

Observation 1 (*Drafting with parallel scoring*)**.** Generating candidate sequences from the draft model and scoring them using the large target model and critic in parallel can be significantly faster compared to generating and scoring responses directly from the target model.

While generating reasoning steps from a draft model is faster, their average quality will be worse since a less capable model was used. Careful draft selection is needed to guarantee a minimal drop in accuracy. Speculative decoding [12, 21, 22] validates reasoning steps by rejection sampling to match the target model distribution. However, in the process, candidates may be unnecessarily discarded, even if they could lead to an accurate reasoning trace. It is reasonable to expect that *soft* verification, which relaxes the distribution matching guarantee, can obtain a further speedup, while still maintaining high accuracy on reasoning tasks.

On the other hand, reasoning datasets typically evaluate model capabilities across a range of skills and difficulty levels. At a finer granularity, certain steps of the solution to a problem might be easy, and a large target model may be overprovisioned to generate steps that the draft model can already solve. For harder steps, it makes sense to defer to the more capable target model for candidate generation. These discussions motivate the following two questions,

RQ1. How do we identify steps where speculative drafting will not significantly hurt accuracy?

RQ2. Given a set of candidate steps how do we best select candidates based on scores from the target model and the critic?

Toward answering the first question, we study how the critic score of reasoning trace evolves over the course of a generation, and how it relates to the final accuracy of the generated response. In Figure 2b we carry out two inference runs, where we start by generating 8 steps from the target model. In one case, we complete the trace using the large target model. In the other, we use the draft model. Conditioning on the traces having high critic score at the 8th step, we observe that completing the trace using the draft model performs nearly as well as using the target model.

Observation 2 (*Draft model can complete high-critic-score traces*)**.** Generating blocks from the draft model costs little accuracy when the critic's score of a partial trace is high.

In Figure 2b we also see that continuing generation from the draft model starting from low-critic-score traces performs poorly. To minimize any decrease in accuracy from speculative drafting, it is reasonable to begin with generating steps from the target model until high-critic-score traces are encountered and use the critic to decide when to switch. With these observations, we present the SPECS algorithm.

2.1 The SPECS Algorithm

SPECS follows a draft-and-select framework where the final response is generated step-by-step. In each iteration, the algorithm uses a generation model $\pi_{\texttt{gen}}$ to propose candidate blocks $\{Y_i\}_{i=1}^n$ to extend the current reasoning trace $\texttt{tr}$ through autoregressive sampling. $\pi_{\texttt{gen}}$ is initialized to be a larger reasoning model $\pi_{\texttt{target}}$, and over the course of generation switches to a smaller draft model $\pi_{\texttt{draft}}$. When $\pi_{\texttt{gen}} = \pi_{\texttt{draft}}$, we call this a speculative drafting step.

The candidate blocks are then evaluated by both a larger, more capable reasoning model $\pi_{\texttt{target}}$ (for speculative drafting step), and a critic function $r_{\texttt{critic}}$. A score (eq. (1)) is assigned to each candidate based on the evaluations, which is then used to select a candidate block (summarized in Algorithm 2). The selected block is appended to the existing response to begin the next iteration.

The score is used to adaptively decide which model generates candidate blocks. This approach is most closely related to "cascades" [23–26] and aims to ensure that the larger model generates candidates for the harder steps of a problem, while the smaller draft model is used for the easier steps.

Algorithm 1 SPECS algorithm

Input: Prompt X, draft model $\pi_{\texttt{draft}}$,
target model $\pi_{\texttt{target}}$, critic $r_{\texttt{critic}}$,
Hyperparameters: Beam-width n,
Block-size ℓ,
Switching threshold τ.
Initialize: Partial response $\texttt{tr} \leftarrow X$
Draft generator $\pi_{\texttt{gen}} \leftarrow \pi_{\texttt{target}}$
// Generation begins with target model.
Repeat
Step I. Generate candidate blocks
Draw n blocks $\sim \pi_{\texttt{gen}}$ of block-size ℓ,

$$\mathbf{Y}_{\text{draft}} = \{Y_i\}_{i=1}^n \overset{\text{i.i.d.}}{\sim} \pi_{\texttt{gen}}(\cdot|\texttt{tr})$$

Step II. Select candidate block
$Y = \textsc{SubSample}(\mathbf{Y}_{\text{draft}}, \beta, \pi_{\texttt{gen}}|\texttt{tr})$,
// Selects a draft using scores in eq. (1)
Step III. Dynamic switching
if $\max_{i\in[n]} r_{\texttt{critic}}(Y_i|\texttt{tr}) > \tau$
$\pi_{\texttt{gen}} \leftarrow \pi_{\texttt{draft}}$ // Switch to draft model
// if critic score $r_{\texttt{critic}}$ is large
$\texttt{tr} \leftarrow (\texttt{tr}, Y)$
Until: Target sequence length is achieved.
Return: $\texttt{tr}$

Candidate selection via SUBSAMPLE. This subroutine carries out the selection of candidate steps $\{Y_i\}_{i=1}^n$. The goal is to select a response among these which simultaneously has high reward (i.e., completions are likely to lead to a correct response) while not having low probability under the larger target model. SUBSAMPLE achieves this by combining evaluations from the large reasoning model and critic via the SPECS score function:

$$\log\left(\frac{\pi_{\texttt{target}}(Y_i|\texttt{tr})}{\pi_{\texttt{gen}}(Y_i|\texttt{tr})}\right) + \beta r_{\texttt{critic}}(Y_i|\texttt{tr}) \tag{1}$$

One among the n candidate responses is selected with probability proportional to the exponential of the above score. When $\pi_{\texttt{gen}} = \pi_{\texttt{target}}$, the score collapses to $\beta r_{\texttt{critic}}$. With $\pi_{\texttt{gen}} = \pi_{\texttt{draft}}$, it also includes the log density ratio between draft and target models. We formalize the choice of score in Section 3.1 through the lens of optimizing the expected reward with KL regularization (to the target model).

Algorithm 2 SUBSAMPLE subroutine

Input: Context $\texttt{tr}$, Draft blocks $\{Y_i\}_{i=1}^n$, Target model $\pi_{\texttt{target}}$, critic $r_{\texttt{critic}}$, Draft generator $\pi_{\texttt{gen}}$,
Hyperparameters: Temp. β
for $i = 1$ **to** n,
$S_i = \log\left(\frac{\pi_{\texttt{target}}(Y_i|\texttt{tr})}{\pi_{\texttt{gen}}(Y_i|\texttt{tr})}\right) + \beta r_{\texttt{critic}}(Y_i|\texttt{tr})$
Sample $i^\star$ w.p., $P(i^\star = i) \propto e^{S_i}$.
Return: $Y_{i^\star}$

Deferrals via dynamic switching. SPECS uses a deferral rule to determine which (draft or target) model to use for generating candidate blocks in each subsequent iteration. Along the lines of Observation 2, the algorithm begins generation from the target model[2], since the critic's score is low to begin with. The algorithm switches to generating reasoning steps from the draft model if high reward partial traces are present among the current set of beams (i.e., their critic score exceeds a threshold). This builds on our finding in Observation 2: it is beneficial from a latency point-of-view to switch to generating from the draft model when we already have traces with high reward.

3 A Theoretical Perspective on SPECS

In this section, we discuss the theoretical motivations and guarantees of SPECS by showing that as the beam-width (i.e., amount of parallel test-time compute) increases, the distribution of output of SPECS converges (up to an unavoidable misspecification error) to the optimal policy of a KL-regularized reward maximization problem [27, 28], which is widely used for reward-guided LLM generation and post-training. With $\pi_{\texttt{target}}$ as the reference model, and with a ground-truth reward function r, the objective is to maximize,

$$\mathcal{L}_\beta(\pi) = \mathbb{E}_{\rho,\pi}[r(Y|X)] - \frac{1}{\beta} D_{\mathsf{KL}}(\pi \| \pi_{\texttt{target}}). \tag{2}$$

Instead of directly maximizing the expected reward, the objective is smoothed by a KL regularization term to encourage the policy to stay close to the target model and avoid reward hacking, making it more robust to imperfect or noisy rewards.

3.1 Motivation Behind Scores in SUBSAMPLE

The optimal solution to eq. (2) has an explicit form [29–31], $\pi^\star_\beta(y|x) \propto \pi_{\texttt{target}}(y|x) e^{\beta r(y|x)}$. For complete responses generated from a different model $\pi_{\texttt{gen}}$, the optimal policy can be written by importance reweighting as,

$$\pi^\star_\beta(y|x) \propto \pi_{\texttt{gen}}(y|x) e^{S(y|x)}, \tag{3}$$

where $S(y|x) = \log\big(\pi_{\texttt{target}}(y|x)/\pi_{\texttt{gen}}(y|x)\big) + \beta r(y|x)$. This identity motivates the block-level score in eq. (1), where SUBSAMPLE uses the critic $r_{\texttt{critic}}$ in place of r. At each prefix, importance reweighting gives the same limiting next-block distribution for either generator. The product of these locally normalized distributions is the policy $\widetilde{\pi}_\beta$ defined below, which can differ from the global optimum $\pi^\star_\beta$.

We defer a more detailed analysis and interpretation of SUBSAMPLE as an instantiation of sequential monte carlo (SMC) or particle filtering [32, 33]. For complete-response candidates, these methods attempt to sample from the distribution in eq. (3) by approximating the partition function, $Z(x) = \mathbb{E}_{Y\sim\pi_{\texttt{gen}}(\cdot|x)}[\exp(S(Y|x))]$, via computing an empirical estimate over a finite set of "particles".

3.2 Theoretical Guarantees for SPECS

In this section, we describe the theoretical guarantees for SPECS. First we formalize some notation.

[2]This is in contrast to several closely related approaches such as [19, 20], which begin with the draft model and escalate to the target model when the reward signal is low.

Notation. Given a (partial) response y, let $(y_1, y_2, \cdots)$ denote the reasoning steps contained within and let $y_{\le t} = (y_1, \cdots, y_t)$ denote the prefix of the first t blocks. By default, we let $y_{\le 0} = \emptyset$. H denotes the horizon: the maximum number of blocks that models are allowed to generate. Let ρ denote a target distribution over prompts. The notation $\mathbb{E}_\rho[\cdot]$ denotes expectation over $X \sim \rho$, and $\mathbb{E}_{\rho,\pi}[\cdot]$ is shorthand for $\mathbb{E}_{X\sim\rho, Y\sim\pi(\cdot|X)}[\cdot]$, where Y is a full response drawn from π.

Information theoretic divergences. For a pair of distributions $p \ll q$, denote the KL divergence between them by $D_{\mathsf{KL}}(p\|q) = \mathbb{E}_{X\sim p}[\log(p(X)/q(X))]$. Define the order-2 Renyi divergence by $D_2(p\|q) = \log(\mathbb{E}_{X\sim p}[p(X)/q(X)])$. More generally, for policies π_1, π_2, denote $D_{\mathsf{KL}}(\pi_1\|\pi_2) = \mathbb{E}_\rho[D_{\mathsf{KL}}(\pi_1(\cdot|X)\|\pi_2(\cdot|X))]$ and $D_2(\pi_1\|\pi_2) = \mathbb{E}_\rho[D_2(\pi_1(\cdot|X)\|\pi_2(\cdot|X))]$.

We study the objective of sampling from a policy that maximizes the KL-regularized reward (eq. (2)). While r in eq. (2) is a ground-truth terminal reward, eq. (1) is implemented using a PRM, $r_{\texttt{critic}}$, which could induce different behaviors compared to the ground truth reward r. Thus, we expect the performance of the algorithm to depend on the quality of $r_{\texttt{critic}}$. Below we introduce a notion called the *misspecification* of the critic, capturing its quality.

Definition 1 (Misspecification of the critic)**.** Define the policy, $\widetilde{\pi}_\beta(y|x) = \prod_{t=1}^{H} \widetilde{\pi}_\beta(y_t|x, y_{\le t-1})$, where each next-block distribution $\widetilde{\pi}_\beta(y_t|x, y_{\le t-1}) \propto \pi_{\texttt{target}}(y_t|x, y_{\le t-1}) e^{\beta r_{\texttt{critic}}(y_t|x, y_{\le t-1})}$. Define the misspecification of the critic by

$$\varepsilon_{\texttt{critic}} = D_2(\widetilde{\pi}_\beta \| \pi^\star_\beta)$$

Remark 1. Informally, the policy $\widetilde{\pi}_\beta(\cdot|x, y_{\le t-1})$ is the limiting distribution over blocks (conditioned on the prompt x and partial trace $y_{\le t-1}$) generated by SubSample as the beam width $n \to \infty$. To interpret $\varepsilon_{\texttt{critic}}$, note that in the special case where $H = 1$, we have $\varepsilon_{\texttt{critic}} \le 2\beta\|r_{\texttt{critic}} - r\|_\infty$, equalling 0 when $r_{\texttt{critic}} = r$. More generally, if $r_{\texttt{critic}}(y_t|x, y_{\le t-1}) = \frac{1}{\beta}\log \mathbb{E}_{Y\sim\pi_{\texttt{target}}(\cdot|x)}[\exp(\beta r(Y|x)) \,|\, Y_{\le t} = y_{\le t}]$, i.e., the *optimal soft value function* [34] under $\pi_{\texttt{target}}$, then $\varepsilon_{\texttt{critic}} = 0$.

With this definition of misspecification in place, we have the following main result.

Theorem 1. *Assume $H \ge 1$, $\beta > 0$, $n \ge 3$, and $0 \le r, r_{\texttt{critic}} \le 1$ pointwise. The policy $\pi_{\texttt{SPECS}}$ returned by* SPECS *with beam-width n satisfies*

$$D_{\mathsf{KL}}(\pi_{\texttt{SPECS}} \| \pi^\star_\beta) \le \mathcal{O}\left(\frac{H C^2_{\texttt{block}} e^{2\beta}}{n^2}\right) + \varepsilon_{\texttt{critic}},$$

where the hidden constant is universal, $\varepsilon_{\texttt{critic}}$ is defined in Definition 1, and

$$C_{\texttt{block}} = \sup_{\substack{x\in\mathcal{X},\ 0\le t<H \\ y_{\le t}}} \left\| \frac{\pi_{\texttt{target}}(\cdot|x, y_{\le t})}{\pi_{\texttt{draft}}(\cdot|x, y_{\le t})} \right\|_\infty \left\| \frac{\pi_{\texttt{draft}}(\cdot|x, y_{\le t})}{\pi_{\texttt{target}}(\cdot|x, y_{\le t})} \right\|_\infty . \tag{4}$$

Here $\pi(\cdot|x, y_{\le t})$ denotes the next-block distribution given the partial trace $y_{\le t}$ and prompt x, and the supremum is over prompts and prefixes in the common support.

The proof of this result is deferred to Section C.

Remark 2. As a sanity check: when $n \to \infty$, $\pi_{\texttt{SPECS}}$ converges to $\widetilde{\pi}_\beta$ and therefore Theorem 1 is tight as $n \to \infty$. The KL to $\pi^\star$ decays *quadratically* in the beam-width, and depends on $C_{\texttt{block}}$, which captures how close the draft model is to the target model.

4 Experiments

We test SPECS on the latency–accuracy tradeoff against several test-time scaling baselines, with a primary focus on math reasoning benchmarks. We also include experiments on multi-disciplinary QA and preference alignment datasets to demonstrate the broad applicability of our method. We state our main results and briefly discuss our main findings on component ablations in the section. We present details on ablation studies, as well as studies on hyperparameter sensitivity, robustness to noisy reward, and latency breakdowns in Section B.

Table 1: **Latency-accuracy tradeoff of SPECS vs baselines on math reasoning benchmarks.** n denotes per-step beam width, 'Lat (s)' is average per-problem wall-clock latency, 'Lat/step (s)' is per-problem latency divided by the per-problem step count, and '% B' is the percentage of steps generated by the big model B averaged across problems. Each number is an average over 3 independent runs, with $\pm$ showing the standard deviation. The bolded accuracies are the highest in each cell, and the underlined accuracies are ones whose confidence intervals overlap with the highest.

Method	n	Acc (%) ↑	Lat (s) ↓	Lat/step (s) ↓	% B	Acc (%) ↑	Lat (s) ↓	Lat/step (s) ↓	% B
		Qwen2.5 Family				Llama3 Family			
AMC23									
BS (S)	4	38.3 ± 4.7	10.3 ± 0.6	0.80 ± 0.01	0	11.7 ± 5.1	5.94 ± 0.24	0.38 ± 0.15	0
BS (B)	4	**59.2 ± 1.2**	18.3 ± 1.5	1.46 ± 0.06	100	**26.3 ± 1.3**	12.7 ± 1.3	0.59 ± 0.17	100
RSD++	4	31.2 ± 17.8	12.6 ± 3.8	0.82 ± 0.60	60.5 ± 1.4	15.0 ± 7.4	8.9 ± 1.1	0.66 ± 0.79	91.8 ± 0.6
BS (B)+SD	4	57.2 ± 1.2	16.64 ± 1.5	2.03 ± 0.06	100	25.1 ± 1.3	8.86 ± 1.3	1.14 ± 0.17	100
GSI	4	44.2 ± 3.8	21.0 ± 1.5	2.98 ± 0.27	60.1 ± 2.6	19.6 ± 4.0	21.9 ± 2.6	1.36 ± 0.06	85.3 ± 1.1
SPECS	4	**59.2 ± 3.1**	14.8 ± 0.5	1.42 ± 0.13	67.9 ± 1.8	**26.3 ± 5.3**	15.4 ± 1.8	0.65 ± 0.03	96.4 ± 1.2
BS (S)	8	46.7 ± 3.1	17.6 ± 2.3	1.14 ± 0.01	0	14.2 ± 4.2	11.85 ± 0.22	0.61 ± 0.46	0
BS (B)	8	59.2 ± 5.1	21.6 ± 1.1	2.00 ± 0.06	100	27.5 ± 2.0	19.1 ± 0.96	0.84 ± 1.39	100
RSD++	8	49.2 ± 1.2	20.3 ± 1.2	1.99 ± 4.43	61.1 ± 0.3	18.3 ± 5.9	10.8 ± 1.9	0.91 ± 0.8	90.0 ± 1.1
BS (B)+SD	8	61.3 ± 5.1	22.66 ± 1.1	2.78 ± 0.06	100	25.7 ± 2.0	11.22 ± 0.96	1.46 ± 1.39	100
GSI	8	48.3 ± 3.8	27.8 ± 2.0	3.62 ± 0.18	51.9 ± 1.7	23.2 ± 1.9	37.6 ± 1.6	1.94 ± 0.09	77.4 ± 0.8
SPECS	8	**62.5 ± 2.0**	20.7 ± 2.0	1.94 ± 0.11	62.8 ± 2.1	**30.0 ± 2.5**	28.0 ± 1.2	1.04 ± 0.05	93.6 ± 1.0
BS (S)	16	50.8 ± 3.1	25.6 ± 3.1	1.81 ± 0.03	0	24.2 ± 4.7	24.47 ± 1.21	1.13 ± 0.95	0
BS (B)	16	**68.3 ± 4.2**	35.0 ± 2.6	2.80 ± 0.13	100	32.5 ± 4.7	32.1 ± 2.4	1.24 ± 1.94	100
RSD++	16	50.0 ± 2.0	25.0 ± 0.8	2.56 ± 1.78	55.6 ± 2.2	20.0 ± 4.1	16.3 ± 0.4	1.18 ± 1.09	90.0 ± 2.0
BS (B)+SD	16	65.3 ± 4.2	34.27 ± 2.6	2.99 ± 0.13	100	32.7 ± 4.7	21.59 ± 2.4	2.31 ± 1.94	100
GSI	16	56.7 ± 3.8	39.7 ± 3.6	3.28 ± 0.56	49.6 ± 1.4	30.1 ± 3.2	60.9 ± 4.0	1.90 ± 0.12	69.2 ± 0.8
SPECS	16	**68.3 ± 2.4**	28.8 ± 3.5	2.70 ± 0.05	70.3 ± 1.7	**36.3 ± 3.8**	44.2 ± 2.9	1.56 ± 0.10	98.1 ± 1.1
MATH500									
BS (S)	4	64.3 ± 2.9	7.9 ± 1.0	0.65 ± 0.04	0	23.1 ± 1.5	4.82 ± 0.40	0.27 ± 3.95	0
BS (B)	4	69.3 ± 4.1	11.2 ± 3.7	1.13 ± 0.01	100	32.0 ± 3.3	8.4 ± 0.1	0.39 ± 0.29	100
RSD++	4	75.8 ± 4.2	10.1 ± 0.2	1.14 ± 0.52	32.3 ± 0.6	35.7 ± 4.1	11.2 ± 0.6	0.55 ± 2.15	72.7 ± 1.1
BS (B)+SD	4	67.2 ± 4.1	14.30 ± 3.7	1.85 ± 0.01	100	**36.1 ± 3.3**	16.07 ± 0.1	1.96 ± 0.29	100
GSI	4	**77.3 ± 5.7**	12.5 ± 0.9	1.65 ± 0.26	59.1 ± 3.2	35.4 ± 3.9	13.7 ± 1.0	0.59 ± 1.74	89.9 ± 1.1
SPECS	4	77.0 ± 1	9.83 ± 0.5	1.11 ± 0.14	49.0 ± 0.8	35.3 ± 3.9	10.8 ± 0.8	0.40 ± 1.17	91.1 ± 0.9
BS (S)	8	66.7 ± 2.1	10.7 ± 0.6	1.01 ± 0.03	0	28.3 ± 1.7	9.55 ± 0.40	0.45 ± 0.68	0
BS (B)	8	71.3 ± 0.5	13.6 ± 1.9	1.51 ± 0.00	100	33.0 ± 2.2	12.1 ± 0.1	0.52 ± 0.3	100
RSD++	8	78.3 ± 0.9	11.8 ± 0.3	1.37 ± 0.83	28.4 ± 0.2	39.3 ± 2.1	15.2 ± 0.5	0.71 ± 2.02	73.9 ± 1.2
BS (B)+SD	8	71.0 ± 0.5	17.99 ± 1.9	2.31 ± 0.00	100	**42.0 ± 2.2**	20.56 ± 0.1	2.38 ± 0.3	100
GSI	8	78.9 ± 2.1	13.1 ± 3.4	1.54 ± 0.42	52.2 ± 1.4	40.4 ± 2.6	14.1 ± 1.5	0.94 ± 0.24	97.8 ± 1.8
SPECS	8	**79.5 ± 0.5**	11.3 ± 0.6	1.31 ± 0.56	45.5 ± 0.3	40.7 ± 2.6	15.2 ± 0.74	0.57 ± 0.09	85.2 ± 1.6
BS (S)	16	74.5 ± 0.5	16.8 ± 1.9	1.57 ± 0.10	0	34.3 ± 0.6	17.42 ± 0.69	0.78 ± 2.42	0
BS (B)	16	75.5 ± 0.5	21.3 ± 0.5	2.15 ± 0.02	100	39.0 ± 2.2	18.9 ± 0.46	0.79 ± 0.21	100
RSD++	16	79.0 ± 2.4	16.3 ± 0.4	1.84 ± 0.64	26.8 ± 0.3	40.0 ± 2.2	23.0 ± 1.1	1.02 ± 1.98	70.3 ± 1.3
BS (B)+SD	16	75.0 ± 0.5	31.77 ± 0.5	4.05 ± 0.02	100	39.2 ± 2.2	29.48 ± 0.46	3.34 ± 0.21	100
GSI	16	79.2 ± 2.1	20.3 ± 3.4	2.61 ± 0.50	44.5 ± 0.7	43.0 ± 0.8	27.8 ± 12.1	3.03 ± 0.67	76.4 ± 1.1
SPECS	16	**81.0 ± 1.9**	15.8 ± 0.06	1.31 ± 0.04	45.5 ± 0.8	**44.0 ± 0.8**	22.9 ± 6.3	0.86 ± 0.19	78.1 ± 1.1
OlympiadBench									
BS (S)	4	25.0 ± 1.6	7.9 ± 0.2	0.81 ± 0.2	0	8.0 ± 1.4	5.58 ± 0.12	0.35 ± 0.26	0
BS (B)	4	39.3 ± 2.1	13.1 ± 0.3	1.26 ± 0.3	100	17.0 ± 1.2	13.0 ± 0.05	0.59 ± 0.18	100
RSD++	4	33.0 ± 2.9	17.3 ± 0.4	1.60 ± 0.70	66.3 ± 0.8	7.3 ± 1.7	13.1 ± 0.4	0.61 ± 0.69	95.2 ± 0.9
BS (B)+SD	4	39.4 ± 2.1	12.9 ± 0.3	2.91 ± 0.3	100	16.8 ± 1.2	18.84 ± 0.05	2.56 ± 0.18	100
GSI	4	30.7 ± 0.6	15.4 ± 2.1	2.91 ± 0.24	60.0 ± 1.4	13.4 ± 2.1	17.5 ± 3.3	1.39 ± 0.07	70.8 ± 0.1
SPECS	4	**41.3 ± 0.9**	13.1 ± 0.3	1.30 ± 3.4	83.1 ± 0.4	**18.0 ± 2.8**	14.9 ± 2.8	0.62 ± 0.03	98.1 ± 0.2
BS (S)	8	31.0 ± 0.8	13.4 ± 0.3	1.22 ± 0.3	0	8.0 ± 3.3	11.09 ± 0.49	0.59 ± 0.44	0
BS (B)	8	41.3 ± 2.1	17.9 ± 0.5	1.81 ± 0.5	100	18.3 ± 0.48	18.9 ± 0.1	0.83 ± 0.14	100
RSD++	8	33.3 ± 1.7	21.7 ± 1.3	1.96 ± 1.78	65.3 ± 1.2	8.0 ± 1.4	18.9 ± 0.7	0.83 ± 0.56	92.9 ± 1.1
BS (B)+SD	8	40.1 ± 2.1	17.5 ± 0.5	1.80 ± 0.5	100	18.2 ± 0.48	26.64 ± 0.1	3.42 ± 0.14	100
GSI	8	35.3 ± 1.5	17.9 ± 5.5	2.01 ± 0.54	55.3 ± 0.7	15.7 ± 0.4	20.9 ± 0.2	0.87 ± 0.74	65.2 ± 0.1
SPECS	8	**41.7 ± 1.2**	16.9 ± 0.5	1.80 ± 1.5	80.3 ± 0.4	**18.5 ± 0.5**	19.7 ± 0.14	0.78 ± 0.66	94.6 ± 0.1
BS (S)	16	31.3 ± 3.3	21.1 ± 1.1	2.02 ± 1.1	0	11.3 ± 1.2	23.17 ± 0.73	1.06 ± 1.23	0
BS (B)	16	43.7 ± 0.5	25.4 ± 0.9	2.57 ± 0.9	100	22.0 ± 1.2	30.1 ± 0.8	1.20 ± 5.3	100
RSD++	16	33.5 ± 0.5	28.7 ± 1.3	2.58 ± 13.29	61.2 ± 1.3	9.5 ± 0.5	25.8 ± 0.5	1.18 ± 2.71	93.2 ± 0.8
BS (B)+SD	16	43.8 ± 0.5	27.9 ± 0.9	2.69 ± 0.9	100	**22.4 ± 1.2**	42.81 ± 0.8	4.93 ± 5.3	100
GSI	16	38.7 ± 1.5	36.4 ± 2.4	3.47 ± 0.26	49.7 ± 1.7	17.6 ± 5.0	46.9 ± 3.6	1.64 ± 0.01	73.6 ± 1.1
SPECS	16	**44.0 ± 2.9**	24.6 ± 1.0	2.56 ± 6.2	61.5 ± 0.3	20.0 ± 5.7	31.7 ± 2.4	1.21 ± 0.01	91.1 ± 1.3

4.1 Experimental Setup

Datasets: We conduct experiments on a diverse suite of evaluation benchmarks covering mathematical reasoning, multi-disciplinary question answering, and preference alignment. For mathematical reasoning, we use AMC23 [35], MATH500 [36], and OlympiadBench [37]. The AMC23 dataset comprises 40 problems testing algebra, geometry, number theory, and combinatorics; MATH500 evaluates broad mathematical problem-solving; and OlympiadBench contains challenging olympiad-level

problems requiring deep reasoning. Following prior works [6, 38], we use a randomly selected subset of 100 questions from MATH500 and OlympiadBench to expedite evaluation and report average performance across multiple seeds. For multi-disciplinary QA, we use the GPQA benchmark [39], which consists of 448 high-quality multiple-choice questions in biology, physics, and chemistry, written by domain experts. For preference alignment, we use the Anthropic HH-RLHF dataset [40] with 100 prompts on the test splits. HH-RLHF prompts have no ground-truth answer, so we report pairwise win-rate against a single generation from the target model, judged by GPT-5.4 in a position-swapped 2-pass protocol to remove position bias.

Models: We evaluate over two draft-target model pairs, Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct [41], as well as Llama3.2-1B-Instruct and Llama-3.1-8B-Instruct [42]. These models vary significantly in capability, allowing us to test the generality of our approach. For reward signal on the math and GPQA benchmarks, we use the Skywork-o1-Open-PRM-Qwen-2.5-7B PRM [43], which is specifically trained for mathematical reasoning. For HH-RLHF, we use the Llama-3 draft-target pair and replace the PRM with the Bradley–Terry preference reward model Skywork-Reward-V2-Llama-3.1-8B, which scores complete responses rather than reasoning steps. It is worth pointing out that the generator (draft/target) models have not been specifically finetuned for math reasoning tasks.

Baselines: We compare against six baselines. **BS draft / BS target** are standard PRM-guided beam search using the small (BS (`S`)) and target (BS (`B`)) models respectively as the generator [5, 9]. **RSD++** is a multi-beam strengthening of Reward-Guided Speculative Decoding [19]: we select the highest-scoring candidate of n draft beams, and if its reward falls below the recommended threshold 0.7 we instead select the best of n beams generated by the target model. **GSI** is the concurrent work of Geuter et al. [20]: a PRM-guided decoder that draws steps from the small model, samples from the reward-tilted distribution, and resamples from the target when the tilted reward falls below a threshold; we use GSI's released `reward-tilted` configuration with β=20. **BS (`B`)+SD** couples beam search with classical token-level speculative decoding [12, 13]: the same recipe as BS (`B`), but each beam is generated by the target served via vLLM token-level SD with the small model as draft. We report results with draft length γ=3 in Table 1 as the Pareto-best horizon; results for the full $\gamma \in \{1, 3, 5\}$ sweep are in Section B.5. We note that for all algorithms that involve a smaller speculative model, the model is colocated with either the target model or the reward model, and hence adding no additional hardware overhead. **Best-of-N** samples N complete responses from the target model in parallel under vLLM, scores each complete response with the PRM, and returns the highest-scoring one. We provide its analysis in Table 6.

Evaluation Metrics: We report accuracy (percentage of correctly solved problems) and average wall-clock latency per problem and per reasoning step for different methods with the beam size $n = 4, 8, 16$. Our primary focus is to optimize the pareto frontier of the latency-accuracy trade-off.

4.2 Main Results: Latency–Accuracy Pareto Frontier

Headline result on mathematical reasoning. Table 1 reports our primary comparison across AMC23, MATH500, and OlympiadBench using both the Qwen2.5 (1.5B/7B) and Llama-3 (1B/8B) draft/target families. Our method matches (within standard deviation) or exceeds the accuracy of other baselines in all comparisons. With the Qwen pair it also has lower latency than the competitive baselines. To demonstrate the superior latency-accuracy tradeoff, in Figure 3 we plot the tradeoff curves for each method with both families. For the Qwen family, as shown in the figure, `SPECS` Pareto-dominates target-model beam search (BS (`B`)) on all datasets, matching or exceeding BS (`B`) accuracy at lower latency with reductions of up to 26% (MATH500 n=16); `SPECS` is also at the Pareto frontier in terms of accuracy-latency tradeoff compared to other baseline methods including RSD++, BS (`B`)+SD, and GSI on all datasets.

With the Llama pair, the thresholds of Table 15 favor accuracy. `SPECS` keeps the target model for 78–98% of steps and is more accurate than BS (`B`) in seven of nine settings, at higher latency. The switching threshold τ moves `SPECS` along the accuracy–latency curve. On MATH500 with n=8, τ=0.7 matches the accuracy of BS (`B`) (32.9% vs. 33.0%) at 16% lower latency (10.2 vs. 12.1 s, Table 9).

`SPECS` with n=4 is more accurate and faster than Best-of-N with N=16 on MATH500 and OlympiadBench (Table 6). On MATH500, Best-of-N reaches 72.1% only at N=1024 and 284.3 s (Table 7). Under 1 to 16 concurrent requests, `SPECS` keeps 9–17% lower latency per request than BS (`B`) (Table 8).

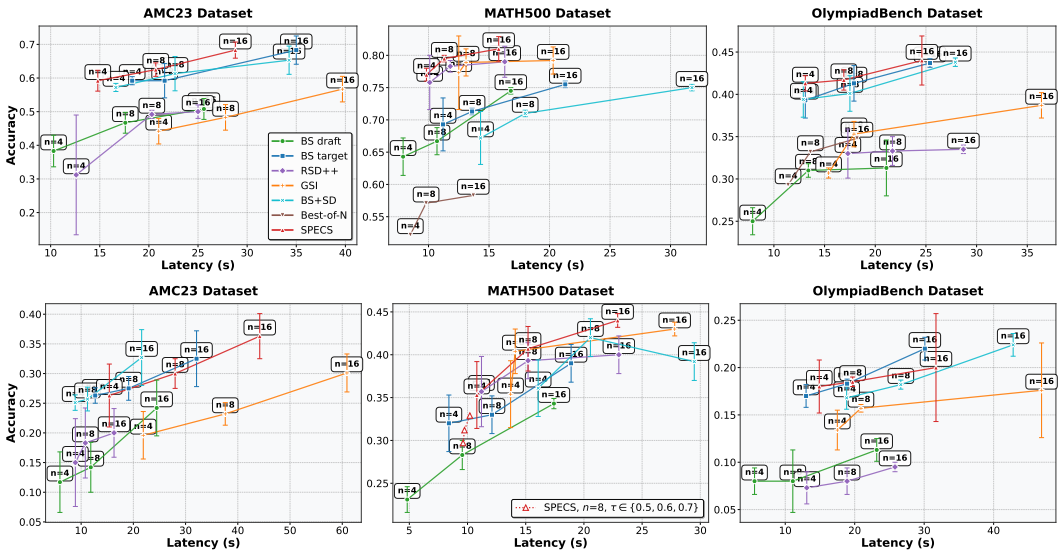


Figure 3: Accuracy vs per-query average latency curves for SPECS and beam search with draft model, target model, RSD++, BS+SD, and GSI, and for Best-of-N. The error bars show the standard deviation computed over 3 independent runs. Top row: Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct. Bottom row: Llama-3.2-1B-Instruct and Llama-3.1-8B-Instruct; the legend of the top row applies. Hollow markers on Llama MATH500 are SPECS with n=8 and $\tau \in \{0.5, 0.6, 0.7\}$.

Table 2: **GPQA accuracy and latency.** Per-step beam width n; Qwen2.5-1.5B/7B-Instruct; Skywork PRM.

n	SPECS		BS (Big)		BS (Small)	
	Acc ↑	Lat ↓	Acc ↑	Lat ↓	Acc ↑	Lat ↓
4	30.8	9.00	34.2	18.6	17.8	7.77
8	32.2	21.83	32.5	23.1	22.6	10.31
16	34.5	22.42	33.3	28.3	24.2	15.08

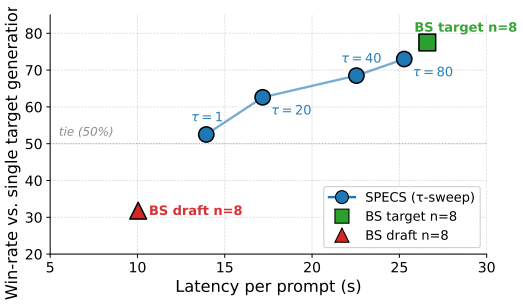


Figure 4: **HH-RLHF helpful-base: SPECS traces a tunable latency–win-rate curve.** Win-rate is pairwise vs. a single target-model generation, judged by GPT-5.4 in a position-swapped 2-pass protocol. Results are averaged across 3 seeds; SPECS uses n=8 over a τ-sweep.

Dynamic switching adapts to problem difficulty. The fraction of steps generated by the large model (**% B**) rises with dataset difficulty (AMC23, OlympiadBench > MATH500), indicating that the switching rule (Algorithm 1, Line 9) correctly allocates target-model compute to harder steps.

Generalization beyond mathematical reasoning The favorable accuracy-latency tradeoff of SPECS generalizes beyond mathematical reasoning. On GPQA (Table 2), SPECS reaches accuracy on par with target-model beam search at substantially lower latency (e.g., 34.5% acc at 22.4 s vs. 33.3% at 28.3 s for n=16). HH-RLHF stresses a fundamentally different reward regime: the scoring model is a Bradley–Terry preference reward (Skywork-Reward-V2-Llama-3.1-8B) that scores complete responses, not a process reward over reasoning steps, demonstrating that SPECS generalizes across reward-model types. On helpful-base, the τ-sweep (Figure 4) traces a smooth tunable curve between BS (S) and BS (B) n=8.

4.3 Component Ablations and Findings

We further conduct component-wise ablations to understand the contribution of each design choice in SPECS, including reward-guided soft verification and the dynamic switching mechanism.

To separate the sources of the gains, an all-target variant keeps SUBSAMPLE but generates every block with the target model (MATH500, Qwen, Table 5). It matches SPECS within 0.8 points, both beat BS (B) by 5.5 to 8.5 points, and SPECS is 11–19% faster. Soft reward-guided selection gives the accuracy gain over beam search, and speculative drafting gives the latency reduction. We elaborate on the effect of draft selection score, dynamic switching and robustness to noisy critics below.

Effect of draft selection score in eq. (1). We perform several ablation studies to study the effect of the draft selection score. We implement a SPECS variant, which ignores the log-likelihood term in the score function (eq. (1)), and uses only the raw critic reward to select from the draft steps. This variant achieves worse accuracy than SPECS, *e.g.,* with $n = 4$ on AMC23, the accuracy is 43.2% compared to SPECS's 59.2% (see Table 4 in Section B). This shows that utilizing the log-likelihood term in the score function is crucial for the performance of SPECS, as it provides a more informative signal for selecting high-quality draft steps. See details of these ablation studies in Section B.1.

Robustness to critic noise. A natural concern for any reward-guided method is whether the gains depend on a perfect PRM. We test this by injecting Gaussian noise into the critic ($\tilde{r}_{\texttt{PRM}} = r_{\texttt{PRM}} + Z$, $Z \sim \mathcal{N}(0, 0.01)$) without retuning any hyperparameters. SPECS retains its accuracy advantage over both BS(Small) and BS(Large) under the noisy reward across AMC23, MATH500, and OlympiadBench at every $n \in \{4, 8, 16\}$, indicating that SPECS tolerates additive critic noise better than beam search based on pure reward; full results are in Section B.11 and table 16.

Effect of dynamic switching. We consider a variant of SPECS where we switch randomly between the draft and target models for generation, such that the average proportion of steps the target model generates matches the target model usage of SPECS. This results in a method with similar average per-step latencies, but with much lower accuracy (with $n = 4$ on AMC23, the accuracy is 42.2% compared to SPECS's 59.2%). We also run a draft-first variant of SPECS that starts from the draft model and switches to the target model (SPECS ($\pi_{\texttt{draft}}$ start) in Table 4). The method results in similar accuracy to SPECS, but with much higher latency. This demonstrates the effectiveness of our dynamic switching scheme. We defer to Section B.1 for a more detailed discussion of these ablations.

5 Conclusion and Limitations

In this work, we address the critical challenge of balancing latency and accuracy in test-time scaling for large language models. We propose SPECS, a latency-aware test-time scaling method for large language models (LLMs) that significantly improves the latency-accuracy tradeoff through speculative drafting with a fast draft model. We propose novel integration strategies, including reward-guided soft verification and a reward-based switching mechanism, to coordinate among the draft model, target model, and the critic, which dynamically balance computational resources with response latency. Empirically, we evaluate SPECS across math reasoning, question answering, and alignment datasets. We show that our algorithm consistently achieves accuracy comparable to or exceeding other test-time scaling methods while delivering substantial reductions in latency (of up to approximately 26%). Our theoretical analysis confirms that SPECS converges to an optimum of the KL-regularized reinforcement learning objective as the beam width increases, up to an error term depending on the misspecification of the critic reward.

Limitations. Due to resource constraints, our evaluation focuses on smaller open-sourced model which could be served on a small GPU topology. Evaluating the effectiveness of our method on larger models is an interesting future direction. Our empirical claims hold for the evaluated Qwen2.5 and Llama-3 pairs, A100 GPUs, and vLLM setup. We measure user-facing latency, not total FLOPs or serving cost. All results use one critic per task, and the switching rule depends on it.

Acknowledgments and Disclosure of Funding

Part of this work was funded by NSF fund CIF-2507523 and ARO fund 062982.

References

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

[2] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. Advances in neural information processing systems, 35:24824–24837, 2022.

[3] Ahmad Beirami, Alekh Agarwal, Jonathan Berant, Alexander D'Amour, Jacob Eisenstein, Chirag Nagpal, and Ananda Theertha Suresh. Theoretical guarantees on the best-of-n alignment policy. arXiv preprint arXiv:2401.01879, 2024.

[4] Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling, 2024. URL https://arxiv.org/abs/2407.21787.

[5] Edward Beeching, Lewis Tunstall, and Sasha Rush. Scaling test-time compute with open models, December 2024. URL https://huggingface.co/spaces/HuggingFaceH4/blogpost-scaling-test-time-compute. Accessed: 2025-04-10.

[6] Jiahao Qiu, Yifu Lu, Yifan Zeng, Jiacheng Guo, Jiayi Geng, Huazheng Wang, Kaixuan Huang, Yue Wu, and Mengdi Wang. Treebon: Enhancing inference-time alignment with speculative tree-search and best-of-n sampling. arXiv preprint arXiv:2410.16033, 2024.

[7] OpenAI. Learning to reason with llms. 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

[8] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

[9] Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In The Thirteenth International Conference on Learning Representations, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

[10] Pratyush Patel, Esha Choukse, Chaojie Zhang, Aashaka Shah, Íñigo Goiri, Saeed Maleki, and Ricardo Bianchini. Splitwise: Efficient generative llm inference using phase splitting. In 2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA), pages 118–132. IEEE, 2024.

[11] Amey Agrawal, Nitin Kedia, Ashish Panwar, Jayashree Mohan, Nipun Kwatra, Bhargav Gulavani, Alexey Tumanov, and Ramachandran Ramjee. Taming Throughput-Latency tradeoff in LLM inference with Sarathi-Serve. In 18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24), pages 117–134, 2024.

[12] Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In International Conference on Machine Learning, pages 19274–19286. PMLR, 2022.

[13] Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling, 2023. URL https://arxiv.org/abs/2302.01318.

[14] Sidharth Mudgal, Jong Lee, Harish Ganapathy, YaGuang Li, Tao Wang, Yanping Huang, Zhifeng Chen, Heng-Tze Cheng, Michael Collins, Trevor Strohman, Jilin Chen, Alex Beutel, and Ahmad Beirami. Controlled decoding from language models. International Conference on Machine Learning, 2024.

[15] Hanshi Sun, Momin Haider, Ruiqi Zhang, Huitao Yang, Jiahao Qiu, Ming Yin, Mengdi Wang, Peter Bartlett, and Andrea Zanette. Fast best-of-n decoding via speculative rejection, 2024. URL https://arxiv.org/abs/2410.20290.

[16] Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. arXiv preprint arXiv:2501.07301, 2025.

[17] Yichao Fu, Xuewei Wang, Hao Zhang, Yuandong Tian, and Jiawei Zhao. Deep think with confidence. In The Fourteenth International Conference on Learning Representations, 2026. URL https://openreview.net/forum?id=8LqHs0KIM7.

[18] Yiming Wang, Pei Zhang, Siyuan Huang, Baosong Yang, Zhuosheng Zhang, Fei Huang, and Rui Wang. Sampling-efficient test-time scaling: Self-estimating the best-of-n sampling in early decoding. In The Thirty-ninth Annual Conference on Neural Information Processing Systems, 2026. URL https://openreview.net/forum?id=BcKYVmh3yH.

[19] Baohao Liao, Yuhui Xu, Hanze Dong, Junnan Li, Christof Monz, Silvio Savarese, Doyen Sahoo, and Caiming Xiong. Reward-guided speculative decoding for efficient llm reasoning, 2025. URL https://arxiv.org/abs/2501.19324.

[20] Jonathan Geuter, Youssef Mroueh, and David Alvarez-Melis. Guided speculative inference for efficient test-time alignment of llms, 2025. URL https://arxiv.org/abs/2506.04118.

[21] Ziteng Sun, Ananda Theertha Suresh, Jae Hun Ro, Ahmad Beirami, Himanshu Jain, and Felix Yu. Spectr: Fast speculative decoding via optimal transport. Advances in Neural Information Processing Systems, 36:30222–30242, 2023.

[22] Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, Chunan Shi, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. Specinfer: Accelerating large language model serving with tree-based speculative inference and verification. ASPLOS '24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400703867. doi: 10.1145/3620666.3651335. URL https://doi.org/10.1145/3620666.3651335.

[23] David Dohan, Winnie Xu, Aitor Lewkowycz, Jacob Austin, David Bieber, Raphael Gontijo Lopes, Yuhuai Wu, Henryk Michalewski, Rif A. Saurous, Jascha Sohl-dickstein, Kevin Murphy, and Charles Sutton. Language model cascades, 2022. URL https://arxiv.org/abs/2207.10342.

[24] Wittawat Jitkrittum, Neha Gupta, Aditya K Menon, Harikrishna Narasimhan, Ankit Rawat, and Sanjiv Kumar. When does confidence-based cascade deferral suffice? Advances in Neural Information Processing Systems, 36:9891–9906, 2023.

[25] Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. Large language model cascades with mixture of thoughts representations for cost-efficient reasoning, 2024. URL https://arxiv.org/abs/2310.03094.

[26] Harikrishna Narasimhan, Wittawat Jitkrittum, Ankit Singh Rawat, Seungyeon Kim, Neha Gupta, Aditya Krishna Menon, and Sanjiv Kumar. Faster cascades via speculative decoding. arXiv preprint arXiv:2405.19261, 2024.

[27] Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. arXiv preprint arXiv:1907.00456, 2019.

[28] Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. Advances in Neural Information Processing Systems, 33:3008–3021, 2020.

[29] Tomasz Korbak, Ethan Perez, and Christopher Buckley. RL with KL penalties is better viewed as Bayesian inference. In Findings of the Association for Computational Linguistics: EMNLP 2022, pages 1083–1091, 2022.

[30] Tomasz Korbak, Hady Elsahar, Germán Kruszewski, and Marc Dymetman. On reinforcement learning and distribution matching for fine-tuning language models with no catastrophic forgetting. Advances in Neural Information Processing Systems, 35:16203–16220, 2022.

[31] Joy Qiping Yang, Salman Salamatian, Ziteng Sun, Ananda Theertha Suresh, and Ahmad Beirami. Asymptotics of language model alignment. arXiv preprint arXiv:2404.01730, 2024.

[32] Donald B Rubin. The calculation of posterior distributions by data augmentation: Comment: A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: The sir algorithm. Journal of the American Statistical Association, 82(398):543–546, 1987.

[33] Christian A Naesseth, Fredrik Lindsten, Thomas B Schön, et al. Elements of sequential monte carlo. Foundations and Trends® in Machine Learning, 12(3):307–392, 2019.

[34] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In International conference on machine learning, pages 1352–1361. PMLR, 2017.

[35] J Douglas Faires and David Wells. The Contest Problem Book VIII: American Mathematics Competitions (AMC 10) 2000–2007, volume 19. American Mathematical Society, 2022.

[36] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. arXiv preprint arXiv:2103.03874, 2021.

[37] Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. arXiv preprint arXiv:2402.14008, 2024.

[38] Ruiqi Zhang, Momin Haider, Ming Yin, Jiahao Qiu, Mengdi Wang, Peter Bartlett, and Andrea Zanette. Accelerating best-of-n via speculative rejection. In ICML 2024 Workshop on Structured Probabilistic Inference {\&} Generative Modeling, 2024.

[39] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In First Conference on Language Modeling, 2024.

[40] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. arXiv preprint arXiv:2204.05862, 2022.

[41] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. arXiv preprint arXiv:2412.15115, 2024.

[42] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. arXiv e-prints, pages arXiv–2407, 2024.

[43] Jujie He, Tianwen Wei, Rui Yan, Jiacai Liu, Chaojie Wang, Yimeng Gan, Shiwen Tu, Chris Yuhao Liu, Liang Zeng, Xiaokun Wang, Boyang Wang, Yongcong Li, Fuxiang Zhang, Jiacheng Xu, Bo An, Yang Liu, and Yahui Zhou. Skywork-o1 open series, November 2024. URL https://doi.org/10.5281/zenodo.16998085.

[44] Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback, 2022. URL https://arxiv.org/abs/2211.14275.

[45] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step, 2023. URL https://arxiv.org/abs/2305.20050.

[46] Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations, 2024. URL https://arxiv.org/abs/2312.08935.

[47] Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Processbench: Identifying process errors in mathematical reasoning, 2024. URL https://arxiv.org/abs/2412.06559.

[48] Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning, 2025. URL https://arxiv.org/abs/2501.07301.

[49] Rui Pan, Yinwei Dai, Zhihao Zhang, Gabriele Oliaro, Zhihao Jia, and Ravi Netravali. Specreason: Fast and accurate inference-time compute via speculative reasoning, 2025. URL https://arxiv.org/abs/2504.07891.

[50] Zhihai Wang, Jie Wang, Jilai Pan, Xilin Xia, Huiling Zhen, Mingxuan Yuan, Jianye Hao, and Feng Wu. Accelerating large language model reasoning via speculative search, 2025. URL https://arxiv.org/abs/2505.02865.

[51] 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.

[52] Jikai Wang, Yi Su, Juntao Li, Qingrong Xia, Zi Ye, Xinyu Duan, Zhefeng Wang, and Min Zhang. Opt-tree: Speculative decoding with adaptive draft tree structure. Transactions of the Association for Computational Linguistics, 13:188–199, 2025.

[53] Zhuoming Chen, Avner May, Ruslan Svirschevski, Yuhsun Huang, Max Ryabinin, Zhihao Jia, and Beidi Chen. Sequoia: Scalable, robust, and hardware-aware speculative decoding. arXiv preprint arXiv:2402.12374, 2024.

[54] Ziyao Wang, Muneeza Azmat, Ang Li, Raya Horesh, and Mikhail Yurochkin. Speculate, then collaborate: Fusing knowledge of language models during decoding, 2025. URL https://arxiv.org/abs/2502.08020.

[55] Yichao Fu, Rui Ge, Zelei Shao, Zhijie Deng, and Hao Zhang. Scaling speculative decoding with lookahead reasoning, 2025. URL https://arxiv.org/abs/2506.19830.

[56] Claudio Mayrink Verdun, Alex Oesterling, Himabindu Lakkaraju, and Flavio P. Calmon. Soft best-of-n sampling for model alignment, 2025. URL https://arxiv.org/abs/2505.03156.

[57] Audrey Huang, Adam Block, Qinghua Liu, Nan Jiang, Akshay Krishnamurthy, and Dylan J. Foster. Is best-of-n the best of them? coverage, scaling, and optimality in inference-time alignment, 2025. URL https://arxiv.org/abs/2503.21878.

[58] Dhruv Rohatgi, Abhishek Shetty, Donya Saless, Yuchen Li, Ankur Moitra, Andrej Risteski, and Dylan J. Foster. Taming imperfect process verifiers: A sampling perspective on backtracking, 2025. URL https://arxiv.org/abs/2510.03149.

[59] Isha Puri, Shivchander Sudalairaj, Guangxuan Xu, Kai Xu, and Akash Srivastava. Rollout roulette: A probabilistic inference approach to inference-time scaling of LLMs using particle-based Monte Carlo methods, 2025. URL https://arxiv.org/abs/2502.01618.

[60] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In Proceedings of the 29th Symposium on Operating Systems Principles, pages 611–626, 2023.

Appendix Contents

A Related work

Reward-guided search. Our algorithm is related to the general literature of process reward modeling [5, 9, 44–48] and reward-guided search [5, 6, 9, 14, 15]. Among these, the closest to ours is reward-guided speculative decoding [19], which also uses a draft model to propose each reasoning step and uses a process reward model to decide on whether to keep the proposal. Closely related are small/large collaboration approaches for reasoning: Pan et al. [49] (SpecReason) drafts steps with a small reasoning model and uses a large model as an LRM-as-judge verifier, and Wang et al. [50] (SpecSearch) integrates a small/large hand-off into tree search via a quality-threshold rejection rule. Compared to these, SPECS uses a process reward model with a soft, KL-regularized aggregation of large-model and reward signals over $n > 1$ candidates rather than discrete LRM judgments or threshold rejection, and admits a formal analysis (Section 3). Empirically, we observe SPECS achieves a better latency-accuracy tradeoff compared to reward-guided speculative decoding.

We would like to mention that existing work that utilizes speculative drafts to reduce latency all start with generations from a speculative model, and switches to a larger when the reward signal is low. While our algorithm starts with the large model and switches to speculative drafts only when the reward signal is high. This approach reduces the time wasted for speculative drafting for low-reward traces, resulting in overall better latency-accuracy tradeoff. This builds on the novel insight that the accuracy gap between a small model and a large model is small for these traces, which could be of independent interest.

Tree-based speculative decoding. Obtaining one sequence from multiple draft sequences has been studied in the speculative decoding literature with the goal of maximizing the likelihood/length of draft tokens accepted while maintaining the same distribution as the large model [21, 22, 51–53]. Compared to these, the goal of our soft verification algorithm is to maximize the success rate of downstream applications such as math-reasoning as guided by both the target model and the reward model. Other soft verification methods for speculative decoding [54] consider token-level verification while we focus on step-level candidate selection. The concurrent work of Fu et al. [55] (Lookahead Reasoning) also performs step-level speculative decoding for reasoning models, but targets distribution-preserving greedy-decoding speedup for a single model, whereas SPECS uses a separate small/large pair under reward guidance to improve the test-time scaling tradeoff.

Formal analysis of best-of-N variants. The theoretical analysis of our soft verification algorithm is related to Verdun et al. [56], Huang et al. [57], Rohatgi et al. [58], which consider alternative candidate selection methods other than Top-1 selection, and to the particle-filtering view of inference-time scaling [59], which gives a parallel soft, particle-based alternative to beam search; our analysis captures the convergence rate in presence of draft models and multiple decoding steps.

Cascading. Cascading techniques [23–26] also utilize multiple language models and decide on the model to used based on certain deferral rule. The switching step in our algorithm can be viewed as imposing a deferral rule based on small, target, and reward models. Additionally, our deferral rule operates at the step-level while previous work usually focuses on individual tokens or the entire generation.

Comparison with concurrent work [20]. The concurrent work of [20] proposes Guided Speculative Inference (GSI), which also considers soft draft verification by sampling from an exponentially tilted distribution (i.e., soft best-of-n). Despite being concurrent, our work provides the following additional novelties compared to [20]: (1) We propose a novel dynamic switching scheme to improve latency-accuracy tradeoff. Similar to prior work, [20] starts by generating steps with the draft model, while SPECS starts with the target model and switches to the draft model only for high-reward/easier steps, this has resulted in favorable gains as demonstrated in our experimental section. (2) Our paper also makes stronger theoretical contributions. In particular, we provide convergence guarantees on the multi-step version of our algorithm, as well as analysis on cases with imperfect reward signals.

While in the main paper we provide a comparison to GSI using Qwen base models, here we also provide a comparison to GSI using the math-instruct models which is what the paper [20] used for

GSI originally. Therefore in Table 3, we include the accuracy of SPECS in the experiment setting of GSI (using Qwen math-instruct models to solve MATH500 and OlympiadBench with $n = 4, 16$) along with the reported numbers for GSI. We can see that SPECS achieves better accuracy than GSI[3].

Table 3: Comparison between SPECS (with Qwen2.5-Math-Instruct models) and GSI [20]

Dataset	**Beam Width** (n)	**SPECS** (Acc)	**GSI** (Acc)
MATH500	4	**82.4**%	81.2%
MATH500	16	**85.7**%	82.2%
OlympiadBench	4	**41.7**%	39.7%
OlympiadBench	16	**44.2**%	40.8%

B Additional Experiments for SPECS

This appendix collects all supplementary experiments. We begin with the component-wise ablation (Section B.1) – the analysis that establishes which design choices in SPECS contribute the gains observed in Table 1. We analyze how SPECS's performance changes with different hyperparameters in Section B.4 and show that SPECS's performance requires minimal hyperparameter tuning to achieve good performance. We then present the detailed comparison to speculative decoding in Section B.5. We further report a per-component latency breakdown (Section B.6), document hardware/framework details and hyperparameters (Sections B.7 and B.9), examine dynamic-switching behaviour qualitatively (Section B.10), and finally evaluate SPECS's robustness to noisy reward signals (Section B.11).

B.1 Analysis and Ablations

We analyse the importance of different components of SPECS and its robustness to hyperparameter choices. The two ablation axes are the design of the soft draft-selection score (Equation (1)) and the dynamic switching scheme; Table 4 reports both. Hyperparameter sensitivity (the choices of β and τ) is reported in Table 13.

Effect of log-likelihood based draft selection score. First, we compare SPECS with a version of the algorithm where we remove dynamic switching, and generate all steps from the draft model with soft verification (in the sense of eq. (1)). Equivalently, we fix $\pi_{\texttt{gen}} = \pi_{\texttt{draft}}$ throughout decoding. We observe much higher accuracy compared to beam search with the small model (BS(S)) while the latency is not much worse (shown in Table 4). This demonstrates the benefit of our score design in eq. (1). The results of this ablation are in Table 4, under "Only Small Gen". We also consider a variant of SPECS where we start with the small model first, switching to the larger model if none of the generations of the small model exceed the threshold τ. We show that this variant outperforms RSD++ in Table 1, showing the utility of our scoring mechanism over hard best-of-n sampling. This is indicated with "SPECS ($\pi_{\texttt{draft}}$ start)" in Table 4. We implement one final variant, "SPECS (no LL)", which uses no log-likelihood term in the score function (eq. (1)). This variant performs worse than SPECS, showing the utility of the log-likelihood ratio in the score.

Effect of dynamic switching. We consider a variant of SPECS where we switch randomly between the draft and target model for generation, in such a way that the average proportion of steps the target model generates matches with the target model usage of SPECS. This results in a method with similar average per-step latencies. We observe that random switching achieves much lower accuracy than dynamic switching based on the reward signal, demonstrating the advantage of our dynamic switching scheme. The results of this ablation are in Table 4, under "Random Switch". Moreover, the fact that SPECS outperforms SPECS ($\pi_{\texttt{draft}}$ start) also shows the advantage of starting by generating from the large model.

τ sweeps the full Pareto frontier. Varying the switching threshold τ (Line 9, Algorithm 1) trades latency for accuracy. The switching rule fires when $\max_i r_{\texttt{critic}}(Y_i|\texttt{tr}) > \tau$, so small τ is easy to satisfy and the algorithm switches to the draft eagerly (lower latency, lower accuracy), whereas large

[3]Table 3 does not report latency numbers, since the work of [20] only reports the per-step accuracy of GSI averaged across a set of 5 benchmarks which do not overlap with the benchmarks we evaluate on.

Table 4: **Each SPECS component contributes; ablating any one degrades the latency–accuracy tradeoff.** Ablation study across AMC23, MATH500, and OlympiadBench using Qwen2.5-Instruct models. 'SPECS' is the main method; 'Only Small Gen' generates all beams from the small model with our soft-verification score; 'Random Switch' switches randomly between small and big model with the same target-usage probability as SPECS; 'SPECS ($\pi_{\texttt{draft}}$ start)' starts from the small model and switches to the target; 'SPECS (no LL)' removes the log-likelihood term from the score.

Method	n	**AMC23**		**MATH500**		**OlympiadBench**	
		Acc (%) ↑	**Lat** (s)	**Acc** (%) ↑	**Lat** (s) ↓	**Acc** (%) ↑	**Lat** (s)
SPECS	4	59.2	14.80	77.0	9.83	41.3	13.10
Only Small Gen	4	51.2	13.68	65.0	7.45	41.1	13.00
Random Switch	4	42.2	10.86	70.0	8.59	30.1	9.43
SPECS ($\pi_{\texttt{draft}}$ start)	4	58.9	18.5	77.5	13.0	41.0	15.3
SPECS (no LL)	4	43.2	12.5	71.5	8.73	32.5	16.6
SPECS	8	62.5	20.70	79.5	11.3	41.7	16.90
Only Small Gen	8	56.2	19.69	66.0	8.15	39.0	16.24
Random Switch	8	50.6	15.45	70.0	12.13	33.5	13.86
SPECS ($\pi_{\texttt{draft}}$ start)	8	61.4	21.9	78.9	12.7	41.3	18.1
SPECS (no LL)	8	47.5	17.9	73.3	10.6	32.8	23.8
SPECS	16	68.3	28.80	81.0	11.748	44.0	24.60
Only Small Gen	16	63.8	26.72	73.0	11.6	39.5	23.84
Random Switch	16	54.7	23.60	74.0	11.3	38.4	21.83
SPECS ($\pi_{\texttt{draft}}$ start)	16	67.9	29.2	81.0	17.2	44.2	27.8
SPECS (no LL)	16	48.6	28.1	74.5	14.5	33.1	27.1

τ is rarely satisfied and the algorithm keeps the target model longer (higher latency, higher accuracy). Figure 5 shows the resulting curve on MATH500 (Qwen2.5-1.5B/7B): every baseline (RSD++, BS draft, BS target) lies inside the convex hull traced by SPECS. As a sanity check, $\tau \to 1$ saturates the switching rule (since $r_{\texttt{critic}} \leq 1$ for the math PRM), so SPECS collapses to target-model beam search and the corresponding points coincide.

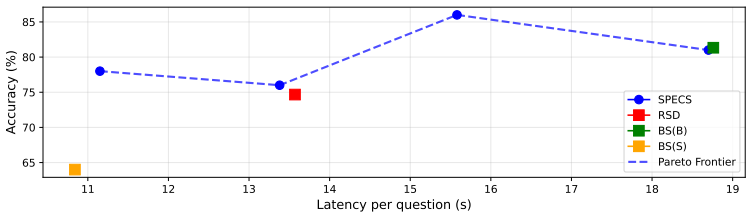


Figure 5: **The τ knob traces the full Pareto frontier on MATH500.** Sweeping τ in SPECS (Qwen2.5-1.5B/7B-Instruct) interpolates between the small- and large-model regimes; RSD++ and beam-search baselines fall strictly inside the convex hull of the SPECS curve. As $\tau \to 1$, SPECS collapses to target-model beam search and the points coincide.

B.2 All-target ablation, Best-of-N, concurrent requests, and the Llama switching threshold

Table 5 gives the all-target ablation of Section 4.3. All-target SPECS uses the SUBSAMPLE rule and the hyperparameters of SPECS but generates every block with the target model.

Best-of-N samples N independent complete responses from the target model, batched under vLLM. The PRM scores each complete response, and we return the highest-scoring one. A deterministic math-equivalence checker grades the returned answer and takes no part in the selection. Table 6 compares Best-of-N with SPECS and target-model beam search at the same N on MATH500 and OlympiadBench with the Qwen pair. Best-of-N is faster than SPECS at the same N but 8.5 to 24.8 points less accurate. Table 7 extends the MATH500 sweep to N=1024. Accuracy stays between 58.3% and 59.4% from N=16 to N=256. It reaches 72.1% at N=1024, which is 4.9 points below

SPECS with n=4 at 29× its latency. We select with the reward model and do not measure oracle pass@N. The plateau may come from reward-model selection errors rather than limited sample coverage.

Table 8 serves 1, 4, 8, and 16 concurrent MATH500 problems with the Qwen pair. We use n=8 for SPECS and target-model beam search, and N=8 for Best-of-N. The models run under vLLM with continuous batching and prefix caching.

Table 9 sweeps the switching threshold τ for the Llama pair (Llama-3.2-1B-Instruct draft, Llama-3.1-8B-Instruct target, Skywork critic) on the 100-problem MATH500 subset, with n=8, β=1000, temperature 0.3, and 256-token blocks. A lower threshold switches to the draft model earlier, and it lowers latency and accuracy together.

Table 10 sweeps the block size ℓ of Algorithm 1, the token budget of one step. We use the Qwen pair on the 100-problem MATH500 subset with n=8, β=1000, τ=0.8, and one seed. These settings differ from Table 1, so only the comparison across block sizes is meaningful. The 256-token blocks of our main experiments give the lowest latency and tie for the highest accuracy. With 128-token blocks, accuracy drops by 7 points. With 512-token blocks, accuracy is unchanged and latency rises by 1.8 s.

Table 5: All-target ablation on MATH500 with the Qwen pair, averaged over 3 seeds. Each cell is accuracy (%) / latency per problem (s).

n	SPECS	All-target SPECS	Target-model beam search
4	77.0 / 9.83	77.8 / 11.0	69.3 / 11.2
8	79.5 / 11.3	79.3 / 13.9	71.3 / 13.6
16	81.0 / 15.8	81.4 / 19.6	75.5 / 21.3

Table 6: Best-of-N against SPECS and target-model beam search with the Qwen pair, at N=n. Each cell is accuracy (%) / latency per problem (s). SPECS and beam search are averaged over 3 runs, as in Table 1.

	MATH500			OlympiadBench		
N	Best-of-N	SPECS	Target BS	Best-of-N	SPECS	Target BS
4	52.2 / 8.5	77.0 / 9.83	69.3 / 11.2	29.3 / 11.4	41.3 / 13.1	39.3 / 13.1
8	57.1 / 9.8	79.5 / 11.3	71.3 / 13.6	33.2 / 13.6	41.7 / 16.9	41.3 / 17.9
16	58.3 / 13.7	81.0 / 15.8	75.5 / 21.3	34.8 / 18.2	44.0 / 24.6	43.7 / 25.4

Table 7: Best-of-N on MATH500 with the Qwen pair (target model Qwen2.5-7B-Instruct). For reference, SPECS with n=4 reaches 77.0% at 9.83 s.

N	4	8	16	32	64	128	192	256	384	512	1024
Acc (%)	52.2	57.1	58.3	58.3	58.9	59.3	59.4	59.4	64.7	66.9	72.1
Lat (s)	8.5	9.8	13.7	16.3	28.3	49.1	56.7	70.8	112.9	140.4	284.3

B.3 Motivating observations across model pairs and datasets

We repeat the experiments behind Observations 1 and 2 (Figure 2) with the Qwen2.5-1.5B/7B-Instruct and Llama-3.2-1B/Llama-3.1-8B-Instruct pairs on MATH500, OlympiadBench, and AMC23.

For Observation 2, the target model generates a partial solution with the beam machinery of SPECS. It uses 256-token blocks delimited by \n\n, temperature 0.3, and selects the candidate with the highest critic score. We save the surviving trace after steps 4, 8, and 12 and score each saved prefix with Skywork-o1-Open-PRM. From each prefix we generate two independent continuations to the final answer, one with the draft model and one with the target model. Both use temperature 0.3 and no critic. A deterministic math-equivalence checker grades the final answers. Table 11 reports prefixes with critic score at least 0.5, the regime in which SPECS switches to the draft model, averaged over the three switch steps. In four of the five settings, the draft model is within 3.3 points of the target

Table 8: Average latency per request (s) under concurrent requests on MATH500 with the Qwen pair. The last column is the change of SPECS relative to target-model beam search.

Requests	Best-of-N	SPECS	Target BS	Change
1	9.6	11.3	13.6	-16.9%
4	10.8	12.9	14.2	-9.2%
8	19.3	24.5	29.2	-16.1%
16	27.8	31.7	36.2	-12.4%

Table 9: SPECS with the Llama pair on MATH500 (n=8) for several switching thresholds τ. τ=0.8 is the setting of Table 1.

Method	Acc (%)	Lat (s)
SPECS, τ=0.5	29.7	9.62
SPECS, τ=0.6	31.2	9.74
SPECS, τ=0.7	32.9	10.2
SPECS, τ=0.8	40.7	15.2
Target BS	33.0	12.1

Table 10: Block size ℓ of SPECS on the 100-problem MATH500 subset (Qwen pair, n=8, β=1000, τ=0.8, one seed).

Block size (tokens)	Acc (%)	Lat (s)
128	45.0	16.6
256	52.0	15.8
512	52.0	17.6

model. The exception is Llama on AMC23, where the draft model is 30.3 points behind. The 1B draft model is weak on AMC23, reaching 24.2% even with draft-model beam search at n=16 (Table 1). In the main Llama AMC23 runs, SPECS generates 93.6–98.1% of steps with the target model.

For Observation 1, we measure the latency of one step on 50 reasoning prefixes per model pair, sampled from the traces above. Each step generates candidate blocks of 256 tokens. We run one measurement at a time after warming the caches of both model servers, and each model runs on one A100 80GB GPU under vLLM. Draft generation alone is about 2 to 3 times faster than target generation. Target verification and critic scoring run concurrently, and their latency on the critical path is the slower of the two rather than their sum.

Table 11: Final-answer accuracy (%) of draft and target completions from target-generated prefixes with critic score at least 0.5, averaged over switch steps 4, 8, and 12.

Pair	Dataset	Draft	Target
Llama	MATH500	48.94	49.13
Llama	AMC23	32.28	62.57
Llama	OlympiadBench	43.14	42.94
Qwen	AMC23	75.93	79.17
Qwen	OlympiadBench	28.37	31.43

Table 12: Mean latency of one step (s) on 50 prefixes per model pair. Target: target-model generation and scoring. Draft: draft generation with parallel target verification and critic scoring.

Pair	Target	Draft
Qwen	1.55	1.25
Llama	1.18	0.84

B.4 Hyperparameter Sensitivity.

We also examine whether SPECS is sensitive to hyperparameter tuning. We find that the method is highly stable: setting $\beta = 2L$ (where L is the token budget) and restricting τ to a coarse set $\{0.7, 0.8, 0.9\}$ recovers nearly all the performance of an exhaustive search. As shown in Table 13, the "Constrained Search" results differ from the optimal search by only marginal amounts, confirming that SPECS is easy to tune and deploy.

B.5 Is the speedup just token-level speculative decoding? Full BS+SD analysis

A natural question is whether SPECS's gains are simply token-level speculative decoding (SD) applied to the target model in disguise. They are not, but the precise comparison depends on the SD horizon γ, so we sweep $\gamma \in \{1, 3, 5\}$. The Pareto-best horizon (most often on the BS+SD frontier) is γ=3, which is the row reproduced in the main results Table 1.

Table 13: SPECS **is robust to hyperparameter choices.** Optimal performance from exhaustive search vs. a constrained search with fixed heuristics ($\beta = 2L$, coarse τ). The performance gap is negligible, so SPECS is easy to tune and deploy. (E) = Exhaustive search, (C) = Constrained search.

Method	n	**AMC23**		**MATH500**		**OlympiadBench**	
		Acc (%) ↑	**Lat** (s) ↓	**Acc** (%) ↑	**Lat** (s) ↓	**Acc** (%) ↑	**Lat** (s) ↓
SPECS (E)	4	59.2	14.8	77.0	9.83	41.3	13.1
SPECS (C)	4	53.1	13.3	77.0	9.80	38.5	12.5
SPECS (E)	8	62.5	20.7	79.5	11.3	41.7	16.9
SPECS (C)	8	58.4	20.2	75.3	11.0	41.0	18.1
SPECS (E)	16	68.3	28.8	81.0	15.8	44.0	24.6
SPECS (C)	16	64.8	26.4	81.0	15.8	42.6	23.8

Per-step and total latency comparison. At batch size 1 on math reasoning, the SD machinery introduces overhead that exceeds the acceptance-based gains: BS+SD per-step latency is higher than BS (B) across every cell (e.g., MATH500 Qwen n=4: BS (B) 1.13 s/step vs. BS+SD γ=3 1.85 s/step; OlympiadBench Qwen n=4: 1.26 s/step vs. 2.91 s/step). SPECS's block-level switching, in contrast, reduces both per-step and total latency relative to BS (B).

Horizon sweep. Table 14 reports BS+SD at all three speculative horizons $\gamma \in \{1, 3, 5\}$ on AMC23, MATH500, and OlympiadBench, for both the Qwen2.5 and Llama-3 draft/target families. Across the 18 cells, γ=3 lies on the BS+SD Pareto frontier in 14 cells, γ=5 in 7, and γ=1 in 5; the γ=3 row is therefore reproduced in the main results Table 1, and $\gamma \in \{1, 5\}$ serve as additional ablations confirming that SPECS's accuracy–latency advantage on harder benchmarks holds across the full sweep.

Vocabulary mismatch. The Qwen2.5-Math 7B and 1.5B vocab dimensions differ by 128 reserved special tokens; we vocab-padded the draft to satisfy vLLM's matching-vocab requirement, leaving generation behaviour on the original vocabulary unchanged.

B.6 Latency Breakdown

In this experiment, we further hope to break down and understand the distribution of latency cost across the various components of SPECS: generation from the draft model (when deferred to), generation from the target model, and scoring using the draft and target models. We evaluate these across all three datasets, AMC23, MATH500 and OlympiadBench when the draft-target pair is Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct. We observe that generation from the target model is still a large component of the latency breakdown, since it is slower than the other steps. However, the overall latency can be brought down via generation from the draft model.

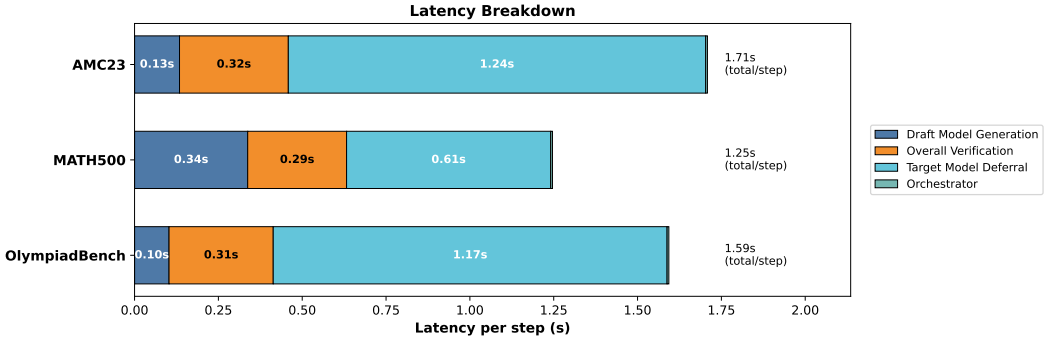


Figure 6: Latency breakdown of SPECS

Table 14: **Full BS+SD speculative-horizon sweep ($\gamma \in \{1, 3, 5\}$).** BS+SD baseline (target served via vLLM token-level speculative decoding using the small model as draft, with horizon γ controlling the number of speculative tokens per target verification step) compared against $BS(\texttt{B})$ on the same setup as Table 1.

Method	n	**Acc** (%) ↑	**Lat** (s) ↓	**Lat/step** (s) ↓	**%** B	**Acc** (%) ↑	**Lat** (s) ↓	**Lat/step** (s) ↓	**%** B
		Qwen2.5 Family				Llama3 Family			
AMC23									
BS (B)	4	59.2 ± 1.2	18.3 ± 1.5	1.46 ± 0.06	100	26.3 ± 1.3	12.7 ± 1.3	0.59 ± 0.17	100
BS+SD (γ=1)	4	56.8 ± 1.4	18.21 ± 1.6	2.24 ± 0.08	100	24.9 ± 1.4	9.51 ± 1.4	1.28 ± 0.20	100
BS+SD (γ=3)	4	57.2 ± 1.2	16.64 ± 1.5	2.03 ± 0.06	100	25.1 ± 1.3	8.86 ± 1.3	1.14 ± 0.17	100
BS+SD (γ=5)	4	57.5 ± 1.5	17.48 ± 1.7	2.16 ± 0.09	100	25.3 ± 1.5	9.39 ± 1.5	1.23 ± 0.20	100
BS (B)	8	59.2 ± 5.1	21.6 ± 1.1	2.00 ± 0.06	100	27.5 ± 2.0	19.1 ± 0.96	0.84 ± 1.39	100
BS+SD (γ=1)	8	60.9 ± 5.3	24.40 ± 1.3	3.05 ± 0.09	100	25.5 ± 2.1	12.25 ± 1.1	1.62 ± 1.45	100
BS+SD (γ=3)	8	61.3 ± 5.1	22.66 ± 1.1	2.78 ± 0.06	100	25.7 ± 2.0	11.22 ± 0.96	1.46 ± 1.39	100
BS+SD (γ=5)	8	61.7 ± 5.4	23.81 ± 1.4	2.95 ± 0.10	100	26.0 ± 2.2	12.02 ± 1.1	1.58 ± 1.47	100
BS (B)	16	68.3 ± 4.2	35.0 ± 2.6	2.80 ± 0.13	100	32.5 ± 4.7	32.1 ± 2.4	1.24 ± 1.94	100
BS+SD (γ=1)	16	64.8 ± 4.4	36.49 ± 2.9	3.23 ± 0.18	100	32.27 ± 4.8	23.10 ± 2.6	2.54 ± 2.03	100
BS+SD (γ=3)	16	65.3 ± 4.2	34.27 ± 2.6	2.99 ± 0.13	100	32.7 ± 4.7	21.59 ± 2.4	2.31 ± 1.94	100
BS+SD (γ=5)	16	65.7 ± 4.5	35.85 ± 2.8	3.15 ± 0.17	100	33.13 ± 4.9	22.71 ± 2.7	2.47 ± 2.01	100
MATH500									
BS (B)	4	69.3 ± 4.1	11.2 ± 3.7	1.13 ± 0.01	100	32.0 ± 3.3	8.4 ± 0.1	0.39 ± 0.29	100
BS+SD (γ=1)	4	66.8 ± 4.3	15.38 ± 3.9	2.04 ± 0.04	100	35.8 ± 3.5	17.42 ± 0.2	2.15 ± 0.33	100
BS+SD (γ=3)	4	67.2 ± 4.1	14.30 ± 3.7	1.85 ± 0.01	100	36.1 ± 3.3	16.07 ± 0.1	1.96 ± 0.29	100
BS+SD (γ=5)	4	67.4 ± 4.4	15.05 ± 3.9	1.98 ± 0.05	100	36.4 ± 3.6	17.06 ± 0.2	2.10 ± 0.34	100
BS (B)	8	71.3 ± 0.5	13.6 ± 1.9	1.51 ± 0.00	100	33.0 ± 2.2	12.1 ± 0.1	0.52 ± 0.3	100
BS+SD (γ=1)	8	70.7 ± 0.7	19.30 ± 2.1	2.56 ± 0.04	100	41.7 ± 2.4	22.16 ± 0.2	2.64 ± 0.35	100
BS+SD (γ=3)	8	71.0 ± 0.5	17.99 ± 1.9	2.31 ± 0.00	100	42.0 ± 2.2	20.56 ± 0.1	2.38 ± 0.3	100
BS+SD (γ=5)	8	71.3 ± 0.8	18.92 ± 2.0	2.49 ± 0.05	100	42.3 ± 2.5	21.79 ± 0.2	2.57 ± 0.36	100
BS (B)	16	75.5 ± 0.5	21.3 ± 0.5	2.15 ± 0.02	100	39.0 ± 2.2	18.9 ± 0.46	0.79 ± 0.21	100
BS+SD (γ=1)	16	74.6 ± 0.7	34.32 ± 0.7	4.39 ± 0.05	100	38.9 ± 2.4	31.84 ± 0.6	3.69 ± 0.26	100
BS+SD (γ=3)	16	75.0 ± 0.5	31.77 ± 0.5	4.05 ± 0.02	100	39.2 ± 2.2	29.48 ± 0.46	3.34 ± 0.21	100
BS+SD (γ=5)	16	75.3 ± 0.8	33.14 ± 0.7	4.28 ± 0.05	100	39.5 ± 2.5	31.10 ± 0.6	3.61 ± 0.28	100
OlympiadBench									
BS (B)	4	39.3 ± 2.1	13.1 ± 0.3	1.32 ± 0.3	100	17.0 ± 1.2	13.0 ± 0.05	0.59 ± 0.18	100
BS+SD (γ=1)	4	39.02 ± 2.3	13.85 ± 0.5	3.18 ± 0.35	100	16.6 ± 1.4	20.35 ± 0.1	2.83 ± 0.23	100
BS+SD (γ=3)	4	39.4 ± 2.1	12.9 ± 0.3	2.91 ± 0.3	100	16.8 ± 1.2	18.84 ± 0.05	2.56 ± 0.18	100
BS+SD (γ=5)	4	39.78 ± 2.4	13.56 ± 0.5	3.11 ± 0.36	100	17.0 ± 1.5	19.95 ± 0.1	2.75 ± 0.24	100
BS (B)	8	41.3 ± 2.1	17.9 ± 0.5	1.81 ± 0.5	100	18.3 ± 0.48	18.9 ± 0.1	0.83 ± 0.14	100
BS+SD (γ=1)	8	39.72 ± 2.3	18.85 ± 0.7	1.95 ± 0.57	100	18.0 ± 0.6	28.52 ± 0.2	3.77 ± 0.20	100
BS+SD (γ=3)	8	40.1 ± 2.1	17.5 ± 0.5	1.80 ± 0.5	100	18.2 ± 0.48	26.64 ± 0.1	3.42 ± 0.14	100
BS+SD (γ=5)	8	40.60 ± 2.4	18.46 ± 0.7	1.92 ± 0.58	100	18.5 ± 0.7	27.98 ± 0.2	3.66 ± 0.22	100
BS (B)	16	43.7 ± 0.5	25.4 ± 0.9	2.57 ± 0.9	100	22.0 ± 1.2	30.1 ± 0.8	1.20 ± 5.3	100
BS+SD (γ=1)	16	43.38 ± 0.7	30.00 ± 1.1	2.92 ± 1.02	100	22.1 ± 1.4	46.03 ± 1.0	5.38 ± 5.65	100
BS+SD (γ=3)	16	43.8 ± 0.5	27.9 ± 0.9	2.69 ± 0.9	100	22.4 ± 1.2	42.81 ± 0.8	4.93 ± 5.3	100
BS+SD (γ=5)	16	44.36 ± 0.8	29.33 ± 1.2	2.85 ± 1.04	100	22.7 ± 1.5	45.38 ± 1.0	5.27 ± 5.70	100

B.7 Hardware and frameworks used for the experiments

The main experiments run on a node with 2 NVIDIA A100 80GB GPUs. The draft model and the PRM share one GPU, and the target model uses the other. Each model is accessed through a vLLM [60] API endpoint. For both beam search and SPECS, we vary the computational budget (e.g., by adjusting the beam width n) to trace out different latency-accuracy scalings.

B.8 Engineering and Performance Optimizations:

In order to make the SPECS algorithm more latency-efficient, we implemented some performance optimizations for increased parallelism and efficiency that are worth noting here. In particular, within the SUBSAMPLE subroutine in Algorithm 2, the log-likelihoods of the generated beams under the target model as well as their scores under the reward model are evaluated. Both of these options can be carried out concurrently to only pay for the latency cost of the slower of these two operations. Moreover, during the generation of new candidate beams for a given step, ideally we would like to use prefix-caching so that we do not need to recompute the logits of the previously decoded steps. However, prefix-caching is disabled in vLLM when we also want to access the log probabilities of the prompt tokens, which is necessary for the SUBSAMPLE subroutine that requires the computation of log-likelihoods of trajectories under the target model. We solve this problem by modifying the

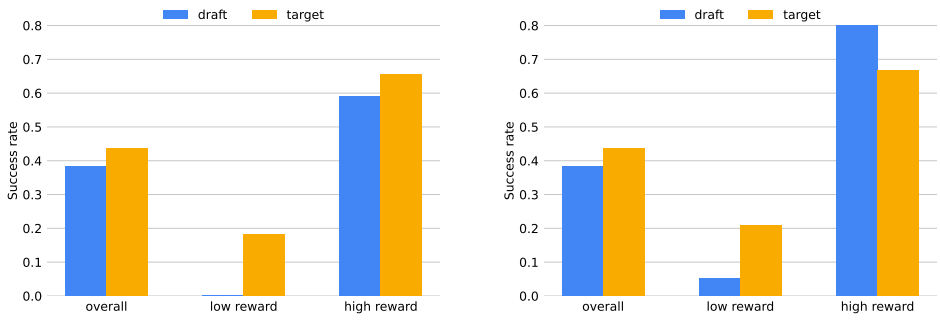
vLLM source code to enable the computation of log probabilities. As a consequence, we do not need to compute the log-probabilities of the tokens that are prefix-cache hit, and only need the log-probabilities of the tokens appearing in the last step. We solve this issue by bypassing the prefix-cache hit tokens for the previous steps and efficiently computing the log-probabilities of the newly generated tokens in the last step, avoiding redundant computations.

B.9 Hyperparameters for SPECS experiments

The hyperparameters used in the experiments in Table 1 are presented in Table 15. Importantly, there are 3 key hyperparameters in this algorithm: the beam-width n, inverse-temperature for soft verification β, and the dynamic switching threshold τ. For the MATH500 and OlympiadBench datasets, hyperparameters are chosen on a separate set of 100 questions from that dataset, treated as a validation set. For AMC23 dataset, since there were only 40 questions, we select the hyperparameters that give the best accuracy vs. latency for different choices of n on the entire dataset, as is common practice for small datasets.

Table 15: Chosen hyperparameters β and τ for different datasets; $\mathbf{n}$ is the beam width.

Family	Dataset	n	β	τ
Qwen Family	AMC23	4	10000	0.8
		8	1000	0.8
		16	1000	0.9
	MATH500	4	1000	0.8
		8	10000	0.75
		16	1000	0.8
	OlympiadBench	4	1000	0.9
		8	100	0.9
		16	1000	0.7
Llama Family	AMC23	4	1000	0.9
		8	1000	0.8
		16	1000	0.85
	MATH500	4	100	0.8
		8	1000	0.8
		16	100	0.7
	OlympiadBench	4	10000	0.8
		8	100	0.6
		16	1000	0.7



(a) "High reward" threshold is 0.6

(b) "High reward" threshold is 0.7

Figure 7: The threshold on reward determining the batch of "high-reward" 8-step partial traces generated by the target model is varied, within the same experimental setup of Figure 2b. The trend persists very strongly: the draft model is much better at completing high-reward partial traces compared to low-reward partial traces generated by the target model.

B.10 Dynamic switching

In this section, we carry out some further analyses of how the accuracy varies as the reward threshold which defines "high-reward" and "low-reward" is varied in the setting of Figure 2b, where it is set as 0.5. As we increase the reward threshold defining high-reward, we observe that the trend grows stronger: the draft model gets increasingly better at solving high-reward traces. The results are described in Figure 7.

We further try to understand the effect of switching from target model as generator to draft model as generator, as a function of the point at which traces are sliced. In Figure 2b we observed that the draft model is very successful at completing 8-step high-reward partial traces generated by the target model, while it is much worse at completing the corresponding low-reward partial traces. In this section, we run the same experiment but set the switchpoint to be earlier, by generating only 4 initial steps of reasoning from the target model instead of 8. The results of this experiment are plotted in Figure 8. We observe that the draft model continues to achieve higher accuracy when completing high-reward traces compared to when completing low-reward traces. However, when we set the switchpoint to be earlier, we observe that completing with either draft or target model achieves similar performance on the low-reward traces, while there is a deterioration in performance on high-reward traces when the draft model is used to complete them.

Based on this experiment, we conclude that the trend observed in Figure 2b is stronger when we switch from the target to the draft model at later points: completing high-reward traces using the draft model is better than using the target model, when the high-rewards are observed later in the trace.

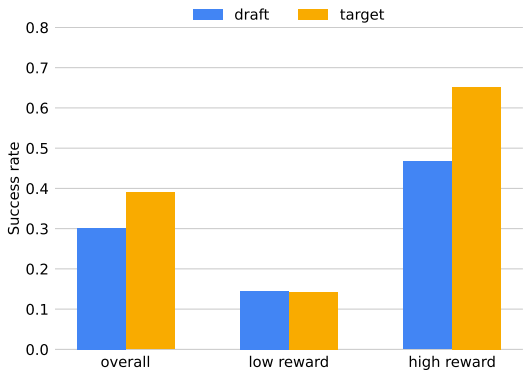


Figure 8: We generate 4 steps of reasoning from the target model. The remaining steps are completed either by the target model, or by the draft model. We bucket the initial 4-step reasoning traces generated by the target model into "low-reward" and "high-reward" based on the PRM reward of the trace, where the threshold is set as reward $= 0.5$. When contrasted with Figure 2b, which is the same experiment, but with 8 initial steps of reasoning generated by the target model, there are some differences.

B.11 Robustness to Critic Noise

A key concern for approaches based on reward-guided generation is the reliability of the Process Reward Model (PRM). To assess, in a controlled manner, how SPECS tolerates imperfections in the PRM, we simulate a degraded reward signal by injecting Gaussian noise: $\tilde{r}_{\texttt{PRM}} = r_{\texttt{PRM}} + Z$, where $Z \sim \mathcal{N}(0, 0.01)$ is an independent zero-mean Gaussian with standard deviation 0.1. As shown in Table 16, SPECS demonstrates remarkable robustness. Even with a noisy signal, the performance gap widens between SPECS and the baseline which is beam search with the small model. At the same time SPECS remains competitive with beam search with the large model. Crucially, we did not retune hyperparameters for this noisy setting, suggesting that SPECS does not require a perfect reward signal to function effectively.

Table 16: Robustness of SPECS under a noisy PRM ($\tilde{r} = r + Z$ where $Z \sim \mathcal{N}(0, 0.01)$). SPECS maintains high success rates compared to baselines, demonstrating that it does not rely on perfect reward signals.

Method	n	AMC23		MATH500		OlympiadBench	
		Acc (%) ↑	**Lat** (s) ↓	**Acc** (%) ↑	**Lat** (s) ↓	**Acc** (%) ↑	**Lat** (s) ↓
BS(Small) w/ $\tilde{r}$	4	36.3	9.55	53.3	7.07	26.0	11.4
BS(Large) w/ $\tilde{r}$	4	43.3	14.4	76.7	11.2	30.7	14.4
SPECS w/ $\tilde{r}$	4	**46.7**	16.5	**78.0**	11.1	**30.7**	14.1
BS(Small) w/ $\tilde{r}$	8	34.4	12.1	62.0	9.77	25.0	15.8
BS(Large) w/ $\tilde{r}$	8	50.8	20.6	**78.7**	14.2	34.0	20.3
SPECS w/ $\tilde{r}$	8	**50.8**	19.6	71.3	12.5	**35.3**	20.0
BS(Small) w/ $\tilde{r}$	16	43.1	19.7	59.3	15.3	29.3	25.4
BS(Large) w/ $\tilde{r}$	16	46.7	26.6	**73.3**	18.6	32.7	25.5
SPECS w/ $\tilde{r}$	16	**49.2**	27.6	72.7	18.8	**37.0**	28.0

C Theoretical analysis of SPECS

In this section, we describe the theoretical analysis of SPECS. We separate the error due to a finite beam-width from the discrepancy induced by the critic. As the beam-width increases, the output distribution approaches the critic-guided policy; when the critic error is zero, this policy is optimal for KL-regularized reward maximization. We first introduce some relevant notation.

Notation. Denote the spaces of prompts and complete responses by $\mathcal{X}$ and $\mathcal{Y}$, and the draft and target models by $\pi_{\texttt{draft}}$ and $\pi_{\texttt{target}}$. Write $y = (y_1, \ldots, y_H)$ for the blocks of a response and $y_{\le t} = (y_1, \ldots, y_t)$ for its prefix, with $y_{\le 0} = \emptyset$. Responses end in an EOS token; shorter responses are padded with deterministic blocks whose critic score is zero. The prompt distribution is ρ, and $\mathbb{E}_{\rho,\pi}[\cdot]$ abbreviates $\mathbb{E}_{X\sim\rho, Y\sim\pi(\cdot|X)}[\cdot]$. For policies π_1, π_2, write $D_{\mathsf{KL}}(\pi_1\|\pi_2) = \mathbb{E}_{X\sim\rho}[D_{\mathsf{KL}}(\pi_1(\cdot|X)\|\pi_2(\cdot|X))]$. Throughout, the reward and critic scores lie in $[0, 1]$. For $\beta > 0$, the objective is to maximize

$$\mathcal{L}_\beta(\pi) = \mathbb{E}_{\rho,\pi}[r(Y|X)] - \frac{1}{\beta} D_{\mathsf{KL}}(\pi\|\pi_{\texttt{target}}). \tag{5}$$

Prior to analyzing the algorithm, we begin with a well-known connection between the KL regularized reward maximization objective and the KL divergence between the learnt policy and the optimal aligned policy [29–31]. We include the proof for the sake of completeness.

Lemma 1 (KL divergence and the objective gap)**.** *For every policy π with finite $D_{\mathsf{KL}}(\pi\|\pi_{\texttt{target}})$,*

$$\mathcal{L}_\beta(\pi^\star_\beta) - \mathcal{L}_\beta(\pi) = \frac{1}{\beta} D_{\mathsf{KL}}(\pi\|\pi^\star_\beta), \tag{6}$$

where $\pi^\star_\beta$ maximizes $\mathcal{L}_\beta$.

Proof. The optimal policy has the form

$$\pi^\star_\beta(y|x) = \frac{\pi_{\texttt{target}}(y|x) e^{\beta r(y|x)}}{Z(x)}, \qquad Z(x) = \mathbb{E}_{Y\sim\pi_{\texttt{target}}(\cdot|x)}[e^{\beta r(Y|x)}]. \tag{7}$$

Taking logarithms gives

$$\begin{aligned} D_{\mathsf{KL}}(\pi\|\pi^\star_\beta) &= D_{\mathsf{KL}}(\pi\|\pi_{\texttt{target}}) - \beta\mathbb{E}_{\rho,\pi}[r(Y|X)] + \mathbb{E}_{X\sim\rho}[\log Z(X)] \\ &= -\beta\mathcal{L}_\beta(\pi) + \mathbb{E}_{X\sim\rho}[\log Z(X)]. \end{aligned}$$

Substituting $\pi = \pi^\star_\beta$ and subtracting proves the claim. □

This result motivates the reverse KL divergence as our performance measure: it is the suboptimality gap of the KL-regularized objective, up to a factor of β.

C.1 Proof of Theorem 1

In order to prove Theorem 1, it will be helpful to consider how SUBSAMPLE behaves in the setting where $H = 1$, with a single prompt x. We will drop x in the conditioning prompt within the policy notation temporarily, and introduce a lemma controlling the rate at which the distribution over responses induced by SUBSAMPLE converges to its limiting distribution.

Lemma 2 (Analysis of SPECS for $H = 1$ and a single prompt)**.** *Fix a draft generator $\pi_{\texttt{gen}}$ and let,*

$$\phi_\beta(y) = \frac{\pi_{\texttt{target}}(y)}{\pi_{\texttt{gen}}(y)} e^{\beta r_{\texttt{critic}}(y)}, \qquad 0 < \phi_{\min} \le \phi_\beta(y) \le \phi_{\max} < \infty.$$

Let $\pi_{\texttt{SMC},n}$ be the marginal distribution over responses of SUBSAMPLE*, with beam size n. Then, for every integer $n \ge 1$,*

$$D_2(\pi_{\texttt{SMC},n} \| \widetilde{\pi}_\beta) \le \frac{9(\phi_{\max}/\phi_{\min} - 1)^2}{n^2}. \tag{8}$$

Proof. The proof of this lemma is discussed in Section C.2.1. □

For either $\pi_{\texttt{gen}} = \pi_{\texttt{target}}$ or $\pi_{\texttt{gen}} = \pi_{\texttt{draft}}$, at every fixed prefix,

$$\frac{\phi_{\max}}{\phi_{\min}} \le C_{\texttt{block}} e^\beta$$

Proof of Theorem 1. Let $\mathcal{F}_{t-1}$ contain the prompt and all sampling randomness preceding iteration t, including the selected generator. Write $\pi_{\texttt{SPECS}}(\cdot|\mathcal{F}_{t-1})$ for the conditional distribution of the next selected block. For each $x, y_{\le t-1}$, Lemma 2 gives, conditional on any preceding history with $X = x$ and $Y_{\le t-1} = y_{\le t-1}$,

$$D_2(\pi_{\texttt{SPECS}}(\cdot|\mathcal{F}_{t-1}) \,\|\, \widetilde{\pi}_\beta(\cdot|x, y_{\le t-1})) \le \frac{9(C_{\texttt{block}} e^\beta - 1)^2}{n^2}.$$

Here $D_2(P\|Q) = \log\big(1 + D_{\chi^2}(P\|Q)\big)$. The next-block distribution conditional only on $x, y_{\le t-1}$ is a mixture over histories. Convexity of D_{χ^2} in its first argument and monotonicity of $\log(1+\cdot)$ give

$$D_2(\pi_{\texttt{SPECS}}(\cdot|x, y_{\le t-1}) \,\|\, \widetilde{\pi}_\beta(\cdot|x, y_{\le t-1})) \le \frac{9(C_{\texttt{block}} e^\beta - 1)^2}{n^2}. \tag{9}$$

By the KL chain rule, $D_{\mathsf{KL}} \le D_2$, and eq. (9),

$$\begin{aligned} D_{\mathsf{KL}}(\pi_{\texttt{SPECS}} \| \widetilde{\pi}_\beta) &= \sum_{t=1}^{H} \mathbb{E}_{\rho,\pi_{\texttt{SPECS}}}[D_{\mathsf{KL}}(\pi_{\texttt{SPECS}}(\cdot|X, Y_{\le t-1}) \,\|\, \widetilde{\pi}_\beta(\cdot|X, Y_{\le t-1}))] \\ &\le \sum_{t=1}^{H} \mathbb{E}_{\rho,\pi_{\texttt{SPECS}}}[D_2(\pi_{\texttt{SPECS}}(\cdot|X, Y_{\le t-1}) \,\|\, \widetilde{\pi}_\beta(\cdot|X, Y_{\le t-1}))] \\ &\le \frac{9H(C_{\texttt{block}} e^\beta - 1)^2}{n^2}. \end{aligned} \tag{10}$$

Finally, apply Lemma 3, stated below, with $(\pi_1, \pi_2, \pi_3) = (\pi_{\texttt{SPECS}}, \widetilde{\pi}_\beta, \pi^\star_\beta)$. By eq. (10) and the definition of $\varepsilon_{\texttt{critic}}$,

$$\begin{aligned} D_{\mathsf{KL}}(\pi_{\texttt{SPECS}} \| \pi^\star_\beta) &\le 2D_{\mathsf{KL}}(\pi_{\texttt{SPECS}} \| \widetilde{\pi}_\beta) + D_2(\widetilde{\pi}_\beta \| \pi^\star_\beta) \\ &\le \frac{18H(C_{\texttt{block}} e^\beta - 1)^2}{n^2} + \varepsilon_{\texttt{critic}} \\ &\le \frac{18H\, C^2_{\texttt{block}} e^{2\beta}}{n^2} + \varepsilon_{\texttt{critic}}. \end{aligned}$$

This proves the theorem. □

Lemma 3 (Comparing policies using Rényi divergence)**.** *For policies π_1, π_2, π_3,*

$$D_{\mathsf{KL}}(\pi_1 \| \pi_3) \le 2D_{\mathsf{KL}}(\pi_1 \| \pi_2) + D_2(\pi_2 \| \pi_3). \tag{11}$$

C.2 Proofs of the auxiliary lemmas

C.2.1 Fixed beam-width sampling

Proof of Lemma 2. Let $Z = \mathbb{E}_{Y\sim\pi_{\texttt{gen}}}[\phi_\beta(Y)]$. The target of the subsampling step is $\widetilde{\pi}_\beta(y) = \pi_{\texttt{gen}}(y)\phi_\beta(y)/Z$. By exchangeability of the n candidates,

$$\frac{\pi_{\texttt{SMC},n}(y)}{\widetilde{\pi}_\beta(y)} = \mathbb{E}[T_y^{-1}], \qquad T_y = \frac{\phi_\beta(y) + \sum_{i=1}^{n-1}\phi_\beta(Y_i)}{nZ}, \qquad Y_i \overset{\text{i.i.d.}}{\sim} \pi_{\texttt{gen}}.$$

Candidates are counted by their indices, so this identity also includes repeated candidate values. Since $T_y \geq \phi_{\min}/Z$, the identity

$$T_y^{-1} = 1 - (T_y - 1) + \frac{(T_y-1)^2}{T_y}$$

implies

$$\begin{aligned}\left|\frac{\pi_{\texttt{SMC},n}(y)}{\widetilde{\pi}_\beta(y)} - 1\right| &\leq \frac{|\phi_\beta(y) - Z|}{nZ} + \frac{(n-1)\operatorname{Var}(\phi_\beta(Y)) + (\phi_\beta(y)-Z)^2}{n^2 Z\phi_{\min}} \\ &\leq 2u_n + u_n^2, \qquad u_n = \frac{\phi_{\max}/\phi_{\min} - 1}{n}.\end{aligned}$$

The last inequality uses $Z \geq \phi_{\min}$, $|\phi_\beta(y) - Z| \leq \phi_{\max} - \phi_{\min}$, and

$$\operatorname{Var}(\phi_\beta(Y)) \leq Z(\phi_{\max} - Z) \leq Z(\phi_{\max} - \phi_{\min}).$$

Consequently,

$$\log\bigl(1 + D_{\chi^2}(\pi_{\texttt{SMC},n}\|\widetilde{\pi}_\beta)\bigr) \leq \log\bigl(1 + (2u_n + u_n^2)^2\bigr) \leq 9u_n^2.$$

For $u_n \leq 1$, the final inequality follows from $2u_n + u_n^2 \leq 3u_n$ and $\log(1+z) \leq z$. For $u_n \geq 1$, use $1 + (2u_n + u_n^2)^2 \leq (1+3u_n^2)^2$ and $2\log\bigl(1+3u_n^2\bigr) \leq 6u_n^2$. □

C.2.2 Comparison with the global optimum

Proof of Lemma 3. Fix a prompt x and suppress the conditioning. By Jensen's inequality,

$$\begin{aligned}D_{\mathsf{KL}}(\pi_1\|\pi_3) - 2D_{\mathsf{KL}}(\pi_1\|\pi_2) &= \mathbb{E}_{Y\sim\pi_1}\left[\log\frac{\pi_2(Y)^2}{\pi_1(Y)\pi_3(Y)}\right] \\ &\leq \log\left(\sum_y \frac{\pi_2(y)^2}{\pi_3(y)}\right) = D_2(\pi_2\|\pi_3).\end{aligned}$$

Averaging this conditional inequality over $X \sim \rho$ proves the lemma. □

Limiting-error statement. Put $f = \log\Bigl(\widetilde{\pi}_\beta/\pi_\beta^\star\Bigr)$. The normalized score and reward bounds give $|f| \leq \beta(H+1)$. By eq. (10), $D_{\mathsf{KL}}(\pi_{\texttt{SPECS}}\|\widetilde{\pi}_\beta) \to 0$ as $n \to \infty$. Pinsker's inequality then gives $\mathbb{E}_{\rho,\pi_{\texttt{SPECS}}}[f] - \mathbb{E}_{\rho,\widetilde{\pi}_\beta}[f] \to 0$. Thus

$$D_{\mathsf{KL}}(\pi_{\texttt{SPECS}}\|\pi_\beta^\star) = D_{\mathsf{KL}}(\pi_{\texttt{SPECS}}\|\widetilde{\pi}_\beta) + \mathbb{E}_{\rho,\pi_{\texttt{SPECS}}}[f] \longrightarrow D_{\mathsf{KL}}(\widetilde{\pi}_\beta\|\pi_\beta^\star) \leq \varepsilon_{\texttt{critic}}.$$

□

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We added limitation of our work in the conclusion section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We provide proofs for all theory results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We explain the necessary packages and hardware in the paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We plan to open-source our code in the final version of the paper.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We explain all the experimental settings, the datasets, which problems and which splits clearly in the paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We do multiple runs in our experiments and report both mean and standard deviation.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification:Yes, we provide the computer resources used (2 80GB A100 GPUs) and report wall clock time of runs clearly.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our paper conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: This work does not have negative societal impacts or noteworthy positive societal impacts.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: There are no significant risk involved in this work that we believe is worth safeguarding.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All creators of code, data and models are credited properly with right terms.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We dont release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer:[N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.