

000 001 002 003 004 005 006 007 008 009 010 011 012 013 014 015 016 017 018 019 020 021 022 023 024 025 026 027 028 029 030 031 032 033 034 035 036 037 038 039 040 041 042 043 044 045 046 047 048 049 050 051 052 053 SPECEXIT: ACCELERATING LARGE REASONING MODEL VIA SPECULATIVE EXIT

Anonymous authors

Paper under double-blind review

ABSTRACT

Despite their strong performance on reasoning tasks, large reasoning models (LRMs) often suffer from overthinking, producing unnecessarily long outputs and incurring high end-to-end latency, a significant limitation to their real-world deployment. To address overthinking, early-exit mechanisms have been proposed to terminate reasoning before typical completion, showing that this approach can effectively shorten generation length with minimal impact on accuracy. However, their reliance on probing mechanisms introduces a detection overhead that limits their end-to-end latency gains and compromises their generalizability across diverse problems. Inspired by the use of hidden states in speculative decoding, we propose **SpecExit**, a novel framework that predicts both future tokens and an early-exit signal directly from a lightweight draft model without probing overhead. Our method offers significant improvements, achieving up to 66% generation length reduction and 2.5 \times end-to-end speedup compared with the speculative decoding baseline, without compromising accuracy. Our method leverages the inherent signals from hidden states to provide effective early-exit signals, suggesting broader use of hidden states for efficient reasoning. Our code is available at: <https://anonymous.4open.science/r/SpecExit-B802>.

1 INTRODUCTION

Large reasoning models (LRMs) such as OpenAI-o1 (OpenAI, 2024), DeepSeek-R1 (DeepSeek-AI et al., 2025) and Qwen (Qwen et al., 2025) have recently achieved state-of-the-art performance in complex tasks. These models follow the test-time scaling law (Snell et al.; Brown et al.; Muenninghoff et al.), where generating longer chain-of-thought (CoT) sequences (Wei et al.) generally enhances model performance. However, this reliance on extended reasoning often leads to an *overthinking* problem, where models produce unnecessarily verbose outputs. This redundancy leads to both excessive token usage and high end-to-end latency, which limits LRMs' practical deployment.

To mitigate overthinking, researchers have proposed both inference-time and training-based strategies (Sui et al., 2025). Inference-time early-exit methods (Yang et al., 2025; Fu et al., 2024) rely on model-generated signals such as intermediate answers or output logits to terminate decoding once sufficient evidence is detected. These methods can shorten reasoning length without harming accuracy, but the probing overhead they introduce limits the actual latency gains. Training-based approaches, such as reinforcement learning (Aggarwal & Welleck, 2025; Yeo et al., 2025) and supervised fine-tuning (Ma et al., 2025b; Munkhbat et al., 2025), incur little runtime overhead during inference but risk altering the model's output distribution. As a result, existing methods struggle to deliver consistent improvements in end-to-end efficiency.

Speculative decoding (Chen et al., 2023; Leviathan et al., 2023) is a promising approach that improves efficiency without altering the target model's outputs. A lightweight draft model proposes multiple candidate tokens in advance, and the target model then verifies these candidates in parallel. This parallel verification allows the system to generate multiple tokens per forward pass of the target model, alleviating the inherent sequential bottleneck of autoregressive decoding and better utilizing modern GPU hardware. However, this strategy alone does not resolve the overthinking problem, as models still generate the full CoT. Recent advanced speculative decoding methods (Li et al., 2024a; Zhang et al., b) exploit hidden states to predict several future tokens. Other studies (Lin et al.; Dong et al.) also show that hidden states encode richer predictive signals beyond next-token probabilities.

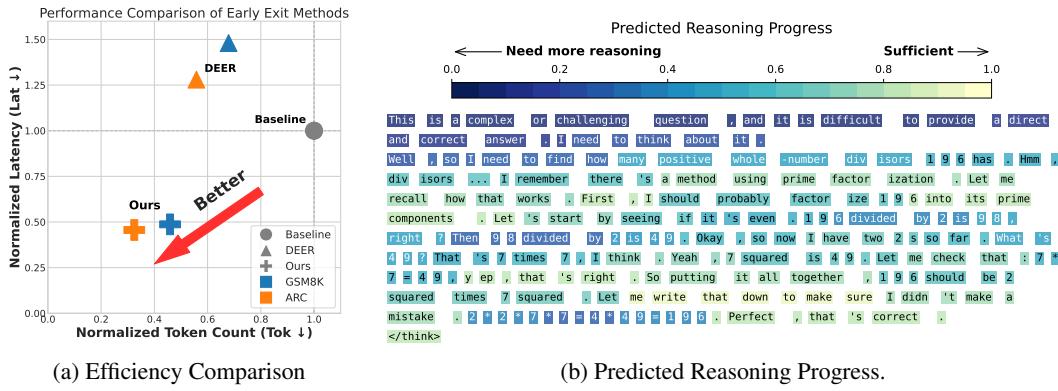


Figure 1: Effectiveness of the proposed method. (a) Statistical comparison showing that our approach produces shorter reasoning chains and faster inference than baselines. (b) visualizes the predicted reasoning progress on a MATH500 example, where darker colors denote insufficient reasoning and lighter colors denote sufficiency, demonstrating valuable signals can be extracted from hidden states regarding the model’s reasoning process.

In this work, we introduce **SpecExit**, a reasoning-aware early-exit framework that leverages draft model hidden states not only to anticipate future tokens but also to predict early-exit signals. Unlike prior probing-based approaches, SpecExit requires no modifications to the target model and incurs no additional detection overhead. Instead, it extends the lightweight draft model with auxiliary prediction heads, enabling it to jointly output token distributions and reasoning-related signals in a single forward pass. By exploiting the latent information embedded in hidden states, SpecExit provides reliable criteria for dynamically terminating chain-of-thought generation when sufficient reasoning has been achieved.

We validate SpecExit on state-of-the-art reasoning models across mathematical, scientific, and logical benchmarks. For Qwen3-4B-Thinking-2507, DeepSeek-R1-Distill-Llama-8B, and Phi-4-reasoning models, SpecExit achieves up to 66% generation length reduction and 2.5x end-to-end latency speedup compared with the speculative decoding baseline. Our contributions can be summarized as follows:

- **Signals Extracted for Early Exit.** We derive early-exit signals from hidden features and integrate them into speculative decoding, enabling reliable early exit for efficient reasoning.
- **General and Practical Framework.** We implement SpecExit, a reasoning-aware early-exit framework, in both PyTorch and vLLM, making it easy to deploy across diverse inference environments.
- **Substantial End-to-End Performance Gains.** SpecExit reduces reasoning length by as much as 66% and delivers as high as 2.5x speedup in end-to-end inference compared with speculative decoding while maintaining accuracy.

2 MOTIVATION

Current early-exit methods face two challenges: runtime overhead from probing and limited generalizability from task-specific prompts. Since hidden states already reflect reasoning sufficiency, we propose leveraging them to replace costly probing for faster and more reliable inference across diverse tasks.

Probing Overhead and Limited Generalizability. Probing-based early-exit methods suffer from both latency overhead and poor generalizability across tasks. Shortened reasoning traces do not effectively translate into latency improvements. As shown in Figure 1a, DEER (Yang et al., 2025) reduces the generation length of GSM8K (Cobbe et al., 2021) and ARC-Challenge(Clark et al., 2018) by 32% and 44% respectively on Qwen3-4B-Thinking-2507, but even increases end-to-end latency than vanilla baseline. The gap arises because probing introduces extra computation, and its effectiveness is highly sensitive to both the task and the model. A probing phrase like “Final Answer is” elicits effective intermediate answers in math problems, but fails in coding tasks where

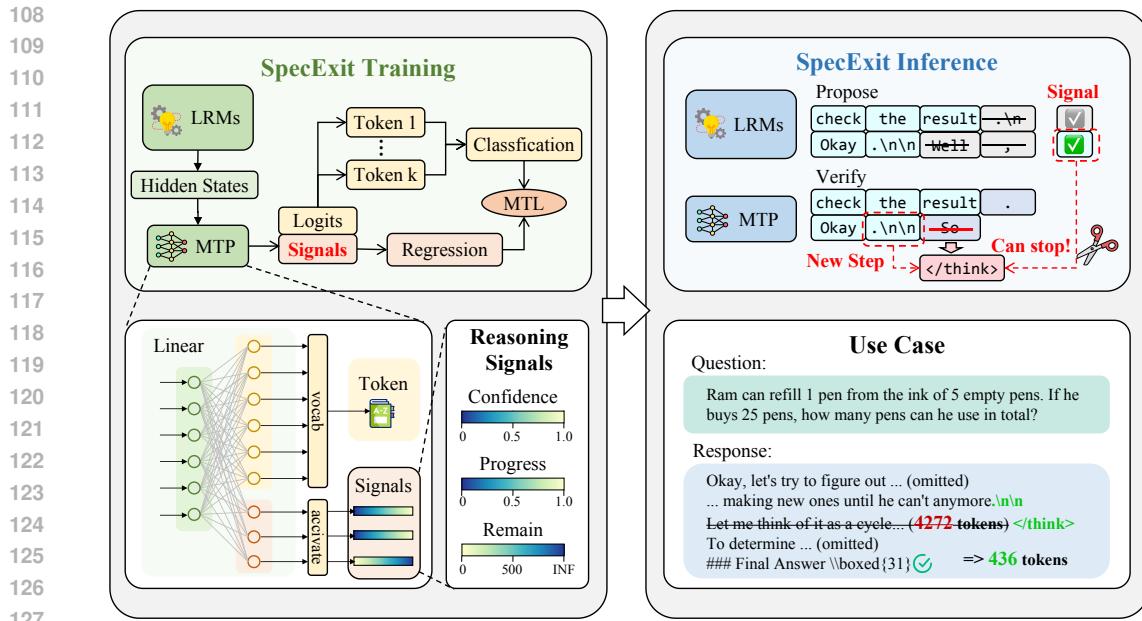


Figure 2: Overall architecture of the proposed SpecExit framework. The Multi-Token Prediction (MTP) layer is augmented to output both token logits and auxiliary signals. Training is performed with Multi-Task Learning (MTL), while at inference these signals guide speculative early stopping without modifying the backbone model. The example illustrates how redundant reasoning steps can be pruned while preserving final answer quality.

extra tokens are still generated. This drawback not only undermines generalizability across domains but also limits true latency savings.

Signals from Hidden States. Our preliminary experiments show that models' hidden states encode informative signals about its reasoning process. As illustrated in Figure 1b, we use a MLP trained on hidden states to predict reasoning progress. For complex tasks, the predicted signals appear darker at the beginning, reflecting the need for continued reasoning, but gradually shift to lighter colors as the model approaches a sufficient chain of thought. This progression suggests that hidden states provide fine-grained indicators of task complexity and reasoning sufficiency. Leveraging these internal signals offers an efficient alternative to costly probing, motivating our approach of utilizing signals from hidden states.

3 METHOD

3.1 OVERALL ARCHITECTURE

SpecExit Framework. The overall design of this work aims to incorporate additional learnable signals into large model reasoning, thereby providing explicit decision-making criteria for early stopping. In the decoding process of large language models, hidden states not only encode semantic and contextual information for next-token prediction but also implicitly contain higher-level cues related to reasoning progress, generation quality, and content completeness. The Multi-Token Prediction (MTP) mechanism leverages these hidden states to project into the vocabulary space and predict multiple future tokens simultaneously, thereby improving inference efficiency. Inspired by this, we extend the MTP layer while keeping the backbone language model unchanged, introducing auxiliary prediction heads that allow the model to explicitly generate reasoning-related signals, including confidence, reasoning progress, and remaining reasoning length, alongside token distributions. This design preserves the original language modeling ability while providing learnable auxiliary variables for inference control, enabling efficient and dynamic reasoning regulation. The overall architecture of the SpecExit framework is shown in Figure 2.

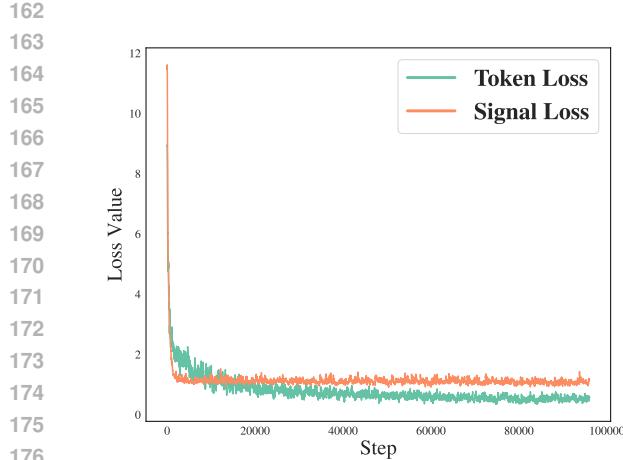


Figure 3: Convergence of token classification loss and signal regression loss during training.

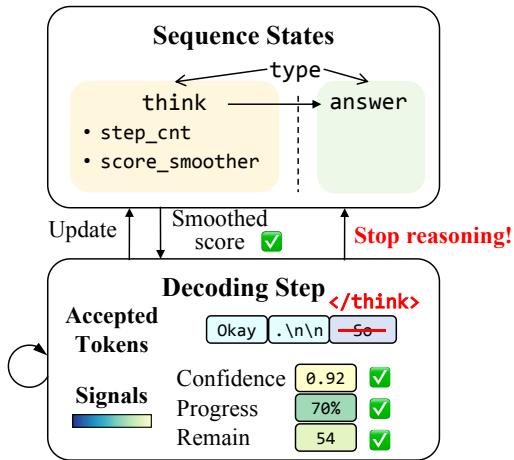


Figure 4: Inference process with signal-guided speculative exit.

Model Structure. We propose a speculative sampling-based early stopping mechanism for reasoning. In the MTP layer of the model, we extend the linear projection to include additional dimensions. In standard decoding, the hidden state $h \in \mathbb{R}^D$ is projected into the vocabulary space to predict the next token distribution, where D is the hidden size of the model. In our method, the linear layer output is extended as:

$$Wh = [W_{\text{tok}}h, W_{\text{conf}}h, W_{\text{prog}}h, W_{\text{rem}}h], \quad (1)$$

where $W_{\text{tok}}h$ produces standard token predictions, while $W_{\text{conf}}h, W_{\text{prog}}h, W_{\text{rem}}h$ predict confidence, reasoning progress, and remaining reasoning length, respectively. These additional signals serve as observable indicators for deciding whether to stop reasoning early, thus reducing redundant computation without compromising output quality.

3.2 SIGNAL-EXTRACTED TRAINING

Data Construction. We first obtain the complete response generated by the base language model and extract the reasoning content enclosed within the `<think>` and `</think>` tokens. To identify the effective reasoning trace, we iteratively attempt to insert the closing marker `</think>` after each paragraph and verify whether the resulting final answer matches the original output. If the answer remains consistent, the subsequent reasoning content is regarded as redundant. Consequently, only the minimal reasoning segment required to produce the correct answer is retained as training data.

Signal Annotation. CONFIDENCE is defined as the geometric mean of the logit probabilities across prediction steps, reflecting the reliability of the generation; REMAINING reasoning length is defined as the number of tokens from the initial `<think>` marker to the earliest valid insertion point of `</think>` that still yields the correct answer; PROGRESS is represented as a normalized value increasing from 0 to 1, capturing the relative progression of the reasoning CoT.

Signal Regression. We propose a cost-efficient extension by introducing a small number of additional dimensions into the linear projection layer of the MTP module for regressing reasoning signals. These dimensions are orthogonal to the vocabulary classification weights, ensuring that signal regression does not interfere with the convergence of speculative decoding training.

The Multi-Task Learning (MTL) overall training objective jointly optimizes token classification and signal regression, defined as:

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \lambda_c \mathcal{L}_{\text{conf}} + \lambda_p \mathcal{L}_{\text{prog}} + \lambda_r \mathcal{L}_{\text{rem}}, \quad (2)$$

216 where \mathcal{L}_{cls} is the standard cross-entropy loss for vocabulary prediction, and $\mathcal{L}_{\text{conf}}$, $\mathcal{L}_{\text{prog}}$, \mathcal{L}_{rem} correspond to the regression losses for confidence, progress, and remaining reasoning length, with 217 λ_c , λ_p , λ_r denoting dynamic weighting coefficients. Specifically:

218 Confidence and progress are optimized using mean squared error (MSE), remaining reasoning length 219 is optimized with mean squared logarithmic error (MSLE):

$$223 \quad \mathcal{L}_{\text{conf}} = \frac{1}{N} \sum_{i=1}^N (\text{sigmoid}(\hat{c}_i) - c_i)^2, \quad (3)$$

$$226 \quad \mathcal{L}_{\text{prog}} = \frac{1}{N} \sum_{i=1}^N (\text{sigmoid}(\hat{p}_i) - p_i)^2, \quad (4)$$

$$229 \quad \mathcal{L}_{\text{rem}} = \frac{1}{N} \sum_{i=1}^N (\hat{r}_i - \log(1 + r_i))^2, \quad (5)$$

233 where \hat{c}_i , \hat{p}_i , and \hat{r}_i represent the model-predicted confidence, progress, and remaining reasoning 234 length, respectively; c_i , p_i , and r_i denote their corresponding ground-truth values; and N is the total 235 number of samples.

236 **Dynamic Weighting.** Since the regression losses of signals converge faster than the token classi- 237 fication loss, we adopt a gradient-based dynamic weighting strategy to balance the contributions of 238 different tasks. This mechanism assigns higher weights to tasks with smaller gradient magnitudes, 239 preventing tasks with larger gradients from dominating the learning process and ensuring all tasks 240 are effectively optimized. Formally, the mechanism is defined as:

$$242 \quad \lambda_j = \frac{\|\nabla_{\theta} \mathcal{L}_j\|}{\sum_k \|\nabla_{\theta} \mathcal{L}_k\|}, \quad \mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cls}} + \sum_j \lambda_j \mathcal{L}_j, \quad (6)$$

245 where $\mathcal{L}_j \in \{\mathcal{L}_{\text{conf}}, \mathcal{L}_{\text{prog}}, \mathcal{L}_{\text{rem}}\}$, $\nabla_{\theta} \mathcal{L}_j$ denotes the gradient of task j with respect to the model 246 parameters, \mathcal{L}_{cls} is the cross-entropy loss for token classification, and λ_j is the dynamically computed 247 weight. This formulation ensures that the gradient contributions of all tasks are balanced, which 248 facilitates stable convergence in multi-task optimization, as shown in Figure 3.

250 3.3 SIGNAL-GUIDED INFERENCE

252 **Overall Procedure.** We build upon the speculative decoding framework, where a smaller draft 253 model first proposes a sequence of candidate tokens, which are then verified in parallel by a larger 254 target model. To evaluate feasibility and efficiency, the inference procedure is implemented on both 255 **PyTorch** and **vLLM** frameworks. The central modification lies in the forward pass of the target 256 model. Beyond computing the logits for the next token, we additionally extract the final hidden state 257 corresponding to the last accepted token. This representation is processed through a lightweight 258 linear layer to generate three signals: a confidence score, a progress indicator, and an estimate of the 259 remaining reasoning length, as shown in Figure 4.

260 **Speculative Decoding Inference Procedure.** In the speculative decoding inference pipeline, each 261 sequence of draft tokens proposed by the draft model is forwarded to the target model for verifica- 262 tion. Only those draft tokens that pass this verification are directly committed to the KV-cache of 263 the target model, and the last accepted draft token is followed by a recover token generated by the 264 target model. The hidden states produced by the target model during verification are then fed back 265 into the draft model to guide the generation of new draft tokens. Subsequently, both the recover 266 token and the newly generated draft tokens are again passed to the target model for verification and 267 potential acceptance. A schematic overview of the speculative decoding pipeline integrated with the 268 proposed reasoning early-exit mechanism is provided in Figure 11(Appendix).

269 **Stopping Conditions.** To ensure that early-exit decisions occur at semantically coherent boundaries, 270 we introduce a class of special markers called *step split tokens*, which indicate natural segmentation

270 **Algorithm 1:** Inference procedure with signal-guided speculative exit

271 **Input:** Draft model M_d , target model M_t , tokenizer, thresholds

272 **Output:** Generated sequence y

273 Define $is_thinking \leftarrow \text{true}$;

274 Define $step_split_tokens \leftarrow \{\text{ids of "\n\n", ".\n\n", ...}\}$;

275 Define $stop_think_token \leftarrow \text{id of } </\text{think}>$;

276 **while** not terminated **do**

277 Extract hidden state of t_{acpt} , generate candidate tokens with M_d ;

278 Compute $signals$ (confidence, progress, remaining);

279 Set $signals \leftarrow$ update smoothed scores;

280 Concat last accepted token with draft candidates, forward through M_t with tree attention;

281 Accept tokens t_{acpt} , accept length l_{acpt} , target model recover token t_{rec} ;

282 **if** $is_thinking$ **and** any($t_{acpt} \in step_split_tokens$) **and** $signals$ exceed thresholds **then**

283 Set $l_{acpt} \leftarrow$ corresponding $step_split_token$ position;

284 Set $t_{rec} \leftarrow stop_think_token$;

285 Update KV-cache and hidden states accordingly;

286 Set $is_thinking \leftarrow \text{false}$;

287 **end**

288 **end**

289

290

291 points in the generated text. Specifically, step split tokens can be divided into two categories: PARAGRAPH DELIMITERS (e.g., $.\n\n$), which mark the end of a paragraph or reasoning unit, and DISCOURSE MARKERS (e.g., "Wait", "But", or "Therefore"), which often signal semantic transitions or logical shifts during reasoning. Since the segmentation strategy based on PARAGRAPH DELIMITERS is more general, this strategy is adopted by default in subsequent experiments. Examples of commonly observed discourse markers in reasoning traces are shown in Figure 12 (Appendix). When a sampled token belongs to the above set, the early-exit logic is triggered. If the smoothed signal exceeds the predefined threshold, the system determines that the reasoning process has been sufficiently explored. The complete inference process is summarized in Algorithm 1. In this case, the accepted output length is truncated at the position of the step split token, and the target model's recover token is replaced with a special reasoning-end marker (e.g., $</\text{think}>$), thereby ensuring that the termination point lies at a natural boundary while maintaining coherence of the generated text.

303 **Signal Smoothing.** Since raw signals may exhibit significant volatility, relying directly on them
 304 risks premature or unstable termination. To enhance robustness, we apply an Exponentially
 305 Weighted Moving Average (EWMA) to smooth the signals across steps. At each iteration, the
 306 smoothed value is updated as a weighted average of the current raw signal and the previous smoothed
 307 value, with the smoothing factor controlling the balance between recent and past observations. A
 308 smaller factor emphasizes historical stability, yielding smoother traces that are less sensitive to trans-
 309 ient noise. This ensures that termination decisions reflect consistent trends rather than isolated
 310 fluctuations.

313 4 EXPERIMENTS

315 4.1 EXPERIMENTAL SETUP

317 To evaluate the effectiveness of our SpecExit framework, we conducted a comprehensive set of
 318 experiments across multiple domains. Specifically, we used the **GSM8K** (Cobbe et al., 2021),
 319 **MATH500** (Hendrycks et al., 2021) and **AIME** (MAA Committees) datasets for mathematical rea-
 320 soning, the **HumanEvalPlus** (Liu et al., 2023) dataset for coding, the **GPQA Diamond** (Rein et al.,
 321 2023) dataset for science, and the **ARC-Challenge** (Clark et al., 2018) dataset for logic. Experi-
 322 ments are conducted on three mainstream LMRs: **Qwen3-4B-Thinking-2507** (Qwen et al., 2025),
 323 **DeepSeek-R1-Distilled-Llama-8B** (DeepSeek-AI et al., 2025) and **Phi-4-reasoning** (Abdin et al.,
 2025).

324
 325
 326
 327
 328
 329
 330
 331
 Table 1: Performance comparison of various reasoning methods on mathematical, scientific, general,
 and coding benchmarks. “Acc” denotes accuracy, “Tok” denotes token count, and “Lat” denotes total
 end-to-end latency. \uparrow indicates that higher values are better, while \downarrow indicates that lower values are
 better. For the early-exit methods (NoThink, DEER, and SpecExit*), the highest and second-highest
 Acc values are marked in **bold** and underline, respectively. Across all methods, the smallest and
 second-smallest Tok and Lat values are marked **bold** and underline, respectively. SpecExit* uses
 default parameter settings consistent with the best variants in ablation studies.

Method	Math						Coding			Science			Logic					
	GSM8K			MATH500			AIME			HUMANEVAL+			GPQA-D			ARC-Challenge		
	Acc \uparrow	Tok \downarrow	Lat \downarrow	Acc \uparrow	Tok \downarrow	Lat \downarrow	Acc \uparrow	Tok \downarrow	Lat \downarrow	Acc \uparrow	Tok \downarrow	Lat \downarrow	Acc \uparrow	Tok \downarrow	Lat \downarrow	Acc \uparrow	Tok \downarrow	Lat \downarrow
Qwen3-4B-Thinking-2507																		
Vanilla	95.3	1414	155.6	96.6	6719	530.1	86.7	19577	243.3	90.9	5079	175.3	68.7	9041	325.8	95.6	1812	156.5
NoThink	95.2	1631	204.2	<u>96.6</u>	6395	488.5	<u>86.7</u>	19816	243.2	<u>88.4</u>	4480	131.5	67.2	<u>8833</u>	276.8	95.1	1889	<u>159.8</u>
DEER	94.3	960	230.3	94.4	4893	519.6	70	17838	218.6	86.6	4079	242.4	67.2	9053	505.2	94.6	1011	200.3
EAGLE3	94.8	1408	140.3	96.6	6670	<u>395.7</u>	80	19792	<u>206.1</u>	87.2	5178	<u>81.7</u>	67.7	8975	212.2	95.7	1822	164.2
SpecExit*	93.8	649	75.8	96.8	4777	367.9	90	17769	187.3	89.6	4319	58.4	68.7	7011	137	94.5	588	71.4
DeepSeek-R1-Distill-Llama-8B																		
Vanilla	76.4	1008	629.4	81.8	6878	857.1	36.7	22170	307	74.4	6287	445.5	43.6	8857	574	49.9	1917	628.5
NoThink	54.6	233	22.2	55.2	1643	262.8	10	<u>8744</u>	<u>184.1</u>	46.3	472	7.3	26.8	1200	166.6	12.6	135	13.6
DEER	<u>74.7</u>	710	484.8	80.8	3533	973.3	40	15619	272.3	<u>79.3</u>	4206	269.2	<u>40.9</u>	8492	521.5	<u>47.5</u>	1029	531.3
EAGLE3	79.3	976	276.9	80.8	6172	593.6	30	25686	228.1	78.7	5312	346.5	43.9	8749	420.1	59.2	1378	496.4
SpecExit*	75.3	<u>333</u>	<u>112.6</u>	<u>80.6</u>	<u>1968</u>	<u>348.3</u>	<u>36.7</u>	8160	176	81.7	<u>3105</u>	<u>118.1</u>	46	<u>6849</u>	<u>307.5</u>	50.3	<u>500</u>	<u>253.7</u>
Phi-4-reasoning																		
Vanilla	95.8	709	207.2	94.9	2122	543.7	74.2	10980	536.3	72.6	2059	300.2	68.7	7544	726.7	96.6	607	193.2
NoThink	<u>95.7</u>	668	197.3	94.1	2051	554.8	<u>70.7</u>	11104	509.4	72.8	1919	297.3	64.7	7334	710.0	96.7	588	178.0
DEER	95.5	<u>582</u>	223.5	92.4	1502	516.0	60.0	7003	507.4	66.3	1420	211.1	<u>65.2</u>	4479	1296.1	<u>96.0</u>	<u>540</u>	183.4
EAGLE3	95.2	707	<u>153.1</u>	94.5	2136	<u>324.2</u>	74.0	10657	<u>308.5</u>	<u>72.0</u>	2035	<u>155.8</u>	68.2	7512	<u>478.2</u>	96.7	615	<u>128.9</u>
SpecExit*	95.8	400	61.0	<u>93.6</u>	<u>1750</u>	271.9	74.7	<u>9988</u>	<u>272.6</u>	<u>72.0</u>	<u>1605</u>	131.8	67.7	<u>6922</u>	422.9	95.8	286	80.0

348
 349
 350
 351
 352
 353
 354
 355
 We compare our SpecExit method against several baselines: **Vanilla**, which represents full generation
 without any early-exit mechanism; **NoThink** (Ma et al., 2025a), which skips the reasoning phase;
DEER (Yang et al., 2025), a dynamic early-exit method; and **EAGLE3** (Li et al., 2025), a speculative decoding baseline. For a fair and consistent comparison, the speculative decoding
 component in our system adopts the same draft-model architecture as EAGLE3, namely a one-layer
 causal model whose hidden size matches the corresponding target model. The draft model is trained
 together with SpecExit signals using the same training procedure to ensure comparable conditions.

363
 364
 365
 Our performance analysis is based on three key metrics, as detailed in Table 1: **Accuracy** (\uparrow), **Token**
 (\downarrow) count and end-to-end **Latency** (\downarrow). All experimental results are obtained by implementing our
 early-exit strategy in vLLM (Kwon et al., 2023), and running inference on an 8xH20 GPU cluster.

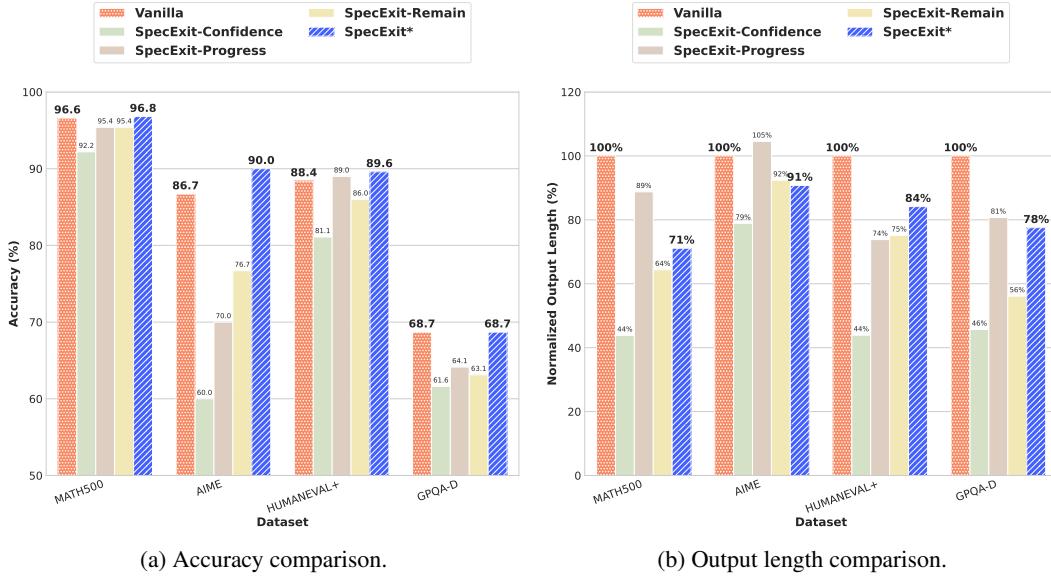
4.2 MAIN RESULTS

366
 367
 368
 369
 We first evaluate the proposed SpecExit against baseline reasoning approaches on mathematical,
 scientific, coding, and logical benchmarks. As shown in Table 1, SpecExit consistently achieves
 substantial reductions in both output length and inference latency while maintaining comparable or
 even higher accuracy.

370
 371
 372
 373
 Across benchmarks, SpecExit significantly shortens reasoning traces, with up to 54% and 53% re-
 duction on GSM8K and ARC-Challenge for Qwen3-4B-Thinking-2507, and up to 66% and 64% re-
 duction for DeepSeek-R1-Distill-Llama-8B. The reduced reasoning length corresponds to mea-
 surable efficiency improvements: SpecExit achieves a up to 1.9x latency reduction with Qwen3-4B-
 Thinking-2507 and up to 2.5x speedup with DeepSeek-R1-Distill-Llama-8B on GSM8K, compared
 with the speculative decoding baseline EAGLE3. Importantly, these gains come only with marginal

accuracy differences, confirming that early termination of redundant reasoning does not harm task performance. By contrast, prior inference-time methods primarily focus on reducing output length, but the latency gains they achieve are relatively modest. In some datasets, the additional computational overhead even leads to slower inference than the standard think mode. Notably, for Qwen3-4B-Thinking-2507 and Phi-4-reasoning models, inserting the `</think>` token at the beginning of reasoning in the NoThink baseline still fails to suppress reasoning, yielding output lengths similar to the vanilla Think mode and occasionally slightly longer.

Overall, these results demonstrate that SpecExit achieves a favorable balance between efficiency and accuracy, highlighting the practicality of integrating reasoning-aware early-exit strategies into LRM inference.



(a) Accuracy comparison.

(b) Output length comparison.

Figure 5: Ablation study of SpecExit signal types on Qwen3-4B-Thinking-2507.

4.3 ABLATION STUDY

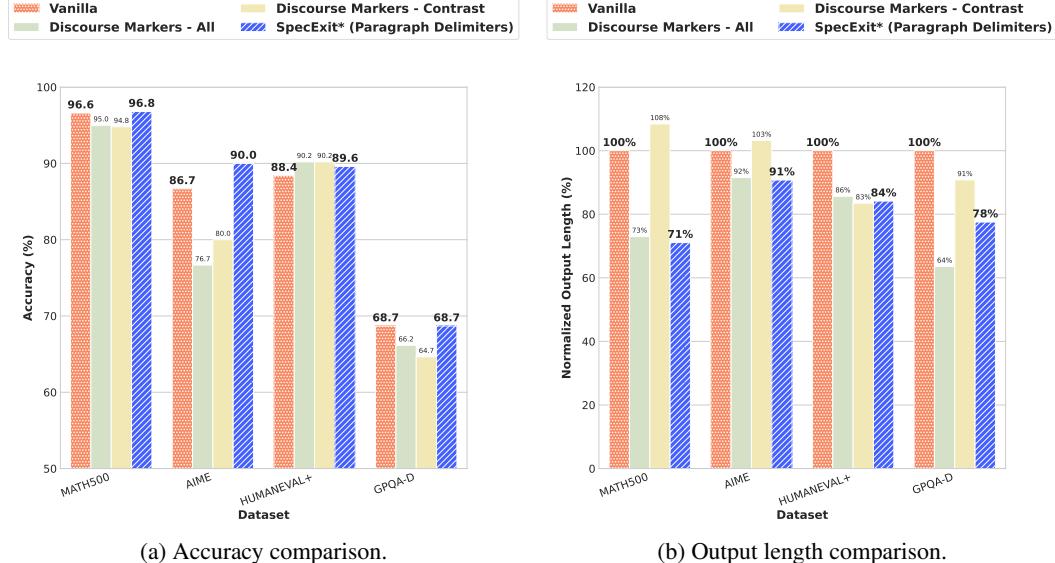
Signal Type. To investigate the impact of individual reasoning signals in SpecExit, we conduct ablation studies on confidence, progress, and remaining token length, along with a combined configuration (SpecExit*) that integrates all three. As shown in Figure 5, the confidence-only variant yields the largest token reduction but overestimates the model’s certainty, resulting in noticeable accuracy drops on complex benchmarks. The predicted reasoning progress increases sharply in the early steps yet continues to fluctuate during iterative reflection. Remaining token length is generally high at the beginning of inference but often triggers premature exits on complicated problems. By integrating all signals, SpecExit* leverages their complementary strengths, preserving competitive accuracy while substantially reducing tokens, demonstrating that multi-signal integration mitigates individual biases and enables more reliable early stopping.

Signal Smoothing. In order to investigate the influence of different smoothing strategies on the stability and performance of early-exit decisions, we conducted a series of ablation experiments comparing multiple approaches. As shown in Table 2, removing smoothing increases the volatility of cognitive signals, leading to inconsistent early exits and increased token consumption. The momentum-based prediction strategy significantly reduces token usage, though it may slightly degrade accuracy due to overly aggressive early termination. Smoothing using sliding-window and paragraph-level averaging offers a better trade-off, maintaining accuracy while improving efficiency. Among all methods, Exponential Weighted Moving Average (EWMA) strikes the most consistent balance, providing both stability and reliability. These results demonstrate that appropriate smoothing is essential for reliable early-exit behavior, as it mitigates the influence of transient fluctuations in the raw cognitive signals.

432 Table 2: Ablation study of different signal smoothing methods on Qwen3-4B-Thinking-2507. The
 433 highest Acc values are marked in bold for the early-exit methods.
 434

435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485	Method	MATH500		AIME		HUMANEVAL+		GPQA-D		Average	
		Acc↑	Tok↓	Acc↑	Tok↓	Acc↑	Tok↓	Acc↑	Tok↓	Acc↑	Tok↓
	Vanilla	96.60	6719	86.67	19577	88.40	5133	68.69	9041	85.09	10118
	NoSmooth	94.20	3608	73.33	17832	92.10	2789	62.12	4066	80.44	7074
	Momentum	91.80	2230	60.00	12427	83.50	2219	64.65	3406	74.99	5071
	Sliding Window	95.20	4444	80.00	19184	86.60	4342	62.12	4738	80.98	8177
	Paragraph Mean	95.40	4285	76.67	18231	84.80	4569	65.66	4726	80.63	7953
	SpecExit* (EWMA)	96.80	4777	90.00	17769	89.60	4319	68.69	7011	86.27	8469

Step Split Tokens. To evaluate the influence of different step split strategies on early-exit performance, we conducted ablation experiments comparing paragraph delimiters, general discourse markers, and a contrastive subset of discourse markers. Discourse markers indicate semantic transitions or reasoning shifts, but are dependent on the underlying data and model, limiting their generality. In prior work on dynamic early-exit methods, contrastive subsets of discourse markers (e.g., “Wait”, “But”, “Alternatively”) are frequently used to capture reasoning-relevant transitions. In contrast, paragraph delimiters (`\n\n`) provide a more general segmentation that does not rely on model-specific or dataset-specific patterns. As shown in Figure 6, using paragraph delimiters achieves competitive accuracy and token reduction, demonstrating that a general segmentation strategy can be effective for early-exit decisions while maintaining coherence in reasoning traces.



477 Figure 6: Ablation study of step split tokens strategies on Qwen3-4B-Thinking-2507.
 478
 479

480 In summary, our ablation studies on signal types, smoothing strategies, and step split methods pro-
 481 vide key insights for improving early-exit decision-making. The integration of multiple signals
 482 strikes the best balance between accuracy and token efficiency, while appropriate smoothing meth-
 483 ods stabilize cognitive signals and enhance the consistency of early exits. Additionally, using general
 484 segmentation strategies, such as paragraph delimiters, improves the generalizability of early-exit
 485 systems across diverse datasets. These findings emphasize the importance of a holistic approach,
 where complementary strategies jointly enhance both efficiency and reliability.

486 5 RELATED WORK

488 **Efficient Reasoning.** To mitigate unnecessary CoT generation in LRM (Chen et al., 2025; Sui et al.,
 489 2025), prior work has explored both training-based and inference-time strategies. Training-based
 490 approaches typically modify model behavior through reinforcement learning with length-sensitive
 491 objectives (Aggarwal & Welleck, 2025; Yeo et al., 2025; Shen et al., 2025) or supervised fine-tuning
 492 on reasoning traces of varying lengths (Ma et al., 2025b; Munkhbat et al., 2025). While effective in
 493 shortening outputs, these methods demand substantial retraining cost and can distort the model’s out-
 494 put distribution, raising concerns about reliability and generalization to unseen tasks. Inference-time
 495 methods avoid retraining and instead attempt to stop reasoning dynamically by monitoring model
 496 signals such as logits (Yang et al., 2025) or intermediate answers (Fu et al., 2024). Although these
 497 methods show that early stopping can reduce reasoning length without degrading accuracy, their
 498 reliance on probing introduces additional computation and often emphasizes token count reduction
 499 rather than true end-to-end latency improvements.

500 **Speculative Decoding and Hidden States.** Speculative decoding (Chen et al., 2023; Leviathan
 501 et al., 2023) is a widely adopted technique for accelerating decoding speed, where a lightweight
 502 draft model proposes candidate tokens that a larger target model verifies in a single pass. Recent
 503 methods (Cai et al., 2024; Li et al., 2024a;b; 2025; Zhang et al., b) leverage hidden states to predict
 504 multiple future tokens. Beyond speculative decoding, several studies (Lin et al.; Zhang et al., c;
 505 Dong et al.; Zhang et al., a), have revealed that hidden states contain broader information about
 506 future outputs, including correctness, response length, and reasoning paths. Building on this insight,
 507 our method extends speculative decoding by training hidden states not only to forecast future tokens
 508 but also to produce an early-exit signal.

509 6 CONCLUSION

511 In this work, we propose **SpecExit**, a reasoning-aware early-exit framework that leverages latent
 512 signals from models’ hidden states to dynamically terminate reasoning processes in LRM. By con-
 513 catenating auxiliary prediction heads to a lightweight draft model, SpecExit simultaneously predicts
 514 future tokens and early-exit signals in a single forward pass, eliminating the probing overhead re-
 515 quired by previous approaches. Our experiments across diverse tasks and models demonstrate that
 516 SpecExit substantially reduces reasoning length by up to 66% and achieves significant end-to-end
 517 latency improvements up to 2.5x without compromising accuracy. The proposed method highlights
 518 the potential of hidden states as informative signals for efficient reasoning and establishes a practical
 519 pathway for deploying LRM in real-world scenarios.

521 7 ETHICS STATEMENT

523 This research does not involve human subjects, sensitive personal data, or applications with foresee-
 524 able negative societal impact. All datasets mentioned are publicly available, and proper licenses and
 525 usage guidelines are respected.

528 8 REPRODUCIBILITY STATEMENT

530 We have taken extensive measures to ensure the reproducibility of our results. The code used to
 531 implement our proposed framework, SpecExit, is publicly available at: <https://anonymous.4open.science/r/SpecExit-B802>. Detailed instructions on how to use and run the code,
 532 including environment setup and dependency installation, are provided in the repository. For the
 533 experiments conducted in this paper, we used publicly available benchmark datasets and models.

536 REFERENCES

538 Marah Abdin, Sahaj Agarwal, Ahmed Awadallah, Vidhisha Balachandran, Harkirat Behl, Lingjiao
 539 Chen, Gustavo de Rosa, Suriya Gunasekar, Mojan Javaheripi, Neel Joshi, et al. Phi-4-reasoning
 technical report. *arXiv preprint arXiv:2504.21318*, 2025.

540 Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with
 541 reinforcement learning. *arXiv preprint arXiv:2503.04697*, 2025.

542

543 Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and
 544 Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling.
 545 URL <http://arxiv.org/abs/2407.21787>. version: 1.

546

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

550

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

554

555 Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuwei Liu,
 556 Mengfei Zhou, Zhusong Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Do not
 557 think that much for 2+3=? on the overthinking of o1-like llms, 2025. URL <https://arxiv.org/abs/2412.21187>.

558

559 Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and
 560 Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge.
 561 *arXiv preprint arXiv:1803.05457*, 2018.

562

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

566

567 DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu,
 568 Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu,
 569 Zhibin Gou, Zhihong Shao, Zhusong Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao
 570 Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan,
 571 Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao,
 572 Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding,
 573 Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang
 574 Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai
 575 Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang,
 576 Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang,
 577 Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang,
 578 Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang,
 579 R. J. Chen, R. L. Jin, Ruyi Chen, Shanghai Lu, Shangyan Zhou, Shanhua Chen, Shengfeng
 580 Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing
 581 Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen
 582 Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong
 583 Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu,
 584 Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xi-
 585 aoshua Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia
 586 Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng
 587 Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong
 588 Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong,
 589 Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou,
 590 Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying
 591 Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda
 592 Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu,
 593 Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu
 594 Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforce-
 595 ment learning, 2025. URL <https://arxiv.org/abs/2501.12948>.

596

597 Zhichen Dong, Zhanhui Zhou, Zhixuan Liu, Chao Yang, and Chaochao Lu. Emergent response
 598 planning in LLMs. URL <http://arxiv.org/abs/2502.06258>.

594 Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Aurick Qiao, and Hao Zhang.
 595 Efficiently serving llm reasoning programs with certainindex. *arXiv e-prints*, pp. arXiv-2412, 2024.
 596

597 Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song,
 598 and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021.
 599 URL <https://arxiv.org/abs/2103.03874>.

600 Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph
 601 Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language
 602 model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Sys-
 603 tems Principles*, SOSP '23, pp. 611–626, New York, NY, USA, 2023. Association for Com-
 604 puting Machinery. ISBN 9798400702297. doi: 10.1145/3600006.3613165. URL <https://doi.org/10.1145/3600006.3613165>.

606 Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative
 607 decoding. In *International Conference on Machine Learning*, pp. 19274–19286. PMLR, 2023.

608

609 Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE: Speculative sampling requires
 610 rethinking feature uncertainty. In *International Conference on Machine Learning*, 2024a.

611 Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-2: Faster inference of lan-
 612 guage models with dynamic draft trees. In *Empirical Methods in Natural Language Processing*,
 613 2024b.

614

615 Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-3: Scaling up inference
 616 acceleration of large language models via training-time test, 2025. URL <https://arxiv.org/abs/2503.01840>.

618 Zhengkai Lin, Zhihang Fu, Ze Chen, Chao Chen, Liang Xie, Wenxiao Wang, Deng Cai, Zheng
 619 Wang, and Jieping Ye. Controlling thinking speed in reasoning models. URL <http://arxiv.org/abs/2507.03704>. version: 1.

621

622 Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chat-
 623 gpt really correct? rigorous evaluation of large language models for code generation. *Advances
 624 in Neural Information Processing Systems*, 36:21558–21572, 2023.

625 Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. Reasoning
 626 models can be effective without thinking. *arXiv preprint arXiv:2504.09858*, 2025a.

627

628 Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. Cot-valve: Length-
 629 compressible chain-of-thought tuning. *arXiv preprint arXiv:2502.09601*, 2025b.

630 MAA Committees. Aime problems and solutions. [https://artofproblemsolving.com/
 631 wiki/index.php/AIME_Problems_and_Solutions](https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions).

632

633 Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke
 634 Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. S1: Simple test-time
 635 scaling. URL <http://arxiv.org/abs/2501.19393>. version: 1.

636 Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. Self-
 637 training elicits concise reasoning in large language models. *arXiv preprint arXiv:2502.20122*,
 638 2025.

639

640 OpenAI. Learning to reason with llms, September 2024. URL [https://openai.com/index/learning-to-reason-with-llms/](https://openai.com/index/

 641 learning-to-reason-with-llms/).

642 Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan
 643 Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang,
 644 Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin
 645 Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li,
 646 Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang,
 647 Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025.
 URL <https://arxiv.org/abs/2412.15115>.

648 David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien
 649 Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a
 650 benchmark, 2023. URL <https://arxiv.org/abs/2311.12022>.

651

652 Yi Shen, Jian Zhang, Jieyun Huang, Shuming Shi, Wenjing Zhang, Jiangze Yan, Ning Wang, Kai
 653 Wang, and Shiguo Lian. Dast: Difficulty-adaptive slow-thinking for large reasoning models.
 654 *arXiv preprint arXiv:2503.04472*, 2025.

655 Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute opti-
 656 mally can be more effective than scaling model parameters. URL <http://arxiv.org/abs/2408.03314>. version: 1.

657

658 Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu,
 659 Andrew Wen, Shaochen Zhong, Na Zou, Hanjie Chen, and Xia Hu. Stop overthinking: A survey
 660 on efficient reasoning for large language models, 2025. URL <https://arxiv.org/abs/2503.16419>.

661

662 Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, and
 663 Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. URL
 664 <http://arxiv.org/abs/2201.11903>. version: 1.

665

666 Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Zheng Lin, Li Cao, and Weiping
 667 Wang. Dynamic early exit in reasoning models. *arXiv preprint arXiv:2504.15895*, 2025.

668

669 Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. Demystifying long chain-
 670 of-thought reasoning in llms. *arXiv preprint arXiv:2502.03373*, 2025.

671

672 Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning
 673 models know when they're right: Probing hidden states for self-verification. a.

674

675 Lefan Zhang, Xiaodan Wang, Yanhua Huang, and Ruiwen Xu. Learning harmonized representations
 676 for speculative sampling, b. URL <http://arxiv.org/abs/2408.15766>.

677

678 Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, Shuohang Wang, Yelong Shen,
 679 and Xin Eric Wang. Soft thinking: Unlocking the reasoning potential of LLMs in continuous
 680 concept space, c. URL <http://arxiv.org/abs/2505.15778>.

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702 **A APPENDIX**
 703

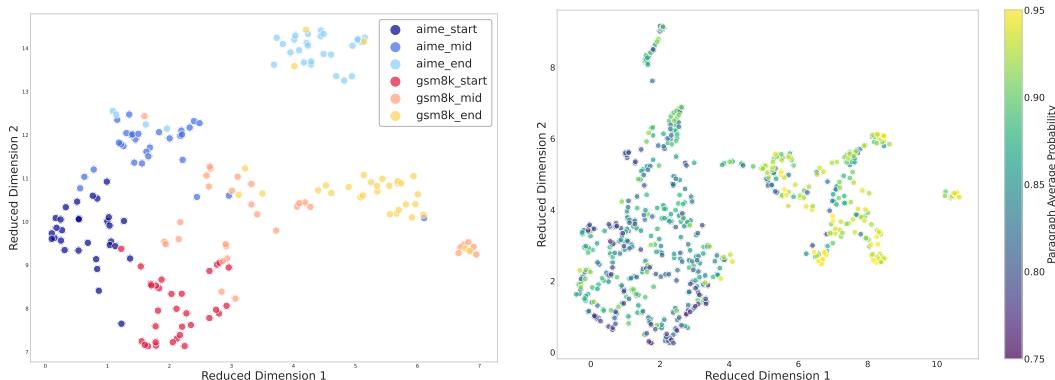
704 This section may contain supplementary materials such as additional experimental details, ablation
 705 studies, hyperparameter settings, and qualitative examples of generated CoT sequences.
 706

707 **A.1 USAGE OF LLMs**
 708

709 We used LLMs(ChatGPT, Gemini) for grammar reviews and style polishing.
 710

711 **A.2 SUPPLEMENTARY EXPERIMENT**
 712

713 **Analysis of Hidden States:** To investigate whether intermediate hidden representations encode dis-
 714 criminative signals relevant to reasoning sufficiency, we analyze hidden states extracted from mul-
 715 tiple depth stages of the reasoning traces. We apply Principal Component Analysis (PCA) and
 716 Uniform Manifold Approximation and Projection (UMAP) to project the hidden states into lower-
 717 dimensional spaces and analyze their spatial distribution. The analysis reveals that, across datasets
 718 of varying difficulty, the hidden states at the “start,” “mid,” and “end” positions exhibit clear clus-
 719 tering patterns. Moreover, the hidden states at the start of the reasoning trace show notable cross-
 720 dataset similarity, as illustrated in Figure 7a. In addition, when examining the relationship between
 721 the embedding representations of hidden states and the paragraph geometric mean of probabilities,
 722 we observe that hidden states also form meaningful clusters under different probability ranges, as
 723 illustrated in Figure 7b. These observations suggest that hidden representations indeed capture infor-
 724 mation related to dataset difficulty, reasoning progress, paragraph-level average probability, thereby
 725 providing preliminary evidence that functions of intermediate hidden states can serve as reliable
 726 proxies for reasoning sufficiency.
 727



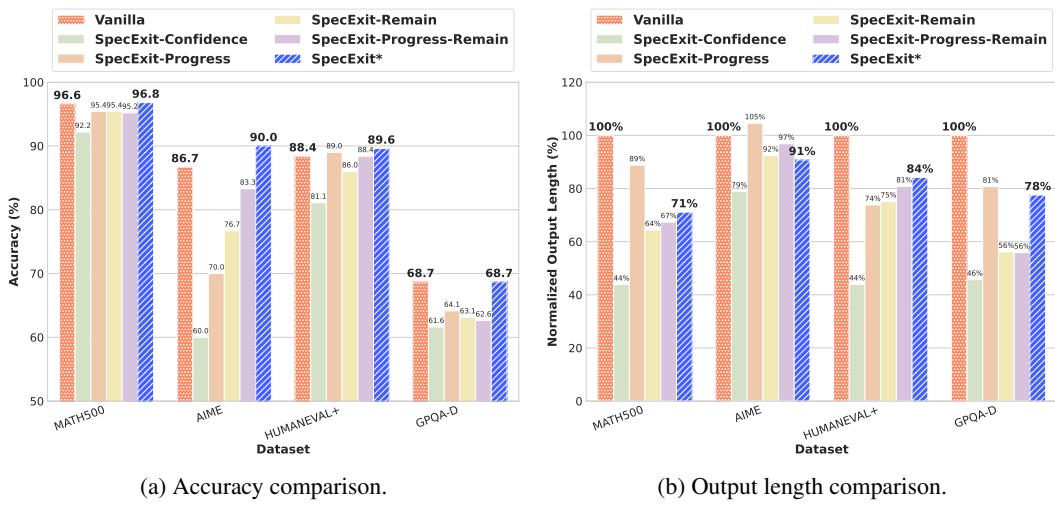
728 (a) Distribution of hidden states on the GSM8K and
 729 AIME datasets. The labels “start,” “mid,” and “end”
 730 correspond to early, intermediate, and final positions
 731 along the reasoning trace, respectively.
 732

733 (b) Relationship between the embedding represen-
 734 tations of hidden states and the paragraph geometric mean
 735 of probabilities, showing clustering patterns under dif-
 736 ferent confidence ranges.
 737

738 Figure 7: Analysis of hidden-state representations on the GSM8K and AIME datasets. (a) shows the
 739 distribution of hidden states along different reasoning positions; (b) shows the relationship between
 740 hidden states and paragraph-level confidence signals.
 741

742 **Signal Type:** In Figure 8, we conduct a systematic comparison of different early-exit signal config-
 743 urations, including the full combination of three signals (confidence, reasoning progress, and remain-
 744 ing reasoning length) and the reduced configuration using only “progress + remaining.” The results
 745 show that relying solely on progress and remaining signals can reduce the reasoning length to some
 746 extent, but this setting consistently underperforms the full three-signals configuration in accuracy,
 747 especially on datasets involving longer reasoning chains or higher task complexity. In contrast, the
 748 complete three-signals design exhibits more stable behavior across datasets, effectively shortening
 749 the output length while maintaining a more favorable accuracy-efficiency trade-off. Overall, these
 750 findings demonstrate that incorporating the confidence signal is essential for constructing stable and
 751

756 generalizable early-exit strategies, enabling more consistent efficiency gains and reliable correctness
 757 compared with the “progress + remaining” setting.
 758



774
 775 Figure 8: Ablation study of SpecExit signal types on Qwen3-4B-Thinking-2507, adding experiment
 776 with only Progress and Remaining signals.
 777

778 Threshold Calibration:

779 In the ablation study of SpecExit signal types, the following thresholds are applied as stopping
 780 conditions for the respective signal types: SpecExit-Confidence requires predicted confidence value
 781 greater than 0.9, SpecExit-Progress requires predicted progress value greater than 0.8, SpecExit-
 782 Remain requires predicted remaining reasoning length value less than 100, and SpecExit* combines
 783 thresholds with a predicted confidence value greater than 0.8, predicted progress greater than 0.3,
 784 and predicted remaining reasoning length less than 200.

785 In addition to the thresholds reported above, we provide two complementary procedures that fur-
 786 ther calibrate the stopping criteria using a small held-out calibration set. Specifically, we sample 90
 787 instances of varying difficulty from the validation split of training data and conduct the following
 788 analyses:

789 (1) Statistical distribution-based thresholding. During data construction, we have access to the short-
 790 est valid reasoning path for each problem, and thus we can determine whether stopping at the end
 791 of any intermediate paragraph would still yield a correct final answer. By examining the empirical
 792 distribution of the predicted signals at these paragraph boundaries and correlating them with
 793 correctness, we obtain the distributions shown in Figure 10. These distributions allow us to derive a
 794 signal-specific threshold that maximizes the retention of correct answers under early stopping.

795 (2) Design space exploration over the threshold search space. We additionally perform a design
 796 space exploration (DSE) over a predefined grid of confidence, progress, and remaining-length
 797 thresholds. Using vLLM to run this evaluation pipeline on the 90-sample calibration set, the full
 798 search requires approximately 2.5 hours on an 8×H20 GPU cluster. Among all threshold combina-
 799 tions, we select those lying on the Pareto frontier that best trade off accuracy preservation against
 800 reduction in reasoning length. The results are shown in Figure 10.

803 A.3 IMPLEMENTATION DETAILS

804
 805 **Model Architectures:** We evaluate SpecExit on three large reasoning models (LRMs): **Qwen3-**
 806 **4B-Thinking-2507** (Qwen et al., 2025), **DeepSeek-R1-Distilled-Llama-8B** (DeepSeek-AI et al.,
 807 2025) and **Phi-4-reasoning** (Abdin et al., 2025). For speculative decoding, the draft models adopt
 808 the EAGLE3 (Li et al., 2025) architecture. The draft models are single-layer causal models whose
 809 hidden sizes match those of the corresponding target models. The input embedding layer of each
 draft model is shared with its corresponding target model to ensure tokenizer compatibility, while the

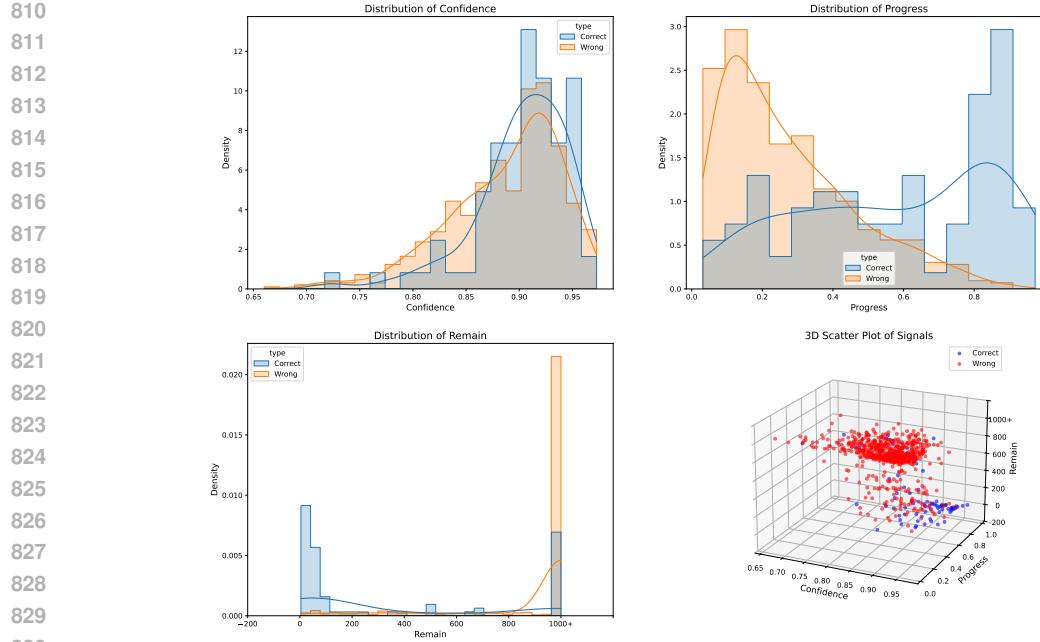


Figure 9: Distribution of signals with correct and wrong answers in calibration data.

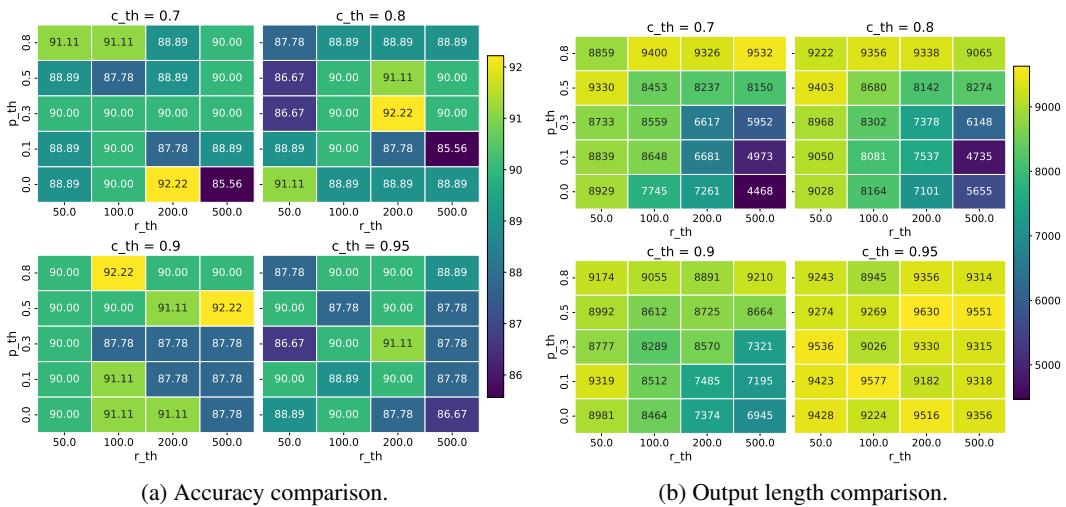


Figure 10: Accuracy and output length comparison in the calibration data under different threshold settings. The parameters c_{th} , p_{th} , and r_{th} denote the thresholds for confidence, progress, and remaining reasoning length estimation, respectively.

output head uses a compact vocabulary of 32k high-frequency tokens. The detailed configurations are summarized in Table 3.

Figure 11 provides an overview of how SpecExit integrates with the EAGLE3 speculative decoding pipeline. During inference, the draft model produces multi-level token predictions, and simultaneously, our early-exit module tracks the evolution of three reasoning-related signals to determine whether the target model can safely terminate the thinking phase. The design is architecture-agnostic and can be combined with other multi-token-prediction frameworks such as Medusa. In practical deployment, the fully-connected layer of the MTP head (e.g., EAGLE3 or Medusa) can be fused with the early-exit module, effectively hiding additional operator invocation latency.

Table 3: Target and Draft Model Configurations.

Model Name	Role	Architecture	Hidden Size	Layers	Vocab Size
Qwen3-4B-Thinking-2507	Target	Qwen3ForCausalLM	2560	36	151936
	Draft	Eagle3LlamaForCausalLM	2560	1	32000
DeepSeek-R1-Distilled-Llama-8B	Target	LlamaForCausalLM	4096	32	128256
	Draft	Eagle3LlamaForCausalLM	4096	1	32000

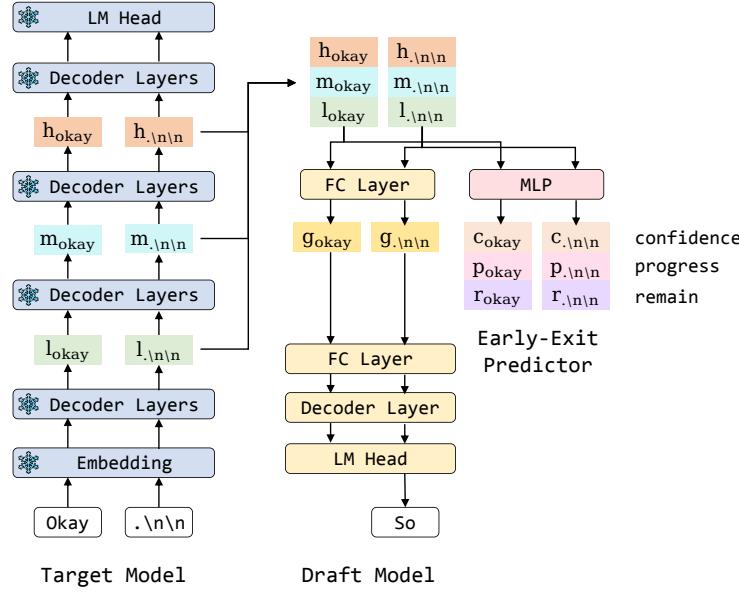


Figure 11: Diagram of the speculative decoding pipeline integrating EAGLE3 (Li et al., 2025) with the proposed reasoning early-exit mechanism.

Signal Smoothing: In the ablation study of signal smoothing strategies, the following methods are implemented to stabilize cognitive signals for early-exit decisions:

- Sliding Window: The sliding window approach smooths the signal by averaging the last N predicted signal values, with N set to 10. The mean score x_t at decoding step t is computed as:

$$x_t = \text{Mean}(s_t, N) = \frac{1}{N} \sum_{i=t-N+1}^t s_i, \quad (7)$$

where s_i denotes the predicted signal value at decoding step i .

- Momentum-based Prediction: This method predicts the next score based on the momentum, which is calculated as the difference between N consecutive signal values, with N set to 10. The predicted score x_t at decoding step t is given by:

$$x_t = \text{Predict}(s_t, N) = s_{t-1} + \frac{1}{N-1} \sum_{i=t-N+1}^{t-1} (s_i - s_{i-1}). \quad (8)$$

- Paragraph Mean: In this approach, the score x_t is calculated as the average of all predicted signal values within the current paragraph:

$$x_t = \frac{1}{T} \sum_{i=1}^T s_i, \quad (9)$$

where T is the total number of steps in the current paragraph.

- Exponential Weighted Moving Average (EWMA): In this approach, the smoothing factor α is set to 0.1. The new score x_t is updated based on the previous score x_{t-1} and the current signal value s_t as:

$$x_t = \text{EWMA}(s_t, x_{t-1}, \alpha) = \alpha \cdot s_t + (1 - \alpha) \cdot x_{t-1}. \quad (10)$$

Discourse Markers: We collect high-frequency words appearing at the beginning of model-generated sentences as discourse markers. As shown in Figure 12, we present examples of discourse markers extracted from Qwen3-4B-Thinking-2507 on the MATH dataset. Among them, transitional words such as "Wait" and "But" can be regarded as a subset of these high-frequency markers.



Figure 12: Discourse marker distribution in Qwen3-4B-Thinking-2507’s responses on the MATH500 (Hendrycks et al., 2021) dataset.

A.4 PREDICTED SIGNAL VISUALIZATION

As shown in Figure 13, for simple problems, the predicted confidence remains high, the predicted remaining reasoning length is relatively short, and the predicted progress rises rapidly within the first few sentences, with only minor drops on a few uncertain words. This indicates that for such problems, the model is able to establish a stable reasoning trace at an early stage. In contrast, as shown in Figure 14, for complicated problems, the model also exhibits high confidence and a short remaining reasoning length in the initial summarization phase, but once it enters the detailed analysis stage, the predicted remaining reasoning length increases significantly, confidence drops. Meanwhile, the predicted progress starts low and rises initially, but then fluctuates markedly during repeated self-reflection, making it difficult to stabilize at a high threshold.

These observations reveal the inherent limitations of relying on individual signals for early exiting. When depending solely on confidence, the model often exhibits overconfidence and terminates too early before sufficient reasoning has been completed, leading to substantial accuracy degradation. When depending solely on the predicted remaining reasoning length, the model may become overly optimistic in the early stages of complicated problems, resulting in premature exits before essential reasoning steps are accomplished. When depending solely on progress, the signal tends to fluctuate and remain unstable in complex reasoning tasks, making it difficult to trigger an appropriate early exit and thereby restricting achievable speedup. In summary, each single signal suffers from the inability to balance accuracy and efficiency across diverse problem types. By integrating multiple signals in a complementary manner, the model can achieve a smoother trade-off between reasoning accuracy and inference acceleration.

Figure 15 illustrates the effect of applying an Exponentially Weighted Moving Average (EWMA) to stabilize predicted signal. EWMA effectively suppresses local noise while preserving the global trend, leading to smoother traces for signal. The smoothed signals reveal clearer overall convergence patterns, providing a more stable and reliable basis for threshold-based early-exit decisions and thereby improving robustness in challenging reasoning scenarios.

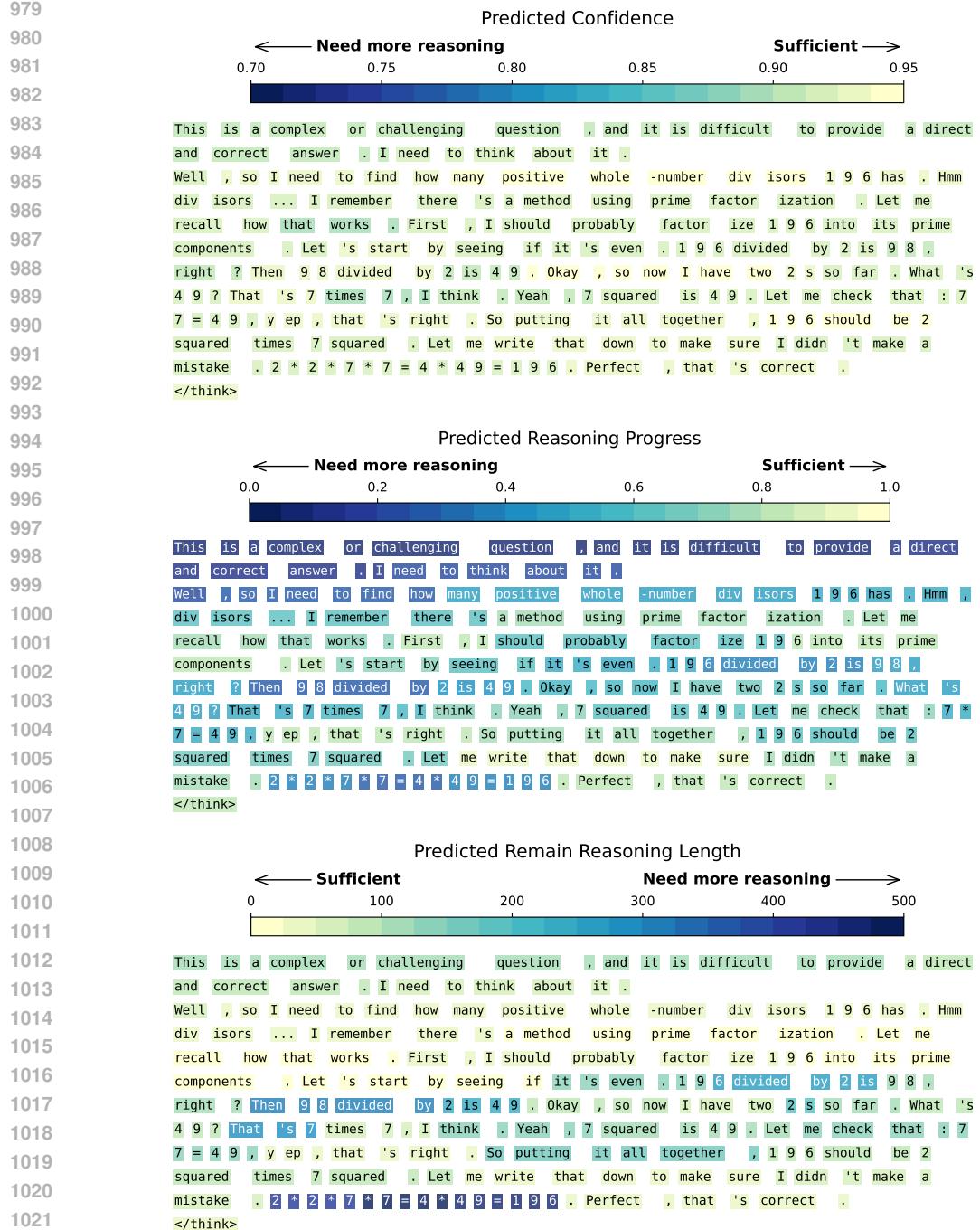


Figure 13: Visualization of reasoning signals for a simple problem, illustrated with an example from the MATH500 (Hendrycks et al., 2021) dataset, where darker colors denote insufficient reasoning and lighter colors denote sufficiency.

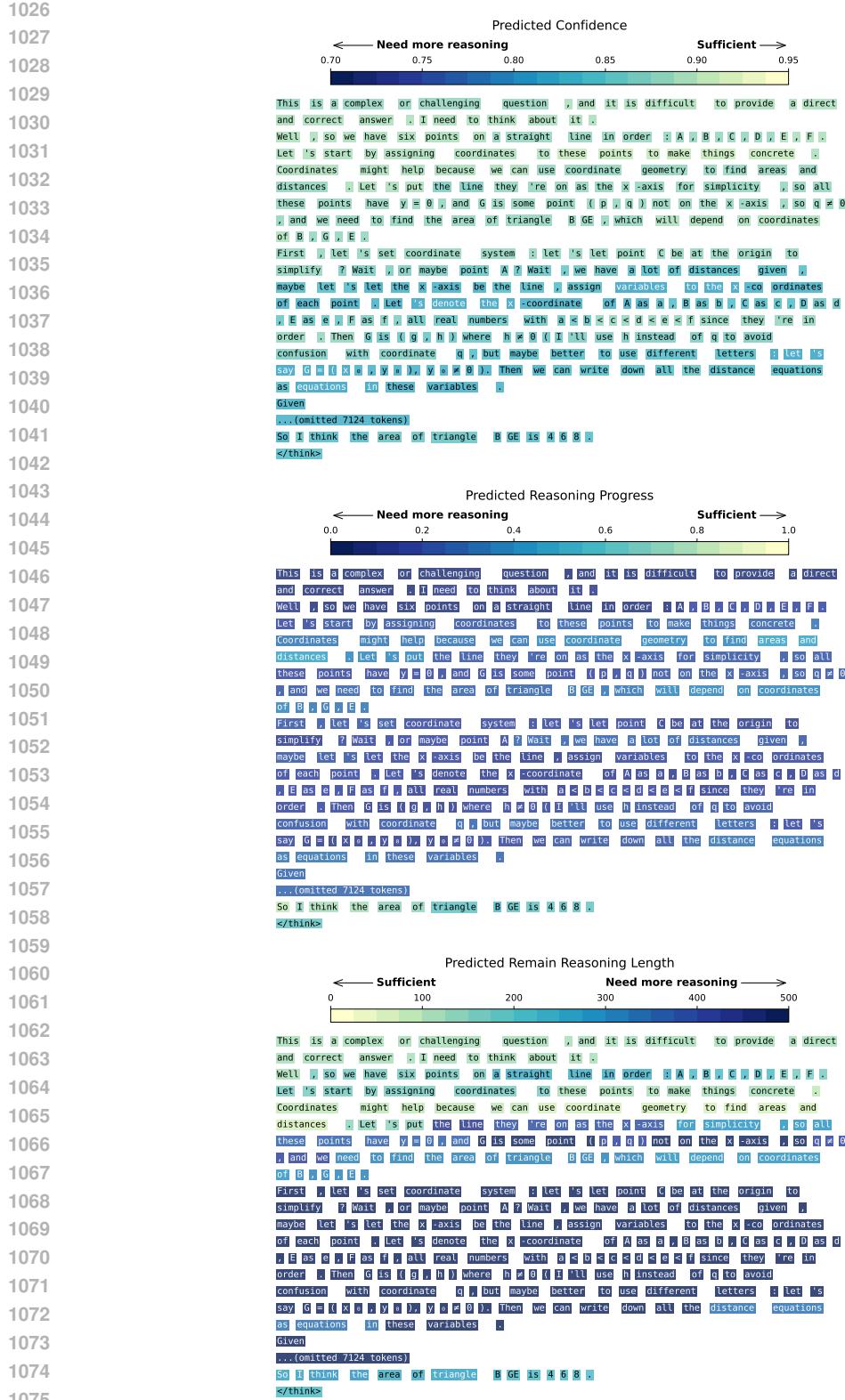


Figure 14: Visualization of reasoning signals for a complicated problem, illustrated with an example from the AIME (MAA Committees) dataset, where darker colors denote insufficient reasoning and lighter colors denote sufficiency.

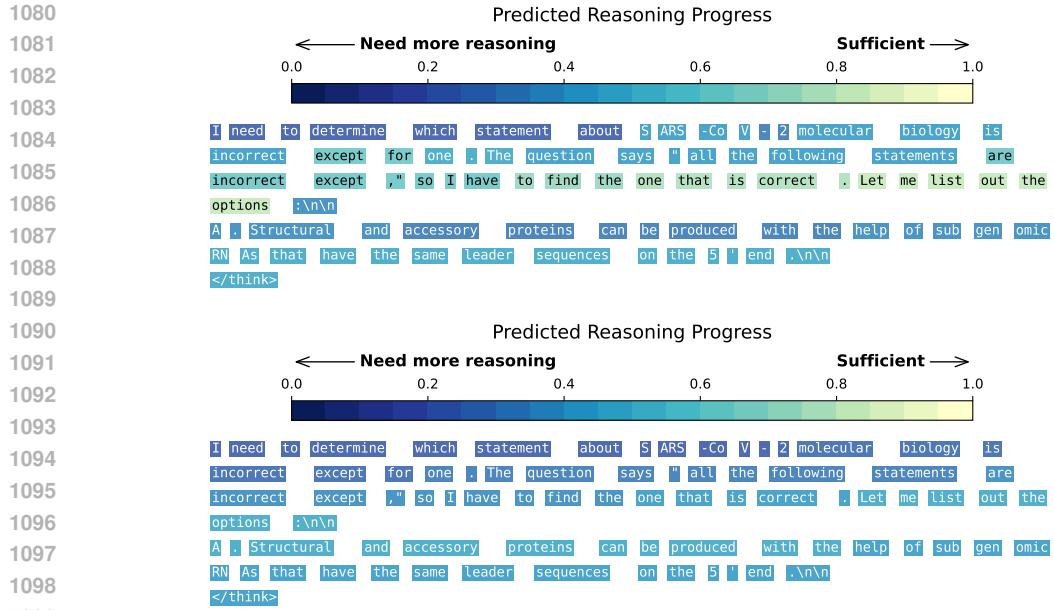


Figure 15: Visualization of the smoothing effect introduced by the EWMA-based signal estimator, illustrated with an example from the GPQA Diamond (Rein et al., 2023) dataset, where the stabilized trace reveals reduced high-frequency fluctuations and clearer monotonic trends in the underlying reasoning signals.

A.5 CASE STUDY DETAILS

Figure 16 presents an example from the GSM8K (Cobbe et al., 2021) dataset, where SpecExit is applied after an initial analysis. In this case, SpecExit inserts a decision to exit reasoning based on the signal magnitude after completing the first paragraph, thus preventing the continuation of redundant reasoning tokens. The process begins with the model evaluating the initial segment of the problem, analyzing the available context and producing intermediate reasoning steps. When SpecExit detects that all the signals exceeds a predefined threshold, it triggers the early exit mechanism. As a result, the model halts further reasoning and transitions directly to the answering phase, thus producing the correct result in a more efficient manner.

By shortening the unnecessary token generation and reducing redundant reasoning, SpecExit leads to significant performance improvements in terms of inference speed while maintaining the correctness of the answer. This early exit strategy not only accelerates the inference process but also avoids the potential confusion that the model might experience from excessive reflection or unnecessary computations. This case highlights the effectiveness of integrating early exit mechanisms in enhancing both the efficiency and reliability of large language model inference.

1134

1135

1136

1137

Question:

1138

1139

1140

Ram uses a lot of pens. He discovered that he can save money by mixing the ink from five empty pens to make one full pen. If he buys 25 pens and then uses them to make new pens when the ink runs low, how many total pens does he get to have?

1141

1142

1143

1144

1145

SpecExit: (421 tokens)

Okay, let's try to figure out how many sterling silver beads Katerina uses per necklace. Hmm, first, let's list out what we know.
She uses 8 topaz gemstones per necklace. Each topaz is 1 inch long, so the total length of the topaz gemstones would be 8 times 1 inch, right? So that's 8 inches.
</think>

Signals: confidence=0.94 progress=41% remain_tokens=56 ✓

1146

1147

1148

1149

To determine how many sterling silver beads Katerina uses per necklace... (omitted)

Final Answer
\$\boxed{68}\$

Katerina uses **68** sterling silver beads per necklace. **correct ✓**

1150

1151

Vanilla: (1521 tokens)**Initial**

1152

1153

1154

Okay, let's try to figure out how many sterling silver beads Katerina uses per necklace. Hmm, first, let's list out what we know.
She uses 8 topaz gemstones per necklace. Each topaz is 1 inch long, so the total length of the topaz gemstones would be 8 times 1 inch, right? So that's 8 inches.

1155

1156

1157

1158

Hmm, let's start by breaking down the problem step by step.
&

Analysis

First, the problem says she uses 8 topaz gemstones per necklace, and each topaz is one inch long. So, the total length of the topaz gemstones would be 8 times 1 inch, right? Let me write that down: 8 topazes * 1 inch each = 8 inches.
&

Now, the necklace is supposed to be a total length of 25 inches. That means the space between the gemstones, which is filled with sterling silver beads, has to make up the rest of the length. So, the total length from the silver beads would be 25 inches minus the length of the topaz gemstones. Let me calculate that: 25 inches - 8 inches = 17 inches. So, the silver beads together are 17 inches long.

1163

1164

1165

1166

But each sterling silver bead is one-quarter of an inch long... (omitted)

Reflection

Wait, hold on. When you have beads between gemstones, do you need to consider how many gaps there are? Because ... (omitted)

Oh! That's a crucial point I might have missed. The problem says "fills the space between gemstones using sterling silver beads." So, the silver beads are in the spaces between the topaz stones. So, if there are 8 topaz stones, there are 7 spaces between them.
&

Let me re-examine the problem statement... (omitted)

Wait, but the problem says "each necklace is made to a total length of 25 inches". So, total length = length of topaz + length of silver beads.
&

But ... (omitted) (continuous reflection over 26 paragraphs)

1173

1174

1175

1176

1177

1178

1179

1180

1181

1182

So, in the original problem, 8 topaz stones (8 inches), total necklace 25 inches, so 17 inches of silver beads. $17 / 0.25 = 68$.
&

Summary

I think that's the answer. The key is that the problem says "fills the space between gemstones using sterling silver beads", so the total length of the beads is the total necklace length minus the topaz length, and then divide by the bead length.
&

I think the confusion might be about the number of gaps, but since the problem doesn't specify that each gap has a certain number of beads, we just take the total length of the beads and divide by the length per bead.
&

So, the answer should be 68.

</think>

1183

1184

1185

1186

1187

Figure 16: Discourse marker distribution in Qwen3-4B-Thinking-2507's responses on the MATH500 (Hendrycks et al., 2021) dataset.