

DySD: Dynamic Speculative Decoding for Evolving LLMs under Rapid Distributional Shifts

Xinglin Liu¹, Bohao Qu^{3,*}, Chenyu Li¹, Yongjie Wang⁴, Qing Guo⁵, Ying Wang^{1,2,*}

¹College of Computer Science and Technology, Jilin University, China

²Key Laboratory of Symbol Computation and Knowledge Engineering (Jilin University), China

³CFAR and IAIC, Agency for Science, Technology and Research (A*STAR), Singapore

⁴Alibaba-NTU Global e-Sustainability CorpLab, Nanyang Technological University, Singapore

⁵VCIP, School of Computer Science, Nankai University, China.

*Corresponding author: Qu_Bohao@a-star.edu.sg, wangying2010@jlu.edu.cn

Abstract

Speculative decoding accelerates large language model (LLM) inference by allowing a lightweight draft model to propose candidate tokens for verification by a target model. In real-world deployments, however, shifts in input distributions and updates to the target model can weaken draft–target alignment, reducing token acceptance and acceleration. Repeatedly retraining the draft model is costly, delays deployment, and may impair performance on previously encountered distributions. Achieving online alignment therefore requires rapid and stable adaptation while incorporating real-time feedback within a limited compute budget. To this end, we propose **DySD**, a framework that aligns the draft model with the target model during inference without offline retraining. DySD combines hierarchical feature alignment with distribution-level matching and uses divergence-triggered dynamic distillation to selectively update the draft model. Experiments across multiple datasets show that DySD achieves 1.6×–2.0× speedups and boosts token acceptance rates by 10%–30% under distribution shifts, enabling speculative decoding to remain fast, stable, and adaptive in ever-changing environments.

1 Introduction

Large Language Models (LLMs) (Chang et al., 2024; Bai et al., 2023; Wei et al., 2022; OpenAI, 2023; DeepSeek-AI, 2025; Touvron et al., 2023) have achieved remarkable success, demonstrating strong reasoning capabilities, broad knowledge coverage, and adaptability across diverse domains. Their high inference accuracy and versatility have driven widespread deployment in online services, ranging from e-commerce assistants to enterprise knowledge systems (Li et al., 2024b; An et al., 2024; Peng et al., 2024; Carta et al., 2023). While the one token at a time nature of classical LLMs

creates a speed bottleneck for latency-sensitive applications, particularly in real-time systems, leading to the development of *Speculative Decoding* (Leviathan et al., 2023; Chen et al., 2023), where a smaller draft model proposes multiple candidate tokens that the larger target model verifies in parallel, thereby accelerating inference without sacrificing quality (Xia et al., 2024).

In real-world scenarios, LLM deployment is inherently dynamic. As illustrated in Figure 1(b), to integrate new domain knowledge and align with shifting preferences, practitioners increasingly adopt Continual Learning for high-frequency target model updates (Shi et al., 2025a). In parallel, as shown in Figure 1(a), live online serving faces rapid concept drift and data shifts; real-world systems frequently encounter severe performance degradation due to unpredictable distributional shifts in the underlying data generating process (Rauba et al., 2024). Either phenomenon creates a moving target: the draft model, trained offline on a static dataset, inevitably becomes misaligned with the evolving target model and the inputs it serves. When this misalignment grows, speculative decoding’s benefits erode, token acceptance rates drop, inference efficiency declines, and in the worst cases, inference latency becomes unstable.

A straightforward response is to retrain the draft model whenever the data distribution changes. Yet, this retraining-centric strategy faces fundamental drawbacks. First, retraining incurs prohibitive computational and operational costs, particularly at the cadence required to keep pace with real-world updates. Second, frequent retraining may cause the draft model to overfit to recent data, potentially degrading its performance on previously encountered distributions and compromising long-term stability. Together, these limitations make static retraining unsustainable for online LLM systems, highlighting the need for a fundamentally different approach: *dynamic alignment*.

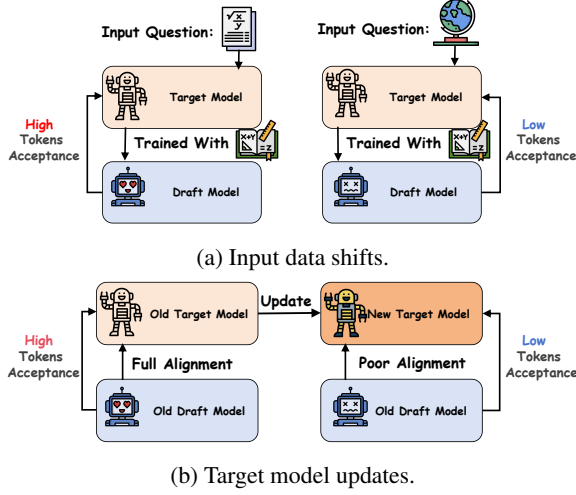


Figure 1: Speculative decoding challenges in dynamic serving environments. (a) Input distributions shift due to evolving user intents (e.g., varying task domains), causing draft model degradation. (b) Target model updates (e.g., version iterations) lead to misalignment as the draft model fails to match the new behavior.

Designing such dynamic alignment is far from trivial and presents three major challenges. ❶ How can we create alignment strategies that quickly and precisely align draft and target models without excessive latency? ❷ How can we dynamically synchronize the behaviors of draft and target models without destabilizing inference? ❸ How can we incorporate real-time feedback efficiently, avoiding prohibitive computational overhead?

To address these challenges, we propose **Dynamic Speculative Decoding (DySD)**, a novel framework that enables real-time draft model alignment without offline retraining. DySD introduces a multi-objective alignment strategy combining hierarchical feature alignment and target alignment. The hierarchical feature alignment leverages the observation that different layers encode distinct linguistic abstractions, establishing correspondence between draft and target models based on relative depth rather than absolute layer matching. The target alignment emphasizes distribution matching over hard label concordance, capturing nuanced decision boundaries. DySD also designs a selective learning mechanism that performs gradient updates solely on incorrect predictions, thereby minimizing computational overhead and resource consumption during online adaptation. Furthermore, DySD incorporates dynamic distillation mechanisms that monitor the output divergence between draft and target models in real-time. Upon detecting signifi-

cant distributional shifts, the system automatically initiates adaptive fine-tuning, seamlessly maintaining alignment throughout the target model’s evolutionary trajectory without requiring computationally expensive offline retraining.

We conducted comprehensive experiments on three widely-used benchmark datasets and one heterogeneous mixed dataset. To thoroughly evaluate DySD’s performance under realistic dynamic conditions, we simulated two prevalent dynamic scenarios commonly encountered in applications: iterative target model updates and continuous input distribution shifts. Experimental results demonstrate that in static scenarios, DySD achieves a remarkable 1.6× to 2.0× speedup in target model decoding compared to static speculative decoding baselines, while significantly improving the token acceptance rate by 10% to 30%. In dynamic scenarios, DySD exhibits rapid alignment capabilities, seamlessly achieving and even surpassing the token acceptance rates of static alignment strategies within a short time window. Furthermore, ablation studies validate the effectiveness of our hierarchical feature alignment and dynamic monitoring strategy. Consequently, DySD proves to be a scalable speculative decoding solution for rapidly evolving environments, ensuring efficient, stable, and adaptable LLM inference.

Our key contributions are as follows:

- We identify the challenge of speculative decoding under dynamic deployment conditions. We analyze how input distribution shifts and target model drift cause draft model misalignment, degrading speed and stability in real-world LLM deployment scenarios.
- We propose DySD, a framework for real-time draft model alignment without offline retraining. DySD enables the draft model to adapt on the fly, avoiding repeated offline retraining while enabling continual adaptation to changing distributions.
- We design a multi-objective alignment strategy with hierarchical feature alignment and distribution matching. This strategy aligns draft and target model representations at multiple abstraction levels while ensuring stable behavior across evolving distributions.
- Through comprehensive experiments across diverse tasks and model architectures, we

demonstrate that DySD achieves substantial inference speedups (up to 1.6×) compared to existing speculative decoding methods particularly in scenarios where data distribution shift over time.

2 Related Work

2.1 Speculative Decoding

Research on speculative decoding (Leviathan et al., 2023) mainly advances three aspects: drafting, verification, and alignment. Drafting methods include learned architectures, such as multi-head prediction in Medusa (Cai et al., 2024) and feature-level autoregression in EAGLE (Li et al., 2024c); multi-token prediction (MTP) (Gloeckle et al., 2024), as subsequently adopted by DeepSeek-V3 (DeepSeek-AI, 2024); retrieval-based drafting, such as NEST (Li et al., 2024a); target-assisted drafting, such as GliDe with CaPE (Du et al., 2024); and self-speculative early exiting, as in Kangaroo (Liu et al., 2024a) and LayerSkip (Elhoushi et al., 2024). Verification methods improve candidate-tree construction and parallel verification, as in Sequoia (Chen et al., 2024) and SpecExec (Svirschevski et al., 2024), or extend speculative verification to beam decoding (Qin et al., 2025). Regarding online alignment, OSD (Liu et al., 2024b) and industrial systems such as ATLAS (Wang et al., 2025) explore runtime draft adaptation. Unlike target-integrated MTP modules, DySD employs an external draft model without modifying the target model and combines hierarchical feature alignment with divergence-triggered selective updates under evolving distributions.

2.2 Knowledge Distillation for LLMs

Knowledge distillation (KD) (Buciluă et al., 2006; Gu et al., 2024) has garnered significant attention as an effective technique for reducing computational demands of LLMs (Gu et al., 2025; Hou et al., 2025; Takrouri et al., 2025; Qiao et al., 2024). MCKD distills student models from few-shot prompted LLMs, achieving superior generalization in semi-supervised tasks (Zhao et al., 2024); GKD (Tan et al., 2023) framework enables various distillation methods for extremely large models (exceeding 10B parameters) on limited GPU resources; MiniLLM (Gu et al., 2024) introduced Reverse Kullback-Leibler divergence to replace forward divergence, preventing overestimation of low-probability regions; In speculative decoding,

DistillSpec (Zhou et al., 2024) pioneered the integration of KD techniques and explored different alignment strategies, while OSD (Liu et al., 2024b) proposes online distillation to update draft models. Existing KD methods for Speculative Decoding primarily focus on output-level alignment without deeply exploring hidden layer features. Building on this foundation, this paper proposes a multi-objective distillation approach to align target and draft models in speculative decoding.

3 Problem Formulation

Speculative decoding addresses computational inefficiencies in autoregressive LLMs by overcoming sequential token generation limitations. This technique employs a smaller draft model to predict multiple candidate tokens that the target LLM verifies in parallel, significantly improving inference speed without compromising output quality. The speculative decoding process can be formalized as follows: Given a target large language model Q and a smaller draft model P_θ , the process begins with an input prompt x of dataset $\mathcal{X}$. The draft model generates a set of candidate tokens of length K , $T = \{t_1, t_2, \dots, t_K\} = P_\theta(x)$ based on x . These tokens are then passed to the target model Q for parallel verification, determining which tokens should be retained in the final output. This process is iteratively repeated until a complete response is generated. Since speculative decoding relies on a small draft model for efficient token generation, we formulate the optimization objective as:

$$\theta^* = \arg \min_{\theta} \mathbb{E}_{x \sim \mathcal{X}} \left[\sum_{i=1}^K \Pr(t_i^Q \neq t_i^{P_\theta} \mid x_{<i}) \right]. \quad (1)$$

where K represents the length of the candidate token sequence. The draft model optimized according to the aforementioned objective can efficiently align with the target model’s outputs. However, the speculative decoding procedure described above assumes that the target model remains static, whereas in practical scenarios, the parameters and input distribution of the target model may vary in response to changing inputs. Consequently, under the assumption that the target model Q updates at time step j , the optimization objective for speculative decoding can be formulated as:

$$\theta_j^* = \arg \min_{\theta} \mathbb{E}_{x \sim \mathcal{X}_j} \left[\sum_{i=1}^K \Pr(t_i^{Q_j} \neq t_i^{P_\theta} \mid x_{<i}) \right]. \quad (2)$$

where Q_j and $\mathcal{X}_j$ denote the target-model state and input distribution at time step j , respectively. Developing innovative speculative decoding approaches to handle dynamic state transitions in target models remains a critical and underexplored research challenge that warrants deeper investigation.

4 Dynamic Speculative Decoding

As illustrated in Figure 2, DySD is built upon three core design principles: **Logits Distillation**, **Hierarchical Feature Alignment**, and **Dynamic Knowledge Distillation**. Specifically, the logits-based loss focuses on ensuring consistency between the output probability distributions of the target and draft models; the hidden layer feature alignment aims to align the intermediate representations between the target and draft models; and the dynamic knowledge distillation strategy dynamically monitors the distributional divergence between the target and draft model outputs to regulate the update frequency of the draft model.

4.1 Target Alignment via Logits Distillation

To enhance the alignment between the draft model’s output and the target model, improve the token acceptance rate of the draft model. We leverage a teacher–student learning framework grounded in knowledge distillation to enable knowledge transfer. Yet, conventional distillation in speculative decoding remains constrained by its reliance on hard-label alignment, which overlooks the nuanced probabilistic reasoning embedded in the target model. DySD overcomes this limitation with a distribution-centric soft alignment strategy, redefining how draft models internalize and adapt the knowledge of their target counterparts.

Our key insight is that speculative decoding’s effectiveness hinges not on exact token matching, but on the draft model’s ability to faithfully reproduce the target’s decision boundaries and uncertainty patterns. Thus, rather than utilizing the teacher model’s final outputs as hard targets for training the draft model, we opt to use the teacher model’s logits as soft training targets. Specifically, we define the logits loss function as follows:

$$\mathcal{L}_{\text{logits}} = D(\sigma(z^Q; \tau), \sigma(z^{P_\theta}; \tau)). \quad (3)$$

where z^{P_θ} and z^Q denote the logits of the draft and target models respectively, $\sigma(\cdot)$ represents the softmax function, and τ is a non-negative temperature hyperparameter, $D(\cdot, \cdot)$ denotes a divergence

measure function that quantifies the discrepancy between two class distributions. As evidenced by the formulation, the value range characteristics of the loss function are primarily determined by the choice of probability distribution divergence metric D . We detail various divergence metric strategies in Appendix A.1.

The logits-based distillation strategy provides an effective foundation for aligning draft and target models through distribution matching, it primarily focuses on the final output layer representations. However, to achieve efficient alignment, the draft model must accurately emulate not just the target’s outputs but also its underlying reasoning process, which is introduced in Section 4.2.

4.2 Hierarchical Feature Alignment Strategy

To improve alignment efficiency beyond merely matching final outputs, DySD builds on the observation that intermediate representations contain rich and transferable knowledge beyond output logits (Shi et al., 2025b). Hidden-state alignment has also been explored in knowledge distillation through stage-wise matching across teacher and student models of different scales. As documented by (Liu et al., 2023), this methodology systematically explores the alignment of distinct hierarchical layers across teacher-student models of varying scales. Similarly, we employ a feature alignment strategy through layer-wise feature matching to align the hidden layer features between the target model and the draft model:

$$\mathcal{L}_{\text{feat}} = D_{\text{feat}}(\phi_Q^l(x), \phi_{P_\theta}^l(x)), \quad (4)$$

where ϕ_Q^l and $\phi_{P_\theta}^l$ represent the features of the l -th layer of the target model and the draft model, respectively. Direct feature alignment in speculative decoding is hindered by architectural discrepancies between the compact draft and extensive target models, where fixed layer correspondences fail to capture equivalent semantic depth. To resolve this, we propose a parametric hierarchical alignment strategy. Instead of assuming a rigid layer-to-layer correspondence, we partition the model depth into three functional stages: representing lexical, syntactic, and semantic/reasoning processing. Formally, we employ a parametric mapping function $\mathcal{M}(l; d_1, d_2)$ that assigns each layer l with relative

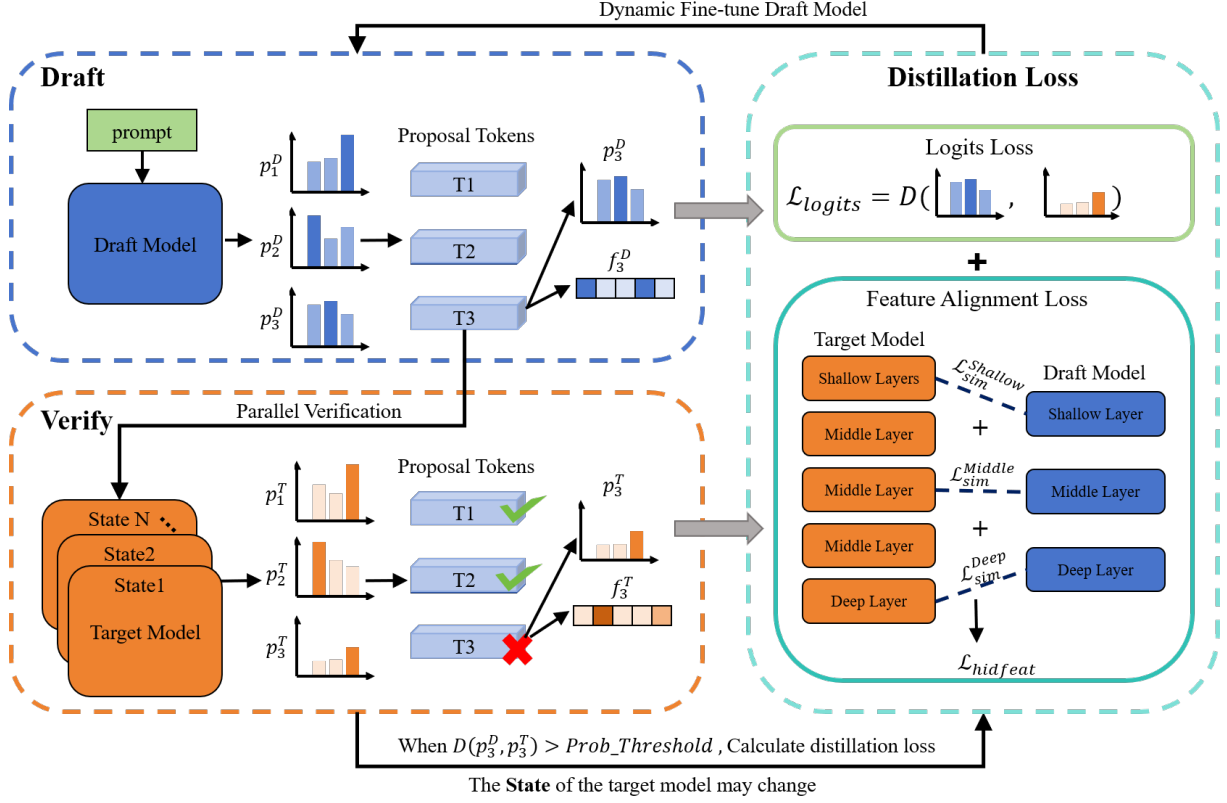


Figure 2: Overview of Dynamic Speculative Decoding: Draft model first predicts K candidate tokens (set to 3 in the figure), followed by parallel verification from the target large language model. When the draft model’s predictions are incorrect and the divergence between the output distributions of both models exceeds a predetermined threshold, our dynamic knowledge distillation mechanism is triggered to update the draft model. Through optimization of a multi-objective loss function comprising logits loss and feature alignment loss, we achieve dynamic alignment between the draft and target models, thereby enhancing the efficiency of speculative decoding.

depth r_l to a semantic stage:

$$\mathcal{M}(l) = \begin{cases} \text{Shallow Layer,} & \text{if } r_l \leq d_1 \\ \text{Middle Layer,} & \text{if } d_1 < r_l \leq d_2 \\ \text{Deep Layer,} & \text{if } r_l > d_2 \end{cases} \quad (5)$$

To align the intermediate representations, we first resolve the dimensional mismatch between the target and draft models. Since their feature dimensions d_Q and d_{P_θ} generally differ, we project the draft features into the target feature space using a learnable linear transformation:

$$\hat{\phi}_{P_\theta}^i(x) = \mathbf{W}\phi_{P_\theta}^i(x) + \mathbf{b}, \quad (6)$$

where $\phi_{P_\theta}^i(x)$ is the i -th layer feature of the draft model, and $\mathbf{W} \in \mathbb{R}^{d_Q \times d_{P_\theta}}$ and $\mathbf{b} \in \mathbb{R}^{d_Q}$ are learnable projection parameters. Rather than being fixed, $\mathbf{W}$ and $\mathbf{b}$ are jointly optimized with the draft model during online distillation, while the target model remains frozen. This allows the projection to adapt to changes in the target model or input distribution. The complete set of trainable parameters is

therefore

$$\Theta = \{\theta, \mathbf{W}, \mathbf{b}\}. \quad (7)$$

We then quantify the discrepancy between the corresponding layers using the squared L_2 norm:

$$\mathcal{L}_{sim}(j, i) = \left\| \phi_Q^j(x) - \hat{\phi}_{P_\theta}^i(x) \right\|_2^2, \quad (8)$$

where $\phi_Q^j(x)$ is the feature of the j -th layer of the target model. Finally, the total hierarchical feature alignment loss is computed by averaging the similarity losses over the set of aligned layer pairs $\mathcal{H}$:

$$\mathcal{L}_{hidfeat} = \frac{1}{|\mathcal{H}|} \sum_{(j,i) \in \mathcal{H}} \mathcal{L}_{sim}(j, i), \quad (9)$$

where, $\mathcal{H} = \{(j, i)\}$ represents the set of matched layer pairs determined by our relative depth mapping strategy. Finally, we integrate logits distillation loss with feature alignment loss to develop a multi-objective knowledge distillation strategy that optimizes the draft model for speculative decoding:

$$\mathcal{L} = \mathcal{L}_{logits} + \lambda \mathcal{L}_{hidfeat}, \quad (10)$$

where λ is feature alignment loss weight hyperparameter. By optimizing this multi-objective loss function, DySD facilitates more efficient alignment between draft model and target model, thereby enhancing the efficiency of speculative decoding.

Through multi-level feature matching and output-layer knowledge distillation, more efficient knowledge transfer is achieved compared to traditional approaches, offering an effective solution for model alignment in static scenarios. However, the current method is still limited by the assumption of static target models and cannot handle the challenges of continuously evolving target models and dynamic data distribution shifts in practical applications. Therefore, extending existing multi-objective alignment strategies to dynamic environments, enabling them to adapt to model updates and distribution drift.

4.3 Dynamic Knowledge Distillation

Although static knowledge distillation effectively aligns target and draft models in speculative decoding, enhancing efficiency, this alignment deteriorates under data distribution shifts or target model update, degrading performance. To address the prohibitive computational costs of continuous retraining, we propose a dynamic speculative decoding framework implementing conditional dynamic knowledge distillation during decoding.

DySD’s dynamic distillation relies on two critical components: the **erroneous tokens set** $\mathcal{R}$ and **update threshold** η . DySD monitors the divergence between target and draft models; when it exceeds η , distillation is triggered. This process computes losses specifically for rejected tokens, creating a focused learning signal. These elements jointly enable real-time adaptation without compromising efficiency.

Erroneous Tokens Set: DySD focuses computational resources exclusively on prediction errors. Crucially, the error buffer size ($|\mathcal{R}|$) is not a fixed hyperparameter but a dynamic variable. Because the set is only cleared following a complete parameter update cycle, this design creates an adaptive batching mechanism inversely proportional to the update frequency. During mild distributional drift, updates are relatively sparse, allowing the buffer to naturally accumulate a richer error history; this effectively increases the update batch size, enabling robust optimization and preventing overfitting to noise. Conversely, under severe drift, updates are triggered immediately for low-latency rapid correc-

tion, optimizing the adaptation rate dynamically to the current context.

Update Threshold: Dynamic knowledge distillation monitors the discrepancy between the target and draft models using a configurable distributional divergence:

$$d_i = D(Q(\cdot | x_i), P_\theta(\cdot | x_i)), \quad (11)$$

where $D(\cdot, \cdot)$ can be instantiated as KL divergence, reverse KL divergence, Jensen–Shannon divergence, or total variation distance. When d_i exceeds a predetermined threshold η , DySD triggers knowledge distillation to align the output distributions and intermediate representations of both models according to Equation 10. This mechanism enables timely adaptation to changes in the target model or input distribution while avoiding unnecessary updates. Since different divergence measures have different numerical ranges, η is selected according to the chosen metric. Smaller thresholds trigger more frequent updates and faster adaptation, whereas larger thresholds reduce computational overhead at the cost of slower recovery. The theoretical relationship between η , update frequency, and divergence reduction is analyzed in Appendix A.2.

Output Distribution Preservation. Although online distillation changes the draft-model parameters, it does not alter the distribution of the final generated output. DySD preserves the standard speculative decoding procedure: the frozen target model verifies every proposed token, and rejected tokens are corrected through the corresponding rejection-sampling mechanism. Therefore, under the standard verification rule, the resulting output distribution remains identical to that of the target model. Dynamic adaptation affects the quality of the draft proposals and the token acceptance rate, but it does not modify the target distribution. The draft-model parameters remain fixed throughout each proposal–verification block and are updated only after the current verification step is completed.

5 Experiments

Model, Datasets and Metrics: We conduct extensive experiments on **Vicuna** (Chiang et al., 2023) with **7B** and **13B** parameters as target models, while employing **LLaMA-160M** (Miao et al., 2023) as the draft model. Its lightweight architecture ensures substantially lower drafting costs than target-model verification, preserving the effi-

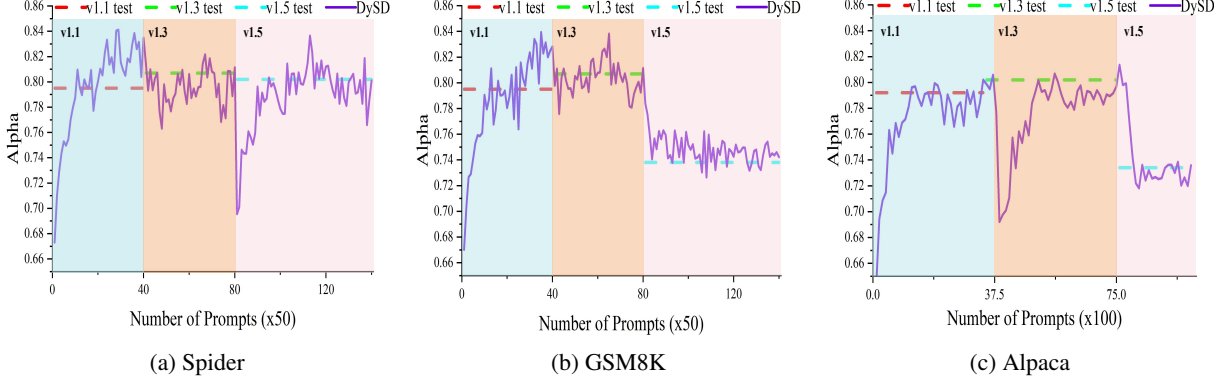


Figure 3: Experimental results of target model updates across different datasets. α is calculated as the average value over 50/100 prompts.

ciency benefits of speculative decoding. To evaluate DySD’s performance across diverse task domains, we utilize established benchmark datasets: **Spider** (Yu et al., 2018) (complex text-to-SQL tasks), **GSM8K** (Cobbe et al., 2021) (mathematical reasoning problems), and **Alpaca** (Bharti, 2023) (financial instruction-following tasks). We employ two evaluation metrics (Leviathan et al., 2023): (1) **Model speedup ratio**, which quantifies inference acceleration compared to conventional autoregressive decoding; and (2) **Token acceptance rate** α , which measures the percentage of draft model-generated tokens accepted by the target model. We employ **DistillSpec** (Zhou et al., 2024) as the static distillation baseline and **OSD** (Liu et al., 2024b) as the dynamic scenario baseline. Additional comparisons with Token Recycling (Luo et al., 2025) under dynamic distribution shifts and EAGLE (Li et al., 2024c) under matched full-data pretraining are reported in Appendices A.6 and A.7, respectively.

Experimental Setup: We freeze the active target-model checkpoint within each evaluation stage and optimize only the draft-side parameters $\Theta = \{\theta, W, b\}$. The LLaMA-160M draft model is initialized from its original pretrained checkpoint without task-specific fine-tuning and updated continuously throughout each dynamic evaluation sequence. Dynamic scenario updates are trained for a single epoch. We set depth partition hyperparameters $d_1 = 0.25$, $d_2 = 0.75$, and maintain a decoding temperature of 1.0. Detailed configurations and hardware specifications are provided in Appendix A.4.

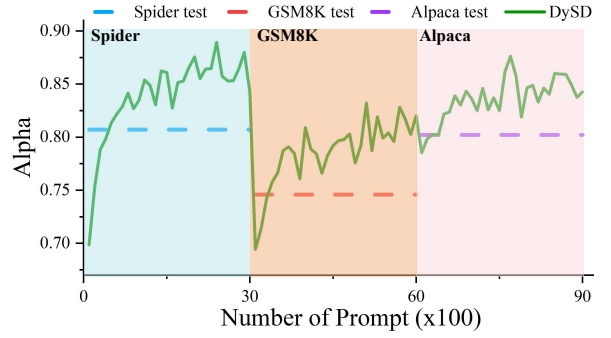


Figure 4: Experimental results on Mix Dataset

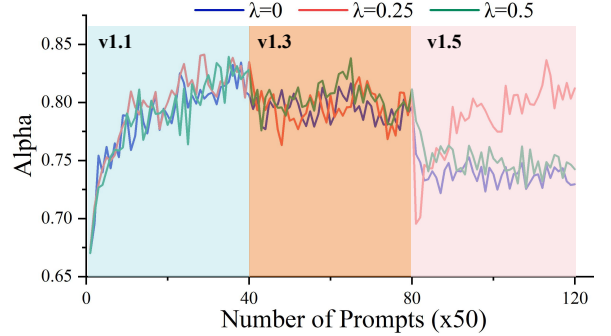


Figure 5: Feature alignment weights analysis on Spider.

5.1 DySD’s Dynamic Adaptation Capabilities

In this subsection, we experimentally validate the performance of DySD under dynamic conditions. We design experiments to simulate two distinct types of distribution shift scenarios commonly observed in real-world platforms: 1) **Target Model Update** 2) **Input Distribution Shifts**.

Target Model Update. We simulate model update using Vicuna-7B v1.1/v1.3/v1.5 across Spider, GSM8K, and Alpaca. As shown in Figure 3, target model updates induce immediate Alpha value drops during online adaptation, revealing draft models’ adaptation struggles. This triggers

DySD’s dynamic distillation to initiate fine-tuning. Within limited iterations, draft models achieve target model alignment, α value rapidly surpassed the baseline algorithm that uses 10% of training data for static knowledge distillation.

② Input Distribution Shift. Figure 4 presents the results of our data distribution shift experiment. To simulate a realistic dynamic stream where user intent shifts over time, we construct a heterogeneous stream where requests transition from Coding (Spider) to Reasoning (GSM8K) and then to Financial Instruction-Following (Alpaca). This mimics a multi-purpose LLM agent serving diverse user needs sequentially. Experimental results show that while distribution shifts initially degrade draft model efficiency, DySD mitigates this via dynamic continuous optimization. Consequently, DySD gradually recovers and eventually matches or exceeds static distillation benchmarks.

5.2 Feature Alignment Experiments

To evaluate the effectiveness of our proposed feature alignment strategy, we conducted experiments under both dynamic and static scenarios. The dynamic scenario used the same experimental setup as the target model update study. For the static scenario, we performed static distillation using a fixed 25% of the training data.

① Dynamic Scenario Analysis. Figure 5 confirms the efficacy of the proposed feature alignment loss, which consistently outperforms the baseline. Notably, $\lambda = 0.25$ yields superior results compared to $\lambda = 0.5$, suggesting that excessive weights induce overfitting to the current target, thereby hindering adaptability to subsequent updates.

② Static Scenario Analysis. As shown in Figure 7, feature alignment loss consistently improves performance in static settings. However, due to scale discrepancies between feature metrics and KL-divergence, efficacy does not increase monotonically with weight magnitude. While disproportionately high weights risk prioritizing feature matching over target alignment, the method demonstrates robust stability and effectiveness across most calibrations.

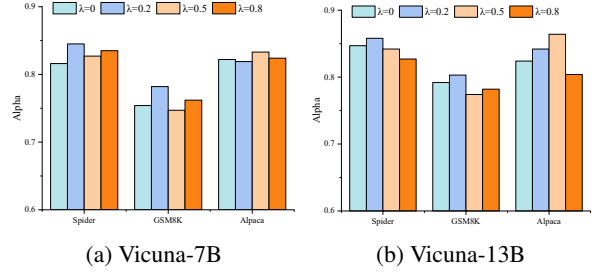


Figure 7: Feature alignment weight analysis.

5.3 Acceleration Performance Experiments

Method	Spider		GSM8K		Alpaca	
	α	Speedup	α	Speedup	α	Speedup
V7B-10%	0.807	1.35	0.746	1.25	0.839	1.35
V7B-25%	0.816	1.52	0.754	1.32	0.822	1.46
V13B-10%	0.834	1.42	0.789	1.34	0.831	1.42
V13B-25%	0.847	1.56	0.792	1.47	0.824	1.57
DySD + V7B-25%	0.845	1.89	0.803	1.72	0.833	1.67
DySD + V13B-25%	0.858	2.01	0.794	1.59	0.852	1.68

Table 1: Experimental results on static scenario. Percentages indicate the proportion of training data used for static distillation.

① Dynamic Scenario Analysis: Figure 6 compares DySD against static and OSD baselines on a continuous data stream. DySD outperforms OSD with significantly faster convergence and superior stability. This advantage stems from our multi-objective strategy, which integrates hierarchical feature alignment to learn target representations more efficiently than OSD’s logit-only approach. Furthermore, DySD’s threshold-controlled mechanism minimizes computational overhead by avoiding continuous updates.

② Static Scenario Analysis. Table 1 shows that DySD outperforms the single-objective baseline in static scenarios. This highlights two advantages: 1) By training only on error tokens when divergence exceeds the threshold, DySD matches full-batch performance with reduced computational cost; and 2) The results validate the effectiveness of our dual-objective distillation strategy.

5.4 Update Threshold Analysis

The update threshold η controls the frequency of online distillation, determining the trade-off between adaptation responsiveness and computational overhead. To examine its influence, we evaluate $\eta \in \{0.07, 0.08, 0.09\}$ across different tasks and target-model scales. In this sensitivity analysis, we instantiate the configurable divergence measure D as Jensen–Shannon divergence.

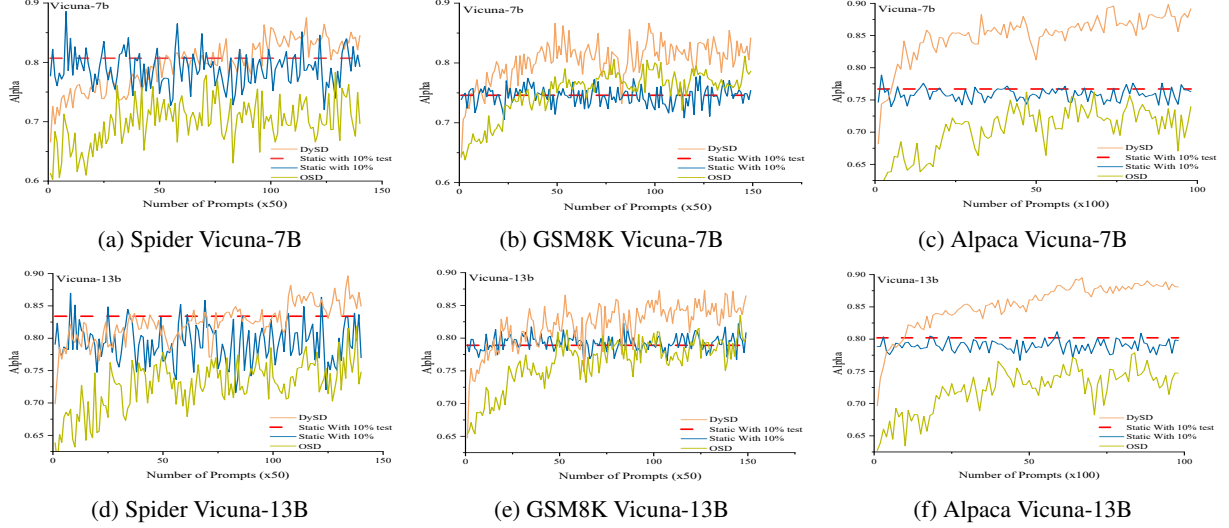


Figure 6: Experimental results under dynamic scenario. α is calculated as the average value over 50/100 prompts.

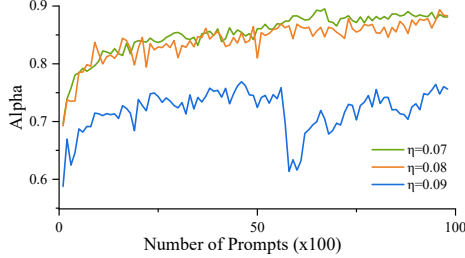


Figure 8: Dynamic adaptation performance under different update thresholds on Alpaca.

Figure 8 presents the dynamic adaptation results on Alpaca. Larger thresholds reduce update frequency, allowing discrepancies between the draft and target models to persist and resulting in greater fluctuations in token acceptance. Conversely, smaller thresholds enable more timely adaptation and maintain a more stable acceptance rate under distribution shifts.

Table 2: Average token acceptance rate α under different update thresholds across tasks and target-model scales.

η	Spider	GSM8K	Alpaca
LLaMA-160M / Vicuna-7B			
0.07	0.834	0.799	0.835
0.08	0.839	0.786	0.836
0.09	0.837	0.783	0.823
LLaMA-160M / Vicuna-13B			
0.07	0.831	0.802	0.827
0.08	0.833	0.788	0.822
0.09	0.798	0.779	0.808

Table 2 further reports the average token acceptance rate across three tasks and two target-model scales. For both Vicuna-7B and Vicuna-13B, $\eta = 0.07$ yields the highest average acceptance rate, although $\eta = 0.08$ performs slightly better on Spider and on Alpaca with Vicuna-7B. This suggests that the optimal threshold varies with the task distribution.

Increasing the threshold to 0.09 generally reduces performance, with a more pronounced decline for Vicuna-13B. For example, its acceptance rate on Spider drops from 0.831 to 0.798, indicating that delayed updates can substantially weaken adaptation under distribution shifts. Overall, relatively small thresholds improve alignment stability, whereas larger thresholds reduce update frequency at the cost of slower recovery.

6 Conclusion

We propose Dynamic Speculative Decoding (DySD) to sustain inference efficiency in dynamic deployment environments characterized by shifting user intents and frequent model updates. By integrating hierarchical feature alignment with dynamic on-the-fly distillation, DySD reduces the need for repeated offline retraining and supports adaptation under distribution shifts. Experiments across diverse shift scenarios demonstrate 1.6 $\times$ –2.0 $\times$ speedups and 10%–30% improvements in token acceptance rates, validating DySD as a robust solution for low-latency LLM inference in evolving domains.

Limitations

DySD introduces additional computation and memory for online adaptation, although the lightweight draft model and divergence-triggered sparse updates keep the measured overhead modest, as reported in Appendix A.5. Since energy consumption depends strongly on hardware, workload, and update frequency, we leave direct energy and lifecycle carbon evaluation to future work. Under abrupt or persistent distribution shifts, the token acceptance rate may temporarily decrease and trigger more frequent updates before the draft model is realigned; nevertheless, target-model verification preserves the correctness of the final output distribution. In addition, the current fixed threshold provides predictable control over adaptation cost but may require calibration across tasks and divergence measures. Adaptive threshold selection based on recent divergence statistics and serving budgets is therefore a promising extension. Finally, our normalized-depth partition provides a simple architecture-agnostic default across the evaluated model scales, while soft or learnable layer correspondence may further improve alignment between more heterogeneous architectures.

Ethical Considerations

DySD accelerates inference without modifying the target model or bypassing its verification procedure. Because online adaptation may process user traffic, practical deployments should protect sensitive data and monitor biased or adversarial inputs. Under standard speculative verification, such inputs may affect drafting efficiency but not the target model’s output distribution. DySD does not eliminate biases or safety risks inherited from the target model; therefore, existing safeguards should remain in place.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (No. 62272191), the Suzhou Science and Technology Plan Project (No.SYG2025096), and the National Research Foundation, Singapore, under its National Large Language Models Funding Initiative (AISG Award No. AISG-NMLP-2024-004). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

References

- Keyu An, Qian Chen, Chong Deng, Zhihao Du, Changfeng Gao, Zhifu Gao, Yue Gu, Ting He, Hangrui Hu, Kai Hu, Shengpeng Ji, Yabin Li, Zerui Li, Heng Lu, Haoneng Luo, Xiang Lv, Bin Ma, Ziyang Ma, Chongjia Ni, and 14 others. 2024. *Funaudiollm: Voice understanding and generation foundation models for natural interaction between humans and llms*. *Preprint*, arXiv:2407.04051.
- Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, and 29 others. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.
- Gaurang Bharti. 2023. *gbharti/finance-alpaca*. <https://huggingface.co/datasets/gbharti/finance-alpaca>. Accessed: 2025-1-25.
- Cristian Bucilua, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.
- Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. 2024. *Medusa: Simple LLM inference acceleration framework with multiple decoding heads*. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 5209–5235. PMLR.
- Thomas Carta, Clément Romain, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. 2023. Grounding large language models in interactive environments with online reinforcement learning. In *International Conference on Machine Learning*, pages 3676–3713. PMLR.
- Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2024. *A survey on evaluation of large language models*. *ACM Trans. Intell. Syst. Technol.*, 15(3).
- Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*.
- Zhuoming Chen, Avner May, Ruslan Svirschevski, Yuh-sun Huang, Max Ryabinin, Zhihao Jia, and Beidi Chen. 2024. *Sequoia: Scalable and robust speculative decoding*. In *Advances in Neural Information Processing Systems*, volume 37, pages 129531–129563. Curran Associates, Inc.
- Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion

- Stoica, and Eric P. Xing. 2023. [Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality](#).
- 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. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.
- DeepSeek-AI. 2024. [Deepseek-v3 technical report](#). Preprint, arXiv:2412.19437.
- DeepSeek-AI. 2025. [Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning](#). Preprint, arXiv:2501.12948.
- Cunxiao Du, Jing Jiang, Xu Yuanchen, Jiawei Wu, Sicheng Yu, Yongqi Li, Shenggui Li, Kai Xu, Liqiang Nie, Zhaopeng Tu, and Yang You. 2024. [GliDe with a CaPE: A low-hassle method to accelerate speculative decoding](#). In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 11704–11720. PMLR.
- Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, Ahmed Aly, Beidi Chen, and Carole-Jean Wu. 2024. [LayerSkip: Enabling early exit inference and self-speculative decoding](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12622–12642, Bangkok, Thailand. Association for Computational Linguistics.
- Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Roziere, David Lopez-Paz, and Gabriel Synnaeve. 2024. [Better & faster large language models via multi-token prediction](#). In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 15706–15734. PMLR.
- Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. [MiniLLM: Knowledge distillation of large language models](#). In *The Twelfth International Conference on Learning Representations*.
- Yuxian Gu, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. 2025. [MiniPLM: Knowledge distillation for pre-training language models](#). In *The Thirteenth International Conference on Learning Representations*.
- Wenlong Hou, Weidong Zhao, Ning Jia, and Xianhui Liu. 2025. [Low-resource knowledge graph completion based on knowledge distillation driven by large language models](#). *Appl. Soft Comput.*, 169:112622.
- Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR.
- Minghan Li, Xilun Chen, Ari Holtzman, Beidi Chen, Jimmy Lin, Wen-tau Yih, and Xi Victoria Lin. 2024a. [Nearest neighbor speculative decoding for llm generation and attribution](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 80987–81015. Curran Associates, Inc.
- Yangning Li, Shirong Ma, Xiaobin Wang, Shen Huang, Chengyue Jiang, Hai-Tao Zheng, Pengjun Xie, Fei Huang, and Yong Jiang. 2024b. [Ecomgpt: Instruction-tuning large language models with chain-of-task tasks for e-commerce](#). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18582–18590.
- Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024c. [EAGLE: Speculative sampling requires rethinking feature uncertainty](#). In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 28935–28948. PMLR.
- Dongqin Liu, Wentao Li, Wei Zhou, Zhaoxing Li, Jiao Dai, Jizhong Han, Ruixuan Li, and Songlin Hu. 2023. [Semantic stage-wise learning for knowledge distillation](#). In *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pages 816–821.
- Fangcheng Liu, Yehui Tang, Zhenhua Liu, Yunsheng Ni, Duyu Tang, Kai Han, and Yunhe Wang. 2024a. [Kangaroo: Lossless self-speculative decoding for accelerating llms via double early exiting](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 11946–11965. Curran Associates, Inc.
- Xiaoxuan Liu, Lanxiang Hu, Peter Bailis, Alvin Cheung, Zhijie Deng, Ion Stoica, and Hao Zhang. 2024b. [Online speculative decoding](#). In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 31131–31146. PMLR.
- Xianzhen Luo, Yixuan Wang, Qingfu Zhu, Zhiming Zhang, Xuanyu Zhang, Qing Yang, and Dongliang Xu. 2025. [Turning trash into treasure: Accelerating inference of large language models with token recycling](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6816–6831, Vienna, Austria. Association for Computational Linguistics.
- Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Rae Ying Yee Wong, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. 2023. [Specinfer: Accelerating generative llm serving with speculative inference and token tree verification](#). Preprint, arXiv:2305.09781.
- OpenAI. 2023. [Gpt-4 technical report](#).
- Bo Peng, Xinyi Ling, Zirui Chen, Huan Sun, and Xia Ning. 2024. [ecellm: Generalizing large language models for e-commerce from large-scale, high-quality instruction data](#). *arXiv preprint arXiv:2402.08831*.

- Shanbao Qiao, Xuebing Liu, and Seung-Hoon Na. 2024. [DistillMIKE: Editing distillation of massive in-context knowledge editing in large language models](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7639–7654, Bangkok, Thailand. Association for Computational Linguistics.
- Zongyue Qin, Zifan He, Neha Prakriya, Jason Cong, and Yizhou Sun. 2025. Dynamic-width speculative beam decoding for llm inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25056–25064.
- Paulius Rauba, Nabeel Seedat, Krzysztof Kacprzyk, and Mihaela van der Schaar. 2024. Self-healing machine learning: A framework for autonomous adaptation in real-world environments. *Advances in Neural Information Processing Systems*, 37:42225–42267.
- Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyan Wang, Yibin Wang, Zifeng Wang, Sayna Ebrahimi, and Hao Wang. 2025a. Continual learning of large language models: A comprehensive survey. *ACM Computing Surveys*, 58(5):1–42.
- Ruxue Shi, Hengrui Gu, Hangting Ye, Yiwei Dai, Xu Shen, and Xin Wang. 2025b. [Latte: Transferring llms’ latent-level knowledge for few-shot tabular learning](#). In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pages 6173–6181. International Joint Conferences on Artificial Intelligence Organization. Main Track.
- Ruslan Svirschevski, Avner May, Zhuoming Chen, Beidi Chen, Zihao Jia, and Max Ryabinin. 2024. [Specexec: Massively parallel speculative decoding for interactive llm inference on consumer devices](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 16342–16368. Curran Associates, Inc.
- Mohannad Takrouri, Nicolas M. Cuadrado, and Martin Takác. 2025. [Knowledge distillation from large language models for household energy modeling](#). *CoRR*, abs/2502.03034.
- Shicheng Tan, Weng Lam Tam, Yuanchun Wang, Wenwen Gong, Shu Zhao, Peng Zhang, and Jie Tang. 2023. [GKD: A general knowledge distillation framework for large-scale pre-trained language model](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 134–148, Toronto, Canada. Association for Computational Linguistics.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. [Llama 2: Open foundation and fine-tuned chat models](#). *Preprint*, arXiv:2307.09288.
- Junxiong Wang, Shirley Wu, Zelei Shao, Vikranth Srivatsa, Jue Wang, Roy Yuan, Qingyang Wu, Alpay Ariyak, Rupert Wu, Wai Tong Chung, Chenfeng Xu, Yonatan Oren, Pragaash Ponnusamy, Yineng Zhang, Avner May, Leon Song, Tri Dao, Percy Liang, Ce Zhang, and Ben Athiwaratkun. 2025. Adaptive-learning speculator system (atlas): A new paradigm in llm inference via runtime-learning accelerators. <https://www.together.ai/blog/adaptive-learning-speculator-system-atlas>. Accessed: 2026-08-24.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. [Chain-of-thought prompting elicits reasoning in large language models](#). In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.
- Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhi-fang Sui. 2024. [Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7655–7671, Bangkok, Thailand. Association for Computational Linguistics.
- Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. [Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.
- Jiachen Zhao, Wenlong Zhao, Andrew Drozdov, Benjamin Rozenoyer, Md Arafat Sultan, Jay-Yoon Lee, Mohit Iyyer, and Andrew McCallum. 2024. [Multistage collaborative knowledge distillation from a large language model for semi-supervised sequence generation](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14201–14214, Bangkok, Thailand. Association for Computational Linguistics.
- Yongchao Zhou, Kaifeng Lyu, Ankit Singh Rawat, Aditya Krishna Menon, Afshin Rostamizadeh, Sanjiv Kumar, Jean-François Kagy, and Rishabh Agarwal. 2024. [Distillspec: Improving speculative decoding via knowledge distillation](#). In *The Twelfth International Conference on Learning Representations*.

A Appendix

A.1 Knowledge Distillation

Knowledge distillation has emerged as a prominent paradigm for model compression and knowledge transfer, finding extensive applications across machine learning domains. The fundamental objective

is to transfer knowledge from a computationally intensive teacher model to a compact student model, thereby substantially reducing computational and memory requirements while preserving model performance. Traditional knowledge distillation methods align the logit distributions between teacher and student models, enabling the student to learn the predictive behavior of the teacher through the following loss function:

$$\mathcal{L}_{\text{KD}}^{\text{logits}} = D(\sigma(z^T; \tau), \sigma(z^S; \tau)). \quad (12)$$

where z^S and z^T denote the logits of the student and teacher models respectively, $\sigma(\cdot)$ represents the softmax function that transforms the output logits into probability distributions, and τ is a non-negative temperature hyperparameter that controls the smoothness of the predicted distributions. $D(\cdot, \cdot)$ denotes a divergence measure function that quantifies the discrepancy between two class distributions, with Kullback-Leibler (KL) divergence being a commonly adopted choice. Beyond logit-based distillation, feature-level distillation methods focus on minimizing the discrepancy between intermediate feature representations of teacher and student models, thereby enabling the student to acquire the internal representations of the teacher. This approach can be formally expressed as:

$$\mathcal{L}_{\text{KD}^{\text{feat}}} = D_{\text{feat}}(T(\phi_T), T(\phi_S)) \quad (13)$$

where ϕ_T and ϕ_S represent the features of the teacher model and the student model, respectively, $T(\cdot)$ denotes the alignment module that maps the feature spaces of teacher and student models to the same dimensionality, ensuring comparability at the feature level, $D_{\text{feat}}(\cdot, \cdot)$ denotes the function which can compute the distance between two feature maps, such as $L1$ – or $L2$ –norm. By jointly leveraging both logits-based and feature-based distillation strategies, we can achieve effective knowledge transfer from the teacher to the student model. Building upon this foundation, we proceed to investigate the integration of knowledge distillation mechanisms with the speculative decoding framework, with particular emphasis on enabling dynamic adaptation to target model updates.

DySD does not require a specific divergence measure. DySD uses a shared configurable divergence measure D for both logits-level alignment and update monitoring. In each deployment, D is instantiated as one of four alternatives: KL divergence, reverse KL divergence, Jensen–Shannon

divergence, or total variation distance. Unless otherwise specified, the same choice of D is used throughout the corresponding experiment:

$$D_{\text{KL}}(p \parallel q) = \sum_i p_i \log \frac{p_i}{q_i} \quad (14)$$

$$D_{\text{RKL}}(p \parallel q) = D_{\text{KL}}(q \parallel p) = \sum_i q_i \log \frac{q_i}{p_i}. \quad (15)$$

$$D_{\text{JSD}}(p \parallel q) = \frac{1}{2} D_{\text{KL}}\left(p \parallel \frac{p+q}{2}\right) + \frac{1}{2} D_{\text{KL}}\left(q \parallel \frac{p+q}{2}\right) \quad (16)$$

$$D_{\text{TVD}}(p \parallel q) = \frac{1}{2} \sum_i |p_i - q_i| \quad (17)$$

These measures provide interchangeable choices for distribution alignment and update triggering. KL and reverse KL capture directional discrepancies, while JSD and TVD offer symmetric alternatives. Because their numerical ranges differ, the update threshold should be calibrated for the selected divergence measure.

A.2 Theoretical Analysis of Threshold-Controlled Updates

To characterize the trade-off between adaptation quality and computational overhead, we analyze the relationship among the update threshold η , the update frequency, and the reduction in draft–target divergence. Let

$$d_t = D(Q_t(\cdot \mid x_t), P_{\theta_t}(\cdot \mid x_t)), \quad (18)$$

denote the divergence between the target and draft distributions at decoding step t , where $D(\cdot \parallel \cdot)$ represents the divergence metric adopted by the update mechanism. The update indicator is defined as

$$u_t(\eta) = \mathbb{I}[d_t > \eta], \quad (19)$$

and the empirical update frequency over T decoding steps is

$$F_T(\eta) = \frac{1}{T} \sum_{t=1}^T u_t(\eta). \quad (20)$$

For any fixed divergence sequence $\{d_t\}_{t=1}^T$, $F_T(\eta)$ is monotonically non-increasing with respect to η . Consequently, a smaller threshold activates online distillation more frequently, whereas a larger threshold reduces the number of parameter updates.

To further examine the benefit of each update, let d_t^- and d_t^+ denote the divergences immediately before and after an update, respectively. Assume that the update satisfies the following contraction condition:

$$d_t^+ \leq \rho d_t^- + \epsilon, \quad 0 \leq \rho < 1, \quad (21)$$

where ρ characterizes the alignment effectiveness and ϵ accounts for optimization error. Because an update is triggered only when $d_t^- > \eta$, the resulting divergence reduction satisfies

$$\begin{aligned} d_t^- - d_t^+ &\geq (1 - \rho)d_t^- - \epsilon \\ &> (1 - \rho)\eta - \epsilon. \end{aligned} \quad (22)$$

Therefore, whenever

$$\eta > \frac{\epsilon}{1 - \rho}, \quad (23)$$

each triggered update yields a positive lower bound on divergence reduction. This result clarifies that the threshold determines not only how often adaptation occurs but also the minimum discrepancy associated with a triggered update.

Under the additional assumption that the divergence drift between consecutive decoding steps is bounded by ν ,

$$d_{t+1} \leq d_t + \nu, \quad (24)$$

a larger threshold generally increases the interval between effective updates. Accordingly, the update frequency scales approximately with

$$F_T(\eta) = \mathcal{O}\left(\frac{\nu}{(1 - \rho)\eta - \epsilon}\right), \quad (25)$$

provided that each update returns the divergence below the triggering threshold and the associated threshold overshoot remains bounded.

This analysis explains the empirical behavior observed in our experiments: smaller thresholds enable faster recovery under distribution shifts, while larger thresholds reduce optimization overhead at the cost of slower adaptation.

A.3 DySD Detailed Algorithm

Algorithm 1 presents the detailed procedure of DySD. For each draft block, the divergence-based update condition is evaluated over all verified token positions, whereas rejected proposals are collected as training samples for online distillation.

Algorithm 1 Dynamic Speculative Decoding

Require: Target model Q , draft model P_θ , input prompt x , draft length K , update threshold η , temperature τ , feature-alignment weight λ

Ensure: Generated sequence T

```

1:  $T \leftarrow x$ 
2:  $\mathcal{R} \leftarrow \emptyset$ 
3: while EOS  $\notin T$  do
4:    $(\hat{y}_{1:K}, \mathbf{p}_{1:K}) \leftarrow \text{DRAFT}(P_\theta, T, K)$ 
5:    $(\mathbf{q}_{1:K}, \mathcal{I}_{\text{ver}}, \mathcal{I}_{\text{rej}}, y_{\text{emit}}) \leftarrow \text{VERIFY}(Q, T, \hat{y}_{1:K})$ 
6:    $T \leftarrow T \oplus y_{\text{emit}}$ 
7:    $\text{update} \leftarrow \text{false}$ 
8:   for  $i \in \mathcal{I}_{\text{ver}}$  do
9:      $d_i \leftarrow D(\mathbf{q}_i, \mathbf{p}_i)$   $\triangleright$  Monitor every verified token
10:    if  $d_i > \eta$  then
11:       $\text{update} \leftarrow \text{true}$ 
12:    end if
13:    if  $i \in \mathcal{I}_{\text{rej}}$  then
14:       $\mathcal{R} \leftarrow \mathcal{R} \cup \{(\mathbf{q}_i, \mathbf{p}_i, T_{<i})\}$ 
15:    end if
16:  end for
17:  if  $\text{update}$  and  $\mathcal{R} \neq \emptyset$  then
18:     $\mathcal{H} \leftarrow \text{SAMPLELAYERPAIRS}(Q, P_\theta)$ 
19:    Update the draft-side parameters  $\Theta = \{\theta, \mathbf{W}, \mathbf{b}\}$  on  $\mathcal{R}$  by minimizing  $\mathcal{L}$  in Equation 10
20:     $\mathcal{R} \leftarrow \emptyset$ 
21:  end if
22: end while
23: return  $T$ 

```

A.4 Detailed Experimental Setup

To comprehensively evaluate the proposed Dynamic Speculative Decoding (DySD) framework, we establish a rigorous experimental setup designed to simulate real-world deployment constraints. Throughout all experiments, the active target-model checkpoint remains frozen within each evaluation stage, while online optimization updates only the draft-side parameters $\Theta = \{\theta, \mathbf{W}, \mathbf{b}\}$. This design choice guarantees that any observed improvements in inference speed and token acceptance rates are solely attributable to our dynamic alignment strategy, while also reflecting practical scenarios where target LLMs are computationally prohibitive to update frequently.

For dynamic environment simulations, we uniformly set the training epoch to 1. This constraint is critical to accurately simulate real-time online

adaptation, forcing the draft model to learn from continuous data streams in a single pass without relying on repetitive offline training cycles that would introduce unacceptable deployment latency. And the draft length $K = 5$, learning rate is $1e-4$.

Regarding the hierarchical feature alignment, the relative depth partition hyperparameters are set to $d_1 = 0.25$ and $d_2 = 0.75$. The objective here is to effectively partition the model layers into shallow, middle, and deep stages, enabling the draft model to structurally capture lexical, syntactic, and abstract semantic representations progressively. While these values are fully tunable, our empirical validation demonstrated that this specific configuration yields robust and consistent alignment across the entire Vicuna model family, minimizing the need for exhaustive hyperparameter searches when swapping target models.

Furthermore, to ensure fair and rigorous comparisons against baseline methods, the decoding temperature is permanently fixed at 1.0. Because temperature scaling artificially alters the entropy of the output probability distributions, it significantly impacts both language model generation behavior and the resulting token acceptance rates. Maintaining a uniform temperature of 1.0 preserves the true confidence levels and original reasoning patterns of the target model. Finally, to ensure reproducibility and uniform latency measurements, all experiments were conducted using a single NVIDIA L40 48GB GPU.

A.5 End-to-End Latency and Overhead Analysis

To evaluate the practical overhead of online adaptation, we profile DySD using Vicuna-13B as the target model and LLaMA-160M as the draft model on a single NVIDIA L40 GPU. Draft generation, target verification, and parameter updates are executed synchronously.

Latency Overhead. DySD performs online updates only when the draft–target divergence exceeds the specified threshold, restricting optimization to the erroneous token set $\mathcal{R}$. As shown in Table 3, a single draft-model update requires 96.23 ms, while the observed update frequency is approximately 10%. The resulting amortized update cost is therefore 9.62 ms per decoding iteration, corresponding to approximately 6.0% of the target-model verification time.

Although individual updates introduce tempo-

Component	Latency / Value
Target verification (Vicuna-13B)	159.78 ms
Draft forward (LLaMA-160M)	15.96 ms
Draft update (LLaMA-160M)	96.23 ms
Update frequency	10.0%
Amortized update cost	9.62 ms

Table 3: Latency breakdown of DySD with synchronous online updates.

Component	Memory
Target model weights	25,025.81 MB
Draft model and optimizer states	981.10 MB
Forward KV cache and activations	30.78 MB
Additional backward overhead	629.31 MB
Total peak memory	26,667.00 MB

Table 4: Peak GPU memory breakdown for Vicuna-13B and LLaMA-160M.

rary latency increases, their low frequency substantially reduces the average computational overhead. Asynchronous updates or double-buffered draft models could further mitigate latency spikes in future deployment scenarios.

Memory Overhead. Table 4 reports the peak GPU memory consumption. The target-model weights account for 25,025.81 MB, while the draft model and optimizer states require 981.10 MB. Online optimization introduces an additional 629.31 MB of backward-related memory, and forward activations contribute 30.78 MB.

Together, the draft model, optimizer states, and backward overhead account for 1,610.41 MB, approximately 6.44% of the target-model weight footprint. Including forward activations, the total additional memory is 1,641.19 MB, or approximately 6.56%. These results indicate that DySD maintains a modest memory overhead relative to the frozen target model while supporting online adaptation.

A.6 Comparison with Training-Free Baselines

To further demonstrate the robustness of our dynamic update mechanism against domain shifts, we extend our evaluation to include strong training-free speculative decoding baselines. Training-free methods, dynamically adjust generation strategies without modifying draft model parameters. For instance, Token Recycling (Luo et al., 2025) iteratively adjusts its adjacency matrix after each decoding step, making it naturally adaptable to continuous data streams.

To provide a rigorous comparison, we evaluate

DySD against **Token Recycling** using the same heterogeneous mixed-dataset stream (comprising Spider, GSM8K, and Alpaca) as described in Figure 4. The primary evaluation metric is the average speedup ratio across these diverse domains.

Table 5: Comparison of average speedup ratios on continuous data streams between Token Recycling and DySD.

Method	Spider	GSM8K	Alpaca
Token Recycling	1.72×	1.68×	1.55×
DySD (Ours)	1.78×	1.72×	1.59×

As detailed in Table 5, DySD consistently outperforms Token Recycling across all evaluated tasks. While Token Recycling effectively handles local structural adjustments via its dynamic adjacency matrix, it fundamentally operates on the surface level of token generation. In contrast, DySD benefits from its hierarchical feature alignment module. By continuously aligning both the output logits and the deep internal representations between the draft and target models, DySD captures the evolving, underlying reasoning processes of the target model more comprehensively. This deeper structural synchronization ensures higher token acceptance rates and, consequently, superior acceleration performance under rapid distributional shifts.

A.7 Comparison with EAGLE under Sufficient Pretraining

To complement the dynamic comparison with Token Recycling, we further evaluate DySD against EAGLE(Li et al., 2024c) in a static setting. Because EAGLE relies on sufficient task-specific training, we pretrain the LLaMA-160M draft model used by DySD on the complete training set rather than the partial-data settings adopted in the main experiments. This configuration ensures that both methods are compared under a sufficiently trained setting.

We report the mean accepted length (MAL), which measures the average number of draft tokens accepted during each speculative verification step. Larger values indicate better alignment between the proposal mechanism and the target model.

As shown in Table 6, DySD achieves higher mean accepted length across all evaluated task-model combinations. The improvements are especially pronounced on Spider, where the target and draft models must align more closely on structured text-to-SQL generation.

Table 6: Comparison between DySD and EAGLE in terms of mean accepted length under sufficient pretraining. Both methods are evaluated with Vicuna-7B and Vicuna-13B as target models.

Method	Spider		GSM8K		Alpaca	
	7B	13B	7B	13B	7B	13B
EAGLE	1.430	1.421	2.010	2.010	1.640	1.690
DySD	1.820	1.827	2.230	2.220	1.650	1.710

These results indicate that combining output-distribution matching with hierarchical feature alignment improves draft-target correspondence even when both approaches are provided with sufficient training data. The comparison complements the dynamic results by showing that the proposed multi-objective alignment strategy is also beneficial under static deployment conditions.