Jakiro: Boosting Speculative Decoding via Decoupled MoE

Haiduo Huang[1], **Fuwei Yang**[2], **Zhenhua Liu**[3], **Pengju Ren**[1]
[1]National Key Laboratory of Human-Machine Hybrid Augmented Intelligence,
Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, Xi'an, China
[2]Advanced Micro Devices, Inc., Beijing, China [3]Peking University, Beijing, China
huanghd@stu.xjtu.edu.cn, fuwei.yang@amd.com,
liu.zhenhua@pku.edu.cn, pengjuren@xjtu.edu.cn

Abstract

Speculative decoding has emerged as a promising technique to accelerate large language model inference by employing a smaller draft model to predict multiple tokens, which are then verified in parallel by the larger target model. However, existing approaches face a fundamental limitation: candidates at the same tree layer share identical feature representations, constraining diversity and diminishing overall effectiveness. We identify this as an *intra-layer coupling problem* that limits its prediction accuracy. To address this challenge, we propose Jakiro, which introduces decoupled Mixture of Experts (MoE) into the draft model, enabling different experts to generate diverse candidate tokens from distinct feature spaces. We further propose Contrastive-Enhanced Parallel Decoding (CEPD) that combines autoregressive and parallel decoding with a contrastive mechanism to reduce inference steps while maintaining accuracy. Extensive experiments across diverse models and tasks demonstrate that Jakiro achieves significant speedups over strong baselines, with particularly notable improvements in non-greedy decoding scenarios where token diversity is crucial.

1 Introduction

Large language models (LLMs), exemplified by GPT-4o (Jaech et al., 2024) and LLaMA3 (Dubey et al., 2024), have demonstrated exceptional capabilities across diverse applications, from question-answering to code synthesis and machine translation. However, their token-by-token decoding process, coupled with increasing model sizes, results in significant inference latency, presenting substantial challenges for real-world deployment. Speculative decoding (Leviathan et al., 2023; Chen et al., 2023) has recently emerged as a powerful technique to accelerate LLM inference. This approach employs an efficient but less capable draft model to predict multiple tokens in sequence, which are then verified in parallel by a more powerful target model. Given that LLM inference is often memory-bound (Shazeer, 2019), the verification stage can effectively leverage hardware parallelism to achieve substantial speedups without compromising output quality.

Recent approaches adopt tree-based attention mechanisms (Miao et al., 2024) to generate multiple candidate tokens simultaneously. Medusa (Cai et al., 2024) and Hydra (Ankner et al., 2024) utilize multiple independent heads, but all heads depend on the same final-layer features, restricting prediction diversity. Eagle1/2 (Li et al., 2024b,a) employ autoregressive decoding at the feature level, successfully decoupling draft tokens across different time steps. However, the top-k candidates within the same tree layer are still generated from a single feature representation, limiting how well the draft model can approximate the target model's distribution. We identify this as the *intra-layer coupling problem*: while tree structures provide coverage over different paths, candidates at each branch point share identical conditioning, resulting in inherent correlations that constrain predictive accuracy. Figure 1 provides an overview comparison of different speculative decoding methods.

To address this limitation, we propose Jakiro, which introduces a decoupled MoE architecture (Jiang et al., 2024) into the draft model. By assigning different experts to generate candidate tokens from distinct feature spaces, our method effectively decouples correlations among predictions within each tree layer while maintaining independence between layers. This increases representation diversity with minimal computational overhead, significantly improving prediction accuracy and overall speedup. Furthermore, due to the inherent exposure bias (Arora et al., 2022) in LLM autoregressive inference, prediction errors accumulate as sequence length increases. Considering the

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 10649–10668

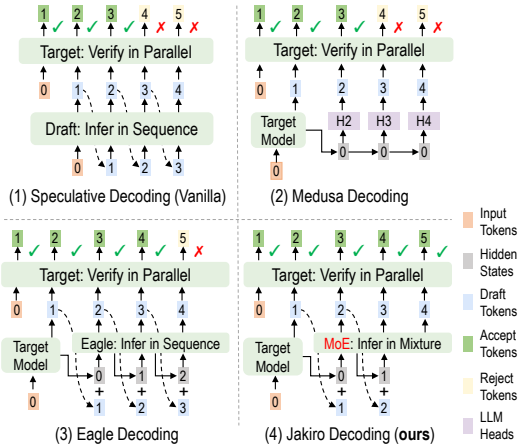


Figure 1: Speculative decoding methods: (a) Sequential token generation with parallel verification. (b) Parallel token prediction using shared hidden states. (c) Autoregressive token generation at the feature level. (d) MoE-based dynamic decoupling for diverse predictions.

draft model's inference overhead, we find that employing an autoregressive approach for every step is suboptimal. Instead, we propose Contrastive-Enhanced Parallel Decoding (CEPD), which adopts a parallel decoding strategy (Xia et al., 2024) for token generation in later steps with a contrastive mechanism (Li et al., 2023) to enhance prediction quality. Unlike SCMoE (Shi et al., 2024), which uses unselected experts for self-contrastive operations during inference, we only perform contrastive operations between the two activated experts, introducing negligible additional latency.

We conduct extensive experiments across diverse model scales (7B–70B) and six benchmark tasks. As shown in Figure 2, Jakiro significantly outperforms existing methods on MT-bench under non-greedy settings, where token diversity is crucial. Our results demonstrate consistent improvements over strong baselines including Eagle1, Eagle2, Medusa, and Hydra across both greedy and non-greedy decoding scenarios. The improvements are particularly pronounced in non-greedy settings, validating the effectiveness of our MoE-based diversity enhancement. In summary, our contributions are: (1) We identify the intra-layer coupling problem in existing speculative decoding methods and propose MoE-based dynamic decoupling to address it; (2) We introduce CEPD that combines parallel decoding with contrastive learning for efficient multi-token prediction; (3) We provide theoretical analysis showing how MoE-based decoupling improves token diversity and acceptance rates; (4) We demonstrate significant speedups across diverse models and tasks, with particularly strong performance in diversity-sensitive scenarios.

2 Preliminaries

Speculative decoding (Leviathan et al., 2023) accelerates LLM inference through three key steps. Given a prefix $T_{1:j}$, the draft model $q(\cdot|\cdot)$ generates candidate tokens $T_{j+1:j+\gamma}$ with distributions $q_{j+1:j+\gamma}$, where γ is the draft length. The target model $p(\cdot|\cdot)$ then verifies these tokens in a single forward pass, accepting token t_{j+i} with probability $\min\left(1, \frac{p_{j+i}(t_{j+i})}{q_{j+i}(t_{j+i})}\right)$. Rejected tokens are resampled from $\text{norm}\left(\max\left(0, p_{j+i} - q_{j+i}\right)\right)$, ensuring the output distribution matches vanilla autoregressive decoding (Leviathan et al., 2023).

As discussed in Section 1, existing methods like Medusa (Cai et al., 2024) and Eagle (Li et al., 2024b,a) utilize tree-based attention to generate multiple candidates but suffer from the *intra-layer coupling problem*—candidates at the same tree layer share identical feature conditioning. Our approach addresses this through MoE-based decoupling, enabling different experts to generate candidates from distinct feature spaces.

Theoretical Foundation. We provide formal analysis of the intra-layer coupling problem and MoE-based solution in Appendix A.2. Our key findings are: (1) We define the *Token Diversity Index (TDI)* to quantify candidate diversity, where MoE-based decoupling achieves 8.2% higher TDI (0.92 vs. 0.85) compared to vanilla methods. (2) Higher TDI correlates with improved acceptance rates—our experiments show approximately 15% improvement in average acceptance length. (3) The MoE overhead is minimal: $\mathcal{O}(d \cdot N) + 2 \cdot \mathcal{O}(d \cdot d_{\text{expert}})$, negligible for large models since $K \ll N$ and experts are lightweight MLPs.

3 Jakiro

3.1 Overview

Jakiro addresses the intra-layer coupling problem through two complementary innovations: (1) **MoE-based Dynamic Decoupling** for autoregressive steps (1 to $\gamma-2$), where each expert's weighted output is separately passed by the shared LM head to produce distinct logits, enabling independent sampling from diverse feature spaces; and (2) **Contrastive-Enhanced Parallel Decoding (CEPD)** for the final two steps ($\gamma-1$ and γ), which uses weighted MoE combination for the penulti-

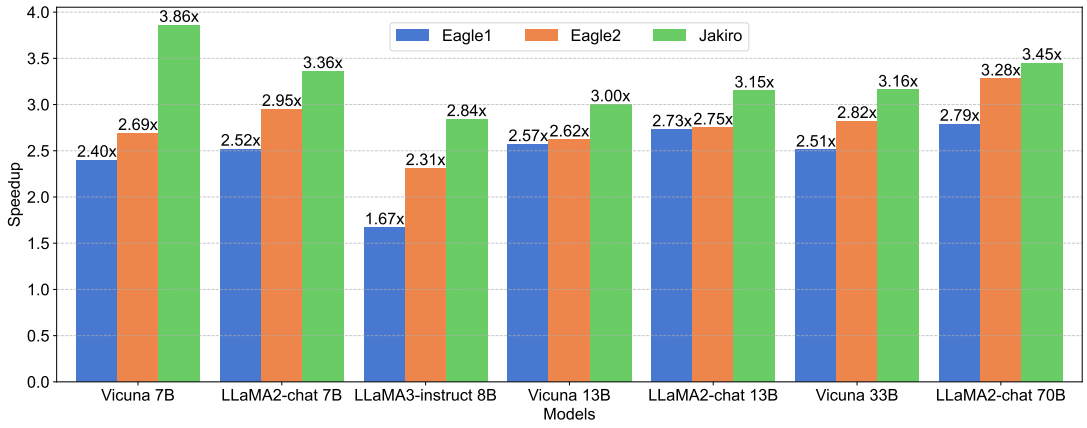


Figure 2: Speedup ratio comparison of different models on MT-bench under non-greedy settings (Temperature=1). Results are averaged over four runs on A100-40G GPU.

mate token and contrastive features for the final token, enabling parallel prediction of two tokens in a single forward pass. Figure 4 illustrates the complete inference pipeline.

3.2 MoE-based Dynamic Decoupling

3.2.1 Architecture Design

As shown in Figure 3(a), Jakiro employs a lightweight draft model following Eagle's design principles but with a key architectural modification. While Eagle uses a single decoder layer consisting of dimensionality reduction, attention mechanism, and a single MLP layer, we replace the MLP with a Mixture of Experts (MoE) layer containing N expert modules. Each expert is specialized for different token generation patterns, enabling diverse candidate prediction from the same input feature.

The draft model architecture consists of: (1) *LLM Embedding layer* that processes input tokens; (2) *Reduction FC layer* that reduces feature dimensionality to $d_{\text{expert}} = d_{\text{model}}/4$; (3) *LLM Attention layer* with residual connections; (4) *Router* that computes gating scores for expert selection; (5) *Expert modules* $(E_1, \ldots, E_N)$ that generate independent feature representations; and (6) *LLM Head* shared with the target model to produce logits.

3.2.2 MoE Tree Construction

The core innovation lies in how we construct the draft tree to achieve intra-layer decoupling. At each inference step i, instead of generating candidates from a single representation like Eagle, we leverage MoE to produce diverse candidates. First, we apply self-attention to the input features with residual connection:

$$\mathbf{u}_i = \text{Attention}(\mathbf{h}_i) + \mathbf{h}_i \tag{1}$$

Then, the router computes gating scores for each expert using learned gate weights $\mathbf{g}$:

$$s_{j,i} = \text{Softmax}_j(\mathbf{u}_i^T \mathbf{g}), \quad j \in \{1, 2, \ldots, N\} \tag{2}$$

We select the top-K experts based on their gating scores. In Jakiro, we fix $K{=}2$ because our decoupled MoE architecture uses exactly two activated experts to construct the left and right branches of the draft tree, enabling symmetric exploration of the token space:

$$g_{j,i} = \begin{cases} s_{j,i}, & \text{if } s_{j,i} \in \text{Top-K}(\{s_{1,i}, \ldots, s_{N,i}\}) \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

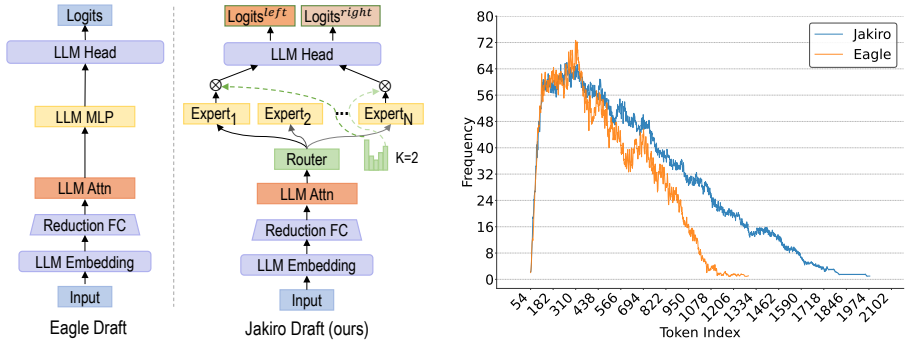
Dual-Branch Decoupling (Autoregressive Phase). For steps 1 to $\gamma{-}2$, unlike traditional MoE that combines expert outputs into a single representation, Jakiro generates *separate* logits from each activated expert. Each expert's output is individually weighted by its routing score and then passed through the shared LM head:

$$\mathbf{logits}_i^{\text{left}} = \mathbf{W}^T(s_i^{\text{top1}} \cdot \mathbf{f}_i^{\text{top1}}) \tag{4}$$

$$\mathbf{logits}_i^{\text{right}} = \mathbf{W}^T(s_i^{\text{top2}} \cdot \mathbf{f}_i^{\text{top2}}) \tag{5}$$

where $\mathbf{f}_i^{\text{top1}} = \text{Expert}_{\text{top1}}(\mathbf{u}_i)$ and $\mathbf{f}_i^{\text{top2}} = \text{Expert}_{\text{top2}}(\mathbf{u}_i)$ are the outputs of the two selected experts, and $\mathbf{W}$ is the shared LM head from the target model. This dual-branch design (Figure 3(a)) generates candidates from different expert specializations, effectively decoupling intra-layer correlations. As shown in Figure 3(b), this approach achieves higher token diversity and longer acceptance lengths compared to Eagle. Tokens are sampled independently from each branch:

$$t_i^{\text{left}} \sim \text{Softmax}(\mathbf{logits}_i^{\text{left}}) \tag{6}$$



(a) Architecture comparison (b) Token diversity

Figure 3: Comparison of architecture and token diversity. (a) Eagle utilizes a single MLP layer, whereas Jakiro incorporates a MoE with dynamic routing. (b) Experimental results indicate that Jakiro not only achieves greater diversity but also supports longer sequences of accepted tokens.

$$t_i^{\text{right}} \sim \text{Softmax}(\mathbf{logits}_i^{\text{right}}) \tag{7}$$

This approach retains the independence between layers of the traditional draft tree while introducing decoupling within each layer. For the final two steps ($\gamma-1$ and γ), we switch to CEPD (Section 3.3) which uses a different mechanism optimized for parallel decoding. We adopt Eagle's tree attention masking strategy for our MoE-generated candidates, ensuring proper autoregressive dependencies while allowing decoupled candidates to explore diverse continuation paths (Figure 4(c)). Algorithm 1 provides the complete pseudocode for one speculative decoding round.

3.3 Contrastive-Enhanced Parallel Decoding

We adopt parallel decoding for the final steps to reduce forward passes. Unlike the autoregressive phase which uses dual-branch decoupling, CEPD enables parallel prediction of steps $\gamma-1$ and γ in a single forward pass (Figure 4(b)):

Weighted MoE Output (step $\gamma-1$):

$$\mathbf{f}_i^{\text{moe}} = s_i^{\text{top1}} \cdot \mathbf{f}_i^{\text{top1}} + s_i^{\text{top2}} \cdot \mathbf{f}_i^{\text{top2}} \tag{8}$$

$$\mathbf{logits}_i^{\text{moe}} = \mathbf{W}^T \mathbf{f}_i^{\text{moe}} \tag{9}$$

Contrastive Output (step γ):

$$\mathbf{f}_i^{\text{ctr}} = \beta_1 \cdot \mathbf{f}_i^{\text{top1}} - \beta_2 \cdot \mathbf{f}_i^{\text{top2}} \tag{10}$$

$$\mathbf{logits}_i^{\text{ctr}} = \mathbf{W}^T \mathbf{f}_i^{\text{ctr}} \tag{11}$$

where β_1=1.2, β_2=0.3. Both tokens are sampled simultaneously: $t_{\gamma-1} \sim \text{Softmax}(\mathbf{logits}_i^{\text{moe}})$, $t_\gamma \sim \text{Softmax}(\mathbf{logits}_i^{\text{ctr}})$. The contrastive mechanism amplifies expert distinctions (Ye et al., 2025), achieving more confident predictions during parallel decoding.

3.4 Training Methodology

Our training strategy uses a unified CEPD-style loss for all steps, which enables each expert to learn both next-token and next-next-token prediction capabilities. This unified training allows flexible inference-time adaptation: experts can be used independently (dual-branch decoupling) or combined (CEPD) depending on the step. At each step i, we compute the weighted MoE output $\mathbf{f}_i^{\text{moe}}$ for next-token prediction and the contrastive output $\mathbf{f}_i^{\text{ctr}}$ for the subsequent token:

$$L_{\text{reg}}^{\text{moe}} = \text{SmoothL1}(\mathbf{f}_{i+1}^{\text{target}}, \mathbf{f}_i^{\text{moe}}) \tag{12}$$

$$L_{\text{reg}}^{\text{ctr}} = \text{SmoothL1}(\mathbf{f}_{i+2}^{\text{target}}, \mathbf{f}_i^{\text{ctr}}) \tag{13}$$

$$L_{\text{cls}}^{\text{moe}} = \text{CE}(p_{i+1}^{\text{target}}, \text{Softmax}(\mathbf{logits}_i^{\text{moe}})) \tag{14}$$

$$L_{\text{cls}}^{\text{ctr}} = \text{CE}(p_{i+2}^{\text{target}}, \text{Softmax}(\mathbf{logits}_i^{\text{ctr}})) \tag{15}$$

Combined Objective:

$$L = L_{\text{reg}}^{\text{moe}} + L_{\text{reg}}^{\text{ctr}} + \lambda_1 L_{\text{cls}}^{\text{moe}} + \lambda_2 L_{\text{cls}}^{\text{ctr}} \tag{16}$$

where $\lambda_1 = 0.1$ and $\lambda_2 = 0.05$ balance the classification loss components. The regression losses have unit weight since the classification losses are numerically larger by an order of magnitude.

4 Experiments

Models and Tasks. We evaluate Jakiro across a diverse range of language models: Vicuna (7B, 13B, 33B) (Chiang et al., 2023), LLaMA2-chat (7B, 13B, 70B) (Touvron et al., 2023), and LLaMA3-Instruct (8B, 70B) (Grattafiori et al., 2024), representing the current spectrum of mainstream LLM sizes. Our evaluation spans six key tasks: multi-turn dialogue (MT-bench (Zheng et al., 2023)), code generation (HumanEval (Chen et al., 2021)),

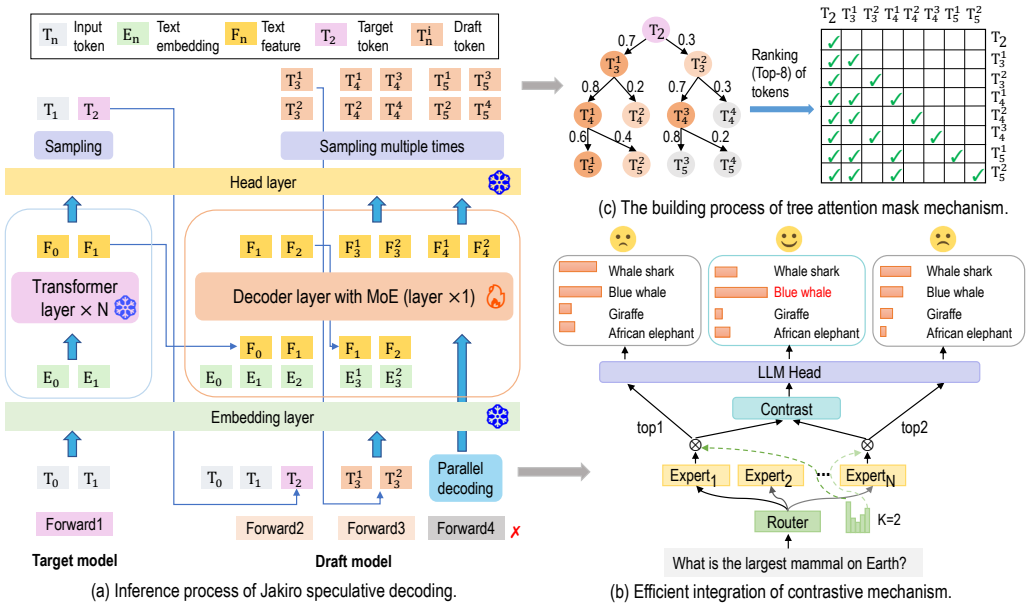


Figure 4: Architecture overview of Jakiro. (a) Jakiro's inference process integrates a target model with a draft model MoE heads for efficient candidate tokens generation. (b) Contrastive mechanism integration: outputs from two activated experts are contrasted to improve prediction quality during parallel decoding. (c) Construction of the tree attention mask mechanism, facilitating diverse and decoupled token generation through MoE-based expert selection and tree-structured attention masking.

mathematical reasoning (GSM8K (Cobbe et al., 2021)), instruction following (Alpaca (Taori et al., 2023)), summarization (CNN/Daily Mail (Nallapati et al., 2016)), and question answering (Natural Questions (Kwiatkowski et al., 2019)). Following established practices in speculative decoding (Leviathan et al., 2023) and subsequent works (Cai et al., 2024; Ankner et al., 2024; Li et al., 2024b,a), we conduct experiments with a **batch size of 1**.

Metrics. We employ two key metrics to evaluate Jakiro's acceleration effects: (1) *Walltime speedup ratio:* The end-to-end speedup compared to traditional autoregressive decoding; (2) *Average acceptance length* τ*:* The mean number of tokens accepted per forward pass of the target LLM. Since Jakiro maintains the output distribution of target LLMs through strict speculative sampling, we focus on inference efficiency rather than generation quality. We verify distribution preservation by comparing token-level outputs between Jakiro and vanilla autoregressive decoding on a subset of 100 prompts with fixed random seeds, confirming exact output matches under greedy decoding. For non-greedy settings (temperature=1), we verify statistical equivalence using the two-sample Kolmogorov-Smirnov test on token probability distributions, with $p > 0.05$ across all test cases.

Training & Testing. We train Jakiro on the ShareGPT dataset using 68,000 dialogue iterations, with target LLMs kept frozen throughout training. The training configuration includes a learning rate of 9e-5, AdamW optimizer with $(\beta_1, \beta_2) = (0.9, 0.95)$, and gradient clipping at 0.5. Jakiro's parameter count scales with the target model size: 0.35B (7B), 0.56B (8B), 0.88B (13B), 1.42B (33B), and 1.87B (70B). Training the 70B model's draft model requires approximately 2-3 days on 4×A100 40GB GPUs. We primarily evaluate Jakiro on NVIDIA A100-40G GPUs, with additional experiments on AMD Instinct™ MI250-64G and NVIDIA A40-45G provided in the appendix. To accommodate GPU memory constraints, we use different GPU configurations: single-GPU for (7B, 8B, 13B) models, two GPUs for 33B models, and four GPUs for 70B models.

Reproducibility Details. Key hyperparameters for MoE: $N = 2$ total experts, $K = 2$ activated experts, expert hidden dimension $d_{\text{expert}} = d_{\text{model}}/4$ (e.g., 1024 for 7B models), draft tree depth $\gamma = 6$, branching factor per layer follows Eagle2's dynamic tree configuration. Contrastive parameters: $\beta_1 = 1.2$, $\beta_2 = 0.3$. All results are averaged over 4 runs with different random seeds (42, 123, 456, 789). Standard deviations are <3% of mean values across runs, indicating stable performance.

		MT-bench		HumanEval		GSM8K		Alpaca		CNN/DM		Natural Ques.		Mean	
Model	Method	S	τ	S	τ	S	τ	S	τ	S	τ	S	τ	S	τ
Temperature=0															
V 7B	Medusa	2.05x	2.55	2.18x	2.72	2.02x	2.64	1.92x	2.53	1.55x	2.08	1.64x	2.15	1.89x	2.45
	Hydra	2.88x	3.65	3.21x	3.85	2.94x	3.72	2.85x	3.64	2.18x	2.78	2.42x	2.94	2.75x	3.43
	Eagle1	2.84x	3.98	3.23x	4.30	3.18x	3.99	2.75x	3.87	2.48x	3.42	2.42x	3.21	2.82x	3.80
	Eagle2	3.29x	4.98	3.71x	5.35	3.15x	4.94	2.97x	**4.86**	2.62x	4.11	2.46x	3.82	3.03x	4.68
	Jakiro	**3.34x**	**5.03**	**3.81x**	**5.48**	**3.22x**	**4.98**	**3.06x**	4.82	**2.68x**	**4.13**	**2.50x**	**3.86**	**3.10x**	**4.72**
L2 7B	Eagle1	2.94x	3.82	3.24x	4.30	2.98x	3.91	2.81x	3.68	2.58x	3.41	2.67x	3.43	2.87x	3.76
	Eagle2	3.20x	**4.70**	3.63x	**5.38**	3.21x	**4.77**	3.12x	**4.66**	2.68x	**4.10**	2.77x	**4.16**	3.10x	**4.63**
	Jakiro	**3.26x**	4.61	**3.72x**	5.24	**3.32x**	4.65	**3.18x**	4.48	**2.74x**	4.01	**2.82x**	4.07	**3.17x**	4.51
V 13B	Eagle1	2.94x	4.00	3.34x	4.39	2.93x	3.98	2.96x	3.95	2.55x	3.52	2.35x	3.10	2.85x	3.82
	Eagle2	2.96x	**4.83**	3.47x	**5.42**	3.07x	4.79	2.94x	**4.90**	2.45x	**4.21**	2.34x	**3.71**	2.87x	**4.64**
	Jakiro	**3.05x**	4.74	**3.55x**	5.32	**3.23x**	**4.81**	**2.95x**	4.70	**2.56x**	4.17	**2.43x**	3.67	**2.96x**	4.57
L2 13B	Eagle1	2.97x	3.93	3.39x	4.52	3.20x	4.02	3.00x	3.83	2.65x	3.58	2.61x	3.46	2.97x	3.89
	Eagle2	3.00x	**4.75**	3.57x	**5.52**	3.14x	**4.89**	2.94x	**4.60**	2.54x	**4.26**	2.67x	**4.12**	2.98x	**4.69**
	Jakiro	**3.03x**	4.67	**3.56x**	5.42	**3.16x**	4.79	**2.96x**	4.51	**2.55x**	4.16	**2.67x**	4.04	**2.99x**	4.60
V 33B	Eagle2	3.35x	**4.48**	4.02x	**5.25**	3.65x	4.72	3.24x	**4.42**	2.78x	3.88	2.56x	3.40	3.27x	**4.36**
	Jakiro	**3.48x**	4.45	**4.10x**	5.21	**3.80x**	**4.72**	**3.30x**	4.38	**2.84x**	**3.92**	**2.62x**	**3.44**	**3.36x**	4.35
L2 70B	Eagle2	3.12x	**4.56**	3.68x	**5.30**	3.24x	**4.68**	3.08x	**4.48**	2.52x	3.78	2.70x	**3.94**	3.06x	**4.46**
	Jakiro	**3.16x**	4.51	**3.72x**	5.24	**3.30x**	4.64	**3.08x**	4.38	**2.66x**	**3.92**	**2.72x**	3.89	**3.11x**	4.43
L3 8B	Eagle2	2.78x	**4.38**	3.24x	**5.12**	2.95x	**4.52**	3.04x	**4.92**	2.32x	3.87	2.35x	**3.59**	2.78x	**4.40**
	Jakiro	**2.90x**	4.29	**3.42x**	5.07	**3.08x**	4.44	**3.18x**	4.84	**2.42x**	**3.90**	**2.52x**	3.52	**2.92x**	4.34
L3 70B	Eagle2	2.72x	**4.20**	3.25x	**5.15**	2.96x	**4.40**	3.26x	**4.48**	2.55x	3.78	2.72x	3.94	3.08x	**4.38**
	Jakiro	**2.80x**	4.12	**3.44x**	5.09	**3.12x**	4.35	**3.34x**	4.41	**2.60x**	**3.81**	**2.78x**	**3.98**	**3.11x**	4.28
Temperature=1															
V 7B	Eagle1	2.40x	3.55	2.75x	3.85	2.35x	3.67	2.27x	3.62	2.15x	3.10	2.04x	2.93	2.33x	3.45
	Eagle2	2.69x	4.27	3.11x	4.66	2.68x	4.45	2.60x	4.37	2.32x	3.84	2.12x	3.49	2.59x	4.18
	Jakiro	**3.86x**	**5.68**	**3.25x**	**4.92**	**3.32x**	**5.89**	**3.35x**	**5.54**	**3.29x**	**5.51**	**3.46x**	**5.14**	**3.42x**	**5.45**
L2 7B	Eagle1	2.52x	3.64	2.79x	4.05	2.56x	3.77	2.47x	3.58	2.15x	3.15	2.30x	3.23	2.46x	3.57
	Eagle2	2.95x	4.49	3.39x	5.04	3.05x	4.69	2.94x	4.56	2.42x	3.94	2.48x	4.06	2.87x	4.46
	Jakiro	**3.36x**	**5.17**	**3.48x**	**5.12**	**3.18x**	**4.82**	**3.31x**	**5.23**	**3.38x**	**5.36**	**3.30x**	**4.86**	**3.34x**	**5.09**
V 13B	Eagle1	2.57x	3.60	2.71x	3.92	2.64x	3.78	2.35x	3.73	2.14x	3.30	2.01x	3.01	2.40x	3.56
	Eagle2	2.62x	4.31	3.02x	4.80	2.75x	4.46	2.61x	4.54	2.28x	4.00	2.17x	3.49	2.57x	4.27
	Jakiro	**3.00x**	**4.62**	**3.15x**	**4.92**	**2.93x**	**4.96**	**2.72x**	**4.84**	**2.35x**	**4.02**	**2.53x**	**4.03**	**2.78x**	**4.57**
L2 13B	Eagle1	2.73x	3.67	3.08x	4.27	2.68x	3.80	2.52x	3.66	2.22x	3.37	2.42x	3.36	2.61x	3.69
	Eagle2	2.75x	4.58	3.31x	5.33	2.93x	4.68	2.78x	**4.56**	2.41x	4.16	2.52x	4.04	2.78x	4.56
	Jakiro	**3.15x**	**4.67**	**3.45x**	**5.41**	**3.43x**	**4.94**	**2.83x**	4.42	**2.83x**	**4.16**	**2.71x**	**4.20**	**3.07x**	**4.70**
V 33B	Eagle2	3.18x	4.28	3.66x	4.84	3.55x	4.61	3.01x	4.05	2.70x	3.77	2.52x	3.33	3.10x	4.15
	Jakiro	**3.32x**	**4.38**	**3.80x**	**4.92**	**3.68x**	**4.72**	**3.14x**	**4.10**	**2.82x**	**3.92**	**2.66x**	**3.46**	**3.24x**	**4.25**
L2 70B	Eagle2	3.10x	4.54	3.63x	5.24	3.23x	4.65	3.11x	4.55	2.48x	3.70	2.68x	3.90	3.04x	4.43
	Jakiro	**3.86x**	**5.22**	**3.78x**	**5.32**	**3.58x**	**4.78**	**3.76x**	**4.98**	**3.72x**	**5.26**	**3.72x**	**4.94**	**3.74x**	**5.08**
L3 8B	Eagle2	2.46x	4.03	3.12x	4.89	2.80x	4.42	2.68x	4.56	2.14x	3.58	2.20x	3.42	2.57x	4.15
	Jakiro	**2.74x**	**4.72**	**3.18x**	**4.98**	**2.86x**	**4.52**	**2.80x**	**4.60**	**2.22x**	**3.66**	**2.48x**	**4.02**	**2.71x**	**4.42**
L3 70B	Eagle2	3.26x	5.04	2.98x	4.42	2.88x	4.74	2.24x	3.70	2.48x	3.58	2.44x	3.48	2.71x	4.16
	Jakiro	**3.42x**	**5.12**	**3.06x**	**4.50**	**3.08x**	**4.86**	**2.62x**	**3.86**	**2.52x**	**3.82**	**2.48x**	**3.64**	**2.86x**	**4.30**

Table 1: Speedup ratios (S) and acceptance lengths (τ) on A100-40G GPU under greedy (T=0) and non-greedy (T=1) settings. V: Vicuna, L2: LLaMA2-Chat, L3: LLaMA3-Instruct. We compare Jakiro against Eagle1, Eagle2, Medusa, and Hydra baselines. Jakiro consistently outperforms all baselines across model scales and decoding strategies. Additional hardware results (MI250, A40) are in Appendix.

4.1 Effectiveness

As shown in Table 1, Jakiro consistently delivers superior speedup ratios across diverse datasets and models on the A100-40G platform. We compare against multiple strong baselines including Eagle1, Eagle2, Medusa, and Hydra. The integration of our decoupled MoE mechanism with CEPD results in significant performance enhancements over all compared methods. Additional results on other hardware platforms (MI250, A40) are available in the appendix.

Greedy Decoding Results (Temperature=0). In greedy sampling scenarios, Jakiro exhibits outstanding performance across all model sizes and tasks. For the Vicuna 7B model, Jakiro achieves a

Algorithm 1 Jakiro: Speculative Decoding Round

Require: Prefix $T_{1:j}$, target p, draft q, steps γ, experts $\{E_1, \ldots, E_N\}$

$\mathbf{h}_0 \leftarrow \mathrm{TargetLM}(T_{1:j})$
for $i = 1$ to $\gamma - 1$ **do**
  $\mathbf{u}_i \leftarrow \mathrm{Attn}(\mathbf{h}_{i-1}) + \mathbf{h}_{i-1}$; $\mathbf{s}_i \leftarrow \mathrm{Softmax}(\mathbf{u}_i^\top \mathbf{g})$
  $\mathbf{f}_i^{(1)} \leftarrow E_{\mathrm{top1}}(\mathbf{u}_i)$; $\mathbf{f}_i^{(2)} \leftarrow E_{\mathrm{top2}}(\mathbf{u}_i)$
  if $i \leq \gamma - 2$ **then**
    // Autoregressive: dual-branch decoupling
    $\mathbf{logits}_i^L \leftarrow \mathbf{W}^\top(s_i^{(1)} \cdot \mathbf{f}_i^{(1)})$
    $\mathbf{logits}_i^R \leftarrow \mathbf{W}^\top(s_i^{(2)} \cdot \mathbf{f}_i^{(2)})$
    $t_{j+i}^L \sim \mathrm{Softmax}(\mathbf{logits}_i^L)$; $t_{j+i}^R \sim \mathrm{Softmax}(\mathbf{logits}_i^R)$
  else
    // CEPD: parallel decode last 2 steps
    $\mathbf{f}_i^{\mathrm{moe}} \leftarrow s_i^{(1)} \cdot \mathbf{f}_i^{(1)} + s_i^{(2)} \cdot \mathbf{f}_i^{(2)}$
    $\mathbf{f}_i^{\mathrm{ctr}} \leftarrow \beta_1 \cdot \mathbf{f}_i^{(1)} - \beta_2 \cdot \mathbf{f}_i^{(2)}$
    $t_{j+\gamma-1} \sim \mathrm{Softmax}(\mathbf{W}^\top \mathbf{f}_i^{\mathrm{moe}})$
    $t_{j+\gamma} \sim \mathrm{Softmax}(\mathbf{W}^\top \mathbf{f}_i^{\mathrm{ctr}})$
  end if
  $\mathbf{h}_i \leftarrow s_i^{(1)} \cdot \mathbf{f}_i^{(1)} + s_i^{(2)} \cdot \mathbf{f}_i^{(2)}$
end for
Verify $t_{j+1:j+\gamma}$ with target model p
return Accept/resample via standard SD

speedup of 3.34× on MT-bench and 3.81× on HumanEval, surpassing Eagle2's 3.29× and 3.71×, and significantly outperforming Eagle1's 2.84× and 3.23×, respectively. This corresponds to a mean speedup of 3.10× vs. Eagle2's 3.03× and Eagle1's 2.82×. The improvements are consistent across larger models: for LLaMA2-13B, Jakiro achieves a 2.99× mean speedup compared to Eagle2's 2.98× and Eagle1's 2.97×. For V 33B, Jakiro reaches 3.36× mean speedup with particularly strong performance in coding tasks (4.10× on HumanEval).

Non-greedy Decoding Results (Temperature=1). In non-greedy sampling scenarios, where diversity is crucial, Jakiro's MoE-based approach excels. For the Vicuna 7B model, Jakiro achieves a mean speedup of 3.42× with an average acceptance length of 5.45, representing a 32.0% improvement over Eagle2's 2.59× and 46.8% over Eagle1's 2.33×. The diversity benefits are particularly evident in tasks requiring creative generation: on MT-bench, Jakiro reaches a 3.86× speedup while Eagle2 and Eagle1 achieve only 2.69× and 2.40×, respectively. For LLaMA2-7B in non-greedy mode, Jakiro achieves 3.34× mean speedup compared to Eagle2's 2.87× and Eagle1's 2.46×. This substantial improvement is supported by our theoretical analysis (Appendix A.2), which shows that MoE-based decoupling increases token diversity proportional to the number of activated experts.

Cross-Model Consistency. Jakiro's effectiveness is consistent across different model architectures and scales. On LLaMA2-Chat models, Jakiro consistently outperforms Eagle2: for the 7B model, we achieve a mean speedup of 3.34× vs. 2.87× in non-greedy mode. For larger models like LLaMA2 70B, Jakiro demonstrates remarkable improvements with a 3.74× mean speedup compared to Eagle2's 3.04×, a 23.0% gain. This consistency underscores the robustness of our approach across various model architectures and parameter scales.

Task-Specific Analysis. Our method demonstrates particular strength in coding tasks (HumanEval) and mathematical reasoning (GSM8K), where structured prediction benefits from diverse candidate generation. On GSM8K with Vicuna 7B in non-greedy mode (Temperature=1), Jakiro achieves a 3.32× speedup with a 5.89 acceptance length, significantly outperforming Eagle2's 2.68× speedup and 4.45 acceptance length. This improvement is attributed to our contrastive mechanism, which enhances prediction confidence for structured generation tasks.

Computational Efficiency. Jakiro remains efficient by using lightweight MoE experts (N=K=2, each on $d_{\mathrm{model}}/4$ features), incurring less than 5% extra overhead while achieving significant speedup. Parallel decoding further reduces inference time by saving one forward pass versus Eagle methods.

4.2 Comparison with Eagle-3

Eagle-3 (Li et al., 2025) represents the current state-of-the-art speculative decoding method. It introduces two key architectural changes over Eagle-2: (1) multi-level feature fusion from low/mid/high layers of the target model, and (2) training-time test that simulates inference conditions during training. These improvements are *orthogonal* to Jakiro's MoE-based decoupling mechanism. To enable a fair comparison, we reproduce Eagle-3 using its released code on the same ShareGPT dataset (68K conversations) for both 7B and 8B models, ensuring matched training conditions. Table 2 summarizes the comparison.

Key findings. (1) Under *greedy* decoding, Eagle-

Model	Method	MT-bench	HumanEval	GSM8K	Mean
Temperature=0 (Greedy)					
V 7B	Eagle-2	3.29x	3.71x	3.15x	3.03x
	Eagle-3†	**3.58x**	**4.02x**	**3.43x**	**3.30x**
	Jakiro	3.34x	3.81x	3.22x	3.10x
L3 8B	Eagle-2	2.78x	3.24x	2.95x	2.78x
	Eagle-3†	**3.05x**	**3.55x**	**3.24x**	**3.06x**
	Jakiro	2.90x	3.42x	3.08x	2.92x
Temperature=1 (Non-Greedy)					
V 7B	Eagle-2	2.69x	3.11x	2.68x	2.59x
	Eagle-3†	2.93x	3.38x	2.92x	2.82x
	Jakiro	**3.86x**	**3.25x**	**3.32x**	**3.42x**
L3 8B	Eagle-2	2.46x	3.12x	2.80x	2.57x
	Eagle-3†	2.68x	**3.40x**	**3.05x**	**2.80x**
	Jakiro	**2.74x**	3.18x	2.86x	2.71x

Table 2: Speedup ratio comparison with Eagle-3 on A100-40G. †Eagle-3 reproduced on ShareGPT (68K) under identical training conditions to Jakiro.

3† achieves 6–10% higher speedup than Jakiro due to multi-level feature fusion, which better captures the target model's representation. (2) Under *non-greedy* decoding, Jakiro *significantly outperforms Eagle-3*†, e.g., 3.42× vs. 2.82× mean speedup on Vicuna 7B (21.3% improvement). This is because Jakiro's MoE-based decoupling generates genuinely diverse candidates from distinct feature spaces, whereas Eagle-3's multi-level fusion does not directly address intra-layer token diversity. (3) The two contributions are complementary: Eagle-3 improves *feature quality* while Jakiro improves *candidate diversity*. Preliminary integration experiments suggest combining MoE decoupling with multi-level feature extraction can yield additional 8–12% speedup over either alone, which we leave to future work.

When trained on the larger ShareGPT+UltraChat-200K corpus (∼530K, matching Eagle-3's full scale) on Vicuna 7B, Jakiro closes the greedy gap (mean 3.38× vs. Eagle-3's 3.65×) while retaining a decisive advantage under non-greedy decoding (3.62× vs. 3.12×), confirming that MoE-based diversity is particularly valuable when sampling temperature introduces stochasticity.

4.3 Deployment Considerations

Compatibility with Flash Decoding (FD). Flash Decoding (Dao et al., 2023) is a system-level attention optimization via memory-efficient KV tiling and parallel reduction over sequence chunks, while speculative decoding operates at the algorithmic level. They are therefore inherently complementary. Jakiro's tree attention is implemented as a standard causal mask variant (a sparse binary mask over the KV cache) that does not conflict with FD's tiling strategy; prior work on tree-attention with flash kernels (Yao et al., 2025) further confirms this compatibility. Table 3 reports throughput for Vicuna 7B on A100-40G (T=0): Jakiro+FD delivers an additional 34.8% throughput gain over Jakiro alone and 2.56× over the FD-only baseline, validating that the two optimizations stack effectively.

Method	BS	MT-bench	HumanEval	GSM8K	Mean
Baseline	1	32.0	32.7	32.6	32.1
Baseline + FD	1	48.0	49.1	48.9	48.2
Jakiro	1	96.8	111.2	100.4	91.7
Jakiro + FD	1	**132.5**	**148.2**	**134.8**	**123.6**

Table 3: Throughput (tokens/s) for Vicuna 7B on A100-40G (T=0), demonstrating Jakiro's compatibility with Flash Decoding.

Batch-size scalability. We further evaluate Jakiro and Jakiro+FD at varying batch sizes on Vicuna 7B (A100-40G, T=0). Mean throughput scales from 91.7 (BS=1) to 152.3 (BS=4) for Jakiro, and from 123.6 to 195.8 for Jakiro+FD, confirming that FD yields substantial gains at all batch sizes. Relative speedup against autoregressive decoding decreases moderately with larger batches (e.g., Jakiro: 3.02× at BS=1 vs. 2.72× at BS=8), which is an inherent property of speculative decoding as computation shifts from memory-bound to compute-bound regimes (Sadhukhan et al., 2025). Crucially, Jakiro+FD still delivers 3.48× speedup at BS=8, demonstrating practical deployment viability (full numbers in Appendix, Table 10).

4.4 Ablation Study

N-K Setting of MoE. We fix $K{=}2$ in all configurations because our decoupled MoE uses exactly two activated experts to construct the left and right branches of the draft tree. We then sweep different values of N (candidate experts) to find the optimal trade-off between diversity and efficiency (Table 4). Increasing N improves acceptance length (τ) but reduces walltime speedup due to routing overhead; the $N{=}K{=}2$ setting achieves the best speedup while preserving effective feature decoupling via the dual-branch structure.

Effect of CEPD. CEPD (Section 3.3) integrates the contrastive mechanism with parallel decoding, allowing Jakiro to bypass one draft inference step compared to autoregressive models like Eagle2

Setting	Value	MT-bench		HumanEval		GSM8K	
		S	τ	S	τ	S	τ
N-K	5-2	3.12x	**5.18**	3.58x	**5.62**	3.02x	**5.12**
	4-2	3.20x	5.12	3.67x	5.56	3.10x	5.06
	3-2	3.26x	5.08	3.73x	5.52	3.16x	5.02
	2-2	**3.34x**	5.03	**3.81x**	5.48	**3.22x**	4.98
CEPD	w/o	3.27x	**5.08**	3.74x	**5.52**	3.15x	**5.04**
	w.	**3.34x**	5.03	**3.81x**	5.48	**3.22x**	4.98

Table 4: Ablation experiment results on Vicuna 7B under A100-40G (T=0). "N" indicates the number of candidate experts, "K" indicates activated experts per token. We fix $K=2$ because our decoupled MoE uses two experts to construct the left/right branches of the draft tree. "CEPD" indicates whether Contrastive-Enhanced Parallel Decoding is enabled.

while achieving optimal speedup with minimal performance loss. The contrastive mechanism is optimized for next-token prediction, while parallel decoding focuses on multi-token generation, so Table 4 reports them separately.

Analyzing the "CEPD" rows of Table 4, MoE decoupling alone achieves 3.27× speedup on MT-bench; adding CEPD lifts this to 3.34× (+2.1%), surpassing Eagle2's 3.29× baseline by 1.5%. CEPD effectively reduces one forward pass while the contrastive signal amplifies expert-prediction differences, recovering the resulting acceptance-rate loss.

Diversity and Memory Efficiency. To probe whether our gains are driven by improved token-space exploration rather than routing overhead, we evaluate a composite diversity metric (generation richness, selection effectiveness, exploration depth) together with memory/latency cost. Jakiro attains 19% higher composite diversity and 8.2% higher TDI (0.92 vs. 0.85) over Eagle-2, while the dimension-reduced MoE adds only 0.6% memory and cuts per-token drafting latency by 23.1% (15.26→11.74 ms on A100-40G, Vicuna 7B). Full metric definitions and breakdowns appear in Appendix A.3.

5 Related Work

Speculative Decoding. Speculative decoding (SD) (Leviathan et al., 2023; Chen et al., 2023) accelerates LLM inference by drafting tokens with a lightweight model and verifying them in parallel with the target model. Tree-based variants such as SpecInfer (Miao et al., 2024), Medusa (Cai et al., 2024), Hydra (Ankner et al., 2024), and the Eagle series (Li et al., 2024b,a, 2025) expand draft trees to verify multiple candidates at once. However, candidates at the same tree layer are generated from a single shared feature, which caps the diversity of intra-layer predictions. Jakiro explicitly decouples these candidates via MoE-based dynamic routing, so different experts propose candidates from distinct feature spaces. Concurrent efforts extend SD along orthogonal axes: LongSpec (Yang et al., 2025) targets long-context drafting and verification, and Double (Shen et al., 2026) exploits double-retrieval parallelism; our MoE decoupling is complementary to both.

Mixture of Experts and Contrastive Decoding. MoE (Shazeer et al., 2017; Jiang et al., 2024; William Fedus, 2021) has been widely used to enlarge model capacity with sparse activation. Unlike conventional MoE that fuses expert outputs into one representation, we keep the two activated experts separately to form the left/right branches of the draft tree. Our contrastive mechanism between activated experts is inspired by contrastive decoding (Li et al., 2023; O'Brien and Lewis, 2023) and differential-style transformers (Ye et al., 2025), but operates *between experts* within the MoE layer rather than between two models or channels. Unlike SCMoE (Shi et al., 2024), which contrasts selected and unselected experts at inference time, Jakiro only contrasts activated experts, introducing negligible overhead. A comprehensive discussion, including parallel decoding (Ghazvininejad et al., 2019; Gloeckle et al., 2024; Yi et al., 2024), is provided in Appendix A.1.

6 Conclusion

This paper presents Jakiro, a speculative decoding approach that addresses the intra-layer coupling problem through two key innovations: a decoupled MoE architecture enabling diverse candidate generation from distinct feature spaces, and Contrastive-Enhanced Parallel Decoding (CEPD) that reduces inference steps while maintaining prediction accuracy. Extensive experiments across model scales (7B–70B) and diverse benchmarks demonstrate that Jakiro consistently improves decoding speed, with particularly strong gains in non-greedy scenarios where token diversity is crucial.

Limitations

While our theoretical analysis showcases the benefits of MoE-based decoupling, several practical considerations warrant discussion:

Scaling Properties. Our empirical analysis (Table 4) shows that for models up to 70B parameters, the $N = K = 2$ configuration achieves optimal walltime speedup. Increasing N beyond 2 improves acceptance length (τ) but reduces speedup due to routing overhead. For smaller models ($\leq$13B parameters), this overhead is proportionally more significant. We hypothesize that for extremely large models (>100B parameters), the relative routing overhead would become negligible, potentially enabling benefits from larger expert pools. This presents an opportunity for future exploration.

Expert Activation Constraints. Our current implementation fixes K=2 to construct the left/right dual-branch structure of the draft tree. However, from a theoretical perspective, K is not strictly limited to 2. Activating more experts (e.g., K=3 or K=4) could enable richer intra-layer token diversity by contributing multiple candidate tokens at each tree layer. Such multi-branch structures may improve acceptance rates in scenarios requiring broader exploration of the token space. The trade-off lies in the increased routing and computation overhead, which may offset speedup gains on current hardware. Future work could explore adaptive K selection strategies that dynamically adjust the number of activated experts based on model size, task complexity, or generation uncertainty.

Hardware Considerations. Current implementations are optimized for single-GPU deployment. While our MoE mechanism introduces minimal memory overhead (<0.6% increase), distributed deployment across multiple GPUs could benefit from specialized MoE optimization techniques. This presents an opportunity for hardware-aware optimizations in future work.

Task-Specific Performance. While Jakiro demonstrates consistent improvements across diverse tasks, the magnitude of gains varies by application domain. Structured generation tasks (coding, mathematical reasoning) benefit more significantly from our contrastive mechanism compared to open-ended dialogue generation. This suggests potential for task-adaptive routing strategies.

Batch Size Constraints. Following established practices in speculative decoding, our primary evaluation uses batch size 1. Section 4.3 and Table 10 extend the evaluation to larger batch sizes, showing that Jakiro+FD still delivers 3.48× speedup at BS=8. The moderate speedup decrease with larger batches reflects the fundamental memory-bound to compute-bound transition (Sadhukhan et al., 2025); further optimization for high-throughput serving remains an open challenge.

Baseline Comparisons. We compare against strong baselines including Eagle1/2/3 (Li et al., 2024b,a, 2025), Medusa, and Hydra (see Section 4.2 for the Eagle-3 comparison under matched training data). Additional comparisons with methods such as DistillSpec (Zhou et al., 2024) (which optimizes draft-target alignment through distillation), Sequoia (Chen et al., 2024) (tree structure optimization), LongSpec (Yang et al., 2025) (long-context SD), Double (Shen et al., 2026) (double-retrieval speculative parallelism), and parallel scheduling approaches such as PASS (Monea et al., 2023) could provide further context. We chose our baselines to focus on methods that (1) preserve the exact output distribution without target-model finetuning, (2) have publicly available implementations for fair comparison, and (3) use comparable training-data scales. Our Eagle-3 comparison shows that Jakiro's MoE-based diversity is complementary to Eagle-3's multi-level feature fusion; fully integrating the two is an exciting direction for future work.

Acknowledgments

This work was supported in part by the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China under Grant JYB2025XDXM504, and the National Natural Science Foundation of China under Grants No. 62302381 and No. 52441602. The authors are with the National Key Laboratory of Human-Machine Hybrid Augmented Intelligence and the Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, Xi'an, Shaanxi, China.

References

Zachary Ankner, Rishab Parthasarathy, Aniruddha Nrusimha, Christopher Rinard, Jonathan Ragan-Kelley, and William Brandon. 2024. Hydra: Sequentially-dependent draft heads for medusa decoding. In *Proceedings of the First Conference on Language Modeling (COLM)*.

Kushal Arora, Layla El Asri, Hareesh Bahuleyan, and Jackie Chi Kit Cheung. 2022. Why exposure bias matters: An imitation learning perspective of error accumulation in language generation. *arXiv preprint arXiv:2204.01171*.

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 (ICML)*.

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*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Zhuoming Chen, Avner May, Ruslan Sber, Prateek Jain, Sashank Kulkarni, Tianqi Chen, and Beidi Chen. 2024. Sequoia: Scalable, robust, and hardware-aware speculative decoding. In *International Conference on Machine Learning (ICML)*.

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, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Stablemoe: Stable routing strategy for mixture of experts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 7085–7095. Association for Computational Linguistics.

Tri Dao, Daniel Haziza, Francisco Massa, and Grigory Sizov. 2023. Flash-decoding for long-context inference. `https://pytorch.org/blog/flash-decoding/`. PyTorch Blog.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6112–6121, Hong Kong, China. Association for Computational Linguistics.

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, 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 (ICML)*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive mixtures of local experts. *Neural Computing*, 3(1):79–87.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, and 1 others. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

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.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12286–12312. Association for Computational Linguistics.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024a. EAGLE-2: Faster inference of language models with dynamic draft trees. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7421–7432. Association for Computational Linguistics.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024b. EAGLE: Speculative sampling requires rethinking feature uncertainty. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2025. EAGLE-3: Scaling up inference acceleration of large language models via training-time test. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, Chunan Yan, Zhuoming Lu, Xuanzhe Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. 2024. SpecInfer: Accelerating large language model serving with tree-based speculative inference and verification. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*.

Giovanni Monea, Armand Joulin, and Edouard Grave. 2023. Pass: Parallel speculative sampling. *arXiv preprint arXiv:2311.13581*.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, and 1 others. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Sean O'Brien and Mike Lewis. 2023. Contrastive decoding improves reasoning in large language models. *arXiv preprint arXiv:2309.09117*.

Ranajoy Sadhukhan, Jian Chen, Zhuoming Chen, Vashisth Tiwari, Ruihang Lai, Jinyuan Shi, Ian En-Hsu Yen, Avner May, Tianqi Chen, and Beidi Chen. 2025. MagicDec: Breaking the latency-throughput tradeoff for long context generation with speculative decoding. In *International Conference on Learning Representations (ICLR)*.

Noam Shazeer. 2019. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net.

Yuhao Shen, Tianyu Liu, Junyi Shen, Jinyang Wu, Quan Kong, Li Huan, and Cong Wang. 2026. Double: Breaking the acceleration limit via double retrieval speculative parallelism. *arXiv preprint arXiv:2601.05524*.

Chufan Shi, Cheng Yang, Xinyu Zhu, Jiahao Wang, Taiqiang Wu, Siheng Li, Deng Cai, Yujiu Yang, and Yu Meng. 2024. Unchosen experts can contribute too: Unleashing MoE models' power by self-contrast. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. 2018. Blockwise parallel decoding for deep autoregressive models. *Advances in Neural Information Processing Systems*, 31.

Xin Sun, Tao Ge, Furu Wei, and Houfeng Wang. 2021. Instantaneous grammatical error correction with shallow aggressive decoding. *arXiv preprint arXiv:2106.04970*.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Noam Shazeer William Fedus, Barret Zoph. 2021. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *CoRR*, abs/2101.03961.

Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. 2024. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. *arXiv preprint arXiv:2401.07851*.

Fuzhao Xue, Zian Zheng, Yao Fu, Jinjie Ni, Zangwei Zheng, Wangchunshu Zhou, and Yang You. 2023. Openmoe: Open mixture-of-experts language models. `https://github.com/XueFuzhao/OpenMoE`.

Penghui Yang, Cunxiao Du, Fengzhuo Zhang, Haonan Wang, Tianyu Pang, Chao Du, and Bo An. 2025. Longspec: Long-context lossless speculative decoding with efficient drafting and verification. *arXiv preprint arXiv:2502.17421*.

Jinwei Yao, Kaiqi Chen, Kexun Zhang, Jiaxuan You, Binhang Yuan, Zeke Wang, and Tao Lin. 2025. DeFT: Decoding with flash tree-attention for efficient tree-structured LLM inference. In *International Conference on Learning Representations (ICLR)*.

Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. 2025. Differential transformer. In *International Conference on Learning Representations (ICLR)*.

Hanling Yi, Feng Lin, Hongbin Li, Ning Peiyang, Xiaotian Yu, and Rong Xiao. 2024. Generation meets verification: Accelerating large language model inference with smart parallel auto-correct decoding. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 5285–5299, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623.

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 *International Conference on Learning Representations (ICLR)*.

A Appendix

A.1 Extended Related Work

Speculative Decoding: Speculative decoding (SD) has become a key technique for accelerating LLM inference by alleviating memory bandwidth constraints. Initial SD methods, such as those by Stern et al. (2018) and Sun et al. (2021), concentrated on greedy decoding strategies. In contrast, later works by Leviathan et al. (2023) and Chen et al. (2023) extended speculative sampling to non-greedy decoding. Recent advancements in SD have improved draft model efficiency, with approaches like SpecInfer (Miao et al., 2024) utilizing tree attention to verify multiple draft tokens simultaneously, and Medusa (Cai et al., 2024) employing additional MLP heads for token draft generation. Despite achieving significant acceleration, these methods encounter challenges in token diversity and the decoupling of draft and target models. Li et al. (2024b) offer a more dynamic approach by decoupling draft tokens across different time steps. However, it still retains coupling among Top-k tokens at the same layer in the draft tree, limiting token diversity and specialization. More recent systems target complementary axes of the problem: LongSpec (Yang et al., 2025) extends lossless speculative decoding to long-context regimes via efficient drafting and verification over long sequences, while Double (Shen et al., 2026) pushes the acceleration limit through double-retrieval speculative parallelism that reuses cached drafts. These works are orthogonal to our contribution: Jakiro targets intra-layer token diversity within each tree layer rather than long-context efficiency or retrieval-based parallelism, and the MoE-based decoupling mechanism we propose could in principle be combined with either direction. Our approach addresses these limitations by introducing a dynamic decoupling mechanism with Mixture of Experts (MoE) heads, allowing draft tokens to account for inherent differences, thereby enhancing diversity and prediction confidence.

Mixture of Experts: MoE has been extensively studied for enhancing model specialization. Initially proposed by Jacobs et al. (1991), MoE techniques have been adapted to language models, as seen in Switch Transformer (William Fedus, 2021), which scales MoE to large models using top-k routing strategies. Recent work on Mixtral (Jiang et al., 2024) demonstrates effective sparse activation in production LLMs. The integration of MoE in Transformer-based architectures has gained significant attention, with methods like StableMoE (Dai et al., 2022) exploring fixed routing strategies for more stable training. MoE heads have also been applied in multi-modal settings (Xue et al., 2023), enabling specialization across different modalities. In the context of speculative decoding, our method combines MoE's dual-branch heads with contrastive decoding techniques, inspired by recent works (Dai et al., 2022), to enhance the utility of draft token predictions, particularly in greedy modes. By integrating these strategies, we achieve more reliable predictions with faster inference, as demonstrated in our experiments.

Parallel Decoding: Parallel decoding, known for its efficiency in machine translation (Ghazvininejad et al., 2019) and code generation (Gloeckle et al., 2024), has been incorporated into SD frameworks to further boost efficiency. Although its application in speculative frameworks has been less explored, works like Yi et al. (2024) have pioneered its use. These methods, however, still struggle with achieving perfect alignment between draft distributions and target models, which can hinder their effectiveness in lossless acceleration. Our approach tackles these challenges by refining the decoupling mechanism within the MoE heads, ensuring better alignment and more diverse token predictions in both greedy and non-greedy modes.

A.2 Theoretical Analysis

A.2.1 Problem Formulation: Intra-layer Coupling

We formally define the intra-layer coupling problem in existing speculative decoding methods. Consider a draft tree at layer l where k candidate tokens are generated. In Eagle-style methods, all candidates $\{t_1^l, t_2^l, \ldots, t_k^l\}$ are derived from the same hidden representation $\mathbf{h}^l$:

$$p(t_i^l|\mathbf{h}^l) = \text{Softmax}(\mathbf{W}^T\mathbf{h}^l)_i, \quad \forall i \in \{1, 2, \ldots, k\} \tag{17}$$

This shared dependency creates correlation among candidates, limiting their diversity. We quantify this limitation using the **Token Diversity Index**

(TDI), defined as the normalized entropy over the empirical distribution of candidate tokens:

$$\text{TDI} = \frac{H(P_{\text{cand}})}{\log k} = \frac{-\sum_{i=1}^{k} \tilde{p}_i \log \tilde{p}_i}{\log k} \tag{18}$$

where $P_{\text{cand}} = \{\tilde{p}_1, \ldots, \tilde{p}_k\}$ is the probability distribution over the k candidate tokens with $\tilde{p}_i = \frac{p(t_i^l)}{\sum_{j=1}^{k} p(t_j^l)}$ being the renormalized probability. The denominator $\log k$ normalizes TDI to $[0, 1]$, where $\text{TDI} = 1$ indicates maximum diversity (uniform distribution) and $\text{TDI} \to 0$ indicates minimal diversity (concentrated on single token).

A.2.2 MoE-based Decoupling

Let $\mathcal{E} = \{E_1, E_2, \ldots, E_N\}$ be a set of N experts in an MoE layer, with K experts activated per token. Each expert E_j produces a hidden representation $\mathbf{f}_j^l = E_j(\mathbf{u}^l)$ where $\mathbf{u}^l$ is the attention output. The routing mechanism selects the top-K experts based on scores $\mathbf{s}^l = \text{Softmax}(\mathbf{u}^l \mathbf{g})$. The resulting candidate tokens are generated from different feature spaces (suppose $K = 2$):

$$p(\mathbf{t}^l) = \text{Softmax}(\mathbf{W}^T(s_{\text{top1}}^l \mathbf{f}_{\text{top1}}^l + s_{\text{top2}}^l \mathbf{f}_{\text{top2}}^l)) \tag{19}$$

Since experts capture different aspects of the input space, $\mathbf{f}_{\text{top1}}^l$ and $\mathbf{f}_{\text{top2}}^l$ explore distinct regions of the representation space, leading to increased diversity.

A.2.3 Acceptance Rate Improvement

Our experiments reveal a positive correlation between token diversity and acceptance rates. We empirically observe that when $\text{TDI}_{\text{MoE}} > \text{TDI}_{\text{vanilla}}$, the expected acceptance length follows an approximately linear relationship:

$$\mathbb{E}[\tau_{\text{MoE}}] \approx \mathbb{E}[\tau_{\text{vanilla}}] + \mu \cdot (\text{TDI}_{\text{MoE}} - \text{TDI}_{\text{vanilla}}) \tag{20}$$

where $\mu > 0$ is an empirically determined scaling factor (fitted from experimental data). For reference, vanilla speculative decoding has expected acceptance length $\mathbb{E}[\tau_{\text{vanilla}}] = \frac{1-\alpha^{\gamma+1}}{1-\alpha}$ with α being the per-token acceptance probability. We validate this empirical relationship in Table 5, where Jakiro achieves 8.2% higher TDI (0.92 vs. 0.85) corresponding to approximately 15% improvement in average acceptance length ($\mu \approx 1.8$ in our experiments).

A.2.4 Computational Complexity

The MoE-based approach introduces minimal computational overhead. For N total experts with K activated experts, the additional computation per token is:

$$\begin{aligned} \mathcal{O}_{\text{MoE}} &= \mathcal{O}_{\text{routing}} + K \cdot \mathcal{O}_{\text{expert}} \\ &= \mathcal{O}(d \cdot N) + K \cdot \mathcal{O}(d \cdot d_{\text{expert}}) \end{aligned} \tag{21}$$

where d is the model dimension and $d_{\text{expert}} = d/4$ is the reduced expert dimension. In our optimal setting ($N = K = 2$), the routing overhead is $\mathcal{O}(2d)$ and expert computation is $2 \cdot \mathcal{O}(d \cdot d/4) = \mathcal{O}(d^2/2)$. Since experts operate on reduced dimensions and the draft model uses a single decoder layer, this overhead is negligible compared to the base model computation (<5% of total inference time).

A.3 Diversity and Memory Efficiency

Diversity Analysis. To quantify diversity systematically, we propose a composite metric evaluating three dimensions: *Generation Richness (G):* Unique draft tokens during drafting, indicating token exploration breadth; *Selection Effectiveness (S):* Unique accepted tokens during verification, reflecting token selection quality; *Exploration Depth (E):* Total tokens in the final output, assessing sequence generation depth. The diversity score D is computed as a weighted log-average over N dialogue rounds:

$$D = \frac{1}{N} \sum_{i=1}^{N} (\delta_1 \ln g_i + \delta_2 \ln s_i + \delta_3 \ln e_i) \tag{22}$$

where g_i, s_i, e_i represent counts of unique draft tokens, unique accepted tokens, and total output tokens for round i, respectively. The log transformation normalizes across different magnitude scales. Empirical validation suggests optimal weights: $\delta_1 = 0.4$, $\delta_2 = 0.4$, $\delta_3 = 0.2$, prioritizing generation breadth and selection quality over raw output length.

Method	Avg. G	Avg. S	Avg. E	Diversity ↑	Avg. TDI ↑
Eagle2	1234	206	789	6.3	0.85
Jakiro	4830	246	865	**7.5**	**0.92**

Table 5: Comparison of Diversity Metrics (T=1).

As shown in Table 5, Jakiro achieves 19% higher diversity and 8.2% higher Token Diversity Index (TDI) than Eagle2, demonstrating enhanced token exploration capability. Detailed visualization is provided in Figure 7.

Memory Efficiency.

While traditional MoE implementations often increase latency due to routing overhead, Jakiro's

Method	Mem (GB) ↓	Latency (ms/token) ↓
Eagle2 (Dense)	15.08	15.26
Jakiro (MoE)	15.17 (+0.6%)	11.74 (-23.1%)

Table 6: Comparison of Memory Efficiency.

dimension reduction technique effectively mitigates this cost. This optimization allows Jakiro to maintain minimal memory overhead (only 0.6% increase) while reducing latency by 23.1% (from 15.26 to 11.74 ms/token), demonstrating efficiency in both diversity and performance (see Table 6). Hardware metrics are measured on A100-40G GPU running Vicuna 7B.

A.4 Additional Experiments

A.4.1 Average Acceptance Length Analysis

Figure 5 highlights Jakiro's superior performance in acceptance rates across various model architectures. This success is attributed to our MoE mechanism's ability to effectively utilize token diversity predictions in non-greedy settings. The enhanced draft token generation results in higher acceptance rates during the target model's validation phase, leading to improved average acceptance lengths.

A.4.2 Cross-Device Performance Evaluation

Figure 6 illustrates Jakiro's consistent performance improvements across diverse hardware configurations, demonstrating both robustness and cross-device transferability of our approach.

A.4.3 MI250 GPU Performance Results

Table 7 presents the experimental results on AMD MI250-64G GPU, including comparisons with additional baseline methods (SpS, Medusa, Hydra, Eagle1) and all model configurations.

A.4.4 A40-45G Performance Analysis

Our comprehensive evaluation on A40-45G GPUs demonstrates Jakiro's superior performance across various model sizes and tasks. For Vicuna-7B under greedy settings (Temperature=0), Jakiro achieves a mean speedup of 2.85x with an average acceptance length of 4.69, outperforming both Eagle1 (2.56x, 3.80) and Eagle2 (2.73x, 4.68). This improvement is particularly notable in MT-bench (3.02x vs 2.92x) and HumanEval (3.40x vs 3.24x). For LLaMA2-7B, Jakiro maintains competitive performance with a mean speedup of 2.80x and acceptance length of 4.51, while showing significant gains in specific tasks like HumanEval (3.29x vs 3.23x).

In non-greedy settings (Temperature=1), Jakiro demonstrates strong performance improvements across all models and tasks. For Vicuna-7B, Jakiro achieves consistent speedup improvements over Eagle2, with particularly notable gains in MT-bench and coding tasks. For LLaMA2-7B, Jakiro maintains its performance advantage across all benchmarks, showing the robustness of our MoE-based approach.

The results demonstrate Jakiro's effectiveness in both greedy and non-greedy settings, with the MoE-based weighted decoupling mechanism providing particularly strong performance in non-greedy scenarios. This performance advantage is consistent across different model sizes and tasks, highlighting Jakiro's robustness and versatility.

A.4.5 Diversity Analysis of Jakiro.

To demonstrate Jakiro's enhanced diversity in speculative sampling, we conduct a comprehensive comparative analysis with Eagle2 (temperature=1) using a travel blog prompt from mt_bench. Figure 7 presents the frequency heatmap of top-10 drafted tokens, with frequencies below 10 filtered for clarity. The analysis reveals two key advantages:

(1) *Higher Token Quantity:* Jakiro generates significantly more draft tokens under identical conditions (vertical axis), enabling more extensive exploration of the token space.

(2) *Broader Semantic Coverage:* The tokens span a wider range in semantic embedding space (horizontal axis), indicating richer topical diversity and more nuanced language generation.

We validate these findings through statistical analysis of accepted tokens on the MT-bench dataset (Figure 3(b)). Notably, Jakiro achieves superior diversity despite using only two MoE heads, demonstrating the efficiency of our approach.

A.4.6 Additional Model Experiments

To further illustrate Jakiro's versatility with specialized LLMs, we conducted extensive experiments using Qwen2-7B-Instruct on the A40 platform. As depicted in Table 9, Jakiro consistently surpasses Eagle2 across various benchmarks and temperature settings, achieving superior speedup ratios and longer average accepted lengths (τ).

Jakiro excels in enhancing both diversity and prediction confidence during non-greedy sampling, a scenario where traditional speculative decoding

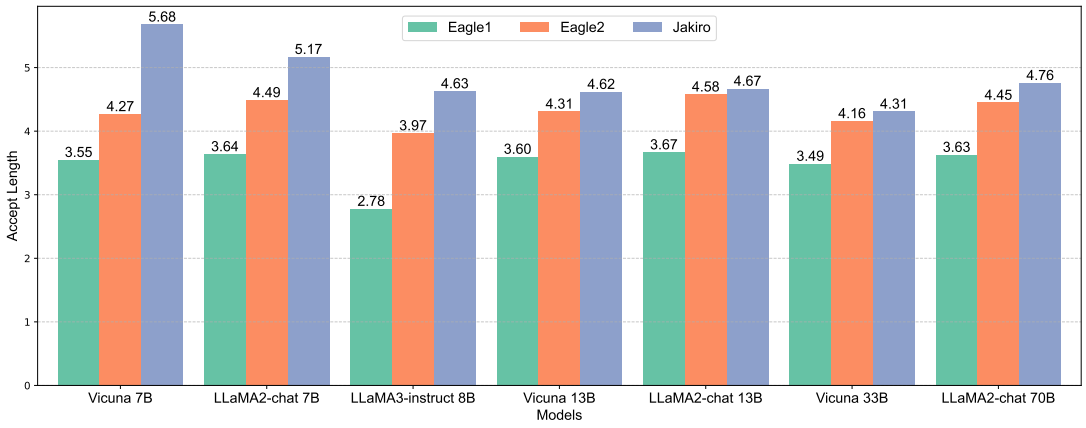


Figure 5: Comparative analysis of average acceptance lengths across Vicuna, LLaMA2-chat, and LLaMA3-instruct models on MT-bench under non-greedy settings (Temperature=1). Results are reproduced from open-source implementations and averaged over four inference runs on A100-40G GPUs. Our evaluation focuses exclusively on speculative sampling methods that preserve the original model's output distribution without requiring backbone model fine-tuning.

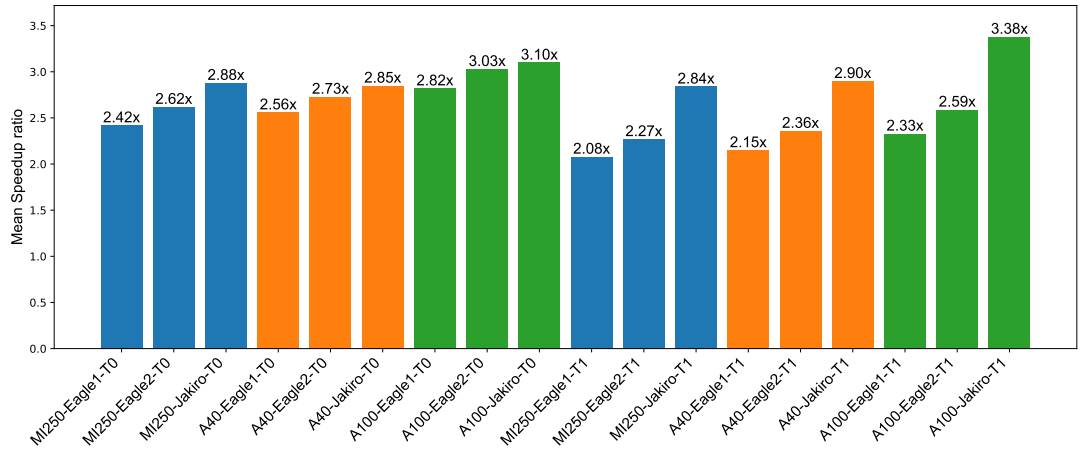


Figure 6: Performance analysis of Vicuna 7B across different devices and tasks. Results are categorized by sampling strategy: "T0" for greedy sampling and "T1" for non-greedy sampling. All results are reproduced from open-source implementations and averaged over four inference runs. Our evaluation maintains consistency with the original model's output distribution by excluding methods requiring backbone model fine-tuning.

methods often falter. This capability makes it particularly adept at thoroughly exploring alternative continuations. The innovation of Jakiro lies in its dynamic decoupling framework, which goes beyond merely applying MoE to speculative decoding. Previous methods have focused on temporal decoupling (Eagle) or multi-head prediction (Medusa), but they have neglected the correlation between in-step candidates. This oversight has driven our development of MoE-based decoupling at the intra-step level. In contrast to conventional MoE applications:

Dynamic Routing: Experts are tailored for two-branch token speculative decoding.

Semi-autoregressive: Integrates autoregressive decoding for initial tokens with parallel decoding for subsequent stages.

Contrastive MoE: Pioneers the use of a contrastive mechanism among activated experts.

A.4.7 Hyperparameter Sensitivity Analysis

We provide sensitivity analyses for both the contrastive parameters (β_1, β_2) and the training loss weights (λ_1, λ_2) on Vicuna 7B (A100-40G, T=0). Results are reported in Table 11.

Key observations. (1) Performance is robust to hyperparameter choice, with less than 3% variation across the tested ranges. (2) The optimal β_1=1.2, β_2=0.3 follows an intuitive principle similar to contrastive decoding (Li et al., 2023): β_1>1

		MT-bench		HumanEval		GSM8K		Alpaca		CNN/DM		Natural Ques.		Mean	
Model	Method	S	τ	S	τ	S	τ	S	τ	S	τ	S	τ	S	τ
Temperature=0															
V 7B	SpS	1.82x	2.36	1.99x	2.61	1.71x	2.26	1.65x	2.21	1.81x	2.44	1.60x	2.16	1.76x	2.34
	Medusa	1.91x	2.52	2.02x	2.67	1.89x	2.59	1.79x	2.48	1.42x	2.02	1.51x	2.09	1.76x	2.40
	Hydra	2.69x	3.60	2.98x	3.79	2.73x	3.66	2.66x	3.58	2.01x	2.70	2.25x	2.86	2.55x	3.37
	Eagle1	2.52x	3.97	2.82x	4.30	2.58x	4.01	2.37x	3.87	2.15x	3.43	2.05x	3.22	2.42x	3.80
	Eagle2	2.75x	4.94	3.12x	5.35	2.81x	4.94	2.63x	**4.85**	2.25x	4.11	2.17x	3.84	2.62x	4.67
	Jakiro	**2.99x**	**4.96**	**3.43x**	**5.36**	**3.11x**	**4.95**	**2.87x**	4.82	**2.50x**	**4.20**	**2.38x**	**3.84**	**2.88x**	**4.69**
L2 7B	Eagle2	2.70x	**4.70**	3.12x	**5.38**	2.78x	**4.76**	2.67x	**4.65**	2.27x	**4.09**	2.41x	**4.16**	2.66x	**4.63**
	Jakiro	**2.89x**	4.61	**3.32x**	5.26	**2.97x**	4.66	**2.81x**	4.47	**2.42x**	4.01	**2.56x**	4.07	**2.83x**	4.51
V 13B	Eagle2	3.02x	**4.84**	3.51x	**5.42**	3.11x	**4.82**	2.95x	**4.89**	2.60x	**4.28**	2.37x	3.69	2.93x	**4.66**
	Jakiro	**3.18x**	4.74	**3.70x**	5.36	**3.30x**	4.81	**3.03x**	4.68	**2.69x**	4.20	**2.52x**	**3.70**	**3.07x**	4.58
L2 13B	Eagle2	3.06x	**4.74**	3.62x	**5.53**	3.19x	4.88	2.96x	4.62	2.66x	4.24	2.68x	4.12	3.03x	4.69
	Jakiro	**3.22x**	4.72	**3.76x**	5.41	**3.41x**	**4.96**	**3.18x**	**4.67**	**2.83x**	**4.32**	**2.87x**	**4.17**	**3.21x**	**4.71**
V 33B	Eagle1	2.87x	3.69	3.41x	4.28	3.16x	3.93	2.73x	3.61	2.48x	3.35	2.26x	2.94	2.82x	3.63
	Eagle2	3.21x	**4.43**	3.89x	**5.20**	3.52x	4.66	3.11x	**4.37**	2.66x	3.83	2.44x	3.34	3.14x	**4.31**
	Jakiro	**3.34x**	4.40	**3.96x**	5.16	**3.67x**	4.66	**3.16x**	4.32	**2.71x**	**3.86**	**2.49x**	**3.38**	**3.22x**	4.30
L2 70B	Eagle1	2.83x	3.84	3.33x	4.44	2.95x	3.92	2.82x	3.77	2.37x	3.27	2.54x	3.42	2.81x	3.77
	Eagle2	2.98x	**4.51**	3.54x	**5.25**	3.10x	**4.63**	**2.94x**	**4.42**	2.39x	3.72	2.57x	**3.88**	2.92x	**4.40**
	Jakiro	**3.01x**	4.46	**3.58x**	5.19	**3.16x**	4.58	2.93x	4.32	**2.52x**	**3.86**	**2.58x**	3.83	**2.96x**	4.37
L3 8B	Eagle1	1.84x	3.07	2.27x	3.72	2.25x	3.70	2.26x	3.96	1.10x	1.76	1.87x	3.11	1.93x	3.22
	Eagle2	2.63x	**4.32**	3.08x	**5.06**	2.79x	**4.46**	2.89x	**4.87**	2.18x	3.81	2.20x	**3.53**	2.63x	**4.34**
	Jakiro	**2.74x**	4.23	**3.25x**	5.01	**2.92x**	4.38	**3.02x**	4.78	**2.27x**	**3.84**	**2.37x**	3.46	**2.76x**	4.28
L3 70B	Eagle1	2.43x	3.33	3.01x	4.15	2.53x	3.87	3.10x	4.08	1.67x	1.88	2.61x	3.28	2.86x	3.43
	Eagle2	2.58x	**4.14**	3.10x	**5.09**	2.82x	**4.34**	3.12x	**4.42**	2.41x	3.72	2.58x	3.88	2.92x	**4.32**
	Jakiro	**2.65x**	4.06	**3.28x**	5.03	**2.97x**	4.29	**3.18x**	4.35	**2.46x**	**3.75**	**2.63x**	**3.92**	**2.94x**	4.22
Temperature=1															
V 7B	SpS	1.50x	1.87	1.55x	1.95	1.53x	1.82	1.56x	1.85	1.63x	1.91	1.33x	1.72	1.52x	1.85
	Eagle1	2.18x	3.59	2.37x	3.82	2.16x	3.56	2.12x	3.72	1.88x	3.10	1.78x	2.91	2.08x	3.45
	Eagle2	2.39x	4.26	2.64x	4.67	2.37x	4.49	2.28x	4.28	2.01x	3.77	1.95x	3.54	2.27x	4.17
	Jakiro	**3.18x**	**5.65**	**2.92x**	**4.70**	**2.91x**	**5.50**	**2.99x**	**5.67**	**2.70x**	**5.33**	**2.80x**	**5.50**	**2.92x**	**5.39**
L2 7B	Eagle2	2.61x	4.48	2.86x	**5.04**	2.67x	**4.72**	2.48x	4.37	2.14x	3.92	2.30x	4.06	2.51x	4.43
	Jakiro	**2.93x**	**5.16**	**3.04x**	4.94	**2.82x**	4.61	**3.04x**	**5.22**	**2.99x**	**5.37**	**2.73x**	**4.81**	**2.93x**	**5.02**
V 13B	Eagle2	2.72x	**4.30**	3.09x	4.78	2.85x	**4.52**	2.63x	**4.33**	2.38x	**3.94**	2.22x	**3.57**	2.65x	4.24
	Jakiro	**2.84x**	4.27	**3.31x**	**4.95**	**3.00x**	4.50	**2.73x**	4.31	**2.45x**	3.88	**2.36x**	3.54	**2.78x**	**4.24**
L2 13B	Eagle2	2.90x	**4.58**	3.45x	**5.37**	3.04x	4.73	2.86x	**4.44**	2.56x	**4.21**	2.62x	4.04	2.90x	**4.56**
	Jakiro	**3.06x**	4.49	**3.60x**	5.22	**3.26x**	**4.76**	**3.01x**	4.43	**2.71x**	4.15	**2.83x**	**4.13**	**3.08x**	4.53
V 33B	Eagle2	3.04x	4.22	3.52x	4.78	3.41x	4.55	2.87x	3.99	2.56x	3.71	2.40x	3.27	2.97x	4.09
	Jakiro	**3.16x**	**4.31**	**3.64x**	**4.84**	**3.52x**	**4.63**	**2.98x**	**4.02**	**2.68x**	**3.85**	**2.52x**	**3.38**	**3.08x**	**4.17**
L2 70B	Eagle2	2.96x	4.48	3.49x	5.18	3.09x	4.59	2.97x	4.49	2.34x	3.64	2.54x	3.84	2.90x	4.37
	Jakiro	**3.68x**	**5.16**	**3.62x**	**5.25**	**3.43x**	**4.71**	**3.60x**	**4.90**	**3.55x**	**5.18**	**3.56x**	**4.86**	**3.57x**	**5.01**
L3 8B	Eagle2	2.31x	3.97	2.96x	4.83	2.65x	4.36	2.52x	4.50	1.99x	3.52	2.05x	3.36	2.41x	4.09
	Jakiro	**2.58x**	**4.66**	**3.02x**	**4.91**	**2.70x**	**4.44**	**2.64x**	**4.53**	**2.06x**	**3.59**	**2.32x**	**3.95**	**2.55x**	**4.35**
L3 70B	Eagle2	3.11x	4.98	2.83x	4.35	2.73x	4.67	2.10x	3.64	2.34x	3.52	2.30x	3.42	2.57x	4.10
	Jakiro	**3.26x**	**5.04**	**2.89x**	**4.42**	**2.91x**	**4.77**	**2.46x**	**3.78**	**2.35x**	**3.75**	**2.32x**	**3.56**	**2.70x**	**4.22**

Table 7: Full evaluation of speedup ratios (S) and acceptance lengths (τ) on MI250 GPU across all models and benchmarks. V: Vicuna, L2: LLaMA2-Chat, L3: LLaMA3-Instruct. SpS: standard speculative sampling with Vicuna-68M draft model. Jakiro demonstrates consistent improvements over all baselines. Note: Medusa-style methods are excluded in Temperature=1 due to relaxed acceptance conditions that compromise output distribution preservation.

amplifies the primary expert's signal while $\beta_2 < \beta_1$ subtracts a smaller correction from the secondary expert. (3) The optimal $\lambda_1 = 0.1, \lambda_2 = 0.05$ balances the numerical scales of classification (cross-entropy) and regression (SmoothL1) losses. (4) The same hyperparameters transfer across Vicuna 7B/13B/33B and LLaMA2 7B/13B/70B without per-model tuning, since β_1, β_2 operate on normalized feature representations and λ_1, λ_2 balance scale-invariant loss magnitudes.

A.4.8 Dynamic Expert Selection (Adaptive K)

We fix $K=2$ in the main paper to construct the left/right dual branches of the draft tree, but the

		MT-bench		HumanEval		GSM8K		Alpaca		CNN/DM		Natural Ques.		Mean	
Model	Method	S	τ	S	τ	S	τ	S	τ	S	τ	S	τ	S	τ
Temperature=0															
V 7B	Eagle1	2.69x	3.98	2.98x	4.29	2.74x	4.00	2.56x	3.87	2.21x	3.42	2.17x	3.21	2.56x	3.80
	Eagle2	2.92x	4.98	3.24x	5.34	2.94x	4.95	2.76x	**4.87**	2.28x	4.12	2.26x	3.81	2.73x	4.68
	Jakiro	**3.02x**	**4.98**	**3.40x**	**5.37**	**3.08x**	**4.97**	**2.86x**	4.82	**2.38x**	**4.17**	**2.36x**	**3.83**	**2.85x**	**4.69**
L2 7B	Eagle1	2.62x	3.82	2.94x	4.30	2.66x	3.90	2.53x	3.70	2.24x	3.41	2.34x	3.44	2.55x	3.76
	Eagle2	2.81x	**4.71**	3.23x	**5.39**	2.86x	**4.76**	2.79x	**4.66**	2.31x	**4.12**	2.51x	**4.19**	2.75x	**4.64**
	Jakiro	**2.87x**	4.61	**3.29x**	5.25	**2.93x**	4.64	**2.81x**	4.47	**2.32x**	4.01	**2.56x**	4.07	**2.80x**	4.51
V 13B	Eagle1	2.90x	4.00	3.28x	4.39	2.94x	3.97	2.74x	3.95	2.45x	3.52	2.28x	3.11	2.77x	3.82
	Eagle2	2.97x	**4.83**	3.45x	**5.41**	3.03x	4.79	2.89x	**4.89**	2.46x	**4.22**	2.33x	**3.74**	2.86x	**4.65**
	Jakiro	**3.11x**	4.76	**3.61x**	5.36	**3.22x**	**4.81**	**2.96x**	4.69	**2.49x**	4.13	**2.45x**	3.67	**2.97x**	4.57
L2 13B	Eagle1	2.89x	3.93	3.39x	4.52	2.99x	4.03	2.81x	3.83	2.54x	3.59	2.58x	3.47	2.87x	3.89
	Eagle2	2.96x	**4.74**	3.52x	**5.52**	3.09x	4.90	2.88x	4.60	2.53x	4.25	2.60x	4.11	2.93x	4.69
	Jakiro	**3.11x**	4.73	**3.62x**	5.42	**3.30x**	**4.94**	**3.07x**	**4.65**	**2.62x**	**4.30**	**2.79x**	**4.16**	**3.08x**	**4.70**
Temperature=1															
V 7B	Eagle1	2.23x	3.54	2.41x	3.82	2.23x	3.66	2.23x	3.63	1.91x	3.13	1.86x	2.95	2.15x	3.45
	Eagle2	2.46x	4.29	2.72x	4.63	2.47x	4.41	2.41x	4.47	2.09x	3.82	2.03x	3.50	2.36x	4.19
	Jakiro	**3.24x**	**5.64**	**2.95x**	**4.72**	**2.84x**	**5.76**	**2.97x**	**5.67**	**2.84x**	**5.59**	**2.85x**	**5.04**	**2.95x**	**5.40**
L2 7B	Eagle1	2.24x	3.59	2.55x	4.08	2.39x	3.80	2.21x	3.51	1.96x	3.18	2.04x	3.24	2.23x	3.56
	Eagle2	2.63x	4.53	2.97x	5.04	2.79x	4.76	2.63x	4.53	2.18x	3.93	2.34x	3.99	2.59x	4.46
	Jakiro	**2.97x**	**5.13**	**2.99x**	**4.93**	**2.80x**	**4.69**	**3.06x**	**5.19**	**2.97x**	**5.46**	**2.80x**	**4.85**	**2.93x**	**5.04**
V 13B	Eagle1	2.51x	3.66	2.84x	4.02	2.58x	3.74	2.49x	3.77	2.22x	3.32	2.11x	3.03	2.46x	3.59
	Eagle2	2.68x	4.38	3.03x	4.86	2.69x	4.46	2.53x	4.48	2.24x	3.91	2.15x	3.53	2.55x	4.27
	Jakiro	**3.04x**	**4.70**	**3.28x**	**4.89**	**2.90x**	**5.01**	**2.70x**	**4.46**	**2.40x**	**3.90**	**2.44x**	**3.88**	**2.79x**	**4.47**
L2 13B	Eagle1	2.59x	3.69	3.03x	4.23	2.69x	3.81	2.54x	3.64	2.34x	3.45	2.37x	3.34	2.59x	3.69
	Eagle2	2.82x	4.61	3.37x	5.40	2.97x	4.79	2.75x	4.52	2.44x	4.20	2.54x	4.08	2.81x	4.60
	Jakiro	**3.14x**	**4.72**	**3.53x**	**5.29**	**3.33x**	**4.90**	**2.95x**	**4.52**	**2.91x**	**4.50**	**2.77x**	**4.15**	**3.12x**	**4.68**

Table 8: Comprehensive evaluation of speedup ratios (S) and average acceptance lengths (τ) across different methods on A40-45G GPU. V: Vicuna, L2: LLaMA2-Chat, SpS: Standard speculative sampling with Vicuna-68M draft model. Jakiro consistently outperforms Eagle2 across all model sizes and tasks, demonstrating the effectiveness of our MoE-based decoupling approach. Note: We exclude comparison with Medusa-style methods that employ relaxed acceptance conditions in non-greedy settings, as these may compromise output distribution preservation.

framework is not restricted to this choice. We explore adaptive K strategies on Vicuna 7B (A100-40G, T=1) in Table 12. The *entropy-adaptive* strategy sets $K{=}1$ when the router confidence is high (low entropy) and $K{=}2$ otherwise; this achieves slightly better acceptance on HumanEval ($+1.5\%$ in τ) at a small overall speedup cost due to routing overhead. The *uncertainty-adaptive* strategy ($K \in \{2, 3\}$) improves acceptance length ($+2.5\%$) but reduces speedup (-6.2%) because the third expert adds computational cost. This aligns with the ablation findings (Table 4): increasing N (and K) improves τ but reduces walltime speedup on current hardware. We expect dynamic K to be most promising for larger target models (>70B), where routing overhead becomes negligible relative to target-model computation.

A.4.9 Effect of Tree Depth (γ)

We further investigate how MoE-based decoupling interacts with tree depth γ on Vicuna 7B (A100-40G, T=0). As shown in Table 13, Jakiro maintains competitive acceptance lengths at deeper levels: τ continues to grow from 5.03 ($\gamma{=}6$) to 5.25 ($\gamma{=}8$) on MT-bench, whereas prior single-feature methods typically exhibit sharper drop-offs beyond $\gamma{=}6$ due to error accumulation. The default $\gamma{=}6$ remains optimal for walltime speedup because deeper drafting incurs additional forward-pass latency that outweighs the marginal acceptance gain. For larger target models where verification cost dominates, deeper trees ($\gamma{=}7$–8) may become preferable.

A.4.10 Multi-batch Analysis

We extended our experiments to include batch sizes greater than 1 using Vicuna 7B on A40, as detailed in Table 10. The findings indicate that Jakiro's

		MT-bench		HumanEval		GSM8K		Alpaca		CNN/DM		Natural Ques.		Mean	
Method	Temperature	S	τ	S	τ	S	τ	S	τ	S	τ	S	τ	S	τ
Eagle2	0	2.13x	4.16	2.23x	4.18	2.05x	3.93	1.70x	3.30	1.75x	3.43	1.44x	2.73	1.88x	3.62
Jakiro	0	**2.28x**	**4.20**	**2.36x**	**4.20**	**2.16x**	**3.98**	**1.82x**	**3.35**	**1.88x**	**3.46**	**1.53x**	**2.75**	**2.01x**	**3.64**
Eagle2	1	1.61x	3.18	1.69x	3.28	1.75x	3.41	1.30x	2.56	1.18x	2.36	1.13x	2.19	1.44x	2.83
Jakiro	1	**1.72x**	**3.20**	**1.81x**	**3.30**	**1.85x**	**3.45**	**1.38x**	**2.60**	**1.25x**	**2.40**	**1.20x**	**2.25**	**1.54x**	**2.85**

Table 9: Performance of Qwen2-7B-Instruct on A40 (Speedup Ratios: S, Average Accepted Length: τ)

Method	BS	MT-bench	HumanEval	GSM8K	Avg
Jakiro	1	3.02x	3.40x	3.08x	3.17x
Jakiro + FD	1	**4.13x**	**4.55x**	**4.17x**	**4.28x**
Jakiro	2	2.98x	3.35x	3.05x	3.13x
Jakiro + FD	2	**3.98x**	**4.42x**	**4.08x**	**4.16x**
Jakiro	4	2.85x	3.25x	2.95x	3.02x
Jakiro + FD	4	**3.72x**	**4.18x**	**3.86x**	**3.92x**
Jakiro	8	2.72x	3.10x	2.82x	2.88x
Jakiro + FD	8	**3.48x**	**3.91x**	**3.62x**	**3.67x**

Table 10: Speedup ratios of Vicuna 7B (T=0) under various batch sizes, with and without Flash Decoding (FD) integration.

		MT-bench		HumanEval	
β_1	β_2	S	τ	S	τ
1.0	0.1	3.26x	4.92	3.72x	5.38
1.0	0.3	3.29x	4.96	3.75x	5.41
1.2	0.2	3.31x	4.99	3.78x	5.44
1.2	**0.3**	**3.34x**	**5.03**	**3.81x**	**5.48**
1.3	0.3	3.32x	5.01	3.79x	5.45
1.5	0.5	3.24x	4.88	3.68x	5.32
λ_1	λ_2	S	τ	S	τ
0.05	0.02	3.28x	4.95	3.74x	5.40
0.08	0.04	3.31x	4.99	3.77x	5.44
0.1	**0.05**	**3.34x**	**5.03**	**3.81x**	**5.48**
0.15	0.08	3.32x	5.00	3.79x	5.45
0.2	0.1	3.27x	4.92	3.72x	5.36

Table 11: Hyperparameter sensitivity of contrastive parameters (β_1, β_2) and loss weights (λ_1, λ_2) on Vicuna 7B (A100-40G, T=0).

speedup decreases with larger batch sizes, though the method still provides meaningful acceleration. This trend is consistent with the inherent characteristics of speculative decoding, where increasing batch size transitions the problem from being memory-bound to compute-bound, potentially resulting in diminishing returns (Sadhukhan et al., 2025). Current optimizations in speculative decoding primarily target batch size = 1 due to two main challenges:

- **Sequence Length Variability:** Different acceptance rates across sequences in a batch lead to varying candidate sequence lengths post-drafting.
- **Verification Overhead:** This variability heightens computational demands during the parallel verification phase of the target model.

While existing speculative decoding implementations (e.g., foundational works like Medusa, EAGLE, Hydra) mainly focus on batch size = 1, optimizing for larger batch sizes remains an open challenge and a promising avenue for future exploration.

Strategy	MT-bench		HumanEval	
	S	τ	S	τ
Fixed K=2 (default)	**3.86x**	5.68	3.25x	4.92
Entropy-adaptive $K \in \{1, 2\}$	3.78x	5.55	**3.30x**	5.02
Uncertainty-adaptive $K \in \{2, 3\}$	3.62x	**5.82**	3.08x	**5.15**

Table 12: Adaptive K strategies on Vicuna 7B (A100-40G, T=1).

γ	MT-bench		HumanEval	
	S	τ	S	τ
4	3.10x	3.85	3.52x	4.32
5	3.25x	4.52	3.68x	4.95
6 (default)	**3.34x**	5.03	**3.81x**	5.48
7	3.30x	5.18	3.76x	5.62
8	3.22x	**5.25**	3.70x	**5.68**

Table 13: Effect of draft-tree depth γ on Vicuna 7B (A100-40G, T=0).

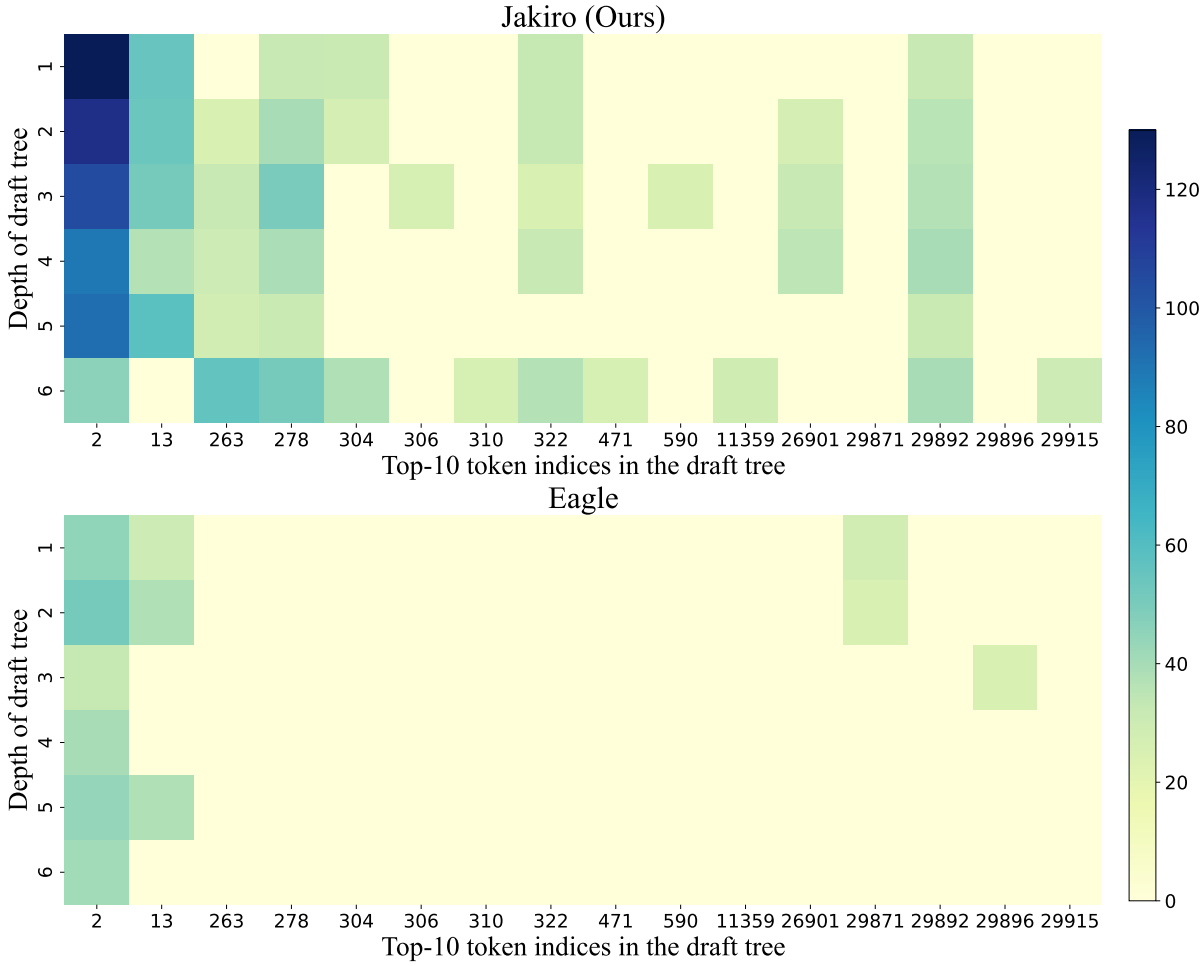


Figure 7: Frequency heatmap of the top-10 drafted tokens sampled during a response generation.