Talon: Breaking the Synchronization Barrier in Speculative Decoding with Hybrid Model-based and Retrieve-based Drafting

Xiangxiang Gao*, Weisheng Xie *†, Lixin, Xuwei Fang, Chen Hang, Changqun Li, Yuhan Lin, Xiaolong Xu

Bestpay AI Lab, Shanghai, 200080 China

{gaoxiangxiang, xieweisheng, lixin, fangxuwei, hangchen, lichangqun, linyuhan, xuxiaolong}@bestpay.com

Abstract

Large Language Models face fundamental deployment challenges due to the computational demands of auto-regressive token-by-token generation. While speculative decoding has emerged as a promising acceleration technique through its draft-then-verify framework, current implementations suffer from two critical limitations: (1) mutual waiting problem caused by sequential dependencies between draft generation and verification phases, and (2) constrained token acceptance rates where retrieval-based drafting methods under-perform in general domains while models-based drafting approaches show reduced efficacy in knowledge-intensive scenarios. To address these challenges, we propose Talon, a novel parallel inference architecture featuring two key innovations: (1) **a novel asynchronous execution paradigm** that decouples draft generation from verification, effectively eliminating synchronization bottlenecks, and (2) **an adaptive hybrid drafting strategy** that dynamically combines model-based and retrieval-based approaches to improve token acceptance rates across diverse domains. Extensive evaluations across standard benchmarks (MT-Bench, HumanEval, GSM8K, Alpaca, CNN/DM) demonstrate Talon's exceptional performance, achieving 4.04x–6.52x acceleration across multiple model families including Vicuna, Deepseek, and LLaMA series. These results represent a significant advancement over existing speculative decoding methods (EAGLE 1-3, Hydra, Medusa, Lookahead, SPS, and PLD), establishing a new paradigm for speculative decoding.

Introduction

Large Language Models (LLMs) such as GPT, LLaMA, and Deepseek (OpenAI et al. 2024; Bommasani et al. 2021; Touvron et al. 2023; DeepSeek-AI et al. 2025) have been widely adopted across various fields. Despite demonstrating near-logarithmic performance gains with increased scale, the accompanying computational requirements create fundamental challenges for efficient deployment. The primary source of their computational demands stems from auto-regressive token-by-token decoding process, which not only creates excessive inference latency but also necessitates quadratic growth in resource requirements with model scale (Shazeer 2019; Ivanov et al. 2020; Pope et al. 2023). This constraint severely limits efficient deployment, particularly in latency-sensitive scenarios where both time and costs become prohibitive factors (Kaplan et al. 2020; Hoffmann et al. 2022).

Speculative decoding addresses this through a draft-then-verify framework consisting of two coordinated phases: (1) drafting phase where drafter generates multiple tokens in parallel, followed by (2) verification phase where target LLMs simultaneously validates these candidates. To guarantee lossless acceleration, the system employs strict acceptance criteria: all non-matching tokens are discarded while only verified tokens identical to target model's computations are retained. By transforming sequential decoding into a parallelization operation, this technique achieves substantial improvements in speed while maintaining accuracy (Zhang, Liu, and Song 2024; Hu et al. 2025; Ryu and Kim 2024).

Although speculative decoding provides significant acceleration benefits, its practical implementation encounters two fundamental limitations: (1) the mutual waiting problem and (2) the constrained token acceptance rate. First, the mutual waiting problem arises from sequential dependencies in the draft-verify paradigm. The target model remains blocked during draft generation, while the draft model cannot proceed during verification phases. This alternating stall pattern introduces synchronization bottlenecks that degrade theoretical throughput improvements (Liu et al. 2025). Second, current implementations face inherent trade-offs in token acceptance rates. Current drafters primarily rely on additional draft models or retrieve context from databases. Retrieval-based approaches consistently demonstrate inferior performance in general domains, while model-based approaches exhibit reduced efficacy in knowledge-intensive domains (Quan et al. 2025; Li et al. 2025a). Limited acceptance rates necessitate excessive drafting-verification cycles, ultimately capping achievable acceleration.

To overcome the inherent limitations of speculative decoding, we propose Talon, a novel parallel inference architecture that integrates: (1) **an asynchronous speculative decoding architecture** eliminates the mutual waiting problem by temporally decoupling the draft generation and verification process. This design achieves full pipeline utilization by eliminating computation stalls between the drafting and verification phases. (2) **a hybrid drafting strategy** that intelligently integrates model-based predictions with

*These authors contributed equally.

†Corresponding Author

retrieval-based methods. This adaptive hybrid drafting approach maintains high token acceptance rates across general tasks to knowledge-intensive domains. This dual optimization simultaneously reduces drafting-verification overhead while improving acceleration performance.

Extensive evaluations across standard benchmarks, including MT-Bench for multi-turn dialogue, HumanEval for code generation, GSM8K for mathematical reasoning, Alpaca for instruction following, and CNN/DM for summarization, demonstrating Talon's exceptional performance, achieving 4.04x to 6.52x acceleration across diverse model families. These results not only significantly outperform existing speculative decoding methods by substantial margins, but also establish a new paradigm for efficient LLM inference through its innovative architectural approach.

Related Works

Model-based Speculative Decoding

In the realm of speculative decoding research, model-based methods have established themselves as the prevailing framework for acceleration strategies, consistently outperforming other approaches in empirical assessments. The field has witnessed several notable advancements: SPS (Chen et al. 2023a) pioneered this direction by employing a 4B parameter drafter to generate draft tokens, which are then verified using a 70B parameter LLM. SpecInfer (Miao et al. 2024) introduces an innovative tree-based inference and verification system that organizes draft tokens into hierarchical structures, employing tree-based decoding to validate all candidate sequences against the LLM. PASS (Monea, Joulin, and Grave 2023) enhances parallel decoding capabilities by introducing trainable token embeddings that are optimized through fine-tuning. Medusa (Cai et al. 2024a) presentes an efficient solution by maintaining the frozen state of the backbone model while integrating lightweight auxiliary heads, enabling simultaneous multi-token generation.

Notably, the EAGLE-Series method currently represents the cutting edge in speculative decoding. The series has evolved through three key iterations: EAGLE-1 (Li et al. 2024b) introduced a novel approach by combining the current embedding input with previous target model features, substantially boosting token acceptance rates. EAGLE-2 (Li et al. 2024a) retained the same core design but replaced static tree structures with dynamic drafting mechanisms during decoding, leading to higher-quality candidate token trees. EAGLE-3 (Li et al. 2025b) shifted from feature prediction to direct token prediction and abandoned top-layer feature dependence in favor of multi-layer feature fusion via a training-time test technique. This enhancement not only improved performance but also allowed the draft model to fully leverage large-scale training data.

Although model-based speculative decoding demonstrates strong performance in general domains, its effectiveness remains limited in knowledge-intensive applications.

Retrieval-based Speculative Decoding

Retrieval-based approaches have emerged as an effective approach for accelerating LLMs in knowledge-intensive scenarios. This training-free paradigm enables seamless integration with existing language models while maintaining computational efficiency, particularly excelling in domains requiring specialized knowledge representation.

Among existing retrieval-based approaches, PLD (Prompt Lookup Decoding) (Saxena 2023a) represents a lightweight solution that retrieves candidate token sequences directly from input prompts through string pattern matching, eliminating the need for dedicated draft models. Lookahead (Zhao et al. 2024) introduces an innovative framework that enables parallel generation of multiple n-grams within a single forward pass, significantly reducing inference latency.

REST (Retrieval-based Speculative Sampling) (He et al. 2024) is a novel algorithm that extends retrieval techniques for speculative execution. Departing from PLD's input-bound retrieval mechanism, REST operates on a pre-constructed knowledge database, dynamically fetching contextually relevant tokens without draft model dependencies. The approach converts existing corpora into structured retrieval libraries, organizes results into verifiable draft trees, and operates without additional draft model parameters. While Retrieval-based Speculative Decoding offers user-friendly deployment advantages, its speculative accuracy currently lags behind model-based approaches in general domains due to limitations in static knowledge retrieval.

Therefore, a hybrid drafting strategy that adaptively combines model-based and retrieval-based drafting approaches helps improve the acceptance rate of draft tokens in both general and knowledge-intensive domains, further improving the inference speed of LLMs.

Synchronous and Asynchronous Architecture

Existing model-based speculative decoding methods predominantly employ synchronous architectures that require alternating between draft generation and verification phases (Chen et al. 2023a). This tight coupling creates inherent pipeline inefficiencies, as each process must wait for the other to complete. While recent advances like Medusa (Cai et al. 2024a) and PASS (Monea, Joulin, and Grave 2023) mitigate drafting latency through parallel multi-token generation, they still maintain sequential dependencies between drafting and verification phases. Furthermore, their inference acceleration remains fundamentally constrained by the drafter's suboptimal token acceptance rates.

Retrieval-based methods (He et al. 2024), while capable of rapid draft tokens generation, suffer from limited token acceptance rates in general domains. This deficiency forces multiple verification iterations, creating cumulative latency that significantly offsets potential acceleration benefits. Consequently, the practical efficiency gains often fall substantially short of theoretical expectations.

To address these, we present an asynchronous architecture that decouples drafting and verification phases, eliminating their mutual blocking constraints. The framework incorporates a hybrid verification mechanism combining retrieval and model-based methods, which synergistically improves token acceptance rates. This dual optimization reduces drafting-verification cycles overhead, achieving superior acceleration compared to existing methods.

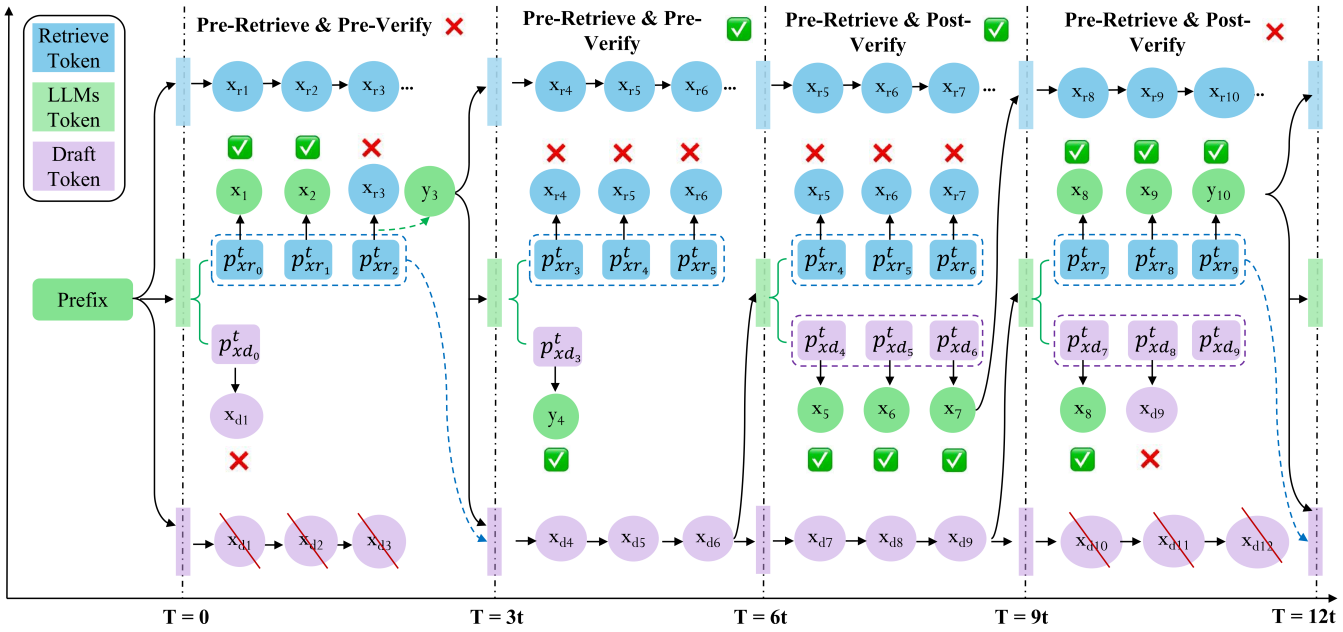

Figure 1: Schematic diagram of Talon. At $T = 0$, M_d generates drafting tokens x_{d1}, x_{d2}, x_{d3} and M_r generates retrieving tokens $x_{r1}, x_{r2}, x_{r3}...$, M_q accepts x_{r1}, x_{r2}, rejects x_{d1}, x_{r3} and resample y_3 with the pre-retrieve and pre-verify strategy. At $T = 3t$, M_p rejects the retrieving tokens $x_{r4}, x_{r5}, x_{r6}...$ and accepts the drafting tokens x_4, and switches to the pre-retrieve and post-verify strategy. At $T = 6t$, M_p rejects all retrieve tokens $x_{r5}, x_{r6}, x_{r7}...$ and accepts all draft tokens x_{d4}, x_{d5}, x_{d6} in the last decoding step, while M_d continues drafting x_{d7}, x_{d8}, x_{d9}. At $T = 9t$, M_q accepts the retrieving tokens x_{r8}, x_{r9}, rejects x_{d9}, x_{r10}, resample y_{10} and then switches to the pre-retrieve and pre-verify strategy. The final output tokens are $[x_1, x_2, y_3, x_4, x_5, x_6, x_7, x_8, x_9, y_{10}]$.

Talon

In this section, we provide a detailed description of the implementation of Talon.

System Architecture Overview

Talon's core architecture employs a multi-engine parallel design, as illustrated in Figure 1, which consists of three key components working collaboratively:

- **Model Drafter** (M_d): A lightweight neural network generating speculative tokens $\{x_{d_i}\}_{i=1}^{n}$
- **Retrieval Drafter** (M_r): A retrieval module that generates contextually grounded tokens $\{x_{r_j}\}_{j=1}^{m}$ through suffix-matching operations on datastore D
- **Verification Decider** (M_q): Implements speculative sampling to verify and select tokens

Although M_r's retrieval and M_q's verification are inherently sequential, we treat them as parallel operations due to M_r's high generation speed. These components implement the speculative execution algorithm through parallelized drafting-verification workflow, with detailed coordination logic described in Algorithm 1.

Asynchronous Execution Paradigm

Talon architecture employs two complementary strategies to maximize hardware utilization during speculative decoding by parallelizing underutilized computational resources.

The first approach, the **pre-retrieve and pre-verify strategy**, addresses idleness in the target model (M_p) during draft token generation. While the draft model M_d sequentially generates γ speculative tokens through γ forward passes, we observe that the target model M_p remains idle despite having access to the current prefix $\mathbf{x}$. To leverage this opportunity, we simultaneously execute three critical operations: (1) initiate $M_r(\mathbf{x})$ for suffix-matching token retrieval from datastore D with near-zero latency ($\leq 0.1t$) due to optimized indexing, (2) maintain M_d's sequential generation of $\{x_{d_i}\}_{i=1}^{\gamma}$, and (3) execute $M_p(\mathbf{x})$ to pre-compute probability distributions for both speculative and retrieved candidates.

As illustrated in Figure 1, at the timestamp of T = 0, the draft model M_d generates draft tokens x_{d1}, x_{d2}, x_{d3} while the retrieval model M_r simultaneously produces retrieved tokens $x_{r1}, x_{r2}, x_{r3}...$ The target model M_p computes the probability distribution p over the combined candidate tokens, rejects the draft token x_{d1} based on the acceptance criterion $\frac{p(x_{d1}|\mathbf{x})}{q(x_{d1}|\mathbf{x})} < \tau$, accepts retrieved tokens x_{r1}, x_{r2} while rejecting x_{r3}, and samples an alternative token y_3 through the pre-retrieve and pre-verify strategy.

At the timestamp of T = 3t, the draft model M_d generates draft tokens x_{d4}, x_{d5}, x_{d6} while the retrieval model M_r produces retrieved tokens $x_{r4}, x_{r5}, x_{r6}..$, and the target model M_p rejects retrieved tokens $x_{r4}, x_{r5}, x_{r6}..$ while accepting the drafting token x_{d4}, triggering a switch to the pre-retrieve and post-verify strategy where x_{d5}, x_{d6} undergo subsequent

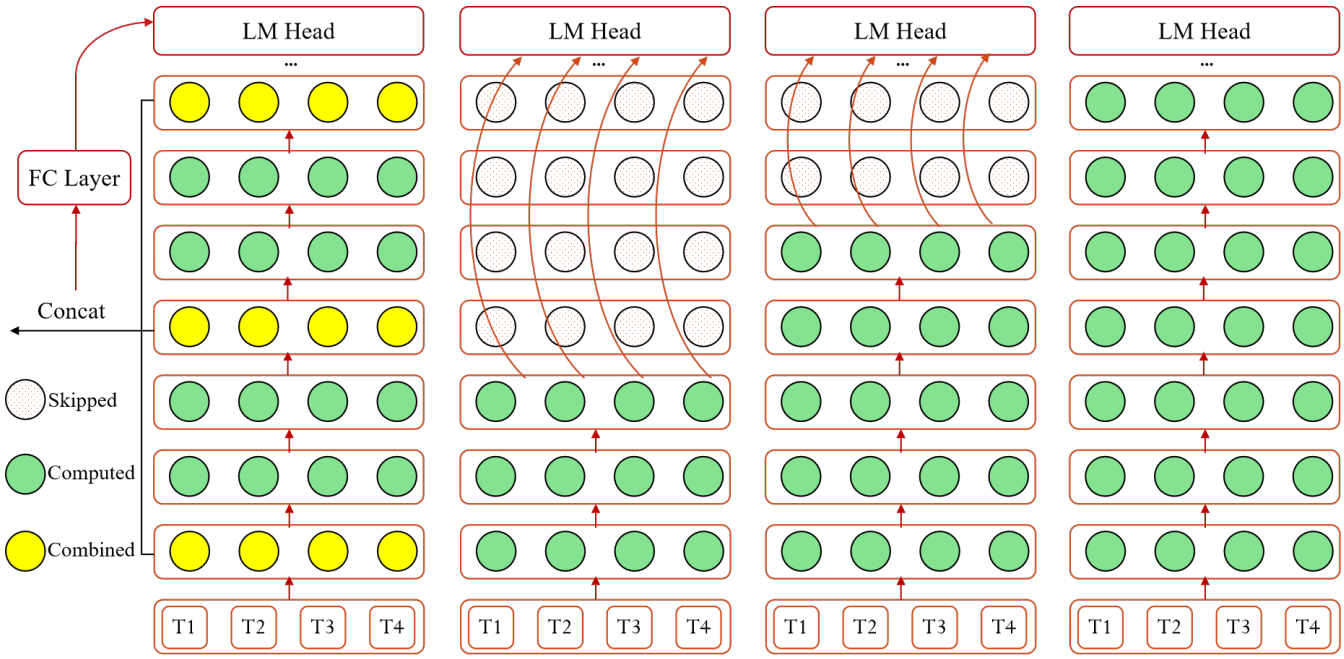

Figure 2: Schematic diagram of knowledge base construction. Green circles denote computed layers, dotted circles indicate skipped layers, and yellow circles represent fused layers. We propose four distinct construction approaches: (1) Shallow layer processing followed by direct output via LLM-Head, (2) Intermediate layer followed by direct output via LLM-Head, (3) Full-layer processing followed by direct output via LLM-Head, and (4) Fusion of multiple intermediate layer hidden states before final output through the LLM-Head module.

verification.

According to this strategy, if the first draft token x_{d1} is rejected, the subsequent draft tokens x_{d2}, x_{d3} are discarded due to their conditional dependency on x_{d1}'s acceptance. The key distinction from conventional speculative decoding lies in the handling of rejected initial tokens: while standard approaches must fall back to generating just one token per step in such cases, our method achieves progressive sequence expansion by incorporating both the accepted retrieved tokens x_{r1}, x_{r2} and the newly sampled token y_3, resulting in $\gamma_{\text{eff}} = k + 1$ effective tokens per step (where k is the count of accepted retrieved tokens). This approach circumvents redundant verification of invalid draft tokens while maintaining decoding throughput through retrieved token integration. The process then immediately transitions to the next drafting phase using the updated prefix $\mathbf{x} \oplus [x_{r1}, x_{r2}, y_3]$, bypassing intermediate verification steps.

The second approach, the **pre-retrieve and post-verify strategy**, eliminates draft model idleness during verification phases. While the target model M_p verifies γ candidate tokens, we observe that the draft model M_d would otherwise remain idle awaiting verification completion. To exploit this resource availability, we concurrently execute three critical operations: (1) initiate $M_d(\mathbf{x} \oplus [\mathbf{c}])$ to generate draft tokens for the subsequent process, $\mathbf{c}$ is candidate tokens from pre-retrieve and pre-verify strategy, (2) invoke $M_r(\mathbf{x} \oplus [\mathbf{c}])$ for suffix-matching token retrieval from datastore D, and (3) compute verification probabilities for subsequent tokens using $M_p(\mathbf{x} \oplus [\mathbf{c}])$.

In Figure 1, at the timestamp of T = 6t, the draft model M_d generates speculative tokens x_{d7}, x_{d8}, x_{d9} while the retrieval model M_r produces retrieved tokens x_{r5}, x_{r6}, x_{r7}..., and the target model M_p completes verification of candidate tokens: it rejects all retrieved tokens x_{r5}, x_{r6}, x_{r7}... and accepts all drafting tokens x_{d7}, x_{d8}, x_{d9}. Consequently, the pre-generated draft tokens x_{d7}, x_{d8}, x_{d9} and retrieved tokens x_{r5}, x_{r6}, x_{r7}... immediately enter the verification phase without an intermediate drafting phase.

At the timestamp of T = 9t, the draft model M_d generates draft tokens $x_{d10}, x_{d11}, x_{d12}$ while the retrieval model M_r produces retrieved tokens x_{r8}, x_{r9}, x_{r10}..., and the target model M_p rejecting draft token x_{d9}, accepting retrieved tokens x_{r8} and x_{r9}, and sampling new token y_{10} though the pre-retrieve and post-verify strategy. The failure to accept all draft tokens triggers a strategic shift to pre-retrieve and post-verify strategy for subsequent draft. Finally, the prefix $x+[x_1, x_2, y_3, x_4, x_5, x_6, x_7, x_8, x_9, y_{10}]$ is input to the next drafting phase.

Taking together the two strategies, our Talon architecture consists of a draft model, a target model, a datastore, and two strategies to decode tokens.

Adaptive Hybrid Drafting Strategy

The adaptive hybrid drafting strategy dynamically integrates model-based token generation and retrieval-based token matching to maximize acceptance rates. This Strategy leverages both the predictive capabilities of the draft model M_d and contextually relevant tokens from the knowledge-

enhanced datastore D.

Model-based token generation employs the draft model M_d to autoregressively predict speculative tokens $\{x_{d_i}\}_{i=1}^{\gamma}$ conditioned on the current prefix $\mathbf{x}$. Each token generation requires a full forward pass of M_d. The sequential dependency chain necessitates that rejection of any x_{d_k} invalidates all subsequent tokens $\{x_{d_j}\}_{j>k}$, this constraint is mitigated through the hybrid design's retrieval augmentation.

For retrieval-based token generation, we employ a datastore to enable efficient token retrieval, utilizing suffix matching to rapidly identify relevant tokens from large-scale knowledge sources. This approach accelerates candidate draft token generation while enhancing draft tokens quality through context-aware knowledge augmentation.

The datastore D is the set of pairs $\{(\mathbf{c}_i, t_i)\}$, where $\mathbf{c}_i$ denotes the prefix context and t_i its corresponding continuation token; its construction is driven by four key innovations:

(1) Tokenizer-based segmentation populates surface-level token distributions; (2) Target model distribution alignment injects tokens generated from M_p's multi-layer outputs to ensure distributional fidelity; (3) Layer-specific token injection generates tokens from distinct transformer layers of M_p:

$$t_i^{(l)} = \arg\max_V (\mathbf{W}_l \cdot \mathbf{h}_l) \quad \forall l \in \{2, 16, 29\} \tag{1}$$

where $\{\mathbf{W}_l\}$ are layer-specific projection heads for low, mid, and high-level representations. The low-layer captures local syntactic patterns and positional relationships. The mid-layer encodes semantic compositionality and medium-range dependencies. The high-layer represents global discourse structure and long-range coherence. (4) Hierarchical feature fusion integrates representations by concatenating hidden states $[\mathbf{h}_2; \mathbf{h}_{16}; \mathbf{h}_{29}]$ followed by a nonlinear transformation, generating enriched tokens that capture cross-layer semantic patterns. These fused tokens combine syntactic precision from lower layers with semantic richness from higher layers.

In this paper, we construct the knowledge-enriched datastore D by systematically integrating multi-domain open-source datasets spanning programming proficiency, mathematical reasoning, instruction adherence, and summarization quality.

Following datastore construction, retrieval operations activate during decoding sequences. During decoding, given current context $\mathbf{s} = (x_1, \ldots, x_t)$, we extract its ℓ-token suffix $\mathbf{c}_{\text{suffix}} = (x_{t-\ell+1}, \ldots, x_t)$ for exact-match retrieval from datastore D. The operation retrieves all context-continuation pairs $\{(\mathbf{c}_i, t_i) \in D \mid \mathbf{c}_i = \mathbf{c}_{\text{suffix}}\}$, then constructs a candidate set $\mathcal{S} = \{(t_j, f_j)\}$ where f_j denotes occurrence count of token t_j across matched pairs. This procedure is formally implemented in Algorithm 1.

Building upon these retrieved candidates, the adaptive hybrid drafting tree generation integrates retrieval-based and model-based token candidates into a unified structure for efficient speculative execution. This process comprises two synergistic stages: retrieval tree construction with intelligent pruning, followed by fusion with the draft model's generation tree.

First, given candidate set $\mathcal{S} = \{(t_j, f_j)\}$ where t_j denotes token sequences, we construct retrieval tree $\mathcal{T}_r$ with each root-to-leaf path representing semantically coherent token sequences. Initialized with a root node, the tree sequentially inserts tokens from candidate sequences $\{t_j\}$ along the tree structure — updating the cumulative frequency f_j for existing paths or creating new branches with initialized frequency counters. When the node count $|\mathcal{T}_r|$ exceeds threshold m, frequency-based pruning activates: recursively removing lowest-cumulative-frequency subtrees bottom-up until $|\mathcal{T}_r| \leq m$. This preserves semantic integrity while maintaining computational efficiency.

Second, we employ tree fusion to solve the quadratic development of forward propagation time in large language models (LLMs), which involves integrating the draft model's generation tree $\mathcal{T}_{\text{draft}}$ with the pruned retrieval tree $\mathcal{T}_{\text{r}}$. This integration eliminates redundant computation through prefix sharing and structural optimization. The core mechanism employs longest prefix matching (LPM) to align branches across both trees. During fusion, we recursively traverse sibling nodes from the root downward, identifying maximal common token sequences between corresponding branches. When a shared prefix $[t_1, \ldots, t_k]$ is detected, we consolidate the matching segment into a single pathway. The terminal node t_k of this common prefix becomes a unified parent node, from which divergent subtrees $\mathcal{T}_{\text{draft}}$ and $\mathcal{T}_{\text{r}}$ extend as children. This hierarchical merging bypasses redundant attention computations for the shared prefix $t_{1:k}$. Through this process, we obtain final draft tree $\mathcal{T}_{combine}$.

Algorithm 1: Retrieval-Based Token Generation Algorithm

Input: Draft model M_d, target model M_p, input prefix $\mathbf{x}$, max tokens L, datastore D, suffix length ℓ
Output: Candidate set $\mathcal{S} = \{(t_j, f_j)\}$
$\mathcal{S} \leftarrow \emptyset$ {Initialize candidate set}
while $\ell > 0$ **do**
  $\mathbf{c}_{\text{suffix}} \leftarrow \mathbf{s}[\max(1, t-\ell+1) : t]$
  $\mathcal{R} \leftarrow \textsc{Search}(D, \mathbf{c}_{\text{suffix}})$ {Get all matched continuation tokens}
  if $\mathcal{R} \neq \emptyset$ **then**
    $\mathcal{T}_{\text{unique}} \leftarrow \emptyset$ {Unique tokens}
    $\mathcal{F} \leftarrow \emptyset$ {Frequency counter}
    for all $t_i \in \mathcal{R}$ **do**
      if $t_i \notin \mathcal{T}_{\text{unique}}$ **then**
        $\mathcal{T}_{\text{unique}} \leftarrow \mathcal{T}_{\text{unique}} \cup \{t_i\}$
        $\mathcal{F}[t_i] \leftarrow 1$
      else
        $\mathcal{F}[t_i] \leftarrow \mathcal{F}[t_i] + 1$
      end if
    end for
    for all $t_j \in \mathcal{T}_{\text{unique}}$ **do**
      $\mathcal{S} \leftarrow \mathcal{S} \cup \{(t_j, \mathcal{F}[t_j])\}$
    end for
    break {Exit after first successful retrieval}
  else
    $\ell \leftarrow \ell - 1$ {Decrease suffix length}
  end if
end while
return $\mathcal{S}$

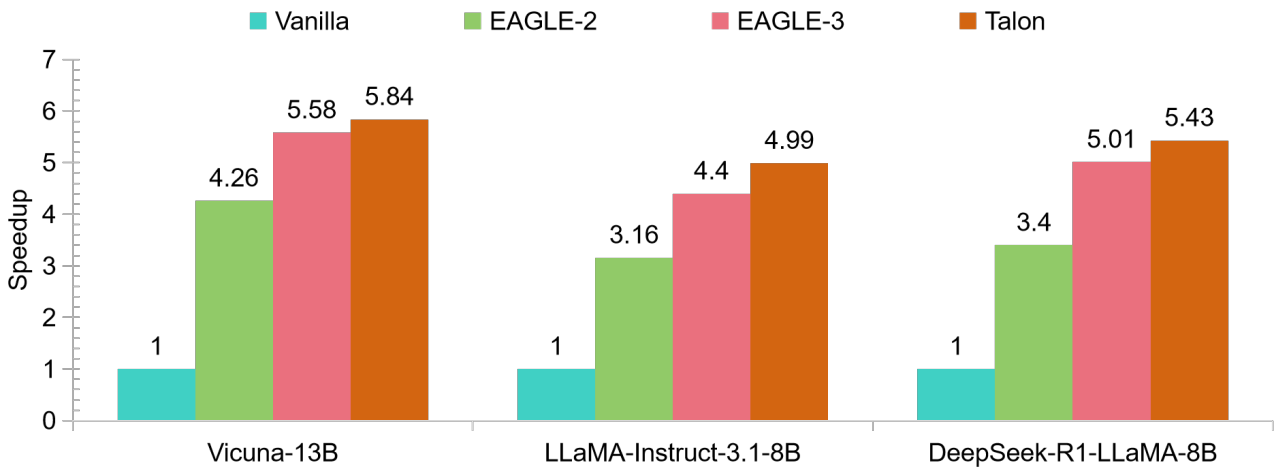

Figure 3: Speedup performance under deterministic decoding (temperature=0) across three representative model-task configurations: Vicuna-13B (multi-turn dialogue on MT-bench), LLaMA-Instruct-3.1-8B (instruction following on MT-bench), and DeepSeek-R1-LLaMA-8B (math reasoning on GSM8K). This figure highlights representative results, complete results are available in Table 1.

Draft Tree Verification

This work employs a recursive verification strategy based on tree attention mechanisms, enabling single-forward-pass computation of token probabilities across all nodes in the draft tree $\mathcal{T}_{\text{combine}}$. The core implementation arranges tree nodes in level-order traversal sequence, utilizing an attention mask matrix $\mathbf{M}$ that simultaneously encodes two critical structural features: positional embeddings representing hierarchical depth and explicit parent-child dependencies. For each node x_i, the SpecInfer acceptance criterion is applied: a node is accepted if and only if $\frac{P_{\text{target}}(x_i|\mathbf{x}_{<i})}{Q_{\text{draft}}(x_i|\mathbf{x}_{<i})} \geq \tau$, where τ denotes a predetermined acceptance threshold, while P_{target} and Q_{draft} represent probability distributions from the target and draft models respectively. Upon node rejection, immediate pruning of the corresponding descendant subtree occurs, with verification automatically transitioning to the next unverified sibling node at the same tree level for continued depth-first processing; when all sibling nodes at a given level are rejected, the system initiates resampling of new tokens from the last accepted ancestor node.

Experiments

Models and Tasks

Our empirical assessment leverages three state-of-the-art open-source language models: Vicuna-13B (Chiang et al. 2023), LLaMA-Instruct 3.3 (8B) (Dubey et al. 2024), and DeepSeek-R1-Distill-LLaMA (8B) (Guo et al. 2025). Comprehensive evaluation employs five established benchmarks spanning distinct capabilities domains: multi-turn conversation (MT-bench (Zheng et al. 2024)), code generation (HumanEval (Chen et al. 2021)), mathematical reasoning (GSM8K (Cobbe et al. 2021)), instruction following (Alpaca (Taori et al. 2023)), and summarization capacity (CNN/Daily Mail (Nallapati et al. 2016)).

Metrics

Falcon leaves the target model's weights untouched and employs greedy decoding to guarantee lossless acceleration, thereby preserving generation quality. Consequently, quality evaluation is unnecessary, and our assessment focuses exclusively on latency-oriented metrics: average acceptance length τ and speedup ratio (SR).

The average acceptance length (τ) quantifies the average count of tokens validated per forward pass of the target LLM, excluding computational overhead from draft token generation. This metric establishes the theoretical upper bound for acceleration potential. Complementarily, the speedup ratio (SR) benchmarks the reduction in real-time latency against standard auto-regressive decoding.

Implementation Details

Our draft model employs the AdamW optimizer with hyperparameters $(\beta_1, \beta_2) = (0.9, 0.95)$, implementing gradient clipping at 0.5 and a constant learning rate of 5×10^{-5}. The training data combines SHAREGPT (68k samples) and ULTRACHAT-200K (464k samples) datasets, where responses are dynamically generated by the target model rather than using static dataset entries. For the reasoning model DEEPSEEK-R1-DISTILL-LLAMA-8B, we augment the training with the OPENTHOUGHTS-114K-MATH dataset to enhance mathematical reasoning capabilities. The target model remains structurally unmodified throughout this process, ensuring consistent generation quality.

Baseline

We establish vanilla autoregressive decoding as the reference baseline (1.00× speed), following standard practice in speculative execution research (Chen et al. 2023b). Our comprehensive evaluation benchmarks TALON against three categories of state-of-the-art speculative decoding methods:

Model	Method	MT-Bench		HumanEval		GSM8K		Alpaca		CNN/DM	
		Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ
Vicuna-13B	SpS	1.93x	2.27	2.23x	2.57	1.77x	2.01	1.76x	2.03	1.93x	2.33
	PLD	1.58x	1.63	1.85x	1.93	1.68x	1.73	1.16x	1.19	2.42x	2.50
	Medusa	2.07x	2.59	2.50x	2.78	2.23x	2.64	2.08x	2.45	1.71x	2.09
	Lookahead	1.65x	1.69	1.71x	1.75	1.81x	1.90	1.46x	1.51	1.46x	1.50
	Hydra	2.88x	3.65	3.28x	3.87	2.93x	3.66	2.86x	3.53	2.05x	2.81
	EAGLE	3.07x	3.98	3.58x	4.39	3.08x	3.97	3.03x	3.95	2.49x	3.52
	EAGLE-2	4.26x	4.83	4.96x	5.41	4.22x	4.79	4.25x	4.89	3.40x	4.21
	EAGLE-3	5.58x	6.65	6.47x	7.54	5.32x	6.29	5.16x	6.17	**5.01x**	6.47
	Talon	**5.84x**	**8.11**	**6.52x**	**8.58**	**6.23x**	**8.10**	**5.65x**	**7.43**	4.67x	**6.60**
LlaMA-3.1 8B	EAGLE-2	3.16x	4.05	3.66x	4.71	3.39x	4.24	3.28x	4.12	2.65x	3.45
	EAGLE-3	4.40x	6.13	4.85x	6.74	4.48x	6.23	4.82x	6.70	3.65x	5.34
	Talon	**4.99x**	**7.03**	**5.33x**	**7.83**	**5.04x**	**7.39**	**4.98x**	**7.10**	**4.04x**	**5.60**
DeepSeek 8B	EAGLE-2	2.92x	3.80	3.42x	4.29	3.40x	4.40	3.01x	3.80	3.53x	3.33
	EAGLE-3	4.05x	5.58	4.59x	6.38	5.01x	6.93	3.65x	5.37	3.52x	4.92
	Talon	**4.93x**	**6.80**	**5.36x**	**7.54**	**5.43x**	**7.37**	**4.21x**	**6.43**	**4.06x**	**5.34**

Table 1: Comparison of speculative decoding methods showing speedup ratios and acceptance lengths (τ). All measurements obtained under greedy decoding (temperature=0).

(1) basic speculative frameworks including standard speculative sampling (Leviathan, Kalman, and Matias 2023; Gante 2023) and Lookahead decoding (Fu et al. 2024); (2) parallel drafting systems such as PLD (Saxena 2023b), HYDRA (Ankner et al. 2024), and MEDUSA (Cai et al. 2024b); and (3) advanced tree-based methods including EAGLE (Li et al. 2024b), EAGLE-2 (Li et al. 2024a), EAGLE-3 (Li et al. 2025b), and HASS (Zhang et al. 2024).

Our evaluation compares Talon with reproduced Eagle-3 results on NVIDIA A800 80GB GPUs, representing the current state-of-the-art in speculative inference. Crucially, both systems employ identical experimental parameters ensuring direct comparability. Performance data for other baselines (Eagle, Medusa, Lookahead, etc.) are derived from their original publications using A100 platforms. All methods are assessed under consistent evaluation protocols (temperature=0, greedy decoding) with identical dataset versions and batch size of 1.

Effectiveness

As quantitatively demonstrated in Figure 3 and Table 1, Talon exhibits statistically significant improvements across all evaluation scenarios. Talon achieves speedups ranging from 4.04× to 6.52× compared to vanilla autoregressive decoding. Talon significantly outperforms prior state-of-the-art speculative decoding methods like EAGLE-3 across diverse tasks and model architectures. Our proposed method demonstrates superior performance than traditional speculative decoding methods, achieving exceptional speedup ratios (4.04x–6.52x) and acceptance lengths (τ = 5.34–8.58).

On the Vicuna-13B architecture, the framework achieves a notable 4.9% average improvement across comprehensive benchmarks, with particularly impressive gains of 15.1% on complex mathematical reasoning tasks (GSM8K). For the LLaMA-3.1 8B model, Talon provides robust acceleration with a consistent 9.0% average speedup, peaking at 11.8% on MT-Bench. Most significantly, Talon achieves breakthrough performance on the DeepSeek-R1 8B model, realizing an exceptional improvement that represents state-of-the-art efficiency gains compared to EAGLE-3. This remarkable advancement is highlighted by 13.0% speed enhancements on Alpaca datasets.

These substantial gains are primarily attributed to Talon's synergistic integration of knowledge-enhanced drafting and parallelized execution. The knowledge-aware retrieval mechanism significantly boosts token acceptance rates in knowledge-intensive domains, while the asynchronous architecture eliminates sequential bottlenecks.

Conclusions

The paper presents Talon, a groundbreaking approach to the speculative decoding community that addresses two critical limitations of existing speculative decoding methods: the mutual waiting problem and constrained token acceptance rates. Talon redefines speculative decoding by introducing an asynchronous execution paradigm that decouples draft from verification, eliminating synchronization bottlenecks, and an adaptive hybrid drafting strategy that dynamically combines model-based predictions with retrieval-based methods to maximize token acceptance rates across diverse domains. Through its innovative architecture, Talon achieves 4.04x–6.52x acceleration while maintaining generation quality, as demonstrated across benchmarks (MT-Bench, HumanEval, GSM8K) and model families (Vicuna, LLaMA, DeepSeek). The system's dual optimization of pipeline efficiency and knowledge-aware drafting sets a new standard for efficient LLM inference, offering a scalable solution for latency-sensitive deployments of LLMs without compromising the inference accuracy.

References

Ankner, Z.; Parthasarathy, R.; Nrusimha, A.; Rinard, C.; Ragan-Kelley, J.; and Brandon, W. 2024. Hydra: Sequentially-dependent draft heads for medusa decoding. *arXiv preprint arXiv:2402.05109*.

Bommasani, R.; et al. 2021. On the Opportunities and Risks of Foundation Models. *CoRR*, abs/2108.07258.

Cai, T.; Li, Y.; Geng, Z.; Peng, H.; Lee, J. D.; Chen, D.; and Dao, T. 2024a. Medusa: Simple LLM Inference Acceleration Framework with Multiple Decoding Heads. arXiv:2401.10774.

Cai, T.; Li, Y.; Geng, Z.; Peng, H.; Lee, J. D.; Chen, D.; and Dao, T. 2024b. Medusa: Simple llm inference acceleration framework with multiple decoding heads. *arXiv preprint arXiv:2401.10774*.

Chen, C.; Borgeaud, S.; Irving, G.; Lespiau, J.-B.; Sifre, L.; and Jumper, J. 2023a. Accelerating Large Language Model Decoding with Speculative Sampling. arXiv:2302.01318.

Chen, C.; Borgeaud, S.; Irving, G.; Lespiau, J.-B.; Sifre, L.; and Jumper, J. 2023b. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*.

Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H. P. D. O.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Chiang, W.-L.; Li, Z.; Lin, Z.; Sheng, Y.; Wu, Z.; Zhang, H.; Zheng, L.; Zhuang, S.; Zhuang, Y.; Gonzalez, J. E.; et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3): 6.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

DeepSeek-AI; Liu, A.; Feng, B.; Xue, B.; Wang, B.; Wu, B.; Lu, C.; Zhao, C.; Deng, C.; Zhang, C.; Ruan, C.; Dai, D.; Guo, D.; Yang, D.; Chen, D.; Ji, D.; Li, E.; Lin, F.; Dai, F.; Luo, F.; Hao, G.; Chen, G.; Li, G.; Zhang, H.; Bao, H.; Xu, H.; Wang, H.; Zhang, H.; Ding, H.; Xin, H.; Gao, H.; Li, H.; Qu, H.; Cai, J. L.; Liang, J.; Guo, J.; Ni, J.; Li, J.; Wang, J.; Chen, J.; Chen, J.; Yuan, J.; Qiu, J.; Li, J.; Song, J.; Dong, K.; Hu, K.; Gao, K.; Guan, K.; Huang, K.; Yu, K.; Wang, L.; Zhang, L.; Xu, L.; Xia, L.; Zhao, L.; Wang, L.; Zhang, L.; Li, M.; Wang, M.; Zhang, M.; Zhang, M.; Tang, M.; Li, M.; Tian, N.; Huang, P.; Wang, P.; Zhang, P.; Wang, Q.; Zhu, Q.; Chen, Q.; Du, Q.; Chen, R. J.; Jin, R. L.; Ge, R.; Zhang, R.; Pan, R.; Wang, R.; Xu, R.; Zhang, R.; Chen, R.; Li, S. S.; Lu, S.; Zhou, S.; Chen, S.; Wu, S.; Ye, S.; Ye, S.; Ma, S.; Wang, S.; Zhou, S.; Yu, S.; Zhou, S.; Pan, S.; Wang, T.; Yun, T.; Pei, T.; Sun, T.; Xiao, W. L.; Zeng, W.; Zhao, W.; An, W.; Liu, W.; Liang, W.; Gao, W.; Yu, W.; Zhang, W.; Li, X. Q.; Jin, X.; Wang, X.; Bi, X.; Liu, X.; Wang, X.; Shen, X.; Chen, X.; Zhang, X.; Chen, X.; Nie, X.; Sun, X.; Wang, X.; Cheng, X.; Liu, X.; Xie, X.; Liu, X.; Yu, X.; Song, X.; Shan, X.; Zhou, X.; Yang, X.; and Li, X. 2025. DeepSeek-V3 Technical Report. arXiv:2412.19437.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv e-prints*, arXiv–2407.

Fu, Y.; Bailis, P.; Stoica, I.; and Zhang, H. 2024. Break the sequential dependency of llm inference using lookahead decoding. *arXiv preprint arXiv:2402.02057*.

Gante, J. 2023. Assisted generation: a new direction toward low-latency text generation.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

He, Z.; Zhong, Z.; Cai, T.; Lee, J.; and He, D. 2024. REST: Retrieval-Based Speculative Decoding. In Duh, K.; Gomez, H.; and Bethard, S., eds., *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 1582–1595. Mexico City, Mexico: Association for Computational Linguistics.

Hoffmann, J.; Borgeaud, S.; Mensch, A.; Buchatskaya, E.; Cai, T.; Rutherford, E.; de Las Casas, D.; Hendricks, L. A.; Welbl, J.; Clark, A.; Hennigan, T.; Noland, E.; Millican, K.; van den Driessche, G.; Damoc, B.; Guy, A.; Osindero, S.; Simonyan, K.; Elsen, E.; Rae, J. W.; Vinyals, O.; and Sifre, L. 2022. Training Compute-Optimal Large Language Models. arXiv:2203.15556.

Hu, Y.; Liu, Z.; Dong, Z.; Peng, T.; McDanel, B.; and Zhang, S. Q. 2025. Speculative Decoding and Beyond: An In-Depth Survey of Techniques. arXiv:2502.19732.

Ivanov, A.; Dryden, N.; Ben-Nun, T.; Li, S.; and Hoefler, T. 2020. Data Movement Is All You Need: A Case Study on Optimizing Transformers. *ArXiv*, abs/2007.00072.

Kaplan, J.; McCandlish, S.; Henighan, T.; Brown, T. B.; Chess, B.; Child, R.; Gray, S.; Radford, A.; Wu, J.; and Amodei, D. 2020. Scaling Laws for Neural Language Models. *CoRR*, abs/2001.08361.

Leviathan, Y.; Kalman, M.; and Matias, Y. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, 19274–19286. PMLR.

Li, Y.; Shi, J.; Feng, S.; Yuan, P.; Wang, X.; Zhang, Y.; Zhang, J.; Tan, C.; Pan, B.; Hu, Y.; and Li, K. 2025a. Speculative Decoding for Multi-Sample Inference. arXiv:2503.05330.

Li, Y.; Wei, F.; Zhang, C.; and Zhang, H. 2024a. Eagle-2: Faster inference of language models with dynamic draft trees. *arXiv preprint arXiv:2406.16858*.

Li, Y.; Wei, F.; Zhang, C.; and Zhang, H. 2024b. Eagle: Speculative sampling requires rethinking feature uncertainty. *arXiv preprint arXiv:2401.15077*.

Li, Y.; Wei, F.; Zhang, C.; and Zhang, H. 2025b. Eagle-3: Scaling up inference acceleration of large language models via training-time test. *arXiv preprint arXiv:2503.01840*.

Liu, T.; Li, Y.; Lv, Q.; Liu, K.; Zhu, J.; Hu, W.; and Sun, X. 2025. PEARL: Parallel Speculative Decoding with Adaptive

Draft Length. In *The Thirteenth International Conference on Learning Representations*.

Miao, X.; Oliaro, G.; Zhang, Z.; Cheng, X.; Wang, Z.; Zhang, Z.; Wong, R. Y. Y.; Zhu, A.; Yang, L.; Shi, X.; Shi, C.; Chen, Z.; Arfeen, D.; Abhyankar, R.; and Jia, Z. 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, Volume 3*, ASPLOS '24, 932–949. New York, NY, USA: Association for Computing Machinery. ISBN 9798400703867.

Monea, G.; Joulin, A.; and Grave, E. 2023. PaSS: Parallel Speculative Sampling. arXiv:2311.13581.

Nallapati, R.; Zhou, B.; Gulcehre, C.; Xiang, B.; et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

OpenAI; Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; Avila, R.; Babuschkin, I.; Balaji, S.; Balcom, V.; Baltescu, P.; Bao, H.; Bavarian, M.; Belgum, J.; Bello, I.; Berdine, J.; Bernadett-Shapiro, G.; Berner, C.; Bogdonoff, L.; Boiko, O.; Boyd, M.; Brakman, A.-L.; Brockman, G.; Brooks, T.; Brundage, M.; Button, K.; Cai, T.; Campbell, R.; Cann, A.; Carey, B.; Carlson, C.; Carmichael, R.; Chan, B.; Chang, C.; Chantzis, F.; Chen, D.; Chen, S.; Chen, R.; Chen, J.; Chen, M.; Chess, B.; Cho, C.; Chu, C.; Chung, H. W.; Cummings, D.; Currier, J.; Dai, Y.; Decareaux, C.; Degry, T.; Deutsch, N.; Deville, D.; Dhar, A.; Dohan, D.; Dowling, S.; Dunning, S.; Ecoffet, A.; Eleti, A.; Eloundou, T.; Farhi, D.; Fedus, L.; Felix, N.; Fishman, S. P.; Forte, J.; Fulford, I.; Gao, L.; Georges, E.; Gibson, C.; Goel, V.; Gogineni, T.; Goh, G.; Gontijo-Lopes, R.; Gordon, J.; Grafstein, M.; Gray, S.; Greene, R.; Gross, J.; Gu, S. S.; Guo, Y.; Hallacy, C.; Han, J.; Harris, J.; He, Y.; Heaton, M.; Heidecke, J.; Hesse, C.; Hickey, A.; Hickey, W.; Hoeschele, P.; Houghton, B.; Hsu, K.; Hu, S.; Hu, X.; Huizinga, J.; Jain, S.; Jain, S.; Jang, J.; Jiang, A.; Jiang, R.; Jin, H.; Jin, D.; Jomoto, S.; Jonn, B.; Jun, H.; Kaftan, T.; Łukasz Kaiser; Kamali, A.; Kanitscheider, I.; Keskar, N. S.; Khan, T.; Kilpatrick, L.; Kim, J. W.; Kim, C.; Kim, Y.; Kirchner, J. H.; Kiros, J.; Knight, M.; Kokotajlo, D.; Łukasz Kondraciuk; Kondrich, A.; Konstantinidis, A.; Kosic, K.; Krueger, G.; Kuo, V.; Lampe, M.; Lan, I.; Lee, T.; Leike, J.; Leung, J.; Levy, D.; Li, C. M.; Lim, R.; Lin, M.; Lin, S.; Litwin, M.; Lopez, T.; Lowe, R.; Lue, P.; Makanju, A.; Malfacini, K.; Manning, S.; Markov, T.; Markovski, Y.; Martin, B.; Mayer, K.; Mayne, A.; McGrew, B.; McKinney, S. M.; McLeavey, C.; McMillan, P.; McNeil, J.; Medina, D.; Mehta, A.; Menick, J.; Metz, L.; Mishchenko, A.; Mishkin, P.; Monaco, V.; Morikawa, E.; Mossing, D.; Mu, T.; Murati, M.; Murk, O.; Mély, D.; Nair, A.; Nakano, R.; Nayak, R.; Neelakantan, A.; Ngo, R.; Noh, H.; Ouyang, L.; O'Keefe, C.; Pachocki, J.; Paino, A.; Palermo, J.; Pantuliano, A.; Parascandolo, G.; Parish, J.; Parparita, E.; Passos, A.; Pavlov, M.; Peng, A.; Perelman, A.; de Avila Belbute Peres, F.; Petrov, M.; de Oliveira Pinto, H. P.; Michael; Pokorny; Pokrass, M.; Pong, V. H.; Powell, T.; Power, A.; Power, B.; Proehl, E.; Puri, R.; Radford, A.; Rae, J.; Ramesh, A.; Raymond, C.; Real, F.; Rimbach, K.; Ross, C.; Rotsted, B.; Roussez, H.; Ryder, N.; Saltarelli, M.; Sanders, T.; Santurkar, S.; Sastry, G.; Schmidt, H.; Schnurr, D.; Schulman, J.; Selsam, D.; Sheppard, K.; Sherbakov, T.; Shieh, J.; Shoker, S.; Shyam, P.; Sidor, S.; Sigler, E.; Simens, M.; Sitkin, J.; Slama, K.; Sohl, I.; Sokolowsky, B.; Song, Y.; Staudacher, N.; Such, F. P.; Summers, N.; Sutskever, I.; Tang, J.; Tezak, N.; Thompson, M. B.; Tillet, P.; Tootoonchian, A.; Tseng, E.; Tuggle, P.; Turley, N.; Tworek, J.; Uribe, J. F. C.; Vallone, A.; Vijayvergiya, A.; Voss, C.; Wainwright, C.; Wang, J. J.; Wang, A.; Wang, B.; Ward, J.; Wei, J.; Weinmann, C.; Welihinda, A.; Welinder, P.; Weng, J.; Weng, L.; Wiethoff, M.; Willner, D.; Winter, C.; Wolrich, S.; Wong, H.; Workman, L.; Wu, S.; Wu, J.; Wu, M.; Xiao, K.; Xu, T.; Yoo, S.; Yu, K.; Yuan, Q.; Zaremba, W.; Zellers, R.; Zhang, C.; Zhang, M.; Zhao, S.; Zheng, T.; Zhuang, J.; Zhuk, W.; and Zoph, B. 2024. GPT-4 Technical Report. arXiv:2303.08774.

Pope, R.; Douglas, S.; Chowdhery, A.; Devlin, J.; Bradbury, J.; Heek, J.; Xiao, K.; Agrawal, S.; and Dean, J. 2023. Efficiently Scaling Transformer Inference. In Song, D.; Carbin, M.; and Chen, T., eds., *Proceedings of Machine Learning and Systems*, volume 5, 606–624. Curan.

Quan, G.; Feng, W.; Hao, C.; Jiang, G.; Zhang, Y.; and Wang, H. 2025. RASD: Retrieval-Augmented Speculative Decoding. arXiv:2503.03434.

Ryu, H.; and Kim, E. 2024. Closer Look at Efficient Inference Methods: A Survey of Speculative Decoding. arXiv:2411.13157.

Saxena, A. 2023a. Prompt Lookup Decoding.

Saxena, A. 2023b. Prompt Lookup Decoding.

Shazeer, N. 2019. Fast Transformer Decoding: One Write-Head is All You Need. *CoRR*, abs/1911.02150.

Taori, R.; Gulrajani, I.; Zhang, T.; Dubois, Y.; Li, X.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023. Alpaca: a strong, replicable instruction-following model; 2023. *URL https://crfm. stanford. edu/2023/03/13/alpaca. html*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; Rodriguez, A.; Joulin, A.; Grave, E.; and Lample, G. 2023. LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971.

Zhang, C.; Liu, Z.; and Song, D. 2024. Beyond the Speculative Game: A Survey of Speculative Execution in Large Language Models. arXiv:2404.14897.

Zhang, L.; Wang, X.; Huang, Y.; and Xu, R. 2024. Learning harmonized representations for speculative sampling. *arXiv preprint arXiv:2408.15766*.

Zhao, Y.; Xie, Z.; Liang, C.; Zhuang, C.; and Gu, J. 2024. Lookahead: An Inference Acceleration Framework for Large Language Model with Lossless Generation Accuracy. arXiv:2312.12728.

Zheng, L.; Chiang, W.-L.; Sheng, Y.; Zhuang, S.; Wu, Z.; Zhuang, Y.; Lin, Z.; Li, Z.; Li, D.; Xing, E.; et al. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.