# *Parallel* LLM Decoding via *Speculative Decoding*

Presented by **Yu-Yang Qian**

**School of Artificial Intelligence, Nanjing University, China**

https://www.lamda.nju.edu.cn/qianyy/

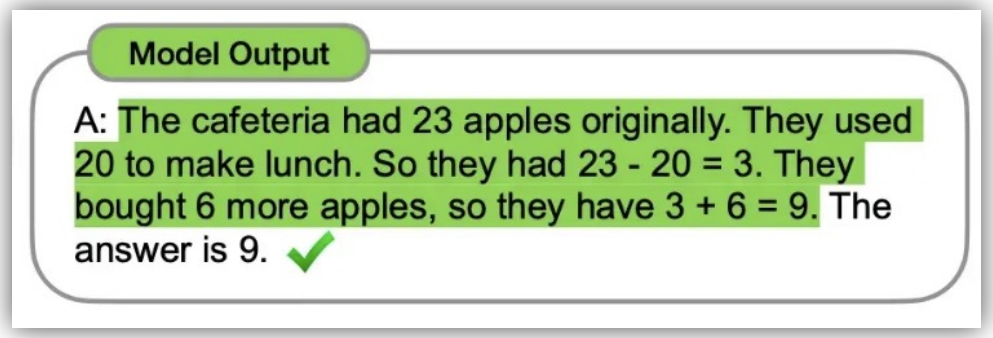
2026.08.08

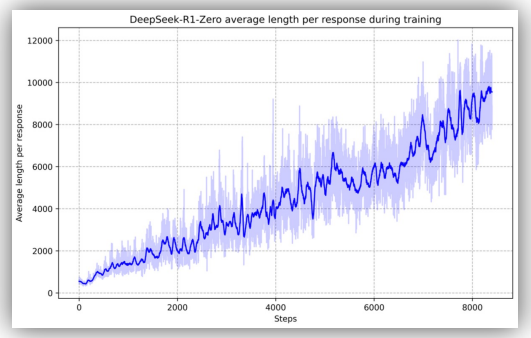
# Background

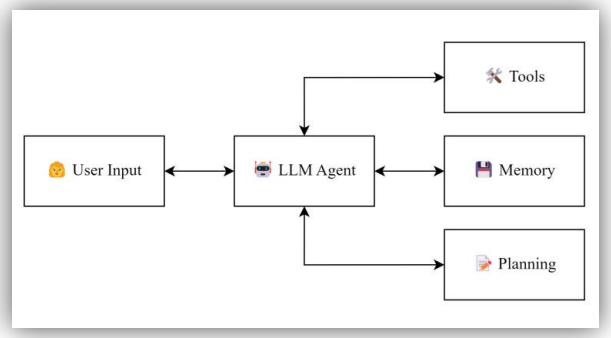
- LLMs have achieved remarkable success across diverse tasks,
  - e.g., ***chatbots***, ***vibe coding***, and long-term ***agentic tasks***.


# Background: LLM's Inference Cost

- LLMs have achieved remarkable success across diverse tasks,
  - e.g., ***chatbots***, ***vibe coding***, and long-term ***agentic tasks***.
- Not only the model size, but also the ***inference cost***, is increasing.
  - Especially with ***Chain of Thought*** (CoT)
  - Recently emerged ***agentic LLMs*** further increase this burden

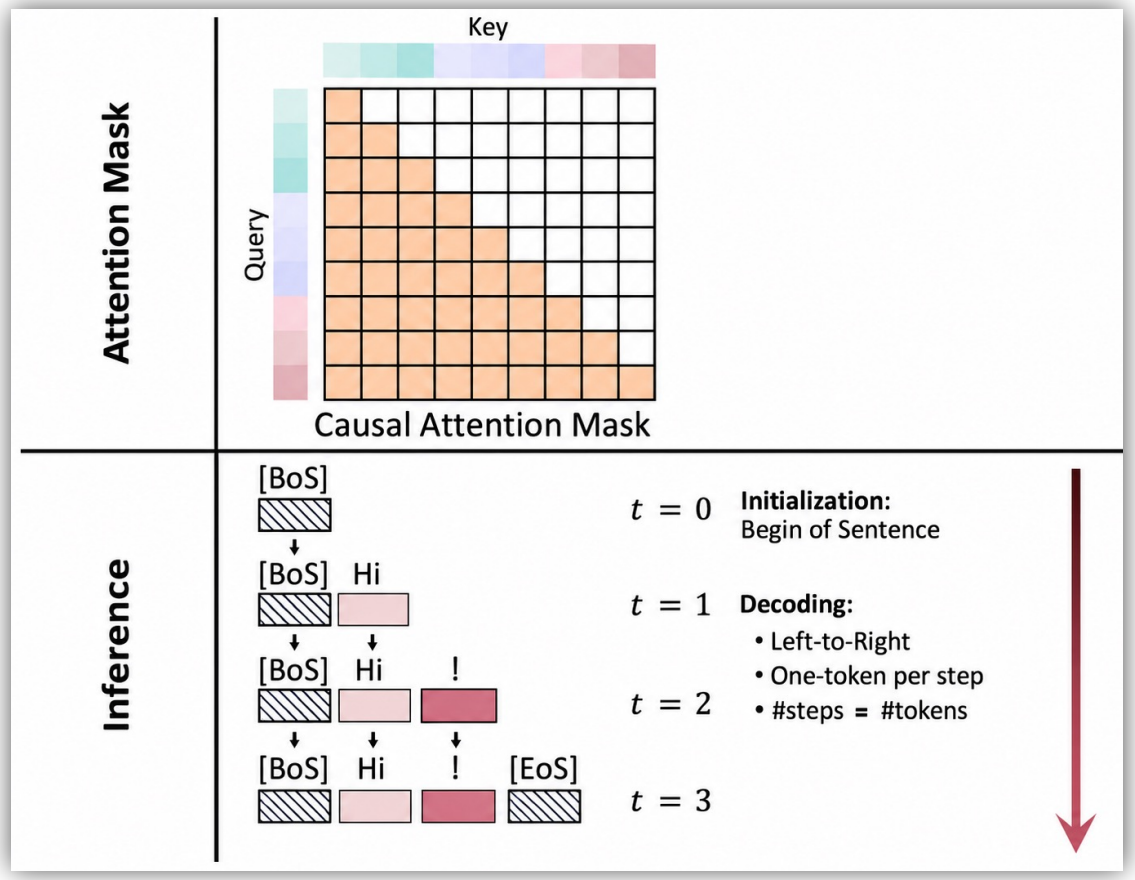



***Inference cost*** and inference latency of LLMs lead to a decline in user experience.

# Background: Latency Bottleneck of AR

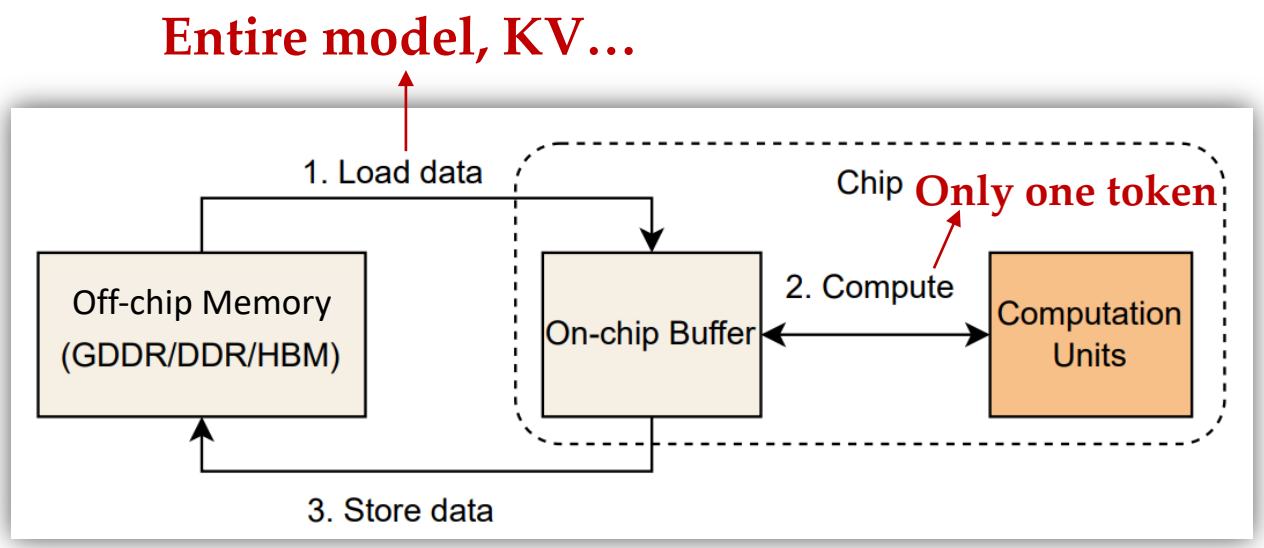


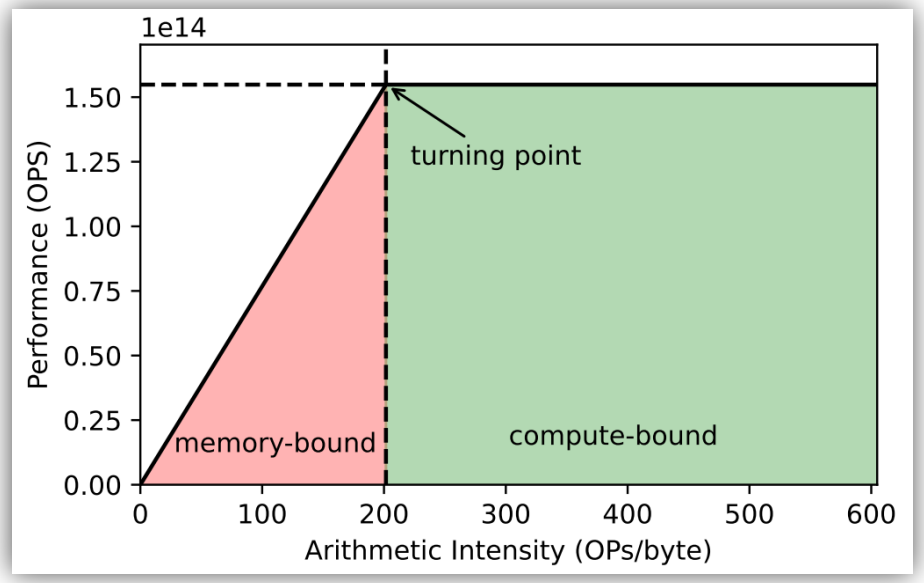

A token sequence probability
$p(x) = \prod_i p(x_i|x_{<i})$

- **Causal attention:** the model predicts the next token.
- It forms the foundation of today's mainstream LLMs such as GPT, LLaMA, and Qwen.

# Background: Latency Bottleneck of AR

- Autoregressive (AR) LLMs generate one token per forward pass.
- AR's decoding is ***memory-bound***, leaving the GPU ***underutilized***.

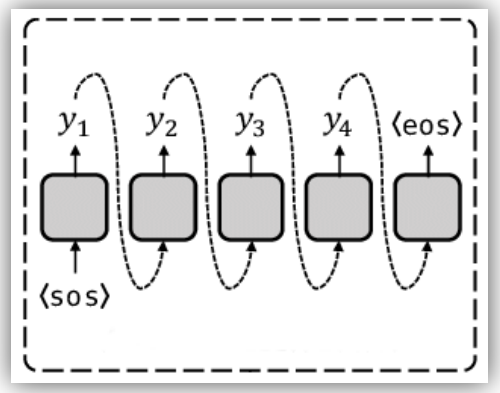



AR's decoding is ***memory-bound***, leaving the GPU ***underutilized***.

# Background: Latency Bottleneck of AR

- LLMs have achieved remarkable success across diverse tasks,
  - e.g., ***chatbots***, ***vibe coding***, and long-term ***agentic tasks***.
- Autoregressive (AR) model's speed bottleneck
  - ***Sequential*** dependency
  - "Next token prediction" paradigm



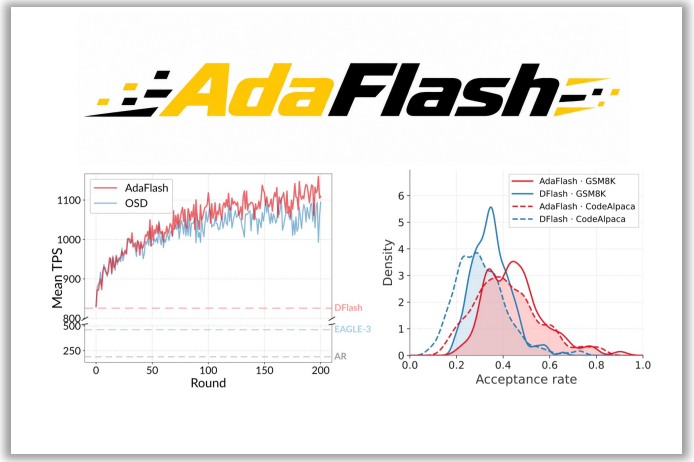

Can we decode ***multiple tokens*** in parallel, not only one at a time?

⇨ Investigating the ***Parallel Decoding*** of LLMs.

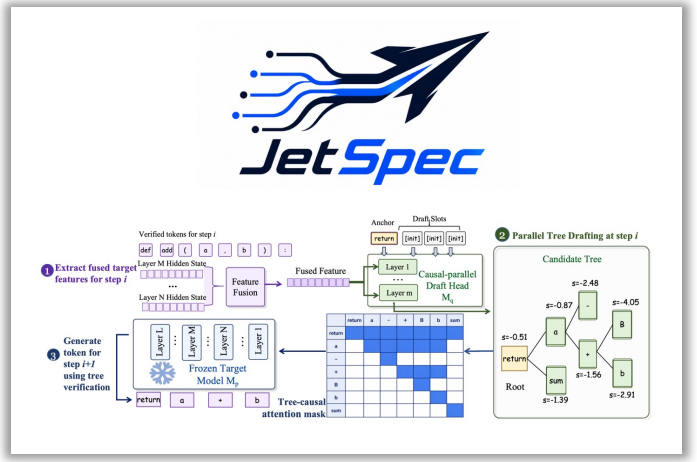
# Outline

- Goal: ***Efficient LLM Inference*** via ***Parallel Decoding*:**


**OnlineSPEC:** ***Online*** evolving
[ICML'26]




**AdaFlash**: Reducing ***variance***
[arXiv'26]




**JetSpec**: Parallel-causal tree
[arXiv'26]

Parallel LLM decoding via ***Speculative Decoding.***

# Outline: *OnlineSPEC*

- Goal: *Efficient LLM Inference* via ***Parallel Decoding*:**


**OnlineSPEC:** ***Online*** evolving
[ICML'26]



AdaFlash: Reducing *variance*
[arXiv'26]



JetSpec: Parallel-causal tree
[arXiv'26]

***Online evolving*** of draft models.

ICML
International Conference
On Machine Learning
Seoul, South Korea

# When Drafts Evolve: Speculative Decoding Meets Online Learning

[ICML'26]

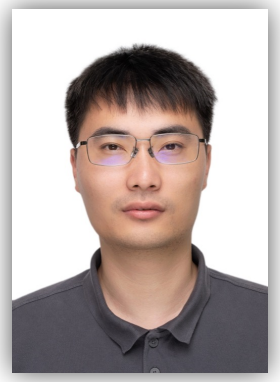
**Yu-Yang Qian**[1,2] **Hao-Cong Wu**[1,2] **Yichao Fu**[3] **Hao Zhang**[3] **Peng Zhao**[1,2]
[1]National Key Laboratory for Novel Software Technology, Nanjing University
[2]School of Artificial Intelligence, Nanjing University [3]University of California, San Diego
{qianyy,zhaop}@lamda.nju.edu.cn hcwu@smail.nju.edu.cn
{yif034,haozhang}@ucsd.edu

Yu-Yang Qian

Hao-Cong Wu

Yichao Fu

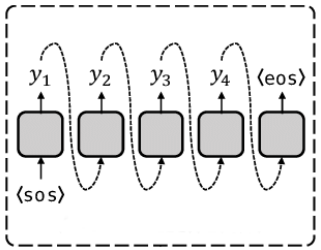
Hao Zhang

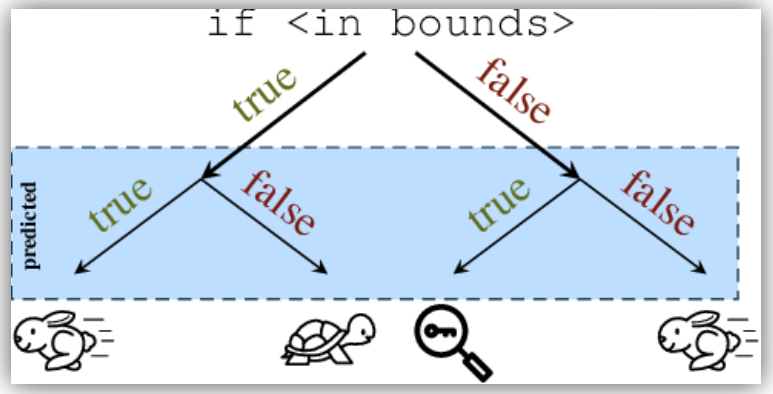


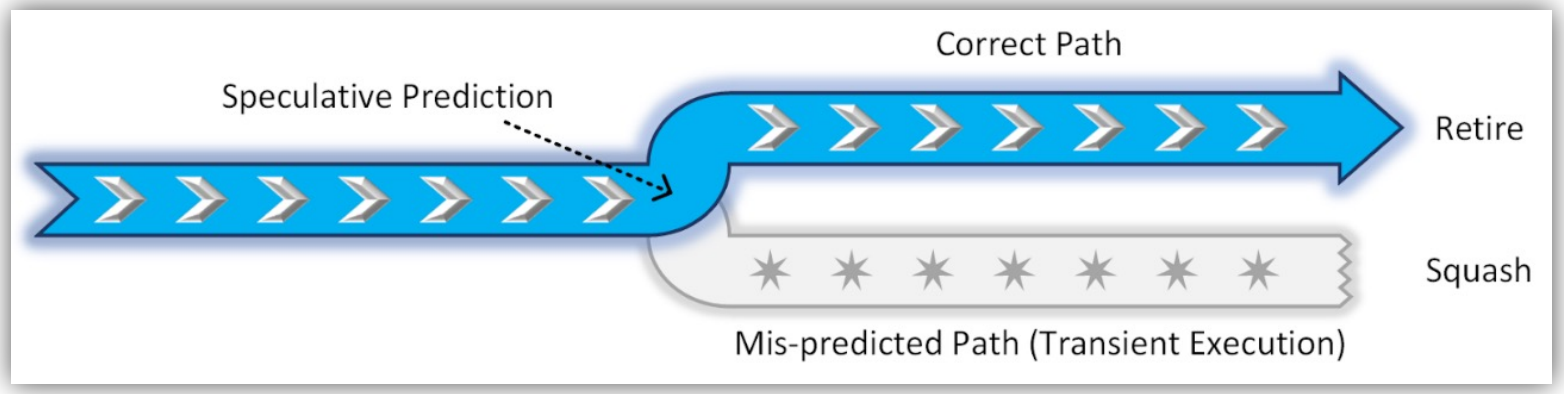
Peng Zhao

# Preliminary: Speculative Execution

- Autoregressive (AR) models speedup bottleneck
  - ***Sequential*** dependency
- Insight: commonly used in modern ***CPU Processors:***
  - Speculative execution



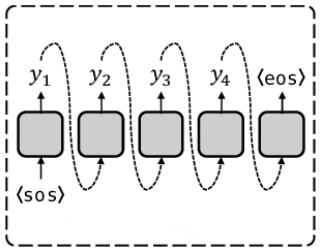




Speculative execution: ***speculatively*** **execute** future instructions: correct predictions yield speedup; while incorrect ones trigger rollback.

# Preliminary: Speculative Decoding

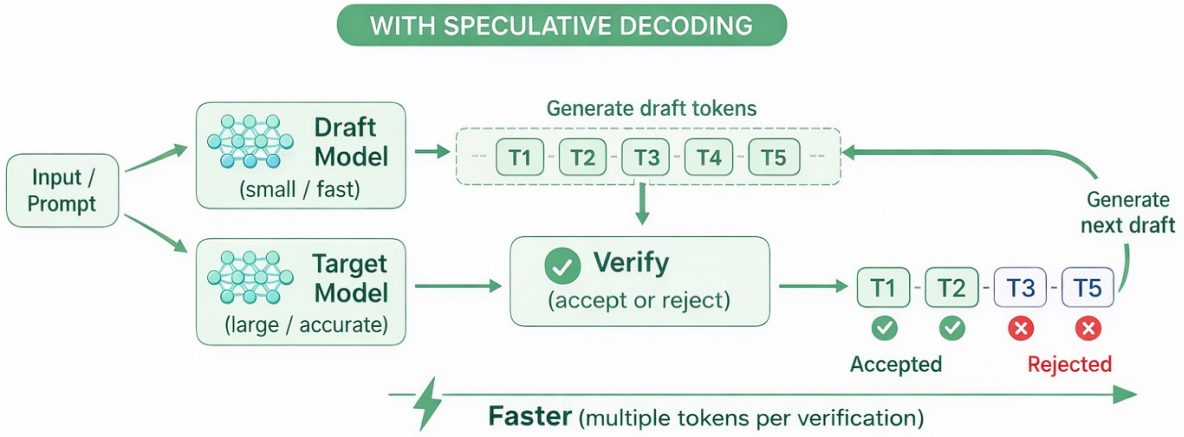
- Autoregressive (AR) models speedup bottleneck
  - ***Sequential*** dependency

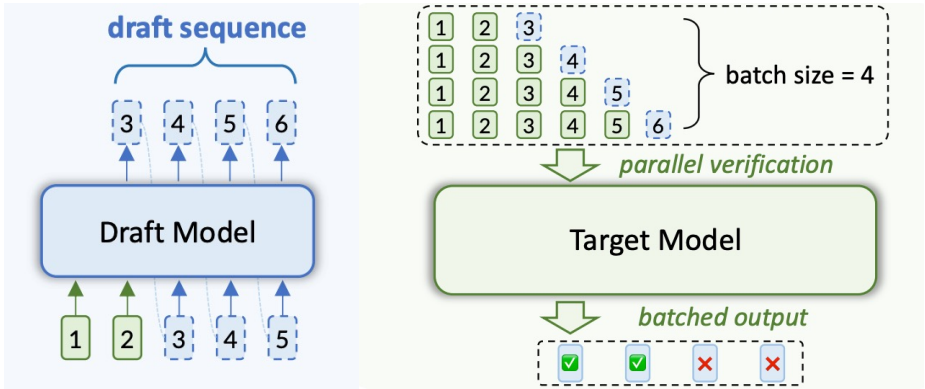



- **Speculative Decoding:** An LLM inference acceleration method.

  Use a small ***"draft model"*** to first quickly generate tokens as drafts;

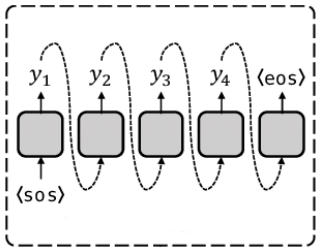
  Use a big ***"target model"*** to ***parallelly*** verify the drafts.



# Preliminary: Speculative Decoding

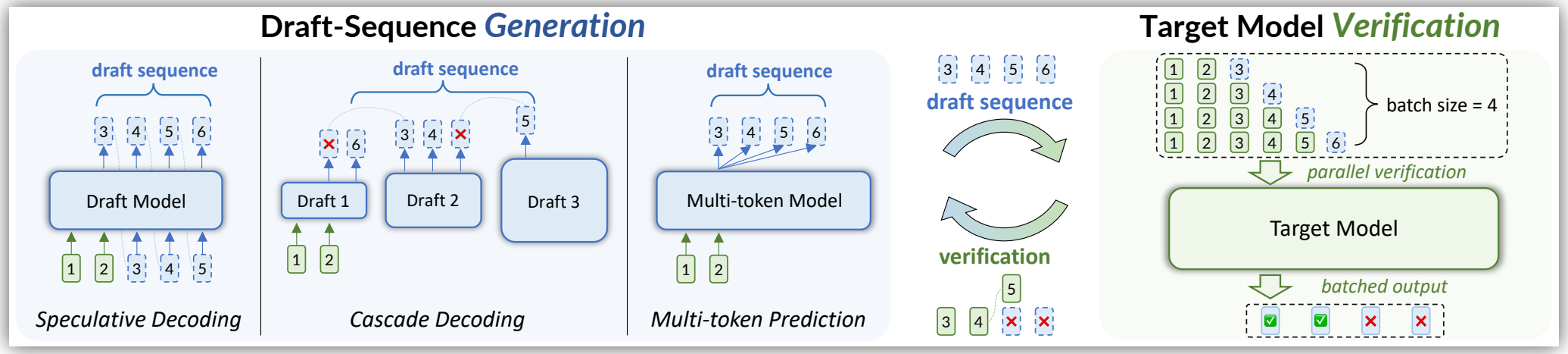
- Autoregressive (AR) models speedup bottleneck
  - *Sequential* dependency



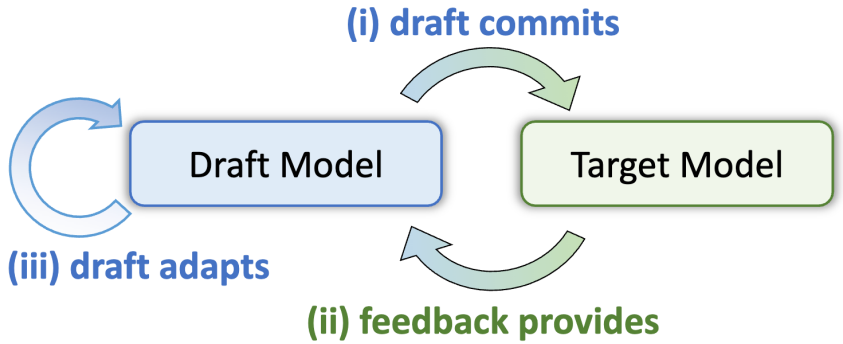

- **Speculative Decoding:** An LLM inference acceleration method.

# Background

- We investigate the "***speculative decoding***" acceleration methods:




- ***Key observation:***
  - Draft model receives timely feedback ***at no cost!*** (white box / black box feedback)
  - We can use these feedback to update draft model(s), exactly match **Online Learning.**




Naturally forms a *"draft commits–feedback provides–draft adapts"* ***evolving loop.***

# A Unified Framework: *OnlineSPEC*

- Formulated as an *online learning problem:*

At each round $t = 1, \ldots, T$:

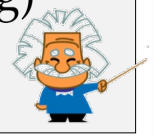
(1) the learner submits draft model(s) $\mathbf{w}_t \in \mathcal{W}$; (draft model generating)

⇩

(2) target model reveals feedback $f_t(\cdot) : \mathcal{W} \mapsto \mathbb{R}$; (parallel verifying)

⇩

(3) the learner suffers loss $f_t(\mathbf{w}_t)$ and updates the draft model(s).



- Performance Metric: ***Dynamic Regret***

$$\mathbf{Reg}_T \triangleq \sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}_t^\star),$$

i.e., the ***cumulative gap*** between the algorithm and a sequence of time-varying comparators $\{\mathbf{w}_t^\star\}_{t=1}^T$.

**Algorithm 1** OnlineSPEC Framework

**Input:** Initial draft model parameter $\mathbf{w}_0 \in \mathcal{W}$ with its predicted distribution $q_{\mathbf{w}_0}(\cdot \mid \mathbf{x})$, target model $\mathbf{v} \in \mathcal{V}$ with its predicted distribution $p_{\mathbf{v}}(\cdot \mid \mathbf{x})$, total steps $T$, candidate length $k$, input prompt $\mathbf{x}_0$.

**Initialize:** Input sequence $\mathbf{x} = \mathbf{x}_0$, draft model $\mathbf{w}_t = \mathbf{w}_0$.

**for** $t = 1, \ldots, T$ **do**

▷ Generate the draft sequence:

**for** $i = 1$ **to** $k$ **do**

$x_i \sim q_{\mathbf{w}_t}(x \mid \mathbf{x}_{<i})$, where $\mathbf{x}_{<i} = \{\mathbf{x}, x_1, \ldots, x_{i-1}\}$.

▷ Verify by target model:

Compute $\{p_{\mathbf{v}}(x_1|\mathbf{x}_{<1}), \ldots, p_{\mathbf{v}}(x_k|\mathbf{x}_{<k})\}$ in parallel.

▷ Determine the accepted token length $n_t$:

Sample $r_1 \sim U(0,1), \ldots, r_k \sim U(0,1)$ uniformly;

$n_t \leftarrow \min(\{j-1 \mid 1 \le j \le k, r_j > \frac{p_{\mathbf{v}}(x_j|\mathbf{x}_{<j})}{q_{\mathbf{w}_t}(x_j|\mathbf{x}_{<j})}\} \cup \{k\})$.

▷ Verify and update the draft sequence:

**if** $n_t < k$ **then**

$p'(x) \propto \max\left(0, p_{\mathbf{v}}(x \mid \mathbf{x}_{<n_t+1}) - q_{\mathbf{w}_t}(x \mid \mathbf{x}_{<n_t+1})\right)$;

**else**

$p'(x) \leftarrow p_{\mathbf{v}}(x \mid \mathbf{x}_{<k+1})$.

$x_{n_t+1} \sim p'(x)$; $\mathbf{x} \leftarrow \mathbf{x} + [x_1, \ldots, x_{n_t}, x_{n_t+1}]$.

▷ Receive interactive feedback: observe the loss function $f_t : \mathcal{W} \mapsto \mathbb{R}$ from the target model $\mathbf{v}$.

▷ Update draft model: $\mathbf{w}_{t+1} \leftarrow \text{Update}(f_t; \mathbf{w}_t)$ as Sec. 3.

**Output:** Token sequence $\hat{\mathbf{x}} \triangleq [x_1, x_2, \ldots]$.

# A Unified Framework: *OnlineSPEC*

- Formulated as an *online learning problem:*

At each round $t = 1, \ldots, T$:

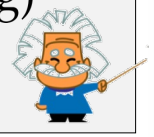
(1) the learner submits draft model(s) $\mathbf{w}_t \in \mathcal{W}$; (draft model generating)

⇩

(2) target model reveals feedback $f_t(\cdot) : \mathcal{W} \mapsto \mathbb{R}$; (parallel verifying)

⇩

(3) the learner suffers loss $f_t(\mathbf{w}_t)$ and updates the draft model(s).



- ***Relationship*** between ***acceleration rate*** and ***regret***:

**Theorem 1** (Acceleration Rate). Under Assumption 1, for OnlineSPEC with a total of $T$ steps, the acceleration rate $\gamma$ is bounded by

$$\frac{k+1}{(\alpha k+1)\left(1+(k+1)\sqrt{\mathrm{Reg}_T/(2T)}\right)} \leq \gamma \leq \frac{k+1}{\alpha k+1},$$

where $\alpha \triangleq \frac{a}{A} \ll 1$ is the ratio of the inference times between the draft model and the target model.

**Algorithm 1** OnlineSPEC Framework

**Input:** Initial draft model parameter $\mathbf{w}_0 \in \mathcal{W}$ with its predicted distribution $q_{\mathbf{w}_0}(\cdot \mid \mathbf{x})$, target model $\mathbf{v} \in \mathcal{V}$ with its predicted distribution $p_{\mathbf{v}}(\cdot \mid \mathbf{x})$, total steps $T$, candidate length $k$, input prompt $\mathbf{x}_0$.

**Initialize:** Input sequence $\mathbf{x} = \mathbf{x}_0$, draft model $\mathbf{w}_t = \mathbf{w}_0$.

**for** $t = 1, \ldots, T$ **do**

▷ Generate the draft sequence:

**for** $i = 1$ **to** $k$ **do**

$x_i \sim q_{\mathbf{w}_t}(x \mid \mathbf{x}_{<i})$, where $\mathbf{x}_{<i} = \{\mathbf{x}, x_1, \ldots, x_{i-1}\}$.

▷ Verify by target model:

Compute $\{p_{\mathbf{v}}(x_1|\mathbf{x}_{<1}), \ldots, p_{\mathbf{v}}(x_k|\mathbf{x}_{<k})\}$ in parallel.

▷ Determine the accepted token length $n_t$:

Sample $r_1 \sim U(0,1), \ldots, r_k \sim U(0,1)$ uniformly;

$n_t \leftarrow \min(\{j-1 \mid 1 \leq j \leq k, r_j > \frac{p_{\mathbf{v}}(x_j|\mathbf{x}_{<j})}{q_{\mathbf{w}_t}(x_j|\mathbf{x}_{<j})}\} \cup \{k\})$.

▷ Verify and update the draft sequence:

**if** $n_t < k$ **then**

$p'(x) \propto \max\left(0, p_{\mathbf{v}}(x \mid \mathbf{x}_{<n_t+1}) - q_{\mathbf{w}_t}(x \mid \mathbf{x}_{<n_t+1})\right)$;

**else**

$p'(x) \leftarrow p_{\mathbf{v}}(x \mid \mathbf{x}_{<k+1})$.

$x_{n_t+1} \sim p'(x)$; $\mathbf{x} \leftarrow \mathbf{x} + [x_1, \ldots, x_{n_t}, x_{n_t+1}]$.

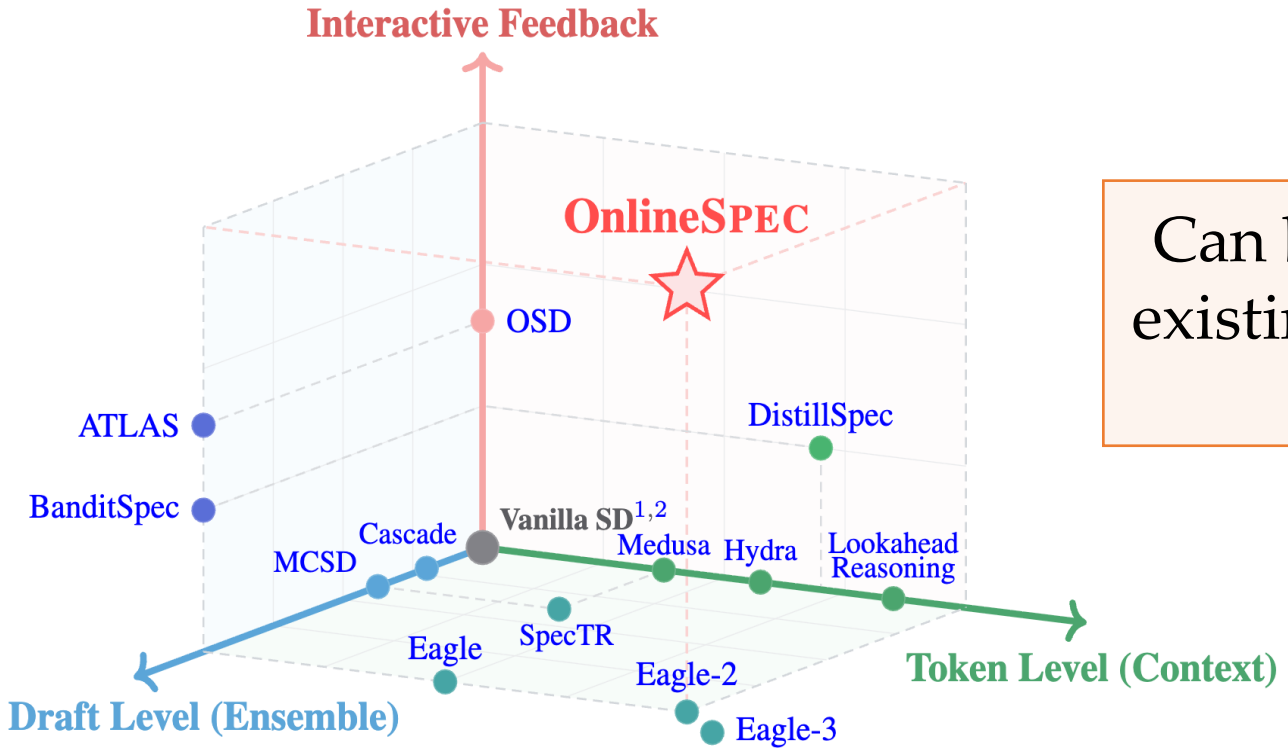
▷ Receive interactive feedback: observe the loss function $f_t : \mathcal{W} \mapsto \mathbb{R}$ from the target model $\mathbf{v}$.

▷ Update draft model: $\mathbf{w}_{t+1} \leftarrow \mathrm{Update}(f_t; \mathbf{w}_t)$ as Sec. 3.

**Output:** Token sequence $\hat{\mathbf{x}} \triangleq [x_1, x_2, \ldots]$.

# *OnlineSPEC* with Instantiations

- *OnlineSPEC* framework can guide the design of new algorithms:
  - A ***unified framework*** to boost speculative decoding
  - By leveraging the rich toolkit from online learning




Can be ***seamlessly*** combined with existing methods to further enhance the acceleration rate.

# *OnlineSPEC* with Instantiations

- ***OnlineSPEC*** framework can guide the design of new algorithms:
  - A ***unified framework*** to boost speculative decoding
  - By leveraging the rich toolkit from online learning
- We demonstrate its generality through three ***instantiations***:

| Base Method | +Feedback via | Instantiation | Acceleration rate |
|---|---|---|---|
| Lookahead Reasoning | Online gradient descent $\mathbf{w}_{t+1} = \Pi_{\mathcal{W}}\left[\mathbf{w}_t - \eta \nabla f_t(\mathbf{w}_t)\right]$ | *Online-LR* | $\gamma = \Omega\left(\frac{1}{\alpha k+1}\min\left\{k+1, \frac{T^{1/4}}{\sqrt{1+P_T}}\right\}\right)$ |
| Hydra | Optimistic online learning $\mathbf{w}_t = \Pi_{\mathcal{W}}[\widehat{\mathbf{w}}_t - \eta \mathbf{h}_t];\ \widehat{\mathbf{w}}_{t+1} = \Pi_{\mathcal{W}}[\widehat{\mathbf{w}}_t - \eta \nabla f_t(\mathbf{w}_t)]$ | *Opt-Hydra* | $\gamma = \Omega\left(\frac{1}{\alpha k+1}\min\left\{k+1, \frac{\sqrt{T}}{(1+\delta_T)^{1/4}\sqrt{1+P_T}}\right\}\right)$ |
| Eagle/Eagle-3 | Online ensemble learning $\mathbf{w}_t = \sum_{i=1}^{N} p_t^i \cdot \mathbf{w}_t^i$ | *Ens-Eagle* | $\gamma = \Omega\left(\frac{1}{\alpha k+1}\min\left\{k+1, \frac{T^{1/4}}{(1+P_T)^{1/4}}\right\}\right)$ |

# Experimental Results

| | | GSM8K | | Spider | | Code-Search | | Alpaca-Finance | |
|---|---|---|---|---|---|---|---|---|---|
| | | AvgLen ↑ | SpeedUp ↑ | AvgLen ↑ | SpeedUp ↑ | AvgLen ↑ | SpeedUp ↑ | AvgLen ↑ | SpeedUp ↑ |
| | | Target model: *lmsys / Vicuna-7B-v1.3* | | | | | | | |
| | Vanilla SD | 1.25±0.01 | 1.00× | 1.20±0.03 | 1.00× | 1.22±0.02 | 1.00× | 1.20±0.01 | 1.00× |
| | **OSD** | **1.52±0.03** | **1.23×** | **1.36±0.02** | **1.12×** | **1.37±0.02** | **1.09×** | **1.30±0.01** | **1.08×** |
| | Hydra | 2.14±0.05 | 1.00× | 2.65±0.31 | 1.00× | 1.82±0.04 | 1.00× | 1.78±0.01 | 1.00× |
| | OSD-Hydra | 2.56±0.06 | 1.19× | 3.11±0.31 | 1.11× | 2.11±0.05 | 1.16× | 2.19±0.03 | 1.25× |
| (OnlineSpec) | **Opt-Hydra** | **2.69±0.08** | **1.26×** | **3.27±0.32** | **1.18×** | **2.37±0.07** | **1.31×** | **2.70±0.03** | **1.55×** |
| | EAGLE | 1.48±0.03 | 1.00× | 1.31±0.03 | 1.00× | 1.41±0.03 | 1.00× | 1.39±0.01 | 1.00× |
| | OSD-EAGLE | 1.92±0.04 | 1.28× | 1.53±0.03 | 1.21× | 1.54±0.04 | 1.07× | 1.51±0.02 | 1.09× |
| (OnlineSpec) | **Ens-EAGLE** | **2.01±0.05** | **1.41×** | **1.60±0.03** | **1.32×** | **1.58±0.04** | **1.15×** | **1.61±0.04** | **1.14×** |
| | EAGLE-3 | 1.78±0.03 | 1.00× | 1.85±0.07 | 1.00× | 1.67±0.04 | 1.00× | 1.62±0.01 | 1.00× |
| | OSD-EAGLE-3 | 2.07±0.04 | 1.20× | 2.10±0.05 | 1.07× | 1.97±0.04 | 1.13× | 2.00±0.04 | 1.16× |
| (OnlineSpec) | **Ens-EAGLE-3** | **2.16±0.03** | **1.26×** | **2.24±0.07** | **1.12×** | **2.01±0.07** | **1.18×** | **2.07±0.04** | **1.27×** |
| | | Target model: *meta-llama / Llama-2-7B-Chat* | | | | | | | |
| | Vanilla SD | 1.25±0.01 | 1.00× | 1.07±0.01 | 1.00× | 1.09±0.01 | 1.00× | 1.20±0.01 | 1.00× |
| | **OSD** | **1.40±0.03** | **1.06×** | **1.47±0.05** | **1.22×** | **1.34±0.02** | **1.26×** | **1.31±0.01** | **1.12×** |
| | Hydra | 1.52±0.02 | 1.00× | 1.48±0.08 | 1.00× | 1.66±0.01 | 1.00× | 1.64±0.01 | 1.00× |
| | OSD-Hydra | 2.55±0.03 | 1.67× | 3.10±0.18 | 1.91× | 2.39±0.04 | 1.43× | 2.25±0.03 | 1.36× |
| (OnlineSpec) | **Opt-Hydra** | **2.94±0.03** | **1.90×** | **3.28±0.16** | **2.03×** | **2.71±0.05** | **1.61×** | **2.78±0.05** | **1.68×** |
| | EAGLE | 1.19±0.01 | 1.00× | 1.15±0.03 | 1.00× | 1.17±0.01 | 1.00× | 1.14±0.01 | 1.00× |
| | OSD-EAGLE | 1.45±0.01 | 1.18× | 1.72±0.06 | 1.46× | 1.47±0.02 | 1.28× | 1.43±0.01 | 1.20× |
| (OnlineSpec) | **Ens-EAGLE** | **1.58±0.01** | **1.33×** | **1.82±0.08** | **1.61×** | **1.62±0.01** | **1.46×** | **1.56±0.02** | **1.41×** |
| | EAGLE-3 | 1.82±0.03 | 1.00× | 1.61±0.03 | 1.00× | 1.69±0.04 | 1.00× | 1.70±0.02 | 1.00× |
| | OSD-EAGLE-3 | 2.25±0.02 | 1.23× | 2.42±0.12 | 1.43× | 2.09±0.10 | 1.27× | 2.13±0.02 | 1.23× |
| (OnlineSpec) | **Ens-EAGLE-3** | **2.33±0.02** | **1.27×** | **2.54±0.14** | **1.57×** | **2.18±0.10** | **1.33×** | **2.15±0.03** | **1.26×** |

🚀 Up to **24%** ***speedup*** over previous SOTA methods.

# Experimental Results

| | GSM8K | | MBPP | | MATH | | MMLU | |
|---|---|---|---|---|---|---|---|---|
| | AvgLen ↑ | Acc (%) ↑ | AvgLen ↑ | Acc (%) ↑ | AvgLen ↑ | Acc (%) ↑ | AvgLen ↑ | Acc (%) ↑ |
| | Target model: *Qwen / Qwen3-8B*, Draft model: *Qwen / Qwen3-0.6B-Base* | | | | | | | |
| Target Model | 1.00 (1.00×) | 94.32 | 1.00 (1.00×) | 53.56 | 1.00 (1.00×) | 91.54 | 1.00 (1.00×) | 83.81 |
| Draft Model | 1.00 (3.60×) | 53.84 | 1.00 (3.56×) | 14.15 | 1.00 (3.54×) | 60.66 | 1.00 (3.56×) | 55.71 |
| LR | 13.25 (1.26×) | 91.04 | 5.76 (1.09×) | 50.65 | 9.56 (1.20×) | **92.84** | 9.40 (1.21×) | 82.86 |
| OSD-LR | 12.57 (1.20×) | **93.84** | 5.66 (1.09×) | 50.54 | 6.21 (1.10×) | 89.87 | 6.67 (1.14×) | 82.14 |
| (OnlineSpec) **Online-LR** | **14.71 (1.41×)** | 92.88 | **7.16 (1.14×)** | **51.19** | **10.63 (1.24×)** | 91.37 | **10.62 (1.26×)** | **84.52** |

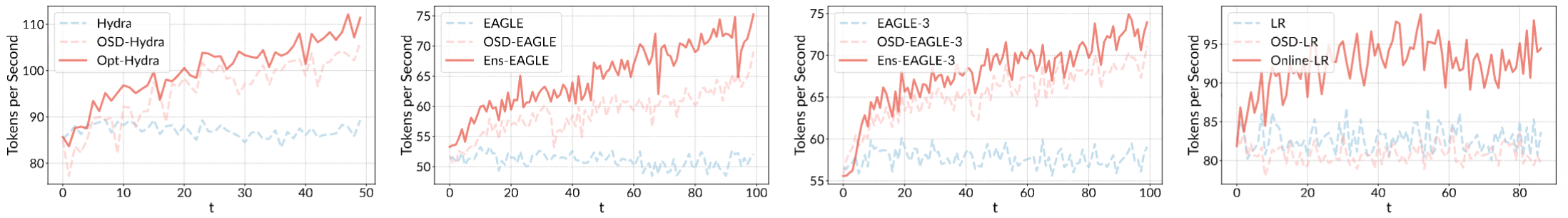



**Figure 3.** Evolution of tokens per second (TPS) on the *GSM8K* dataset: (a) *Opt-Hydra* with *lmsys/Vicuna-7B-v1.3*, (b) *Ens-EAGLE* with *lmsys/Vicuna-7B-v1.3*, (c) *Ens-EAGLE-3* with *lmsys/Vicuna-7B-v1.3*, and (d) *Online-LR* with *Qwen/Qwen3-8B*. This demonstrates consistent performance improvements via online learning, validating the effectiveness of our OnlineSpec during deployment.

🚀 ***Continuously*** improving the draft model and inference efficiency.

# Scaling to larger models

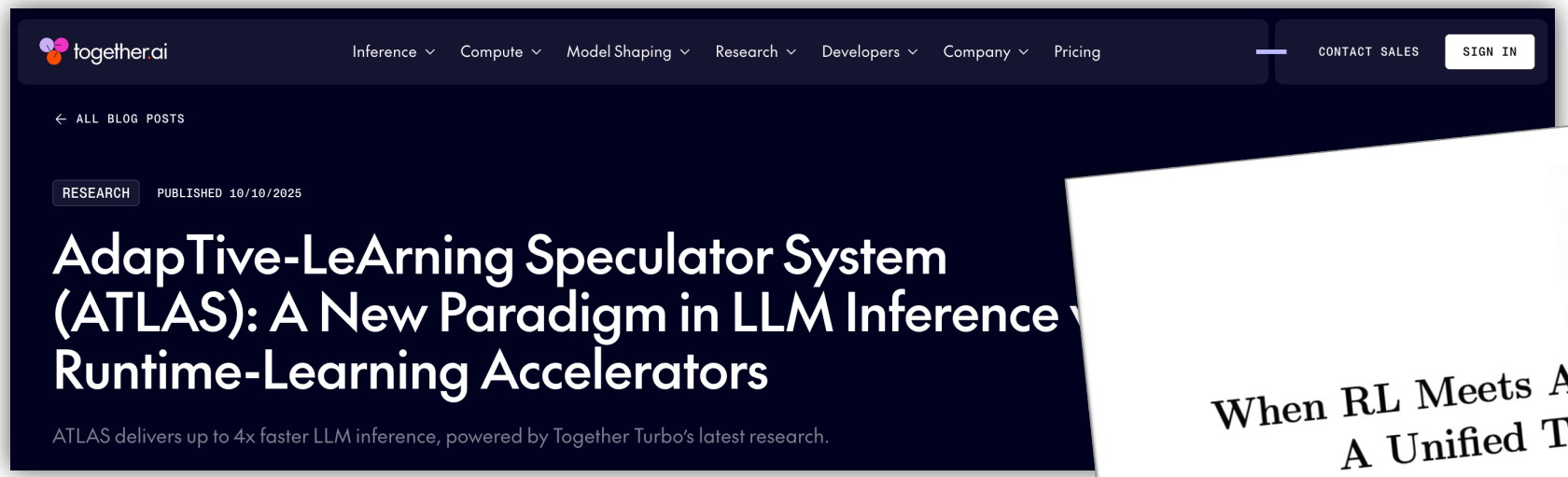
**Table 3.** Results on larger target models. We report SPEEDUP ↑ (with TPS), AVGLEN ↑, and the accuracy (ACC).

| Model | Method | SPEEDUP ↑ | AVGLEN ↑ | ACC (%) |
|---|---|---|---|---|
| Vicuna-13B | Standard AR | 1.00 (40.40) | 1.00 | – |
| | Vanilla SD | 1.20 (48.52) | 1.93 | – |
| | OSD-EAGLE-3 | 1.37 (55.52) | 2.24 | – |
| | **Ens-EAGLE-3** | **1.46** (59.03) | **2.35** | – |
| Vicuna-13B | Standard AR | 1.00 (40.40) | 1.00 | – |
| | Vanilla SD | 1.50 (60.61) | 2.22 | – |
| | OSD-Hydra | 1.69 (68.36) | 2.50 | – |
| | **Opt-Hydra** | **1.84** (74.37) | **2.73** | – |
| Qwen3-32B | Standard AR | 1.00 (35.53) | 1.00 | 96.08 |
| | LR | 1.11 (39.39) | 3.66 | 95.76 |
| | OSD-LR | 1.09 (38.63) | 3.18 | 95.44 |
| | **Online-LR** | **1.31** (46.71) | **9.98** | 95.68 |

🚀 Apply to 13B/32B, ***OnlineSPEC*** still ***outperforms*** other methods.

# Concurrent Work from **together.ai**

- Our paper and concurrent studies (@together.ai) arrive at similar insights.
- All focus on exploiting ***"online updates"*** in inference-acceleration systems.





together.ai

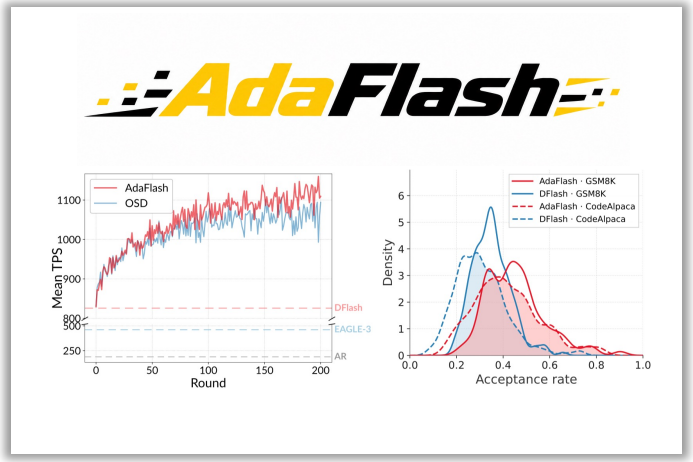
(Co-founded by Tri Dao, Percy Liang, Chris Re, and Ce Zhang)

# Outline: *AdaFlash*

- Goal: ***Efficient LLM Inference*** via ***Parallel Decoding*:**




**OnlineSPEC**: *Online* evolving
[ICML'26]




**AdaFlash**: Reducing ***variance***
[arXiv'26]




**JetSpec**: Parallel-causal tree
[arXiv'26]

Reducing the ***variance*** in diffusion drafters.

# AdaFlash: Adaptive Speculative Decoding via On-Policy Distilled Diffusion Drafters

**Yu-Yang Qian**[1,2,*] **Hao-Cong Wu**[1,2,*] **Chen Chen**[3] **Jiacheng Sun**[3] **Zhenhua Dong**[3]
**Peng Zhao**[1,2,†] **Zhi-Hua Zhou**[1,2]
[1]State Key Laboratory for Novel Software Technology, Nanjing University
[2]School of Artificial Intelligence, Nanjing University [3]Huawei Foundation Model Dept

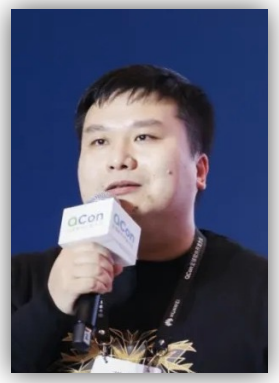
*arXiv:2607.19223* **2026, Jul 21**

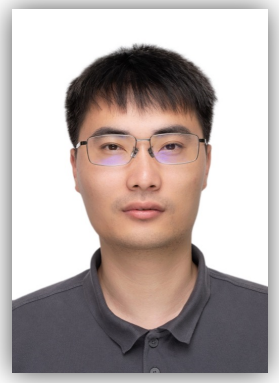

**Yu-Yang Qian***


**Hao-Cong Wu***


**Chen Chen**


**Jiacheng Sun**

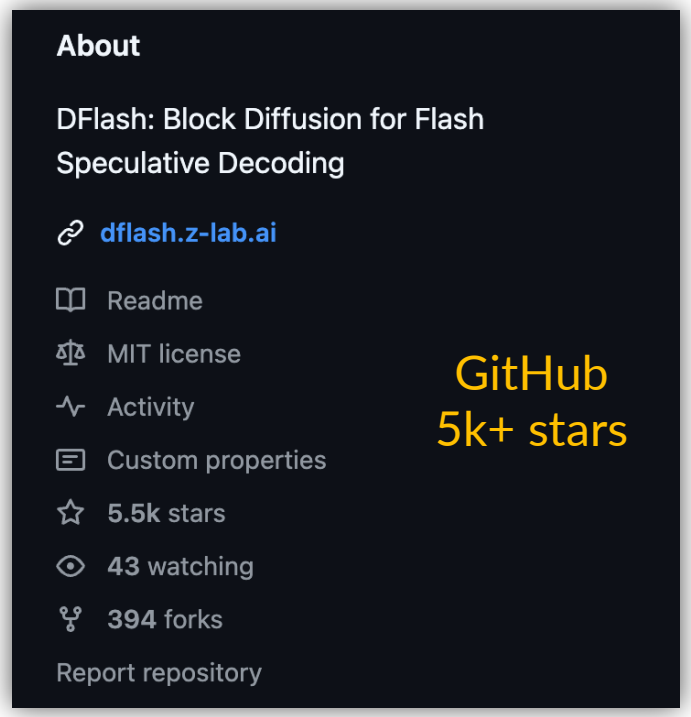

**Zhenhua Dong**


**Peng Zhao**


**Zhi-Hua Zhou**

# Current SOTA Method: DFlash

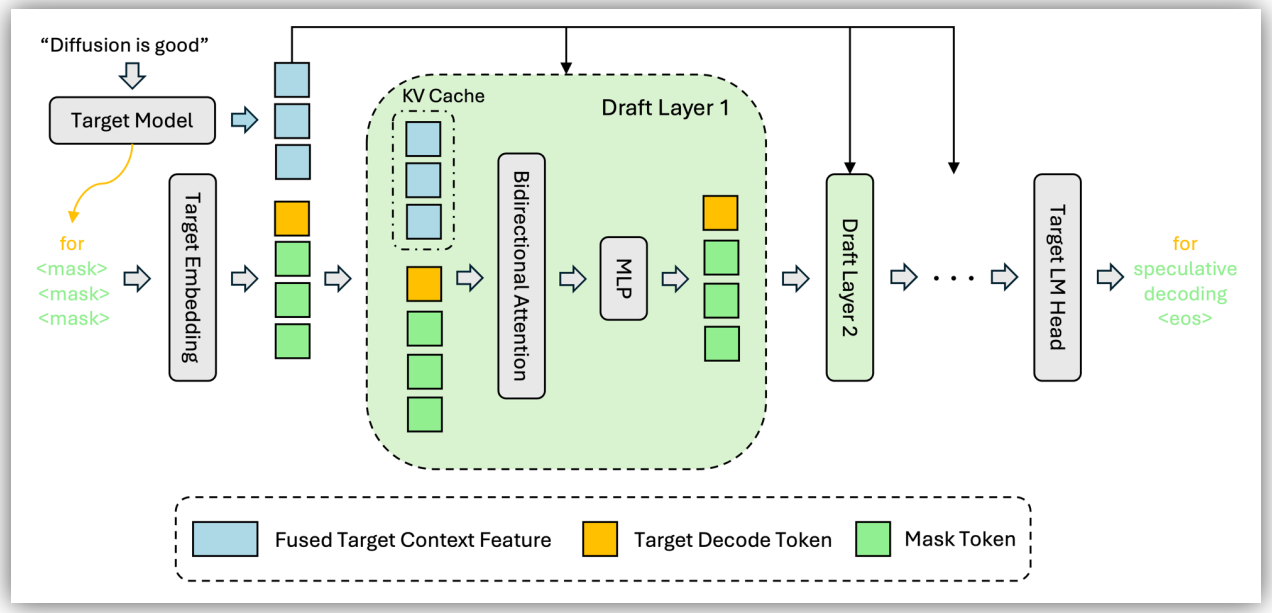
- SOTA of Speculative decoding: **5-6x** speedup compared to AR:


- DFlash is adopted by Xiaomi Mimo, DeepSeek, etc.

# Current SOTA Method: DFlash

- DFlash's key idea: employ diffusion model as drafter
  - **Parallel drafting:** generate a block of tokens in one forward pass;
  - **Feature fusion:** hidden states are compressed into one ***fused context feature;***
  - ***KV injection:*** injecting fused context feature as diffusion model's each layer's input.




Model architecture of DFlash

# Drawbacks of Diffusion Drafters

- We uncover a ***central pitfall*** of diffusion drafters:

Bidirectional attention in diffusion drafters is a ***double-edged sword.***

dLLM's bi-directional attn introduces ***high variance*** in draft sequences.

✅ Pros of ***diffusion drafter:***

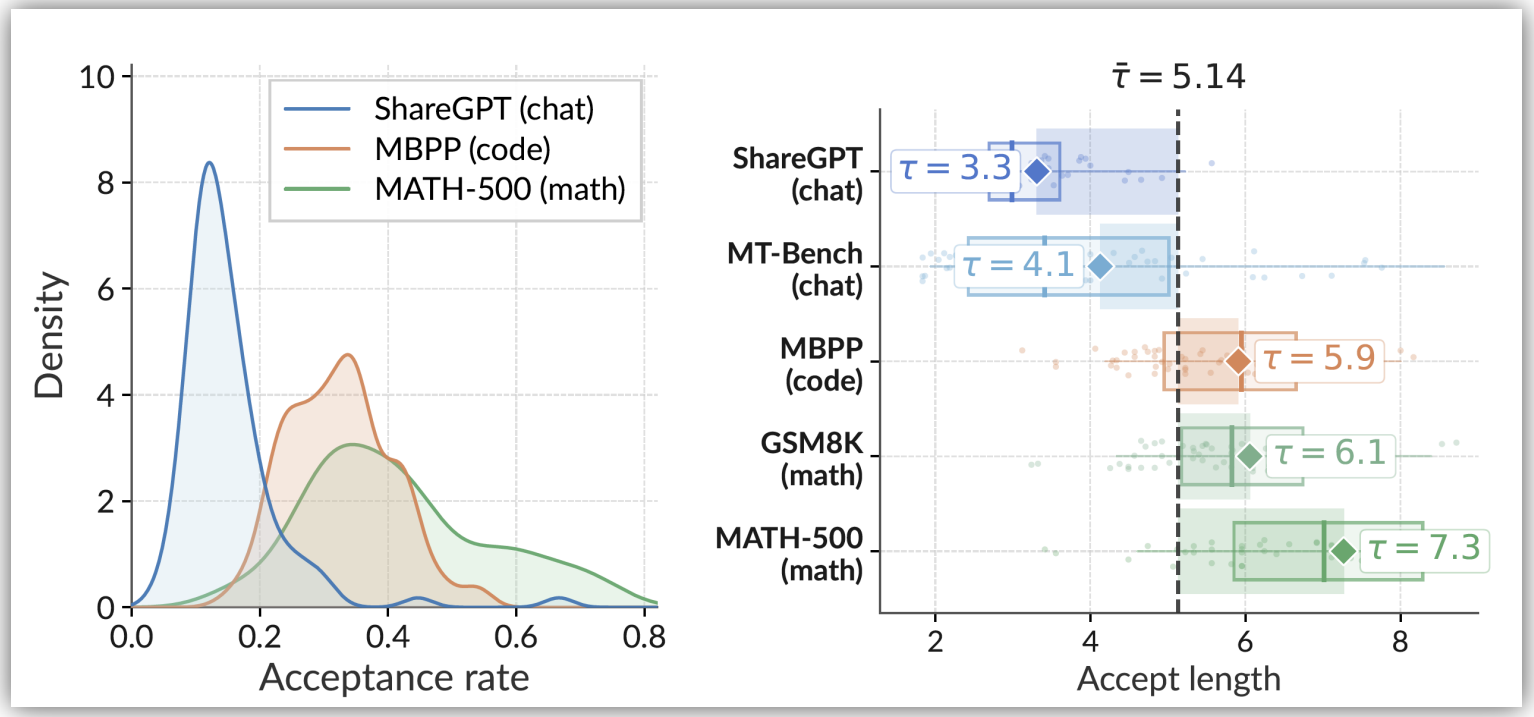
- Global contextual modeling
- One-pass parallel generation

❌ Cons: high variance

- Domain-level variance
- Token-level variance

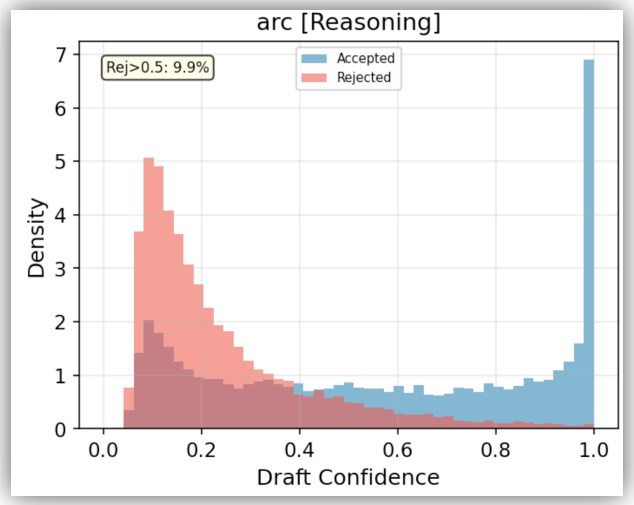
# Observation 1: Domain-Level Variance

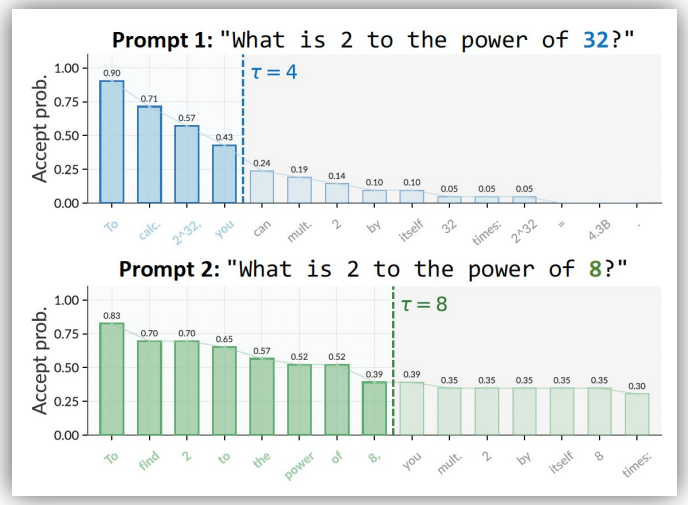
- Drafter's acceptance rate varies on different task domains:




**Motivation**: ***online update*** the draft model during deployment.

# Observation 2: Token-Level Variance



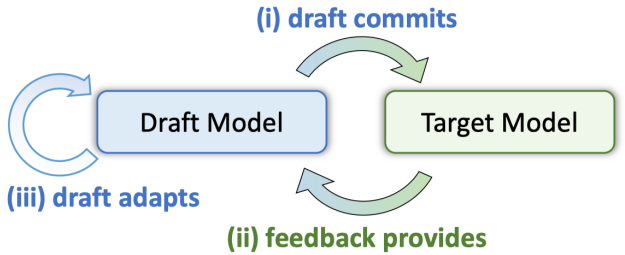

- Verifying all k tokens is consuming, especially under ***high concurrency.***
- Draft quality varies along positions, do not need to verify all *k* tokens

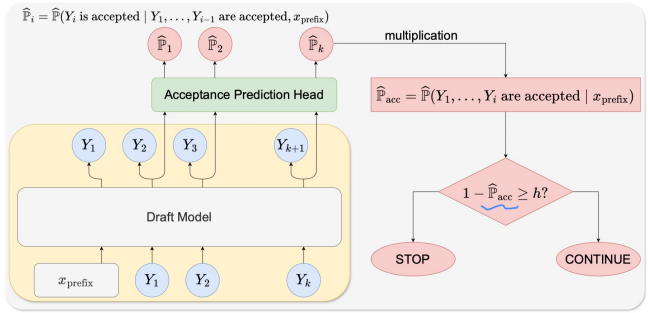
The verify length should be adjusted according to ***task / prompt.***

**Motivation:** target model's verify length should be ***adaptive.***

# *AdaFlash*: Overview

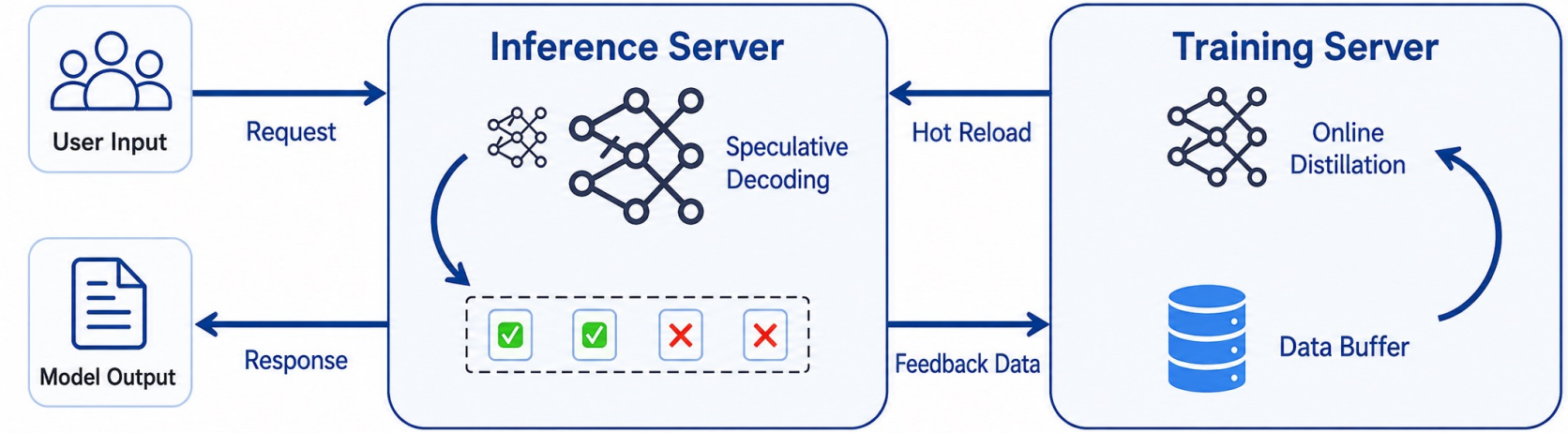
- Diffusion Drafter: bi-directional attention
  - Pros: parallel drafting;
  - **Cons:** ***high variance, domain/task specific.***
- Real-world Deployment: domain inconsistency, high concurrency
  1. For the domain-level variance:
     - Utilize ***interactive feedback***
     - On-policy distillation to adapt to new domain
  2. For the token-level variance:
     - ***Adaptive length head*** for dynamic verify length
     - Reducing unnecessary verification



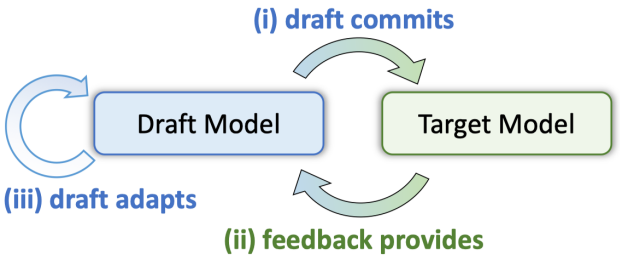



# Part I: *OPD for Diffusion Drafters*

- Utilize the ***free feedback*** in inference to help the draft model adapt:




- Continuously improve the draft model ***on-the-fly***:
  - Acceptance length ↑
  - Throughput (Speedup) ↑



# Part I: *OPD for Diffusion Drafters*

- Mixture OPD loss with ***reverse-KL:*** $\ell_{\text{OPD}} = \alpha \ell_{\text{hard}} + (1-\alpha)\ell_{\text{rkl}}$

$$\ell_{\text{rkl}} = \frac{1}{k}\sum_{i=1}^{k} \text{KL}\big([q_{\mathbf{w}}(\cdot \mid \mathbf{x}_{<t}, \mathbf{z})]_i \,\|\, p_{\mathbf{v}}(\cdot \mid \mathbf{x}_{<t+i})\big)$$

Reverse KL loss: mitigate "*mode covering*"

$$\ell_{\text{hard}} = -\frac{1}{k}\sum_{i=1}^{k} \log q_{\mathbf{w}}(x_i^* \mid \mathbf{x}_{<t+i}), \quad (x_i^* = \arg\max_x p_{\mathbf{v}}(x \mid \mathbf{x}_{<t+i}))$$

Hard label loss: strengthen the top-1 token signal

- Entry-wise ***divergence clipping***:

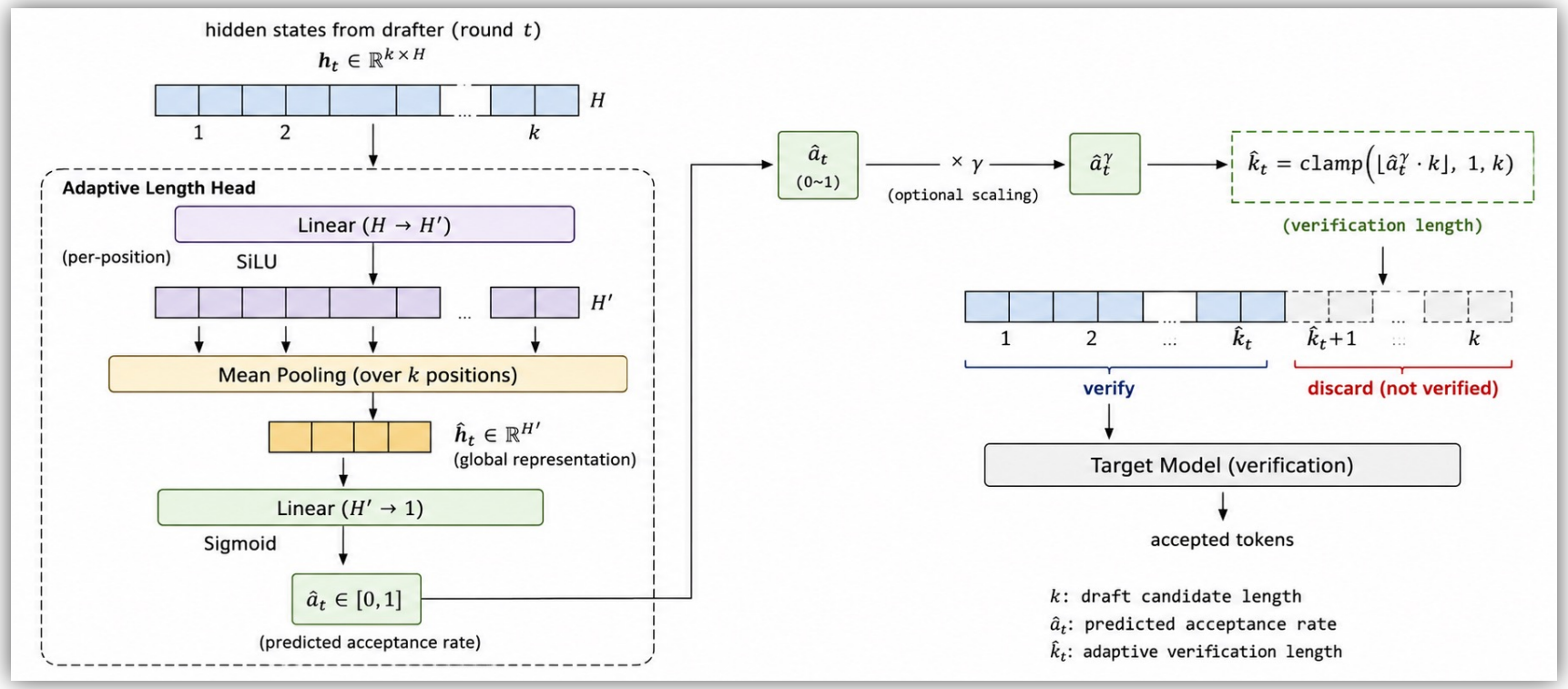

Prevent the diffusion drafter from being misled by outlier *tokens*

$$\ell_{\text{rkl}}^{\text{clip}} = \frac{1}{k}\sum_{i=1}^{k}\sum_{y\in\mathcal{V}} \min\left\{[q_{\mathbf{w}}(y \mid \mathbf{x}_{<t}, \mathbf{z})]_i \log \frac{[q_{\mathbf{w}}(y \mid \mathbf{x}_{<t}, \mathbf{z})]_i}{p_{\mathbf{v}}(y \mid \mathbf{x}_{<t+i})}, \delta\right\}$$

$$\ell_{\text{OPD}} = \alpha\, \ell_{\text{hard}} + (1-\alpha)\, \ell_{\text{rkl}}^{\text{clip}}$$

# Part II: *Adaptive Length Head*

- A lightweight ***adaptive length head*** that predicts the acceptance ratio on the fly.




Truncates the draft sequence to ***reduce unnecessary verification.***

# Part III: *Infrastructure Design*

- Target model verification candidate lengths: *Fixed* → ***Adaptive length***

- Exponential moving average (**EMA**) dynamically estimates the number of affordable requests of the current task.

```python
if self.use_prob_head:
    # run prob head on a secondary CUDA stream, overlapping with LM head
    prob_stream = _get_prob_head_stream(self.device)
    main_stream = torch.cuda.current_stream(self.device)
    prob_stream.wait_stream(main_stream)  # ensure draft_hidden is ready

    if self.tp_rank == 0:
        with torch.cuda.stream(prob_stream):
            with torch.no_grad():
                prob_logits = self.draft_model.prob_head(draft_hidden)  # [bs, block_size,
1]
                # torch.compile fuses sigmoid + cumprod + candidate_lens
                prob_fn = _get_compiled_prob_head_fn()
                candidate_lens = prob_fn(
                    prob_logits,
                    self.prob_head_mul_threshold,
                    self.prob_head_candidate_len_min,
                    self.block_size,
                )
```

- ***Infra*** optimization: operator fusion, concurrent launch, torch compile…

- The scheduler returns requests to pending queue when exceeding the budget.

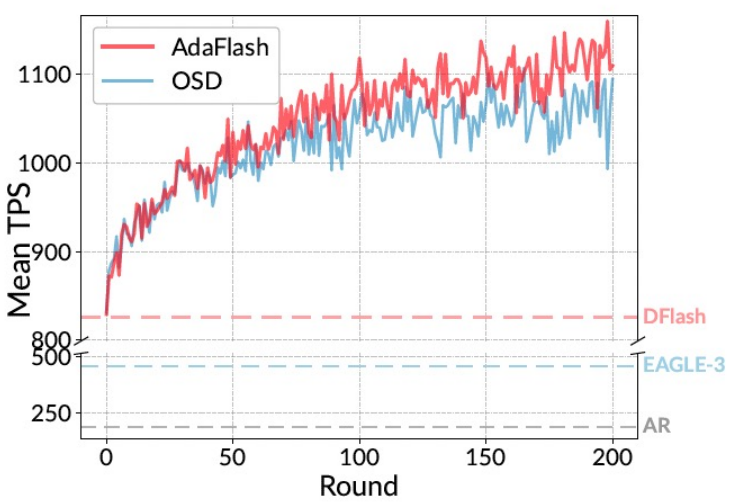
# Experimental Results

| Model | Method | Conc. | MATH | | | | CODE | | | | CHAT | | HYBRID | | *Avg.* | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | MathQA | | GSM8K | | OpenCodeInstruct | | CodeAlpaca | | ShareGPT | | Blend | | | |
| | | | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ |
| Q3-8B | EAGLE-3 | 1 | 2.45× | 4.60 | 2.55× | 4.75 | 2.20× | 4.06 | 2.37× | 4.58 | 2.09× | 3.89 | 2.36× | 4.54 | 2.34× | 4.40 |
| | | 32 | 0.72× | 4.60 | 0.70× | 4.76 | 0.68× | 4.07 | 0.67× | 4.58 | 0.61× | 3.90 | 0.70× | 4.54 | 0.68× | 4.41 |
| | | 64 | 0.46× | 4.59 | 0.43× | 4.75 | 0.44× | 4.07 | 0.40× | 4.57 | 0.40× | 3.90 | 0.45× | 4.54 | 0.43× | 4.40 |
| | | 128 | 0.34× | 4.59 | 0.35× | 4.75 | 0.33× | 4.07 | 0.32× | 4.58 | 0.30× | 3.90 | 0.33× | 4.54 | 0.33× | 4.41 |
| | DFlash | 1 | 4.38× | 7.09 | 3.84× | 6.27 | 4.08× | 6.69 | 3.39× | 5.63 | 2.11× | 3.40 | 3.39× | 6.07 | 3.53× | 5.86 |
| | | 32 | 1.86× | 7.10 | 1.59× | 6.28 | 1.77× | 6.68 | 1.46× | 5.66 | 1.05× | 3.42 | 1.51× | 6.04 | 1.54× | 5.86 |
| | | 64 | 1.24× | 7.11 | 1.03× | 6.29 | 1.17× | 6.67 | 0.91× | 5.65 | 0.70× | 3.41 | 0.98× | 6.06 | 1.01× | 5.86 |
| | | 128 | 0.90× | 7.11 | 0.82× | 6.28 | 0.86× | 6.67 | 0.74× | 5.66 | 0.51× | 3.40 | 0.74× | 6.04 | 0.76× | 5.86 |
| | OSD | 1 | 5.13× | 9.44 | 4.33× | 7.45 | 4.78× | 8.21 | 3.65× | 6.59 | 2.18× | 3.78 | 3.60× | 6.83 | 3.95× | 7.05 |
| | | 32 | 2.16× | 9.42 | 1.80× | 7.47 | 2.03× | 8.20 | 1.55× | 6.61 | 1.08× | 3.79 | 1.57× | 6.82 | 1.70× | 7.05 |
| | | 64 | 1.45× | 9.44 | 1.16× | 7.43 | 1.34× | 8.20 | 0.97× | 6.60 | 0.72× | 3.80 | 1.07× | 6.85 | 1.12× | 7.05 |
| | | 128 | 1.05× | 9.47 | 0.92× | 7.44 | 0.97× | 8.20 | 0.78× | 6.62 | 0.53× | 3.81 | 0.76× | 6.82 | 0.83× | 7.06 |
| | ADAFLASH | 1 | **5.32**× | 9.83 | **4.54**× | 7.75 | **4.87**× | 8.40 | **3.75**× | 6.80 | **2.24**× | 3.92 | **3.66**× | 7.00 | **4.06**× | 7.28 |
| | | 32 | **2.27**× | 9.84 | **1.83**× | 7.76 | **2.03**× | 8.39 | **1.58**× | 6.79 | **1.10**× | 3.92 | **1.61**× | 6.97 | **1.74**× | 7.28 |
| | | 64 | **1.65**× | 9.51 | **1.32**× | 7.28 | **1.51**× | 8.02 | **1.18**× | 6.31 | **1.04**× | 3.61 | **1.26**× | 6.56 | **1.33**× | 6.88 |
| | | 128 | **1.34**× | 9.51 | **1.27**× | 7.27 | **1.25**× | 8.05 | **1.11**× | 6.30 | **0.87**× | 3.58 | **1.05**× | 6.55 | **1.15**× | 6.88 |
| Q3-30B | EAGLE-3 | 1 | 1.79× | 4.39 | 1.56× | 4.06 | 1.96× | 5.15 | 1.93× | 4.94 | 1.32× | 3.08 | 1.79× | 4.49 | 1.73× | 4.35 |
| | | 32 | 1.33× | 4.40 | 1.15× | 4.06 | 2.85× | 5.15 | 1.71× | 4.93 | 1.15× | 3.08 | 1.49× | 4.50 | 1.61× | 4.35 |
| | | 64 | 1.03× | 4.40 | 0.86× | 4.06 | 1.44× | 5.15 | 1.33× | 4.93 | 0.91× | 3.08 | 1.16× | 4.48 | 1.12× | 4.35 |
| | | 128 | 0.71× | 4.41 | 0.63× | 4.06 | 1.15× | 5.15 | 1.10× | 4.93 | 0.60× | 3.08 | 0.77× | 4.49 | 0.83× | 4.35 |
| | DFlash | 1 | 2.09× | 5.15 | 1.84× | 4.85 | 2.56× | 6.86 | 2.39× | 6.03 | 1.04× | 2.79 | 2.07× | 5.43 | 2.00× | 5.19 |
| | | 32 | 1.61× | 5.14 | 1.41× | 4.85 | 3.80× | 6.84 | 2.15× | 6.01 | 1.14× | 2.79 | 1.74× | 5.42 | 1.98× | 5.18 |
| | | 64 | 1.26× | 5.16 | 1.05× | 4.84 | 1.95× | 6.84 | 1.68× | 6.01 | 0.93× | 2.81 | 1.39× | 5.42 | 1.38× | 5.18 |
| | | 128 | 0.87× | 5.15 | 0.77× | 4.84 | 1.55× | 6.84 | 1.40× | 6.00 | 0.60× | 2.80 | 0.92× | 5.42 | 1.02× | 5.18 |
| | OSD | 1 | 2.79× | 8.38 | 2.25× | 6.44 | 2.96× | 8.28 | 2.58× | 7.09 | 1.30× | 3.48 | 2.24× | 6.38 | 2.35× | 6.68 |
| | | 32 | 2.04× | 8.39 | 1.60× | 6.43 | 4.30× | 8.27 | 2.26× | 6.94 | 1.23× | 3.35 | 1.83× | 6.37 | 2.21× | 6.63 |
| | | 64 | 1.59× | 8.36 | 1.26× | 6.43 | 2.21× | 8.27 | 1.77× | 6.95 | 0.98× | 3.37 | 1.43× | 6.35 | 1.54× | 6.62 |
| | | 128 | 1.11× | 8.39 | 0.89× | 6.43 | 1.75× | 8.27 | 1.47× | 6.95 | 0.65× | 3.39 | 0.99× | 6.35 | 1.14× | 6.63 |
| | ADAFLASH | 1 | **2.86**× | 8.53 | **2.32**× | 6.69 | **2.99**× | 8.46 | **2.65**× | 7.21 | **1.32**× | 3.52 | **2.27**× | 6.56 | **2.40**× | 6.83 |
| | | 32 | **2.05**× | 8.53 | **1.59**× | 6.69 | **4.32**× | 8.43 | **2.33**× | 7.08 | **1.23**× | 3.42 | **1.85**× | 6.53 | **2.23**× | 6.78 |
| | | 64 | **1.60**× | 8.08 | **1.32**× | 6.42 | **2.35**× | 7.96 | **1.98**× | 6.63 | **1.08**× | 3.12 | **1.51**× | 6.16 | **1.64**× | 6.40 |
| | | 128 | **1.57**× | 8.11 | **1.28**× | 6.43 | **2.57**× | 7.96 | **2.32**× | 6.59 | **1.04**× | 3.11 | **1.33**× | 6.14 | **1.69**× | 6.39 |

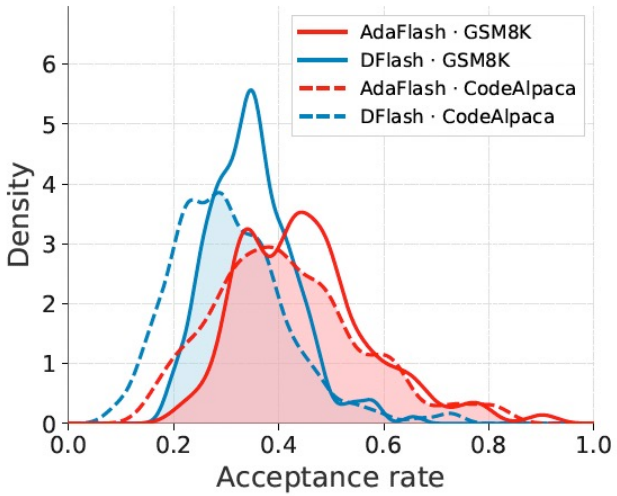
***AdaFlash*** consistently improves speedup.

Significant gains in ***high-concurrency*** scenarios.

# Experiment: Analysis




(a)




(b)


(c)

**Figure 2:** Analysis of ADAFLASH using Qwen3-8B as target model: (a) Speedup vs. online adaptation rounds on the MathQA dataset, showing continuous improvement over time. (b) Domain-level variance of DFlash and ADAFLASH, illustrating how on-policy distillation mitigates domain-level variance. (c) Token-level variance of DFlash and ADAFLASH on the MathQA dataset, showing acceptance probability along token position.

***AdaFlash*** reduces domain-level and token-level variance.

# Experiment: Cross-Domain

**Table 5:** Cross-domain generalization on Qwen3-8B. We compare offline DFlash with AdaFlash trained offline on PerfectBlend, a mixed-domain dataset distinct from each test benchmark, and evaluated with fixed weights without further online adaptation. In-domain AdaFlash results are reported in Table 1.

| Method | Conc. | Math | | | | Code | | | | *Avg.* | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MathQA | | GSM8K | | OpenCodeInstruct | | CodeAlpaca | | | |
| | | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ |
| DFlash | 1 | 4.38× | 7.09 | 3.84× | 6.27 | 4.08× | 6.69 | 3.39× | 5.63 | 3.92× | 6.42 |
| | 32 | 1.86× | 7.10 | 1.59× | 6.28 | 1.77× | 6.68 | 1.46× | 5.66 | 1.67× | 6.43 |
| | 64 | 1.24× | 7.11 | 1.03× | 6.29 | 1.17× | 6.67 | 0.91× | 5.65 | 1.09× | 6.43 |
| | 128 | 0.90× | 7.11 | 0.82× | 6.28 | 0.86× | 6.67 | 0.74× | 5.66 | 0.83× | 6.43 |
| AdaFlash (cross-domain) | 1 | 4.88× | 8.37 | 4.65× | 8.10 | 4.11× | 6.98 | 3.43× | 5.93 | 4.27× | 7.35 |
| | 32 | 2.07× | 8.39 | 1.84× | 8.11 | 1.76× | 6.96 | 1.48× | 5.95 | 1.79× | 7.35 |
| | 64 | 1.55× | 7.91 | 1.37× | 7.79 | 1.38× | 6.65 | 1.13× | 5.53 | 1.36× | 6.97 |
| | 128 | 1.29× | 7.93 | 1.35× | 7.76 | 1.15× | 6.64 | 1.07× | 5.54 | 1.22× | 6.97 |

AdaFlash generalizes well across different domains.

# Experiment: Long-Sequence Generation

**Table 6:** Long-sequence generation on MATH-500 (32K context, Qwen3-8B, thinking mode enabled).

| Method | Conc. | Speedup | $\tau$ | Acc. Rate |
|---|---|---|---|---|
| EAGLE-3 | 1 | 2.26× | 4.25 | 0.105 |
| | 32 | 1.82× | 4.37 | 0.109 |
| DFlash | 1 | 2.60× | 3.97 | 0.198 |
| | 32 | 2.46× | 4.04 | 0.203 |
| OSD | 1 | 3.51× | 5.28 | 0.285 |
| | 32 | 3.40× | 5.31 | 0.287 |
| AdaFlash | 1 | **3.67×** | **5.37** | **0.291** |
| | 32 | **3.52×** | **5.14** | **0.276** |

**Table 7:** Long-sequence generation on AIME25 (32K context, Qwen3-8B, thinking mode enabled).

| Method | Conc. | Speedup | $\tau$ | Acc. Rate |
|---|---|---|---|---|
| EAGLE-3 | 1 | 2.30× | 4.16 | 0.102 |
| | 32 | 1.68× | 4.26 | 0.105 |
| DFlash | 1 | 1.89× | 3.32 | 0.155 |
| | 32 | 1.54× | 3.36 | 0.157 |
| OSD | 1 | 3.01× | 5.21 | 0.281 |
| | 32 | 2.52× | 5.24 | 0.283 |
| AdaFlash | 1 | **3.18×** | **5.32** | **0.288** |
| | 32 | **2.77×** | **4.98** | **0.265** |

AdaFlash maintains its advantage in ***long-sequence generation***.

# Outline: *JetSpec*

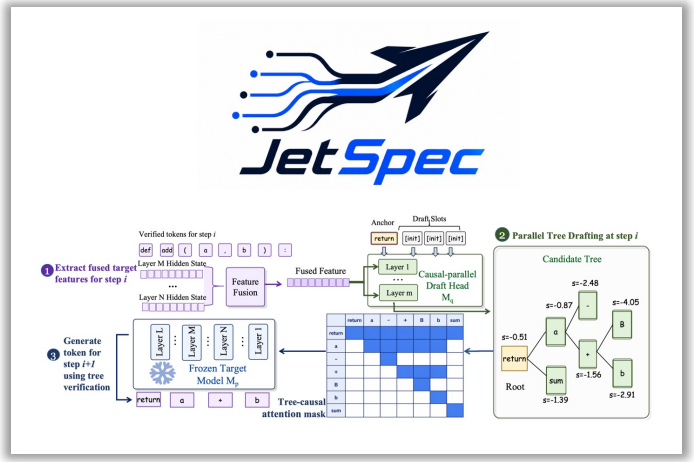
- Goal: ***Efficient LLM Inference*** via ***Parallel Decoding*:**




**OnlineSPEC**: *Online* evolving [ICML'26]




**AdaFlash**: Reducing *variance* [arXiv'26]

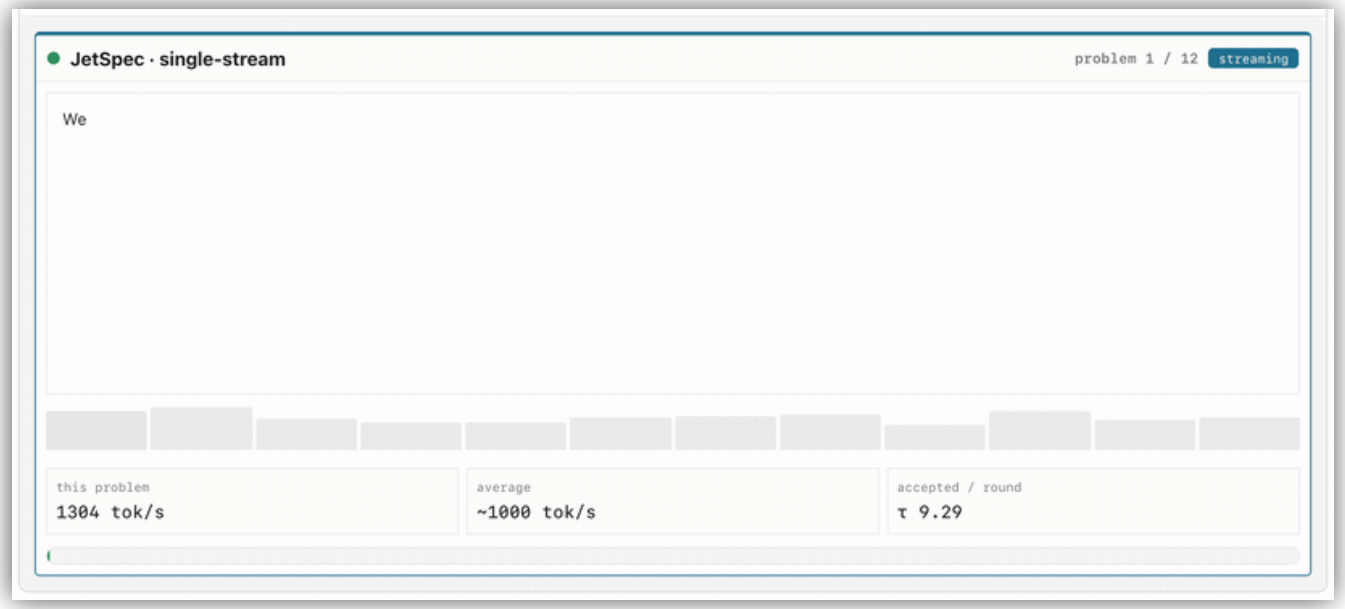



**JetSpec**: Parallel-causal tree [arXiv'26]

Explore the ***"Causality"*** in diffusion-based draft models.

# JetSpec: Breaking the Scaling Ceiling of Speculative Decoding with Parallel Tree Drafting

**Lanxiang Hu**[1] **Zhaoxiang Feng**[1] **Yulun Wu**[2] **Haoran Yuan**[3] **Yujie Zhao**[1] **Yu-Yang Qian**[4]
**Bojun Wang**[5] **Peng Zhao**[4] **Daxin Jiang**[5] **Yibo Zhu**[5] **Tajana Rosing**[1] **Hao Zhang**[1]
[1]UC San Diego [2] Zhejiang University [3]UIUC [4]Nanjing University [5]StepFun




*arXiv:2606.18394*
**2026, Jun 25**

# JetSpec: Motivation

- Key requirement of ***tree drafting***: each draft token's distribution must be conditioned on its own branch prefix.
- DFlash: all draft tokens decoded at once, ***no causal order***.

**(a) Causal head,** $\gamma = 0$

rank 1 ✓ `are` `told` `that` gap $= -\mathbf{0.34}$ *faithful* $\Rightarrow$ verifier accepts **6** tokens

rank 3 × `are` `given` `that` `the2product` gap $= +42.50$

**(b) Diffusion head,** $\gamma = 0$

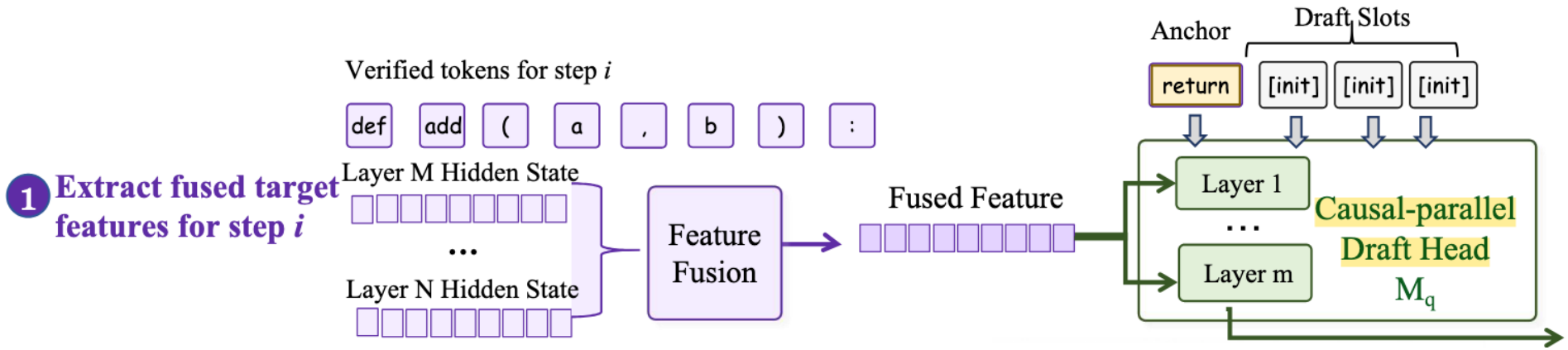
rank 1 × `given` `told` `that` gap $= +\mathbf{59.56}$ *incoherent* $\Rightarrow$ verifier accepts **4** tokens

rank 3 ✓ `are` `given` `that` `the` gap $= -3.69$ *faithful, but at rank 3*

Failure pattern of diffusion drafters in ***tree drafting***.

# JetSpec: Causal-Parallel Drafting

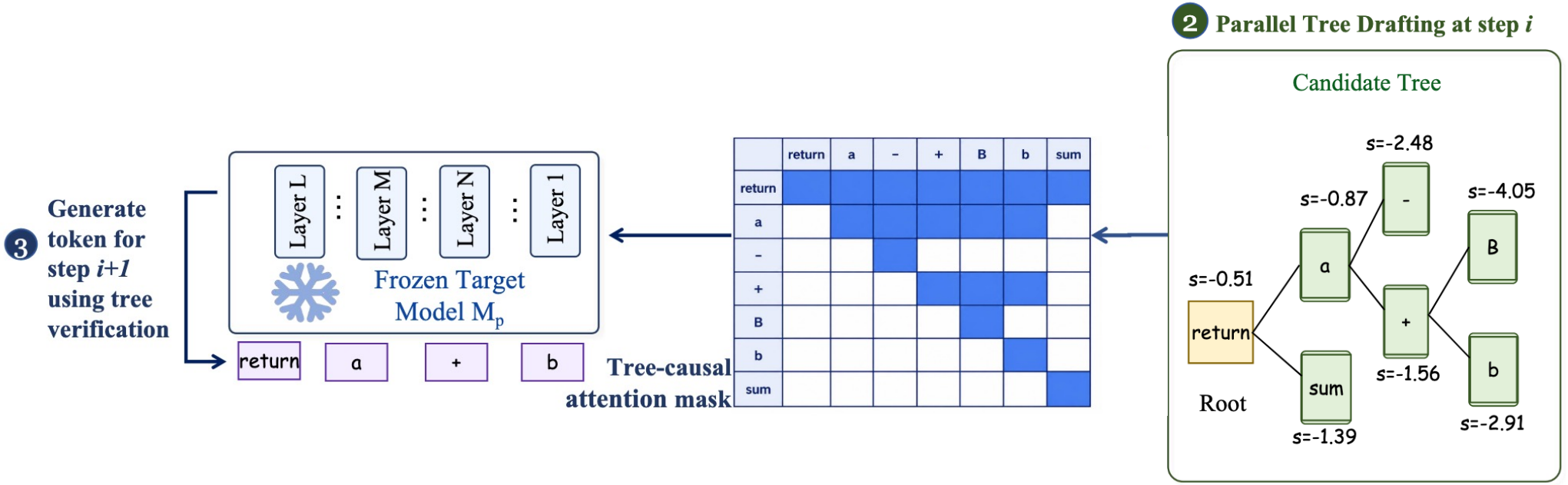
- JetSpec: ***tree drafting + tree verification***: all candidate tokens in one draft's forward pass, then verified in one target's forward pass.




- Applies ***a tree-causal attention mask*** to the draft head.
- Keeps drafting process in parallel, while each tree branch follows an autoregressive-like causal structure.

# JetSpec: Causal-Parallel Drafting

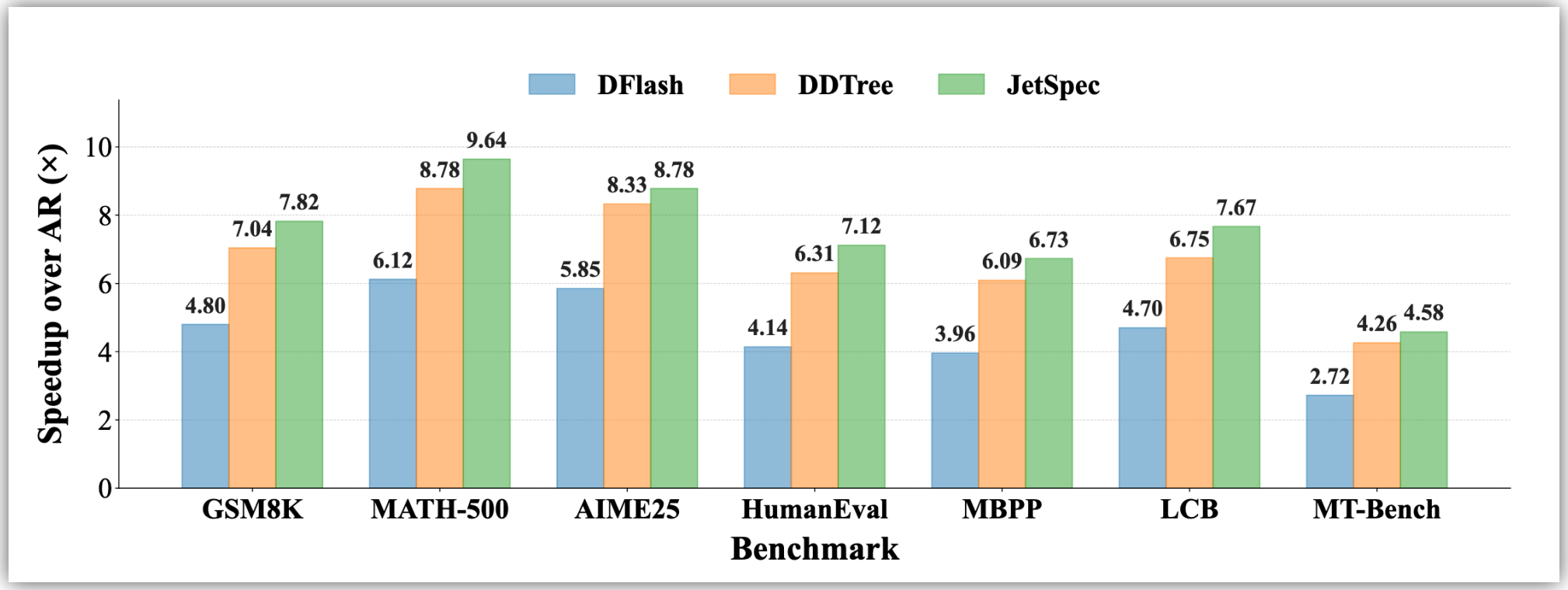
- JetSpec: ***tree drafting + tree verification***: all candidate tokens in one draft's forward pass, then verified in one target's forward pass.




StepFun with JetSpec

# Experimental Results

- ***JetSpec*** preserved branch-wise ***causal conditioning*** through block-level causal attention over hidden states, achieving the ***highest*** speedup ratio.



# Experimental Results

- JetSpec outperforms the contenders under different ***budgets***.

Table 1: Low-budget regime comparison of JetSpec and baselines trained with the same Qwen3-8B model and data recipe. We report results on math, coding, and chat benchmarks using non-thinking mode with a 3072 max tokens. We report end-to-end decoding speedup over standard AR decoding and average accepted length $\tau$ on H100 GPU.

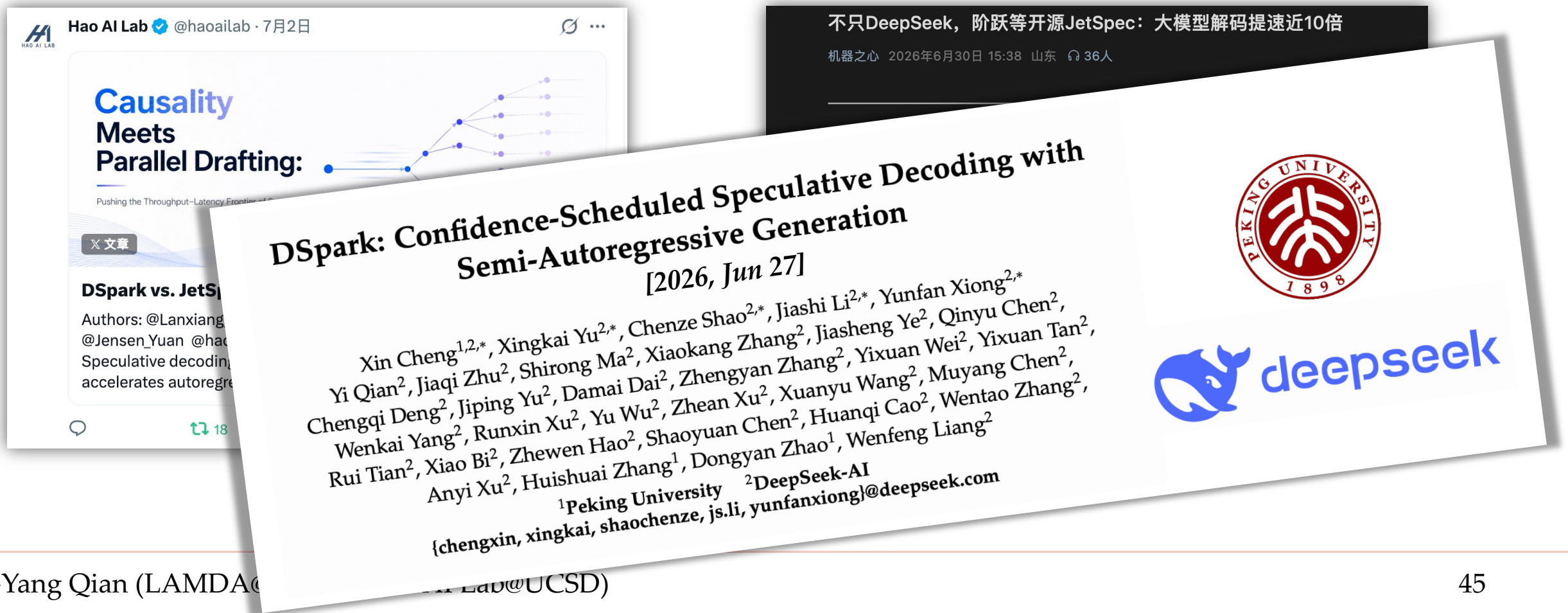
| Method | Budget | GSM8K | | MATH-500 | | AIME25 | | HumanEval | | MBPP | | LCB | | MT-Bench | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ |
| Temperature = 0 | | | | | | | | | | | | | | | |
| EAGLE-3 | 16 | 2.24 | 3.78 | 2.10 | 3.61 | 2.08 | 3.55 | 2.18 | 3.72 | 1.95 | | | | | |
| | 32 | 2.39 | 4.03 | 2.23 | 3.87 | 2.22 | 3.80 | 2.34 | 4.00 | 2.09 | | | | | |
| DFlash | 16 | 4.80 | 6.01 | 6.12 | 7.83 | **5.85** | **7.34** | 4.14 | 5.18 | 3.96 | | | | | |
| | 32 | 4.21 | 5.27 | 5.39 | 6.89 | 5.15 | 6.38 | 3.63 | 4.53 | 3.51 | | | | | |
| JetSpec | 16 | 4.80 | 6.00 | 6.06 | 7.75 | 5.78 | 7.23 | 4.26 | 5.32 | 3.97 | | | | | |
| | 32 | **4.89** | **6.14** | **6.35** | **8.23** | 5.75 | 7.25 | **4.29** | **5.35** | **4.03** | | | | | |
| Temperature = 1 | | | | | | | | | | | | | | | |
| EAGLE-3 | 16 | 2.16 | 3.65 | 2.01 | 3.48 | 1.89 | 3.24 | 2.02 | 3.58 | 1.89 | | | | | |
| | 32 | 2.26 | 3.90 | 2.12 | 3.74 | 2.01 | 3.49 | 2.19 | 3.86 | 2.02 | | | | | |
| DFlash | 16 | 4.32 | 5.44 | 4.87 | 6.30 | 3.55 | 4.69 | 3.56 | 4.45 | 3.52 | | | | | |
| | 32 | 3.85 | 4.84 | 4.29 | 5.55 | 3.44 | 4.48 | 3.22 | 4.00 | 3.17 | | | | | |
| JetSpec | 16 | 4.32 | 5.44 | 4.77 | 6.18 | 3.48 | 4.54 | 3.70 | 4.63 | 3.52 | | | | | |
| | 32 | **4.40** | **5.58** | **4.91** | **6.44** | **3.64** | **4.76** | **3.72** | **4.66** | **3.60** | | | | | |

Table 2: High-budget comparison on Qwen3-8B with at least 64 draft tokens. EAGLE-3 uses tree mode with max depth 8; larger budgets give minimal or worse gains due to training mismatch.

| Method | Budget | GSM8K | | MATH-500 | | AIME25 | | HumanEval | | MBPP | | LCB | | MT-Bench | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ | Speedup | $\tau$ |
| Temperature = 0 | | | | | | | | | | | | | | | |
| EAGLE-3 | 64 | 2.53 | 4.31 | 2.36 | 4.13 | 2.35 | 4.04 | 2.49 | 4.26 | 2.22 | 3.81 | 2.09 | 3.62 | 2.19 | 3.88 |
| DDTree | 64 | 5.63 | 6.18 | 6.51 | 7.16 | 6.40 | 6.96 | 5.08 | 5.57 | 4.99 | 5.49 | 5.47 | 6.06 | 3.74 | 4.51 |
| | 128 | 6.63 | 7.31 | 8.27 | 9.19 | 7.93 | 8.66 | 5.93 | 6.52 | 5.70 | 6.28 | 6.42 | 7.25 | 4.12 | 5.14 |
| | 256 | 7.04 | 7.77 | 8.78 | 9.81 | 8.33 | 9.24 | 6.31 | 6.96 | 6.09 | 6.70 | 6.75 | 7.72 | 4.26 | 5.41 |
| JetSpec | 64 | 5.98 | 6.56 | 6.76 | 7.42 | 6.47 | 7.00 | 5.53 | 6.06 | 5.34 | 5.88 | 5.95 | 6.59 | 3.97 | 4.77 |
| | 128 | 7.34 | 8.05 | 8.93 | 9.95 | 8.26 | 9.10 | 6.66 | 7.28 | 6.31 | 6.95 | 7.29 | 8.21 | 4.37 | 5.52 |
| | 256 | **7.82** | **8.62** | **9.64** | **10.76** | **8.78** | **9.82** | **7.12** | **7.78** | **6.73** | **7.43** | **7.67** | **8.79** | **4.58** | **5.94** |
| Temperature = 1 | | | | | | | | | | | | | | | |
| EAGLE-3 | 64 | 2.35 | 4.17 | 2.23 | 4.00 | 2.11 | 3.73 | 2.35 | 4.13 | 2.13 | 3.70 | 1.99 | 3.49 | 1.96 | 3.63 |
| DDTree | 64 | 5.28 | 5.77 | 5.68 | 6.26 | 4.94 | 5.46 | 4.65 | 5.09 | 4.65 | 5.11 | 5.28 | 5.82 | 3.50 | 4.22 |
| | 128 | 6.11 | 6.72 | 6.79 | 7.60 | 5.40 | 5.98 | 5.29 | 5.76 | 5.22 | 5.75 | 5.99 | 6.71 | 3.66 | 4.59 |
| | 256 | 6.41 | 7.17 | 7.10 | 8.13 | 5.26 | 6.20 | 5.43 | 6.03 | 5.49 | 6.12 | 6.26 | 7.17 | 3.81 | 4.88 |
| JetSpec | 64 | 5.63 | 6.19 | 5.97 | 6.60 | 4.95 | 5.48 | 5.02 | 5.52 | 5.00 | 5.51 | 5.76 | 6.38 | 3.63 | 4.37 |
| | 128 | 6.76 | 7.49 | 7.39 | 8.36 | 5.76 | 6.49 | 5.82 | 6.39 | 5.78 | 6.39 | 6.84 | 7.74 | 3.99 | 5.01 |
| | 256 | **7.16** | **8.03** | **7.83** | **9.01** | **5.94** | **7.06** | **6.19** | **6.85** | **6.11** | **6.83** | **7.25** | **8.29** | **4.06** | **5.22** |

# JetSpec

- Concurrent work: **DSpark** (@DeepSeek), released alongside our JetSpec, identifies the same ***lack of causal ordering issue*** in diffusion drafters.



# Summary

- In this talk, we investigate the inference efficiency in LLMs:
  - LLM's ***Parallel Decoding*** via Speculative Decoding


[OnlineSPEC]: *Online updates* of draft models during deployment time. Continuously update the drafter's parameter and improve acc length.


[AdaFlash]: *Reducing the variance* in diffusion drafters, with OPD updates and adaptive length head for deployment-time acceleration.


[JetSpec]: Introduce *causal-parallel drafting* for tree-structured diffusion drafters, further improve the speedup of speculative decoding.

***Speculative decoding*** is a promising paradigm for efficient inference!

# Summary

- In this talk, we investigate the inference efficiency in LLMs:
  - LLM's ***Parallel Decoding*** via Speculative Decoding

***Speculative decoding*** is a promising paradigm for efficient inference!

**Yu-Yang Qian**
PhD @ NJU (LAMDA),
visiting PhD @ UCSD

OnlineSPEC https://arxiv.org/abs/2603.12617

AdaFlash https://arxiv.org/abs/2607.19223

JetSpec https://arxiv.org/abs/2606.18394

https://www.lamda.nju.edu.cn/qianyy/

Thanks !