TapOut: A Bandit-Based Approach to Dynamic Speculative Decoding

Anonymous ACL submission

Abstract

Speculative decoding accelerates LLMs by using a lightweight draft model to generate tokens autoregressively before verifying them in parallel with a larger target model. However, determining the optimal number of tokens to draft remains a key challenge limiting the approach's effectiveness. Dynamic speculative decoding aims to intelligently decide *how many* tokens to draft to achieve maximum speedups. Existing methods often rely on hand-tuned, sensitive thresholds (e.g., token entropy), which are costly to set and generalize poorly across models and domains. We propose TapOut, an *online, training-free, plug-and-play* algorithm for dynamic speculation algorithm selection using Multi-Arm Bandits. Our approach employs a meta-algorithm which selects among multiple parameter-free dynamic speculation strategies based on past reward and exploration. We conduct extensive experiments across diverse model pairs and datasets, showing that TapOut achieves competitive or superior speedups compared to well-established dynamic speculation baselines *without any hyperparameter tuning*.

1 Introduction & Related Work

Large Language Models generate text in an autoregressive manner, where the generation of each token depends on the entire preceding context. Speculative decoding (Leviathan et al., 2023; Chen et al., 2023) accelerates decoder-only models by using a cheap draft model to generate tokens autoregressively while a larger target model performs verification of these tokens in parallel. This approach leads to massive gains in throughput as target model forward passes are minimized.

Ideally, drafting should continue until the next token is not accepted by the target model for maximum speedups. However, most previous works (Li et al., 2025; Cai et al., 2024; He et al., 2024; Fu et al., 2024) rely on a static draft length. Since oracle draft length exhibits high variance (Agrawal et al., 2024; Mamou et al., 2024) a static draft length is inherently suboptimal and results in missed speedup opportunities.

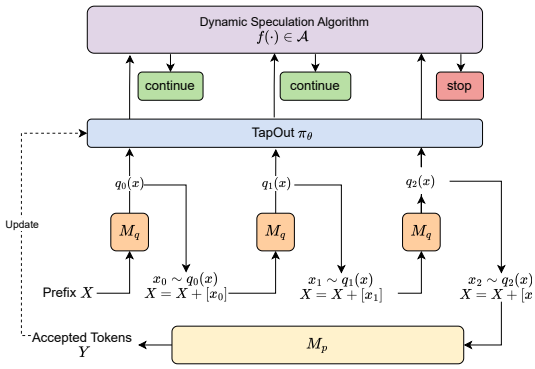


Figure 1: Dynamic speculative decoding with TapOut. After the draft model M_q generates predictions for the next token, TapOut π_θ uses a bandit algorithm to select a dynamic speculation algorithm $f(\cdot)$ which returns a decision to stop or continue drafting. If $f(\cdot)$ chooses to stop drafting, the target model M_p performs verification of all generated tokens and selects a subset which match its predictions. The accepted tokens are used to update the bandit policy.

Previous dynamic speculation techniques have sought to address this challenge. *Training-free* methods propose entropy-based stopping criteria (Zhang et al., 2025; Agrawal et al., 2024), while *training-based* methods train a classifier model to make probabilistic stopping decisions (Mamou et al., 2024; Huang et al., 2025). However, these techniques introduce hyperparameters, such as thresholds, which are costly to tune. Furthermore, none of them act in an *online* manner and thus cannot adapt to shifts in the prompt distribution.

Recently, BanditSpec (Hou et al., 2025), reframed speculative decoding hyperparameter selection as a Multi-Arm Bandit (MAB) problem. However, our work is distinct: BanditSpec aims to perform **static speculative decoding configu-**

ration selection (e.g., fixed draft length, specific draft model) for the full generation process. In contrast, our method addresses **the selection of the dynamic stopping policy** and uses a bandit to make token-wise stop/continue decisions *within a single draft*. In this work, we design TapOut, an online dynamic speculation algorithm based on MABs. Our method selects amongst *parameter- and training-free dynamic speculation techniques* and is thus extremely cheap to deploy. In addition, the application of MABs provides strong interpretability by tracking online parameters, such as arm values. Our contributions can be summarized as follows.

- We formulate the dynamic speculative decoding problem from a Multi-Arm Bandit perspective;
- We analyze Upper Confidence Bound (UCB) and Thompson Sampling (TS) bandit algorithms and propose practical implementation strategies, such as reward signals;
- We perform extensive experiments using several combinations of model families and datasets to evaluate our method. TapOut shows competitive or superior performance to tuning-required dynamic speculation baselines in all cases despite being *tuning-free*.

2 Methodology

2.1 Problem Setup

A Large Language Model (LLM) defines a probability distribution over sequences of tokens. Given a context sequence $x_{<t} = (x_1, x_2, \ldots, x_{t-1})$, the model autoregressively estimates the probability of the next token x_t via $P(x_t \mid x_{<t}; \theta)$, where θ denotes the model parameters.

In the context of speculative decoding, a lightweight draft model first proposes a sequence of tokens $X = (x_1, x_2, \ldots, x_\gamma)$, which are then verified in parallel by a more accurate target model. The target model accepts a prefix of X, denoted $Y = (y_1, y_2, \ldots, y_m)$, where $m \leq \gamma$ and $Y \subseteq X$.

A Multi-Arm Bandit problem is defined over a 2-tuple $(\mathcal{A}, \mathcal{R})$. Here, $\mathcal{A}$ is the set of actions that the agent with policy π can take at each timestep, and $\mathcal{R}^a$ is the stationary reward distribution for action $a \in \mathcal{A}$, with expected reward $\mathbb{E}[R^a] = \mathbb{P}(r \mid a)$. For a finite horizon of time steps T, the agent aims to maximize its cumulative reward, $\sum_{t=1}^{T} r_t$, where $r_t \sim R^{a_t}$ is the reward received at time step t based on the action taken, a_t. In our case, each $a \in \mathcal{A}$ is a dynamic speculation technique, represented as a function $f(\cdot)$ that returns a decision $d \in \{\texttt{stop}, \texttt{continue}\}$. We consider two setups with different action granularity: 1) *sequence-level* bandits where actions are chosen at the beginning of each drafting session and all generation steps use the same dynamic speculation technique, and 2) *token-level* bandits where each position in the drafting sequence is itself a MAB.

2.2 Reward Formulation

The agent must manage a core trade-off between speculation aggressiveness and efficiency. Maximizing the acceptance length requires drafting longer, more speculative sequences, while maximizing the acceptance rate requires drafting shorter, higher-confidence sequences to avoid rejection.

We test two reward formulations for the sequence-level bandit, r^{simple} and r^{blend}, where, $r^{simple} = \frac{|Y|}{\gamma}$ and $r^{blend} = \alpha \cdot \frac{|Y|}{\gamma} + (1-\alpha) \cdot \frac{|Y|}{|X|}$. Note that r^{simple} is the normalized acceptance length and r^{blend} blends both normalized acceptance length and acceptance rate. Speedup is a direct outcome of maximizing acceptance length and rate, and thus r^{blend} serves as an effective proxy reward. In practice, we fix $\alpha = 0.5$ for equal weighting on the two objectives. For the token-level bandit, the reward is simply $r = 1$ if the token is accepted and $r = 0$ otherwise.

2.3 TapOut

Figure 1 demonstrates the overall flow while abstracting the specific bandit algorithm π_θ used. After the draft model generates logits for the next token, TapOut is queried and selects a dynamic speculation algorithm which returns a stopping decision. If the algorithm chooses to stop, the target model performs verification of the candidate tokens and the bandit's parameters are updated. Algorithmic pseudocode is provided in Appendix A.1.

We implement Upper Confidence Bound (UCB) and Thompson Sampling algorithms. For UCB, we compare UCB1 (Auer et al., 2002) and UCB-Tuned algorithms. At each timestep t, UCB1 selects the arm a maximizing the sum of the empirical mean reward and an exploration bonus, $a_t = \arg\max_{a \in \mathcal{A}} \left(\hat{\mu}_a(t) + \sqrt{\frac{2 \log t}{N_a(t)}} \right)$, where $\hat{\mu}_a(t) = \frac{1}{N_a(t)} \sum_{s=1}^{t-1} r_s \mathbf{1}_{\{a_s = a\}}$, is the empirical mean reward of arm a, and $N_a(t)$ is the number of times arm a has been played before time t.

Algorithm	Stopping Condition	Threshold h
Max-Confidence	$p(x_t = \hat{x}_1 \mid x_{<t}) < h$	0.8
SVIP (Zhang et al., 2025)	$\sqrt{\mathcal{H}(p(x_t \mid x_{<t}))} > h$	0.6
AdaEDL (Agrawal et al., 2024)	$1 - \sqrt{\gamma \cdot \mathcal{H}(p(x_t \mid x_{<t}))} < \lambda_t$	N/A
SVIPDifference	$\sqrt{\mathcal{H}(p(x_t \mid x_{<t})) - \sqrt{\mathcal{H}(p(x_{t-1} \mid x_{<t-1}))}} > h$	0.2
LogitMargin	$p(\hat{x}_1 \mid x_{<t}) - p(\hat{x}_2 \mid x_{<t}) < h$	0.2

Table 1: TapOut arm algorithms with stopping condition and, if applicable, threshold. $\hat{x}_1$ and $\hat{x}_2$ represent the top-1 and top-2 logit indices respectively.

UCB-Tuned refines the exploration process by incorporating variance estimates for each arm's rewards to incentivize early exploitation for low-variance arms while retaining a larger exploration bonus for high variance arms. More details can be found in Appendix A.2.

Thompson Sampling (TS) maintains a posterior distribution over the expected reward θ_a for $a \in \mathcal{A}$. At each timestep t, it samples a reward estimate from the posterior, $\tilde{\theta}_a(t) \sim p(\theta_a \mid \mathcal{D}_t)$, where $\mathcal{D}_t$ is the history of observed actions and rewards up to t. The action selected is $a_t = \arg\max_{a \in \mathcal{A}} \tilde{\theta}_a(t)$. After playing arm a_t and observing reward r_t, the posterior is updated via Bayes' rule, $p(\theta_{a_t} \mid \mathcal{D}_{t+1}) \propto p(r_t \mid \theta_{a_t})\, p(\theta_{a_t} \mid \mathcal{D}_t)$. The token-level problem provides token-wise binary rewards for each bandit, and thus we can use the standard Beta-Bernoulli Thompson Sampling configuration. In contrast, the sequence-level problem uses a continuous reward $r_t \in [0, 1]$ representing the quality of an entire generated sequence, so here we use a Gaussian prior with a known noise variance.

3 Experiments

We evaluate acceptance length m, acceptance rate $\%$, and speedup s over vanilla speculative decoding with a fixed draft length of $\gamma = 6$. For dynamic speculation strategies, the max draft length is set to 128 tokens as a proxy for unbounded draft lengths. Table 1 describes the bandit's arm algorithms with set thresholds. Note that these are fixed and not tuned on any dataset. SVIPDifference and LogitMargin are new techniques which we describe in Appendix A.3.

3.1 Ablations

We run the Llama-3.2 1B/3.1 8B model pair on SpecBench (Xia et al., 2024) for all ablations. Extra ablations can be found in Appendix A.4.

Reward Formulation Table 2 shows the performance of a sequence-level UCB1 bandit for both reward formulations. The blended reward, as described by r^{blend} in Section 2.2, results in a larger speedup and higher acceptance rates compared to the acceptance length reward, r^{simple}. This discrepancy arises because the r^{simple} reward function is an incomplete proxy for throughput. By rewarding only the total number of accepted tokens while ignoring the acceptance rate, it incentivizes the bandit to speculate aggressively, as this strategy maximizes the reward signal even at the cost of many rejected tokens, as shown in Figure 2. A category-wise comparison is also shown in Appendix A.5. Thus, we fix the reward to r^{blend} for all future experiments.

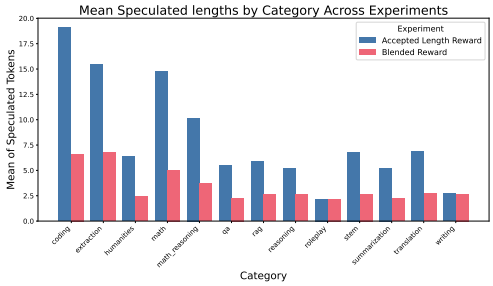


Figure 2: Comparison of speculated length across reward types. The accepted length reward (r^{simple}) causes the agent to aggressively speculate while the blended reward (r^{blend}) acts more conservatively.

Reward Type	m	$\%$	s
Accepted Length Reward	4.13	0.53	1.02
Blended Reward	2.74	**0.80**	**1.22**

Table 2: Comparison of reward strategies across acceptance and speedup metrics. Blended reward outperforms the simple accepted length reward.

UCB1 vs UCB-Tuned Next, we ablate on the UCB algorithm. Although the oracle draft length exhibits a high variance (Agrawal et al., 2024; Mamou et al., 2024), which provides UCB-Tuned an advantage, we find that the standard UCB1 algorithm provides a 12% greater speedup and thus adopt it for future experiments. We attribute this to the significantly lower variance of the r^{blend} reward than r^{simple}. This stability obviates the need for UCB-Tuned's sophisticated variance-aware exploration, favoring the simpler yet more effective strategy of UCB1.

3.2 Results

We evaluate our method against several existing training-free baselines, including Max-Confidence (MC), SVIP (Zhang et al., 2025), and AdaEDL (Agrawal et al., 2024). We use the Llama (Grattafiori et al., 2024) 3.2 1B/3.1 70B,

Model	Method	Tuning Required?	MT-Bench			HumanEval		
			m	%	s	m	%	s
Llama-3 1B/70B	Static-6	No	3.51	0.59	1.00	4.58	0.76	1.00
	AdaEDL	Yes	2.77	0.88	0.93	4.39	0.85	1.03
	SVIP	Yes	3.63	0.86	1.04	6.11	0.82	1.13
	MC	Yes	5.30	0.82	**1.16**	8.58	0.73	*1.14*
	TapOut - Seq TS	No	1.04	0.78	0.76	4.12	0.84	1.04
	TapOut - Seq UCB1	No	5.29	0.82	*1.15*	8.48	0.73	**1.19**
	TapOut - Token TS	No	0.92	0.86	0.72	2.30	0.88	0.84
	TapOut - Token UCB1	No	2.65	0.82	1.03	4.38	0.84	1.06
Llama-3 1B/8B	Static-6	No	3.47	0.58	1.00	4.35	0.72	1.00
	AdaEDL	Yes	1.89	0.88	1.15	3.64	0.89	*1.07*
	SVIP	Yes	2.44	0.87	**1.25**	3.49	0.91	1.06
	MC	Yes	3.14	0.82	*1.18*	4.63	0.86	**1.09**
	TapOut - Seq TS	No	1.10	0.81	1.04	1.39	0.88	0.85
	TapOut - Seq UCB1	No	3.14	0.82	**1.25**	4.62	0.86	**1.09**
	TapOut - Token TS	No	1.89	0.80	1.15	4.63	0.86	**1.09**
	TapOut - Token UCB1	No	2.24	0.83	1.11	2.96	0.88	0.97
Olmo-2 1B/32B	Static-6	No	1.91	0.32	1.00	2.13	0.35	1.00
	AdaEDL	Yes	0.67	0.67	1.11	0.69	0.69	1.15
	SVIP	Yes	1.17	0.67	1.24	1.29	0.67	*1.35*
	MC	Yes	1.38	0.60	1.25	1.57	0.60	**1.38**
	TapOut - Seq TS	No	0.80	0.62	1.12	0.76	0.67	1.16
	TapOut - Seq UCB1	No	1.37	0.59	**1.27**	1.57	0.60	1.25
	TapOut - Token TS	No	1.38	0.60	*1.26*	1.20	0.66	1.21
	TapOut - Token UCB1	No	1.28	0.64	1.25	1.53	0.62	1.33
Gemma3 270M/27B	Static-6	No	3.81	0.63	**1.00**	4.37	0.73	1.00
	AdaEDL	Yes	0.66	0.66	0.76	4.17	*0.88*	**1.52**
	SVIP	Yes	1.77	0.65	0.85	4.82	0.80	1.49
	MC	Yes	2.56	0.63	0.89	7.62	0.76	*1.50*
	TapOut - Seq TS	No	0.73	0.63	0.77	1.25	0.91	1.05
	TapOut - Seq UCB1	No	2.62	0.63	*0.90*	7.85	0.77	**1.52**
	TapOut - Token TS	No	1.28	0.65	0.85	1.73	0.86	1.21
	TapOut - Token UCB1	No	1.62	0.56	0.81	3.42	0.81	1.32

Table 3: Evaluation of dynamic speculation methods on MT-Bench and HumanEval. **Bolded** and *italicized* values represent best and second-best results for a given model family and dataset, respectively.

Model	Method	Training Required?	m	%	s
Llama-3 1B/8B	Static-6	No	3.29	0.55	1.00
	SpecDec++	Yes	2.73	0.65	0.99
	TapOut - Seq TS	No	1.00	0.78	0.90
	TapOut - Seq UCB1	No	2.76	0.81	**1.22**
	TapOut - Token TS	No	2.76	0.81	1.05
	TapOut - Token UCB1	No	2.23	0.82	*1.13*

Table 4: Comparison of our training-free bandit techniques to the training-based SpecDec++ using Llama-3.2 1B/3.1 8B on SpecBench.

Llama-3.2 1B/3.1 8B, Gemma3 (Team et al., 2025) 270M/27B[1] and OLMo-2 (OLMo et al., 2025) 1B/32B model pairs on the MT-Bench (Zheng et al., 2023) and HumanEval (Chen et al., 2021) datasets. Results on SpecBench (Xia et al., 2024) are also provided in Appendix A.6. We also train a SpecDec++ (Huang et al., 2025) classifier and provide a comparison with our bandit techniques in Table 4. All models run on H100 GPUs. For baseline methods with a threshold, we perform a grid search using Llama-3.2 1B/3.1 8B on the SpecBench (Xia et al., 2024) dataset and fix the optimal values for all other model pairs and datasets.

3.3 Analysis

As shown in Tables 3 and 4, in all except one instance, **TapOut sequence-level UCB1 provides top-2 speedup performance despite being tuning-free**. This result is particularly noteworthy as the baselines were given a significant advantage: their hyperparameters were tuned directly on the SpecBench benchmark, a process from which TapOut was exempt. On out-of-distribution tests, such as Olmo-2 on MT-Bench, the adaptive nature of TapOut enables it to outperform these now-suboptimal fixed thresholds, demonstrating the robustness of our approach. Furthermore, when the best baseline varies across datasets (e.g., Llama-3.2 1B/3.1 8B), TapOut matches performance by adaptively choosing stopping policies.

Token-level bandits likely underperform because each individual bandit receives no feedback on throughput and only binary acceptance/rejection. This can cause the bandit to become over-optimistic, as described in Section 3.1. Bandits also offer interpretability for routing decisions. As shown in Figure 3, the order of arm values exactly matches the order of speedup gains that each baseline obtains in Table 3 for Gemma3 on HumanEval (e.g., AdaEDL > MC > SVIP). Further examples are provided in Appendix A.8.

[1]Gemma 3 employs sparse attention mechanisms that are not fully optimized in the Transformers library (Wolf et al., 2020), which can lead to limited or no speedup in some cases.

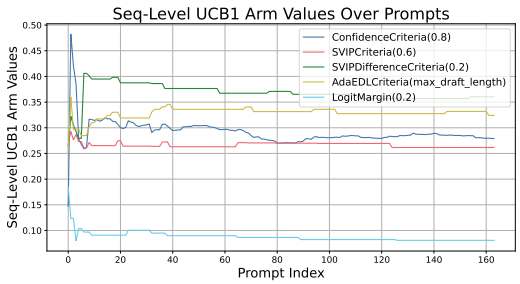


Figure 3: TapOut - Seq UCB1 Arm Value (μ_i) progression for Gemma3 on HumanEval.

4 Conclusion

In this work, we propose TapOut, an online, tuning-free approach to dynamic speculative decoding. Our method uses a MAB to select an algorithm from a pool of training-free dynamic speculation approaches based on past performance, which in turn provides a binary stopping decision. TapOut is simple to implement, highly interpretable, and effectively leverages online feedback to prioritize the best-performing strategies in context. We evaluate our approach using several model families and datasets which demonstrate that the sequence-level UCB1 provides speedup which is competitive or superior to all tuning-required baselines.

5 Limitations

While our work demonstrates the effectiveness and interpretability of bandits for dynamic speculative decoding, there are a few limitations. For instance, the overall performance of bandits does not always exceed the best baseline method, and the datasets we use to evaluate our approach are relatively small. Larger evaluation settings, such as the Alpaca (Stanford Center for Research on Foundation Models, 2023) dataset, may provide a better opportunity for online learning. Further, the performance of the bandit is inherently upper-bounded by the individual performances of the dynamic speculation algorithms used as arms. It requires a set of reasonably strong and diverse heuristics in order to make effective online decisions. An interesting follow-up work could investigate other reinforcement learning approaches which leverage context information, such as contextual bandits.

References

Sudhanshu Agrawal, Wonseok Jeon, and Mingu Lee. 2024. Adaedl: Early draft stopping for speculative decoding of large language models via an entropy-based lower bound on token acceptance probability. *Preprint*, arXiv:2410.18351.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning*.

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

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

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. 2024. Break the sequential dependency of llm inference using lookahead decoding. *Preprint*, arXiv:2402.02057.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Zhenyu He, Zexuan Zhong, Tianle Cai, Jason D. Lee, and Di He. 2024. Rest: Retrieval-based speculative decoding. *Preprint*, arXiv:2311.08252.

Yunlong Hou, Fengzhuo Zhang, Cunxiao Du, Xuan Zhang, Jiachun Pan, Tianyu Pang, Chao Du, Vincent Y. F. Tan, and Zhuoran Yang. 2025. Banditspec: Adaptive speculative decoding via bandit algorithms. *Preprint*, arXiv:2505.15141.

Kaixuan Huang, Xudong Guo, and Mengdi Wang. 2025. Specdec++: Boosting speculative decoding via adaptive candidate lengths. *Preprint*, arXiv:2405.19715.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *Proc. of ICML*.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2025. Eagle: Speculative sampling requires rethinking feature uncertainty. *Preprint*, arXiv:2401.15077.

Tianyu Liu, Yun Li, Qitan Lv, Kai Liu, Jianchen Zhu, Winston Hu, and Xiao Sun. 2025. PEARL: Parallel speculative decoding with adaptive draft length. In *Proc. of ICLR*.

Jonathan Mamou, Oren Pereg, Daniel Korat, Moshe Berchansky, Nadav Timor, Moshe Wasserblat, and Roy Schwartz. 2024. Dynamic speculation lookahead accelerates speculative decoding of large language models. *Preprint*, arXiv:2405.04304.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, and 21 others. 2025. 2 olmo 2 furious. *Preprint*, arXiv:2501.00656.

Yuhao Shen, Junyi Shen, Quan Kong, Tianyu Liu, Yao Lu, and Cong Wang. 2025. Speculative decoding via hybrid drafting and rollback-aware branch parallelism. *Preprint*, arXiv:2506.01979.

Stanford Center for Research on Foundation Models. 2023. Alpaca: A strong, replicable instruction-following model. `https://crfm.stanford.edu/2023/03/13/alpaca.html`. Accessed: 2025-10-01.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard

Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proc. of EMNLP: System Demonstrations*.

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

Ziyin Zhang, Jiahao Xu, Tian Liang, Xingyu Chen, Zhiwei He, Rui Wang, and Zhaopeng Tu. 2025. Draft model knows when to stop: Self-verification speculative decoding for long-form generation. *Preprint*, arXiv:2411.18462.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Preprint*, arXiv:2306.05685.

A Appendix

A.1 TapOut Pseudocode

Algorithm 1 TapOut

Require: Draft model M_q, Target model M_p, Bandit π_θ, Prefix `prefix`, Max draft length γ

```
Initialize draft sequence X ← []
for i = 1 to γ do
    Sample token x_i ~ M_q(prefix + X)
    Append x_i to X
    if π_θ signals stop at step i then
        break
    end if
end for
Verify X with M_p, accept maximal matching prefix Y
Sample final token x_t ~ M_p(prefix + X_[:n])
Update bandit: update(π_θ, |Y|, |X|, γ)
return prefix + X_[:n] + [x_t]
```

A.2 UCB-Tuned

UCB-Tuned selects arms using the following equation:

$$a_t = \arg\max_{a \in \mathcal{A}} \left(\hat{\mu}_a(t) + \sqrt{\frac{\log t}{N_a(t)} \min\left(\frac{1}{4}, V_a(t)\right)} \right)$$

where the variance estimate is

$$V_a(t) = \hat{\sigma}_a^2(t) + \sqrt{\frac{2 \log t}{N_a(t)}},$$

and $\hat{\sigma}_a^2(t)$ is the empirical variance of the rewards observed for arm a.

A.3 Arm Algorithms

Our bandits use five different algorithms as arms which are described briefly below. Let $p(x_t \mid x_{t-1}, x_{t-2}, \ldots)$ denote the draft token distribution at timestep t, and let $\mathcal{H}(p)$ denote the entropy of this distribution.

- **SVIP**
  Decision rule:
  $$\sqrt{\mathcal{H}(p(x_t \mid x_{<t}))} > h$$
  where h is a fixed entropy threshold.

- **SVIP-Difference**
  Decision rule:
  $$\sqrt{\mathcal{H}(p(x_t \mid x_{<t}))} - \sqrt{\mathcal{H}(p(x_{t-1} \mid x_{<t-1}))} > h$$
  This captures a spike in uncertainty between two consecutive token steps.

- **Confidence**
  Decision rule:
  $$p(x_t = \hat{x}_t \mid x_{<t}) < h$$
  where $\hat{x}_t = \arg\max_x p(x \mid x_{<t})$, and h is a confidence threshold.

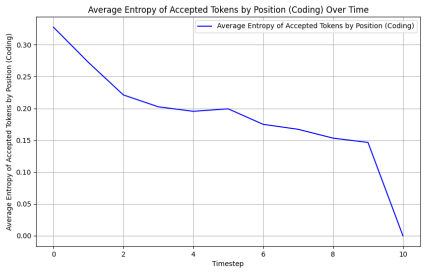
- **Logit Margin**
  Decision rule:
  $$p(x_t = \hat{x}_1 \mid x_{<t}) - p(x_t = \hat{x}_2 \mid x_{<t}) < h$$
  where $\hat{x}_1$ and $\hat{x}_2$ are the top-1 and top-2 predicted tokens respectively.

- **AdaEDL**
  Decision rule:
  $$1 - \sqrt{\gamma \cdot \mathcal{H}(p(x_t \mid x_{<t}))} < \lambda_t$$
  Update rule: After each full draft:
  $$\begin{aligned} r_t &= \frac{n_{\text{acc}}}{n_{\text{drafted}}} \\ \text{accept_rate}_{t+1} &= \beta_1 \cdot \text{accept_rate}_t \\ &\quad + (1 - \beta_1) \cdot r_t \\ \lambda_{t+1} &= \beta_2 \cdot \lambda_t \\ &\quad + (1 - \beta_2) \cdot (\lambda_t + \\ &\quad + \epsilon \cdot \text{sign}(\alpha - r_t)) \end{aligned}$$
  where:
  - n_{acc}: Number of accepted tokens
  - n_{drafted}: Number of drafted tokens
  - $\alpha, \beta_1, \beta_2, \gamma, \epsilon$: Hyperparameters

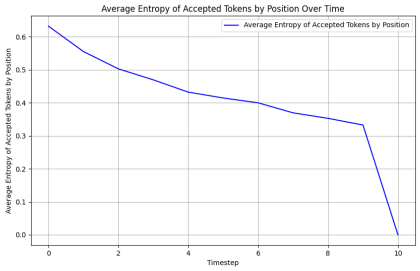
A.4 Extra Ablations

Entropy Thresholding

We provide a practical motivation for our method by analyzing draft model entropy at accepted positions for coding and non-coding prompts. As shown in Figure 4, 1) coding prompts cause significantly lower entropy levels than non-coding prompts and 2) entropy decays as the generation length increases (the draft model becomes more confident as it keeps generating). Thus, a static threshold across all prompts and positions in the drafting sequence, such as the one used in SVIP (Zhang et al., 2025), is suboptimal. An online method is necessary to adapt on-the-fly to changes in the context and possibly favor a different technique.



(a) Coding prompts.



(b) Non-coding prompts.

Figure 4: Draft model $\sqrt{\mathcal{H}(p(x_t \mid x_{<t}))}$ by position t for accepted tokens in responses to coding and non-coding prompts.

Adding More Possible Arms

We also experiment with the possible arms the bandit has can use. Specifically, for each dynamic speculation algorithm, we add several arms with different thresholds (e.g., entropy thresholds at 0.2, 0.4, and 0.6 for SVIP) rather than a single arm to see whether to opportunity to explore more arms results in better speedups. However, we find that the original setup with one threshold per dynamic speculation technique provides 12% stronger speedups overall and adopt it for all future experiments.

A.5 Reward Comparison by Category

Category	Accepted Length Reward		Blended Reward	
	Acceptance Rate	Speedup	Acceptance Rate	Speedup
coding	0.48	0.78	**0.83**	**1.18**
extraction	0.48	0.78	**0.79**	**1.09**
humanities	0.51	0.99	**0.78**	**1.23**
math	0.53	0.82	**0.86**	**1.13**
math_reasoning	0.57	0.89	**0.87**	**1.13**
qa	0.54	1.07	**0.79**	**1.27**
rag	0.50	1.04	**0.79**	**1.22**
reasoning	0.59	1.11	**0.79**	**1.26**
roleplay	**0.77**	**1.28**	**0.77**	**1.28**
stem	0.52	0.95	**0.80**	**1.22**
summarization	0.50	1.08	**0.76**	**1.30**
translation	0.42	0.99	**0.73**	**1.34**
writing	0.79	1.25	**0.80**	**1.26**

Table 5: Comparison of accepted length and speedup across reward types (r^{simple} and r^{blend}, see Section 2.2) for the UCB1 bandit on SpecBench. We find r^{blend} outperforms r^{simple} in all categories.

A.6 SpecBench Results

Model	Method	Tuning Required?	m	%	s
Llama-3 1B/70B	Static-6	No	3.49	0.58	1.00
	AdaEDL	Yes	2.50	0.88	0.96
	SVIP	Yes	3.37	0.85	1.07
	Max-Confidence	Yes	4.70	0.80	**1.18**
	Seq-Level TS	No	2.57	0.73	1.02
	Seq-Level UCB1	No	4.70	0.81	*1.16*
	Token-Wise TS	No	1.47	0.83	0.89
	Token-Wise UCB1	No	3.05	0.84	1.07
Llama-3 1B/8B	Static-6	No	3.29	0.55	1.00
	AdaEDL	Yes	1.53	0.86	1.08
	SVIP	Yes	2.18	0.85	0.68
	Max-Confidence	Yes	2.77	0.81	*1.21*
	Seq-Level TS	No	1.00	0.80	0.90
	Seq-Level UCB1	No	2.76	0.81	**1.22**
	Token-Level UCB1	No	2.23	0.82	1.13
	Token-Level TS	No	1.19	0.79	0.92
Olmo 2 1B/32B	Static-6	No	1.93	0.32	1.00
	AdaEDL	Yes	0.67	0.67	1.10
	SVIP	Yes	1.15	0.68	1.23
	Max-Confidence	Yes	1.37	0.62	**1.28**
	Seq-Level TS	No	0.75	0.64	1.10
	Seq-Level UCB1	No	1.36	0.62	1.25
	Token-Wise TS	No	0.75	0.65	1.11
	Token-Wise UCB1	No	1.33	0.65	*1.26*
Gemma3 270M/27B	Static-6	No	3.32	0.55	1.00
	AdaEDL	Yes	0.66	0.66	0.94
	SVIP	Yes	1.67	0.63	*1.04*
	Max-Confidence	Yes	2.16	0.59	1.02
	TapOut - Seq TS	No	0.73	0.63	0.93
	TapOut - Seq UCB1	No	2.29	0.60	**1.06**
	TapOut - Token TS	No	1.52	0.57	0.97
	TapOut - Token UCB1	No	1.71	0.57	0.99

Table 6: Evaluation of dynamic speculation methods on SpecBench across Llama-3, SmolLM2, and Olmo-2 models families. **Bolded** results are the best and *italicized* results are 2nd best.

A.7 UCB1 vs UCB-Tuned: Speedup Comparison

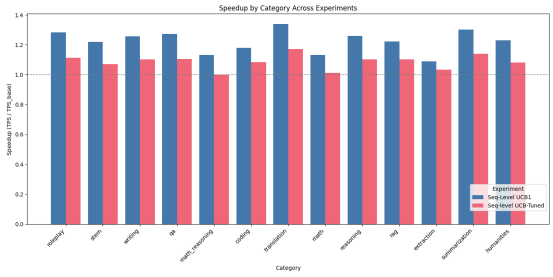
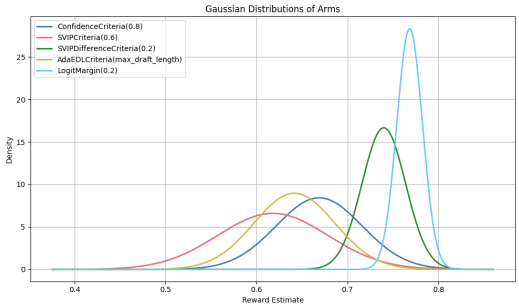
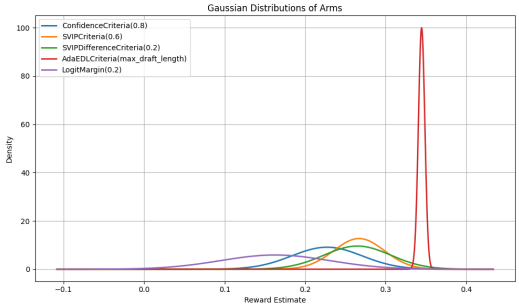


Figure 5: Comparison of speedup between UCB1 and UCB-Tuned. UCB1 provides better performance across all prompt categories.

A.8 Thompson Sampling Plots



(a) Llama-3.2 1B/3.1 70B.



(b) Olmo-2 1B/32B.

Figure 6: Comparison of Seq-Level TS arm distributions after processing all prompts in HumanEval.

As shown in Figure 6, the arm distributions for TS are quite different across model families for the same dataset. These results offer insights into the decision-making process and inherent alignment between draft and target model. For instance, note that in Table 3 the acceptance rates for Olmo-2 1B/32B are quite low across all datasets, indicating poor alignment between the draft and target model. Correspondingly, as seen in Figure 6, the bandit chooses the most conservative strategy (AdaEDL) and exploits it heavily, leading to the peaked distribution. Conversely, the Llama-3.2 1B/3.3 70B models appear much better aligned with higher acceptance rates overall, and thus the bandit favours the more aggressive stopping policies of LogitMargin and SVIPDifference.

A.9 Additional Related Work

Speculative decoding (Leviathan et al., 2023; Chen et al., 2023) algorithms have been well-studied in the literature with several different approaches for drafting and verification. EAGLE (Li et al., 2025) and related works (Cai et al., 2024) eliminate the use of a separate draft model and modify the target model itself to support cheap approximations of the full forward pass. Other approaches, such as REST (He et al., 2024), use simple data structures, like Tries, instead of neural networks as candidate sequence generators. We refer the interested reader to (Xia et al., 2024) for a comprehensive survey and comparison of popular techniques. These techniques largely cover *how* to draft and verify tokens.

Dynamic speculative decoding is a much more recent branch of literature concerning *how long* to draft tokens before performing verification. Early works can largely be categorized as *training-based* and *training-free*. Training-based works (Huang et al., 2025; Mamou et al., 2024) train a classifier to decide whether to keep drafting tokens. Training-free works either use token-level signals, such as entropy (Zhang et al., 2025; Agrawal et al., 2024), to make stopping decisions or completely modify the speculative decoding paradigm to run draft and target models in parallel (Liu et al., 2025; Shen et al., 2025). In this work, we focus on training-free dynamic speculation strategies which use token-level signals as they are simple to implement in existing speculative decoding pipelines and quite versatile, but suffer from sensitive thresholds.