BEYOND THE TIP OF THE ICEBERG: DEPTH-AWARE TOP-K MARGIN DISTILLATION FOR DIFFUSION SPECULATIVE DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

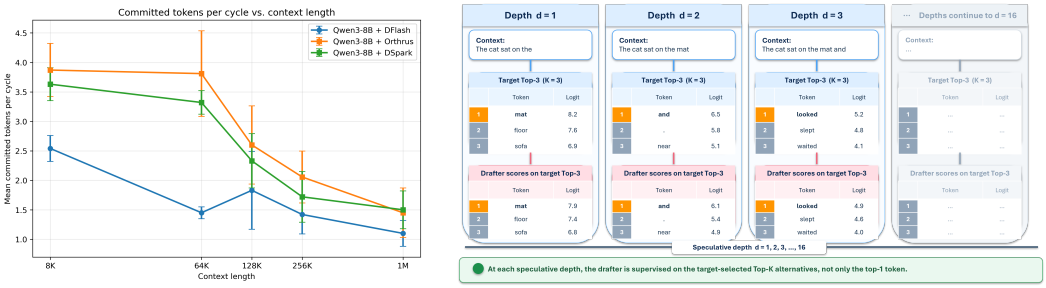
Speculative decoding accelerates LLM inference by using a lightweight drafter to propose future tokens for parallel verification. Diffusion-based and block-parallel drafters can generate long speculative sequences efficiently, but their acceptance often degrades rapidly with speculative depth, particularly at longer context lengths. We introduce Depth-Aware Top-K Margin Distillation, a training objective designed to improve drafter–verifier alignment by accounting for the unequal importance of speculative positions. At each depth, our objective distills the verifier's distribution over its Top-K candidate tokens and aligns the logit margins between adjacent ranked candidates, while assigning larger weights to earlier positions whose rejection invalidates the remaining speculative suffix. The formulation requires no modification to the drafter architecture or inference procedure and can, in principle, be applied to different speculative drafter families, while being particularly well suited to diffusion and parallel drafting where depth-wise acceptance degradation is pronounced. We apply our objective to a block-diffusion drafter with three target models (Qwen3-8B, Qwen3-14B, and Gemma-4-31B) on five benchmarks. With Qwen3-8B and Gemma-4-31B, it improves the average acceptance length by 15–16% over the same drafter trained with its original objective. With Qwen3-14B, it improves acceptance length by 28% over a published baseline that uses a smaller proposal budget. These gains come with correspondingly higher end-to-end speedup. The benefit grows with context length: on BABILong, the relative gain in acceptance length rises from 28% at 8K tokens to 55% at 1M tokens. These gains come with no additional inference-time overhead.

1 INTRODUCTION

Autoregressive decoding is the dominant cost of serving large language models (LLMs). Each new token requires a full forward pass of the model, and because that pass is bound by memory bandwidth rather than compute, most of the accelerator sits idle while weights and KV-cache entries stream through memory (Pope et al., 2023). *Speculative decoding* (Leviathan et al., 2023; Chen et al., 2023) turns this idle compute into throughput. A lightweight *drafter* proposes a block of Γ future tokens, and the target model, acting as a *verifier*, scores all of them in a single parallel pass. It keeps the longest prefix consistent with its own distribution, then appends one corrected token. Because the verifier has the final say, the output distribution is provably unchanged, and the speedup is governed almost entirely by one quantity: *how many drafted tokens are accepted per verification step* (Xia et al., 2024).

From sequential to parallel drafting. Early drafters were small autoregressive LMs, which pay one sequential forward pass per drafted token. This cost bounds how far ahead they can usefully speculate. Later work shrank the drafter into extra decoding heads (Stern et al., 2018; Cai et al., 2024; Ankner et al., 2024) or into a feature-level autoregressive module attached to the verifier (Li et al., 2024a;b; 2025b), but drafting still unrolls token by token. A recent line of work removes that dependency entirely. *Block-parallel drafters* predict all Γ positions in one forward pass (Gloeckle et al., 2024; An et al., 2026), possibly followed by a lightweight sequential module (Cheng et al., 2026), and *diffusion drafters* generate the whole block by parallel denoising (Christopher et al.,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Long-context collapse of parallel drafters. (b) Depth-aware Top-K supervision (ours).

Figure 1: **The problem and our idea. (a)** Mean committed tokens per verification cycle ($\pm$ one standard deviation) versus context length for DFlash, DSpark, and Orthrus, all with a Qwen3-8B verifier. Speculative efficiency degrades sharply as context grows, and at 1M tokens every method approaches one token per cycle. **(b)** *Depth-aware Top-K supervision for DFlash speculative decoding.* At each speculative depth, the target model provides its Top-K candidate tokens rather than only the top-1 prediction. The drafter is trained on the same teacher-selected alternatives, allowing it to learn not only the most likely token but also other plausible candidates and their relative confidence structure across deeper speculative positions.

2025; Li et al., 2025a; Sandler et al., 2026; Chen et al., 2026; Van Nguyen et al., 2026), building on advances in discrete and block diffusion language models (Sahoo et al., 2024; Arriola et al., 2025; Nie et al., 2025). Because the drafting cost of these models is nearly independent of Γ, long drafts of $\Gamma{=}16$ tokens or more become essentially free to *propose*. For instance, DFlash (Chen et al., 2026) drafts a full block in a single pass and outperforms strong autoregressive drafters such as EAGLE-3.

Long drafts are only as good as their prefix. A long draft is valuable only if the verifier accepts a long prefix of it, and acceptance is *prefix-gated*: once position d is rejected, positions $d{+}1, \ldots, \Gamma$ are discarded regardless of their quality. Write β_d for the probability that position d is accepted given that positions $1, \ldots, d{-}1$ were accepted. The expected number of accepted draft tokens is then

$$\mathbb{E}[\tau] \;=\; \sum_{d=1}^{\Gamma} \prod_{j=1}^{d} \beta_j. \tag{1}$$

To see what this implies, consider a drafter with $\beta_d{=}0.9$ at every depth and $\Gamma{=}16$. It yields $\mathbb{E}[\tau] \approx 7.33$ accepted tokens. Lowering the acceptance at the *first* position to 0.85 scales every term of Eq. (1) and costs about 0.41 tokens per step. The same drop at the *last* position costs about 0.01 tokens, roughly $40\times$ less. Speculative positions are therefore far from interchangeable. An error near the front of the block wipes out the whole suffix, while an error near the end is rarely even reached. Parallel drafters face this asymmetry at its worst. Since every position is predicted from the same context without seeing its own drafted predecessors, uncertainty compounds with depth and the per-position acceptance β_d decays rapidly. This collapse is sharpest at *long context lengths*, which is exactly the regime where speculative decoding matters most (Sun et al., 2024; Sadhukhan et al., 2025). Figure 1a measures three recent parallel drafters on the same Qwen3-8B verifier: DFlash (Chen et al., 2026), the semi-autoregressive DSpark (Cheng et al., 2026), and the dual-view diffusion head of Orthrus. As the context grows from 8K to 1M tokens, the mean number of tokens committed per verification cycle falls from 3.87 to 1.45 for Orthrus and from 2.54 to 1.10 for DFlash. At 1M tokens every method approaches one committed token per cycle, the rate of plain autoregressive decoding, so almost the entire draft is wasted while the drafting and verification overhead is still paid. This extends the degradation that Chen et al. (2026) already report for their base drafter beyond 4K tokens. The three drafters differ in architecture, so we target what they have in common: the objective that trains each depth of the draft. Figure 1b previews our remedy. At every speculative depth, the drafter is supervised on the verifier's Top-K candidates and their relative confidences, not only on its top-1 prediction.

Why existing training objectives fall short. Current training objectives capture this structure only in part. Parallel and diffusion drafters are typically fit with token-level cross-entropy on hard

next-token labels. The loss is averaged uniformly over the draft block (Gloeckle et al., 2024; An et al., 2026), or, in DFlash, reweighted by an exponentially decaying position weight, introduced as a heuristic that speeds up convergence (Chen et al., 2026). DSpark applies the same position weights to cross-entropy, a full-distribution matching loss, and a confidence loss (Cheng et al., 2026), so depth-weighted distillation itself is already in use. Knowledge distillation (Hinton et al., 2015; Kim & Rush, 2016) offers a richer signal, and DistillSpec (Zhou et al., 2024) and online speculative decoding (Liu et al., 2024) show that aligning the drafter to the *verifier's* distribution, rather than to the data, directly raises acceptance. HASS (Zhang et al., 2025) further restricts distillation to the verifier's Top-K tokens, where nearly all acceptance mass lives, but it does so for autoregressive drafters and without depth priority. For deep parallel drafts, these objectives leave three limitations. **(1) Depth priority is not tied to the verifier's candidate structure:** existing depth-weighted objectives apply their weights to hard labels or to full-vocabulary distribution matching. They never target the verifier's Top-K candidates or their ranking at each depth, and they offer no principled way to choose how fast the weights should decay. With finite drafter capacity, the drafter needs *both*: to know which positions matter most, and to receive the most decision-relevant signal about the verifier at exactly those positions. **(2) They supervise the wrong target set:** hard-label cross-entropy reveals only the single realized token and ignores the verifier's plausible alternatives. Full-vocabulary divergences spend effort on a long tail of tokens that never affect verification. **(3) They match probabilities, not the geometry of the decision:** whether a draft token survives verification depends on the verifier's *ranking* of candidates and how decisively it separates them. A drafter whose top-1 and top-2 tokens are nearly tied where the verifier's are well separated, or vice versa, flips its decision under small perturbations, even when its matched probabilities look close on average.

Two very recent works confront depth directly, and both illustrate the gap we target. SpecDiff-2 (Sandler et al., 2026) observes that autoregressive-style alignment concentrates gains at early positions, and proposes *streak distillation*, which optimizes the expected accepted streak through products of per-position acceptance terms. AdaFlash (Qian et al., 2026) attributes position-wise variance to bidirectional attention in diffusion drafters. It addresses this with on-policy reverse-KL distillation (Gu et al., 2024; Agarwal et al., 2024) and an additional adaptive-length head. These approaches either couple gradients across all positions through a product objective, or require on-policy drafter rollouts and extra architectural components. What is still missing is a simple, decomposable, per-position objective that (i) encodes the prefix-gated priority of Eq. (1) explicitly, (ii) concentrates supervision on the candidates the verifier actually considers, and (iii) matches the verifier's decision boundary rather than only its probabilities. It should also do all of this without touching the drafter architecture or the inference procedure. To address these limitations, we propose **Depth-Aware Top-K Margin Distillation** (DTopK), a drop-in training objective for speculative drafters with one component answering each limitation above. *First*, a **depth weight** α_d that decays with speculative depth and is shared by *every* loss term, both supervised and distilled. It weights each position by how much a gain there can contribute to the accepted length. We show that under a constant acceptance rate the marginal value of position d decays geometrically. This justifies the exponential schedule that prior work adopted empirically (Chen et al., 2026; Cheng et al., 2026) and ties its decay constant to the drafter's acceptance rate (Section 2.2). Depth weighting is thus not our novelty in itself. Our contribution is *what* is weighted: the verifier's Top-K candidates and their adjacent-rank margins. *Second*, a **Top-K distillation** term that matches the drafter's distribution to the verifier's distribution over the verifier's own Top-K candidates at every depth. This extends the Top-K idea of HASS (Zhang et al., 2025) from autoregressive drafters to the full parallel draft block. *Third*, a **margin-matching** term that aligns the logit gaps between adjacent ranked candidates. If the verifier sees a near-tie between its top two tokens, the drafter should too; if the verifier is decisive, so should the drafter be. Unlike raw logit regression (Ba & Caruana, 2014), the margin loss is invariant to per-position logit shifts. Unlike KL, it provides gradient signal for low-probability candidates and near-ties, where it matters most for rank stability (Appendix A.1). The three terms share a single depth-weighted normalization. They add no parameters, no inference-time computation, and no change to the verification rule, and the verifier's Top-K logits can be cached offline so training cost is nearly unchanged.

Contributions.

- We identify *depth-wise acceptance collapse* as a key bottleneck of parallel and diffusion drafters, show it worsens with context length, and trace it to training objectives that ei-

ther treat prefix-gated positions as interchangeable or supervise them without regard to the verifier's candidate ranking (Sections 1 and 2.2).

- We propose DTopK, which aligns the drafter with the verifier's Top-K distribution and adjacent-rank logit margins at every depth, under depth weights shared with hard-label cross-entropy. It is architecture-agnostic and requires no change at inference (Section 2).
- We analyse the objective, deriving the depth weighting from the sensitivity of expected accepted length and characterizing when margin matching and Top-K distillation agree and where their gradients differ (Section 2.2 and Appendix A.1).
- Applied to a DFlash drafter, DTopK improves the average acceptance length by 15–16% in controlled comparisons on Qwen3-8B and Gemma-4-31B, and by 28% on Qwen3-14B against a published baseline with a smaller proposal budget, with higher speedup in every case. On BABILong the relative gain grows from 28% at 8K to 55% at 1M tokens (Section 3).

2 METHOD

We first formalize speculative decoding with a parallel drafter and show why speculative depths carry unequal value (Section 2.1 and Section 2.2). We then present the three components of DTopK (Section 2.3). The full training procedure (Algorithm 1) and an analysis of the objective are given in Appendix A.

2.1 PRELIMINARIES: SPECULATIVE DECODING WITH A PARALLEL DRAFTER

Let M_t denote the target (verifier) model and M_s the drafter, sharing a vocabulary of size V. Given a context $\boldsymbol{x}_{\leq t}$, a parallel drafter produces, in a single forward pass (or a fixed number of denoising steps for diffusion drafters), a logit vector $\mathbf{z}_d^s \in \mathbb{R}^V$ for each speculative depth $d \in \{1, \ldots, \Gamma\}$, i.e., for tokens $x_{t+1}, \ldots, x_{t+\Gamma}$. Draft tokens $\hat{x}_{t+1:t+\Gamma}$ are obtained from these logits, and the verifier evaluates all of them in one parallel pass, producing logits $\mathbf{z}_d^t$ at each drafted position. Under greedy verification, draft token $\hat{x}_{t+d}$ is accepted if it equals the verifier's argmax. Under lossless speculative sampling (Leviathan et al., 2023; Chen et al., 2023) it is accepted with probability $\min\big(1, q(\hat{x}_{t+d})/p(\hat{x}_{t+d})\big)$, where q and p are the verifier and drafter distributions, so the expected acceptance is $\sum_v \min(p_v, q_v) = 1 - \mathrm{TV}(p, q)$. In both cases the verifier keeps the longest accepted prefix and appends one token of its own.

Let $\tau \in \{0, \ldots, \Gamma\}$ be the number of accepted draft tokens and let $\beta_d \triangleq \Pr(\text{depth } d \text{ accepted} \mid \tau \geq d-1)$ be the *conditional acceptance rate* at depth d. By the chain rule, $\Pr(\tau \geq d) = \prod_{j \leq d} \beta_j$, which gives the identity already used in Eq. (1):

$$\mathbb{E}[\tau] = \sum_{d=1}^{\Gamma} \Pr(\tau \geq d) = \sum_{d=1}^{\Gamma} \prod_{j=1}^{d} \beta_j. \tag{2}$$

This identity is exact; it requires no independence between positions. Each verification step emits $\mathbb{E}[\tau] + 1$ tokens on average. For a parallel drafter the drafting cost is essentially independent of Γ, so the end-to-end speedup is driven by $\mathbb{E}[\tau]$, and the goal of drafter training is to maximize it.

2.2 NOT ALL DEPTHS ARE EQUAL

Equation (2) shows how much an improvement at each depth is worth. Differentiating with respect to β_d gives

$$\frac{\partial \mathbb{E}[\tau]}{\partial \beta_d} = \frac{1}{\beta_d} \sum_{n=d}^{\Gamma} \prod_{j=1}^{n} \beta_j = \underbrace{\prod_{j=1}^{d-1} \beta_j}_{\Pr(\text{reach } d)} \cdot \underbrace{\Big(1 + \sum_{n=d+1}^{\Gamma} \prod_{j=d+1}^{n} \beta_j\Big)}_{\text{value of the suffix it unlocks}}. \tag{3}$$

The first factor is the probability that position d is ever verified. The second is the expected number of tokens that depth d unlocks, counting itself. Both factors shrink as d grows.

Proposition 1 (Geometric decay of depth value)**.** *If $\beta_j = \beta \in (0,1)$ for all j, then*

$$\frac{\partial\, \mathbb{E}[\tau]}{\partial \beta_d} = \beta^{\,d-1}\, \frac{1-\beta^{\,\Gamma-d+1}}{1-\beta},$$

which is strictly decreasing in d and, away from the end of the block, is proportional to $\exp\big(-(d-1)/\gamma^\star\big)$ with $\gamma^\star = -1/\ln\beta$.

Proof. Substitute $\beta_j = \beta$ into Eq. (3): the first factor is β^{d-1} and the second is the geometric sum $\sum_{m=0}^{\Gamma-d} \beta^m = (1-\beta^{\Gamma-d+1})/(1-\beta)$. Both factors are decreasing in d, and $\beta^{d-1} = \exp\big((d-1)\ln\beta\big)$. □

Proposition 1 motivates our depth weights. When drafter capacity is limited, the loss at depth d should be weighted in proportion to the value of that depth, which decays roughly exponentially. We therefore use

$$\alpha_d = \exp\Big(-\frac{d-1}{\gamma_{\text{depth}}}\Big), \qquad d = 1, \ldots, \Gamma, \tag{4}$$

where γ_{depth} controls how quickly importance decays. The proposition also gives this hyperparameter a direct interpretation: $\gamma_{\text{depth}} \approx -1/\ln\bar{\beta}$ for a typical acceptance rate $\bar{\beta}$. For example, $\bar{\beta} = 0.9$ gives $\gamma^\star \approx 9.5$ and $\bar{\beta} = 0.8$ gives $\gamma^\star \approx 4.5$, and $\gamma_{\text{depth}} \to \infty$ recovers uniform weighting. The same exponential form was adopted empirically for the cross-entropy of DFlash drafters, with $\gamma = 7$ for $\Gamma = 16$ (Chen et al., 2026), and for all loss terms of DSpark (Cheng et al., 2026). Through Proposition 1, this choice corresponds to a typical acceptance rate of $\bar{\beta} = e^{-1/7} \approx 0.87$, and the proposition gives a way to retune γ_{depth} when the drafter, the context length, or the domain changes the acceptance profile. We treat Eq. (4) as a soft prior rather than an exact prescription. It never sets a depth's weight to zero, so deep positions still receive gradient. Alternative schedules, such as weights proportional to the empirical $\Pr(\text{reach } d)$ of a baseline drafter, fit the same framework.

2.3 Depth-aware Top-K margin distillation

We now define the objective for a mini-batch of B training samples.

Setup. For each sample $b \in \{1, \ldots, B\}$ and speculative depth $d \in \{1, \ldots, \Gamma\}$ (e.g., $\Gamma = 16$), the drafter produces a logit vector $\mathbf{z}^s_{b,d} \in \mathbb{R}^V$ and the verifier produces the corresponding teacher logit vector $\mathbf{z}^t_{b,d} \in \mathbb{R}^V$. The teacher logits are the verifier's next-token logits at position $t+d$, computed with a single teacher-forced pass over the training continuation. If the continuation is generated by the verifier itself, this is exactly the context in which depth d is verified when all earlier drafts are accepted. The hard next-token label is $y_{b,d}$. A binary mask $m_{b,d} \in \{0,1\}$ marks positions that are valid for training. It excludes, for example, padding, positions beyond the end of a sequence, or, for masked-diffusion drafters, positions that are not predicted in the current training step.

Depth weighting. Each (b,d) term is weighted by α_d from Eq. (4). All three losses share the normalization

$$Z = \sum_{b=1}^{B} \sum_{d=1}^{\Gamma} m_{b,d}\, \alpha_d, \tag{5}$$

so the loss scale does not depend on γ_{depth} or on the number of valid positions, and the λ coefficients below keep the same meaning across schedules.

Depth-weighted cross-entropy. We retain the standard hard-label objective, reweighted by depth:

$$\mathcal{L}_{\text{CE}} = -\frac{1}{Z} \sum_{b=1}^{B} \sum_{d=1}^{\Gamma} m_{b,d}\, \alpha_d\, \log \operatorname{softmax}\big(\mathbf{z}^s_{b,d}\big)_{y_{b,d}}. \tag{6}$$

This term acts over the full vocabulary. It is the only term that penalizes tokens *outside* the verifier's Top-K set, so it prevents the drafter from placing mass on candidates the verifier would never accept.

Top-K distribution distillation. Acceptance is decided almost entirely among a handful of tokens the verifier considers plausible. We therefore extract the verifier's Top-K candidates at every (b, d):

$$\mathcal{I}^t_{b,d} = \{i^t_{b,d,1}, i^t_{b,d,2}, \ldots, i^t_{b,d,K}\} = \mathrm{TopK}(\mathbf{z}^t_{b,d}), \tag{7}$$

ordered so that $i^t_{b,d,1}$ is the verifier's top-1 token, $i^t_{b,d,2}$ its top-2 token, and so on. The teacher and drafter distributions *restricted to this common index set* are

$$q^t_{b,d,k} = \frac{\exp(z^t_{b,d,i^t_{b,d,k}}/\tau_{\mathrm{KD}})}{\sum_{\ell=1}^{K}\exp(z^t_{b,d,i^t_{b,d,\ell}}/\tau_{\mathrm{KD}})}, \qquad p^s_{b,d,k} = \frac{\exp(z^s_{b,d,i^t_{b,d,k}}/\tau_{\mathrm{KD}})}{\sum_{\ell=1}^{K}\exp(z^s_{b,d,i^t_{b,d,\ell}}/\tau_{\mathrm{KD}})}, \tag{8}$$

where τ_{KD} is the distillation temperature. The drafter's probabilities are computed on the *teacher's* indices, not its own. The drafter is therefore asked to rank and weigh exactly the alternatives the verifier prefers at each depth, whatever its own current top candidates are. The Top-K distillation loss is

$$\mathcal{L}_{\mathrm{TopK}} = -\frac{\tau^2_{\mathrm{KD}}}{Z}\sum_{b=1}^{B}\sum_{d=1}^{\Gamma} m_{b,d}\,\alpha_d \sum_{k=1}^{K} q^t_{b,d,k}\,\log p^s_{b,d,k}, \tag{9}$$

where the factor τ^2_{KD} keeps gradient magnitudes comparable across temperatures (Hinton et al., 2015). Compared with hard labels, Eq. (9) exposes the verifier's full preference over plausible continuations. This matters most at deep positions and long contexts, where the realized token is only one of several near-equally likely options. Compared with full-vocabulary KD, it spends no capacity on the tail.

Adjacent-rank margin matching. Distributions can match on average while disagreeing on the decisions that determine acceptance. We therefore also match the *gaps* between consecutively ranked candidates. The verifier's adjacent-rank logit margins are

$$\Delta^t_{b,d,k} = z^t_{b,d,i^t_{b,d,k}} - z^t_{b,d,i^t_{b,d,k+1}}, \qquad k = 1, \ldots, K-1, \tag{10}$$

which are non-negative by construction. The drafter's margins are measured on the same teacher indices:

$$\Delta^s_{b,d,k} = z^s_{b,d,i^t_{b,d,k}} - z^s_{b,d,i^t_{b,d,k+1}}, \qquad k = 1, \ldots, K-1. \tag{11}$$

A negative $\Delta^s_{b,d,k}$ means the drafter inverts the verifier's ranking at rank k. The margin-matching loss is

$$\mathcal{L}_{\mathrm{Margin}} = \frac{1}{Z}\sum_{b=1}^{B}\sum_{d=1}^{\Gamma} m_{b,d}\,\alpha_d\,\frac{1}{K-1}\sum_{k=1}^{K-1}\left(\Delta^s_{b,d,k} - \Delta^t_{b,d,k}\right)^2. \tag{12}$$

This term asks the drafter to identify the verifier's Top-K alternatives *and* to reproduce how far apart they are in logit space. If the verifier's top-1 and top-2 tokens are close, as at a genuine branching point of the continuation, the drafter should keep them close rather than overconfidently commit to one. If the verifier strongly separates top-1 from top-2, the drafter should learn the same decisive separation, so that its greedy choice is stable.

Full objective. The complete DTopK loss is

$$\mathcal{L}_{\mathrm{DTopK}} = \mathcal{L}_{\mathrm{CE}} + \lambda_{\mathrm{TopK}}\,\mathcal{L}_{\mathrm{TopK}} + \lambda_{\mathrm{Margin}}\,\mathcal{L}_{\mathrm{Margin}}, \tag{13}$$

where λ_{TopK} and $\lambda_{\mathrm{Margin}}$ control the strengths of Top-K distillation and margin matching. Two familiar objectives are special cases. Setting $\lambda_{\mathrm{TopK}} = \lambda_{\mathrm{Margin}} = 0$ recovers the depth-weighted cross-entropy of DFlash (Chen et al., 2026), and additionally letting $\gamma_{\mathrm{depth}} \to \infty$ recovers the uniform cross-entropy used to train most parallel drafters. Comparing a DFlash drafter trained with DTopK to one trained with its original objective, under the same configuration, therefore isolates the effect of the objective (Section 3). Algorithm 1 in Appendix A summarizes one training step.

3 Experiments

Our experiments answer three questions. **(RQ1)** Does DTopK improve acceptance and end-to-end speedup over state-of-the-art drafters under standard evaluation (Section 3.2)? **(RQ2)** Do the gains grow with context length, where parallel drafters collapse (Section 3.3)? **(RQ3)** Where in the draft block do the gains come from, and how do they relate to the prefix-gated analysis of Section 2.2 (Section 3.4)?

3.1 EXPERIMENTAL SETUP

Models and baselines. We use Qwen3-8B as the primary verifier, matching Figure 1a, and additionally evaluate Qwen3-14B (Team, 2025) and Gemma-4-31B (Team, 2026) to test generality across scales and families. Verifiers are frozen. We compare against five drafters that cover the main drafting paradigms: the autoregressive feature-level drafter **EAGLE-3** (Li et al., 2025b); **MTP**, parallel multi-token prediction heads (Gloeckle et al., 2024); the block-diffusion drafter **DFlash** (Chen et al., 2026), trained with depth-weighted cross-entropy; the semi-autoregressive **DSpark** (Cheng et al., 2026); and **Orthrus** (Van Nguyen et al., 2026), a dual-view diffusion head that shares the verifier's KV cache.

DTopK is a training objective, not an architecture. We therefore apply it to the *DFlash* drafter with block size $\Gamma = 16$, keeping the DFlash architecture and inference procedure unchanged. On Qwen3-8B and Gemma-4-31B, the DFlash baseline uses the same $\Gamma = 16$ configuration, so the difference between the two rows reflects the training objective. On Qwen3-14B, the EAGLE-3, DFlash, and DSpark rows are taken from the published results of Cheng et al. (2026). That work uses a smaller proposal budget (block size 7 for DFlash and DSpark, and a training-time-test horizon of 7 for EAGLE-3) and sampling temperature 1.0, so the Qwen3-14B comparison reflects both the objective and the larger proposal budget of our $\Gamma = 16$ drafter, and we do not interpret it as objective-only. The Qwen3-8B EAGLE-3 (draft-tree budget 60) and DFlash rows are taken from Chen et al. (2026), and the Qwen3-8B DSpark row from Cheng et al. (2026), under the same settings. For standard-context evaluation we follow prior work (Li et al., 2025b; Chen et al., 2026) and cover three domains: mathematical reasoning (GSM8K, MATH-500), code generation (HumanEval, MBPP), and multi-turn conversation (MT-Bench). For long-context evaluation we use BABILong (Kuratov et al., 2024) at context lengths of 8K, 64K, 128K, 256K, and 1M tokens, the same grid as Figure 1a. Where a length exceeds the verifier's native context window, we extend it with RoPE-based context scaling.

Metrics and implementation. (1) **Acceptance length** $\bar{L}$ is the mean number of tokens committed per verification cycle, i.e., $\mathbb{E}[\tau] + 1$ including the bonus token, as in Figure 1a. It is hardware-independent. (2) **Speedup** is wall-clock throughput relative to autoregressive decoding with the same verifier, measured within each setting. For rows taken from published results, speedups come from the respective paper's hardware and runtime, so cross-row speedup comparisons are indicative rather than strictly controlled. (3) **Per-depth acceptance** β_d (Section 2.1) diagnoses where gains arise. Speculative decoding is lossless, so output quality is identical across methods and is not reported. All our experiments run on $4\times$ NVIDIA H100 GPUs.

3.2 MAIN RESULTS

Table 1 compares all six drafters at standard context lengths. The key comparison is DTopK against DFlash. On Qwen3-8B and Gemma-4-31B the two share architecture, block size, and inference procedure, so their gap isolates the training objective. On Qwen3-14B it also includes the difference in proposal budget (Section 3.1).

Consistent gains over the base drafter. DTopK improves DFlash on every model and every benchmark. The average acceptance length rises from 6.22 to 7.20 on Qwen3-8B ($+15.8\%$), from 4.48 to 5.73 on Qwen3-14B ($+27.9\%$), and from 6.10 to 7.04 on Gemma-4-31B ($+15.4\%$). In the two controlled comparisons (Qwen3-8B and Gemma-4-31B), the gain is consistent across domains. On Qwen3-8B it ranges from $+15.1\%$ to $+16.8\%$. The larger gain on Qwen3-14B also reflects the smaller proposal budget of the published baseline, so we do not attribute it to the objective alone. Because inference is unchanged, the acceptance gains translate into average speedups of $5.09\times$, $4.43\times$, and $5.37\times$, which are 7.2%, 29.5%, and 11.4% higher than DFlash, respectively. Each cycle also pays a fixed drafting and verification cost, so relative speedup gains can be smaller than relative acceptance gains. Multi-turn conversation (MT-Bench) is the hardest domain for every drafter, yet DTopK still improves acceptance length there by 15–16% in the controlled comparisons.

Highest speedup among all drafters. DTopK achieves the highest speedup on all 15 model–benchmark pairs in Table 1. Because some baseline rows come from published results with different settings, we view this ranking as indicative. Orthrus often accepts longer drafts, for example 11.70 tokens per cycle on MATH-500 with Qwen3-8B, but this does not carry over to wall-clock speed.

Table 1: **Main results at standard context length.** Acceptance length $\bar{L}$ (committed tokens per cycle, ↑) and wall-clock speedup over autoregressive decoding (Spd., ↑). DTopK uses the DFlash architecture and training setup and changes only the training objective. Mean and standard deviation are taken over the five benchmarks. Best result per column and model in **bold**. †Reported by Chen et al. (2026) ($\Gamma = 16$, greedy decoding). ‡Reported by Cheng et al. (2026) (block size 7; EAGLE-3 training-time-test horizon 7; temperature 1.0). These rows are not controlled against our $\Gamma = 16$ drafter.

	GSM8K		MATH-500		HumanEval		MBPP		MT-Bench		**Mean ± Std.**	
Method	$\bar{L}$	Spd.	$\bar{L}$	Spd.	$\bar{L}$	Spd.	$\bar{L}$	Spd.	$\bar{L}$	Spd.	$\bar{L}$	Spd.
Target: Qwen3-8B												
EAGLE-3 (tree=60)†	3.71	2.23	3.49	2.05	3.65	2.17	3.25	1.93	3.26	1.90	3.47 ± 0.21	2.06 ± 0.14
MTP	3.56	2.14	3.53	2.07	3.33	1.98	3.22	1.91	2.57	1.50	3.24 ± 0.40	1.92 ± 0.25
DSpark‡	6.17	4.86	5.78	4.47	5.52	4.37	5.16	4.03	3.72	2.41	5.27 ± 0.94	4.03 ± 0.95
Orthrus	**9.60**	5.22	**11.70**	5.95	6.28	3.68	**6.90**	3.90	**4.92**	2.31	$\mathbf{7.88 \pm 2.73}$	4.21 ± 1.42
DFlash†	6.54	5.15	7.87	6.08	6.50	5.14	5.95	4.65	4.24	2.75	6.22 ± 1.31	4.75 ± 1.23
DTopK (ours)	7.60	**5.47**	9.09	**6.55**	**7.59**	**5.53**	6.85	**4.95**	4.88	**2.95**	7.20 ± 1.53	$\mathbf{5.09 \pm 1.33}$
Target: Qwen3-14B												
EAGLE-3‡	5.24	3.15	4.60	2.70	4.14	2.46	3.81	2.26	2.62	1.53	4.08 ± 0.98	2.42 ± 0.60
MTP	5.03	3.02	4.65	2.73	3.78	2.24	3.77	2.24	2.07	1.21	3.86 ± 1.14	2.29 ± 0.69
DSpark‡	6.21	5.58	5.74	4.08	**5.43**	3.68	5.26	4.04	3.70	2.68	5.27 ± 0.95	4.01 ± 1.04
Orthrus	**7.94**	4.32	**7.20**	3.66	4.43	2.60	5.15	2.91	3.60	1.69	5.66 ± 1.84	3.04 ± 1.01
DFlash‡	5.41	4.26	4.84	3.74	4.59	3.63	4.44	3.47	3.10	2.01	4.48 ± 0.85	3.42 ± 0.84
DTopK (ours)	6.68	**5.95**	6.59	**4.32**	5.33	**4.21**	**5.98**	**4.65**	**4.07**	**3.02**	$\mathbf{5.73 \pm 1.07}$	$\mathbf{4.43 \pm 1.05}$
Target: Gemma-4-31B												
EAGLE-3	3.18	1.79	4.24	2.12	3.97	1.86	3.03	1.84	3.33	1.73	3.55 ± 0.53	1.87 ± 0.15
MTP	4.40	3.30	4.41	3.37	4.43	3.67	3.95	2.79	3.24	2.34	4.09 ± 0.51	3.09 ± 0.53
DSpark	5.07	4.62	5.30	4.20	5.05	3.98	5.17	4.34	3.06	2.29	4.73 ± 0.94	3.89 ± 0.92
Orthrus	**11.05**	5.37	**10.77**	5.68	**7.73**	4.01	7.11	3.69	**4.91**	2.52	$\mathbf{8.31 \pm 2.59}$	4.25 ± 1.29
DFlash	7.53	5.30	6.59	5.80	6.00	5.60	6.13	4.40	4.23	3.00	6.10 ± 1.20	4.82 ± 1.15
DTopK (ours)	8.78	**5.67**	7.19	**6.13**	7.21	**6.08**	**7.12**	**5.79**	**4.91**	**3.20**	7.04 ± 1.38	$\mathbf{5.37 \pm 1.23}$

Its average speedup on Qwen3-8B is $4.21\times$ versus $5.09\times$ for DTopK, indicating a higher per-cycle cost that offsets its longer accepted drafts. On Qwen3-14B, DTopK also has the longest average acceptance length (5.73 versus 5.66 for Orthrus). The autoregressive EAGLE-3 drafter and the multi-token prediction heads of MTP trail the block-parallel and diffusion drafters, with average speedups of at most $3.09\times$.

3.3 Scaling to long contexts

Table 2 tests our hypothesis (Section 2.3) that the gains should *grow* with context length, where a single hard label carries less information and the Top-K and margin signals carry more. The results confirm this. All drafters lose acceptance as the context grows, but DTopK degrades most gracefully. Its relative gain over DFlash rises monotonically, from $+28.0\%$ at 8K to $+36.1\%$ at 128K and $+54.5\%$ at 1M tokens. The effect is starker in accepted *draft* tokens per cycle, $\bar{L} - 1$. At 1M tokens DFlash accepts only 0.10 draft tokens per cycle, effectively reverting to autoregressive decoding, whereas DTopK still accepts 0.70, seven times more. Orthrus is the strongest drafter up to 128K. Beyond that point, however, DTopK yields the longest acceptance of all six drafters, 1.92 at 256K and 1.70 at 1M, compared with 1.48 and 1.45 for Orthrus.

3.4 Where do the gains come from? Per-depth acceptance

Table 3 decomposes the acceptance length into the conditional acceptance rate β_d at selected depths, for DFlash and DTopK at a short (8K) and a long (128K) context. We make three observations.

First, DTopK raises acceptance at every depth. Down-weighting deep positions does not sacrifice them. The absolute gain in β_d even grows with depth, from $+0.08$ at d=1 to $+0.15$ at d=16

Table 2: **Long-context results on BABILong** (Kuratov et al., 2024) with a Qwen3-8B verifier. Acceptance length $\bar{L}$ (committed tokens per verification cycle, ↑) as the context length increases from 8K to 1M tokens. The last row reports the relative improvement of DTopK over DFlash, which uses the same drafter architecture.

Method	8K	64K	128K	256K	1M
EAGLE-3	2.80	1.95	1.60	1.35	1.12
MTP	2.60	1.82	1.50	1.28	1.08
DSpark	3.50	2.45	2.00	1.62	1.30
Orthrus	**3.80**	**2.70**	**2.20**	1.48	1.45
DFlash	2.50	1.80	1.55	1.30	1.10
DTopK (ours)	3.20	2.39	2.11	**1.92**	**1.70**
Gain vs. DFlash	+28.0%	+32.8%	+36.1%	+47.7%	+54.5%

Table 3: **Per-depth conditional acceptance** β_d (↑) of DFlash and DTopK (Qwen3-8B verifier, $\Gamma = 16$) at a short and a long context length. Δ is the absolute difference. The last column is the resulting acceptance length $\bar{L}$.

Context	Method	d=1	d=2	d=3	d=4	d=8	d=12	d=16	$\bar{L}$
8K	DFlash	0.65	0.62	0.58	0.54	0.42	0.34	0.28	2.50
	DTopK (ours)	0.73	0.71	0.69	0.66	0.56	0.48	0.43	3.20
	Δ	+0.08	+0.09	+0.11	+0.12	+0.14	+0.14	+0.15	+0.70
128K	DFlash	0.38	0.32	0.27	0.22	0.12	0.07	0.05	1.55
	DTopK (ours)	0.57	0.53	0.49	0.45	0.36	0.31	0.29	2.11
	Δ	+0.19	+0.21	+0.22	+0.23	+0.24	+0.24	+0.24	+0.56

at 8K. This is consistent with the role of the Top-K and margin terms, which provide the most information at deep positions, where the realized next token is least predictable and a hard label is least informative (Section 2.3).

Second, gains compound through the prefix gate. By Eq. (2), the probability that the first d drafts are all accepted is $\prod_{j \le d} \beta_j$. At 8K, the probability of accepting at least four draft tokens rises from 0.13 to 0.24 ($1.9\times$). At 128K it rises from 0.007 to 0.067 ($9.2\times$). As predicted by Eq. (3), most of the gain in acceptance length is realized at the front of the block. The first four depths account for 0.43 of the $+0.70$ tokens gained at 8K, and for 0.55 of the $+0.56$ tokens at 128K.

Third, the gains are larger at long context. At 128K, $\Delta\beta_d$ is 1.6–$2.4\times$ larger than at 8K at every depth ($+0.19$ to $+0.24$), mirroring the growing relative gains in Table 2.

4 Conclusion

The value of a speculative block is decided by its accepted prefix, which shrinks rapidly with depth and context length. Depth-Aware Top-K Margin Distillation trains every depth against the verifier's Top-K candidates and their adjacent-rank logit margins, weighted by how much each depth can contribute to the accepted length. It leaves the drafter architecture, the verification rule, and the inference cost unchanged. Applied to a block-diffusion drafter, it improved the average acceptance length by 15–16% in controlled comparisons on Qwen3-8B and Gemma-4-31B. It also improved acceptance length by 28% on Qwen3-14B against a published baseline with a smaller proposal budget, and it raised end-to-end speedup in every setting. Its advantage grew with context length, reaching a 55% gain in acceptance length at 1M tokens. A per-depth analysis showed that these gains appear at every depth and compound through the prefix gate.

Limitations and future work. We applied DTopK to one drafter family, and several baseline results come from published work with different proposal budgets, temperatures, and hardware. Other drafter families, a fully controlled re-evaluation under a common budget, and broader long-context studies are natural next steps.

AI USE STATEMENT

Generative AI tools were used to assist with literature search, drafting and editing the manuscript text, and LaTeX formatting. All AI-assisted content, including every citation, was reviewed and verified by the authors. The authors take responsibility for the content of this paper.

REPRODUCIBILITY STATEMENT

The complete training objective is specified in Section 2.3 (Eqs. (4)–(13)), and one training step is given as pseudocode in Algorithm 1 (Appendix A). DTopK changes only the training loss of the DFlash drafter (Chen et al., 2026), so the architecture, data pipeline, and inference procedure follow that work. Target models, benchmarks, context lengths, and metrics are described in Section 3.1.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations (ICLR)*, 2024.

Zihao An, Huajun Bai, Ziqiong Liu, Dong Li, and Emad Barsoum. PARD: Accelerating LLM inference with low-cost PARallel draft model adaptation. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2504.18583.

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

Marianne Arriola, Aaron Gokaslan, Justin T. Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *International Conference on Learning Representations (ICLR)*, 2025.

Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. Medusa: Simple LLM inference acceleration framework with multiple decoding heads. In *International Conference on Machine Learning (ICML)*, 2024.

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

Jian Chen, Yesheng Liang, and Zhijian Liu. DFlash: Block diffusion for flash speculative decoding. *arXiv preprint arXiv:2602.06036*, 2026.

Xin Cheng, Xingkai Yu, Chenze Shao, Jiashi Li, Yunfan Xiong, Yi Qian, Jiaqi Zhu, Shirong Ma, Xiaokang Zhang, Jiasheng Ye, et al. Dspark: Confidence-scheduled speculative decoding with semi-autoregressive generation. *arXiv preprint arXiv:2607.05147*, 2026.

Jacob K. Christopher, Brian R. Bartoldson, Tal Ben-Nun, Michael Cardei, Bhavya Kailkhura, and Ferdinando Fioretto. Speculative diffusion decoding: Accelerating language generation through diffusion. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics (NAACL)*, pp. 12042–12059, 2025.

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. Better & faster large language models via multi-token prediction. In *International Conference on Machine Learning (ICML)*, 2024.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. *Advances in Neural Information Processing Systems*, 37:106519–106554, 2024.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning (ICML)*, 2023.

Guanghao Li, Zhihui Fu, Min Fang, Qibin Zhao, Ming Tang, Chun Yuan, and Jun Wang. DiffuSpec: Unlocking diffusion language models for speculative decoding. *arXiv preprint arXiv:2510.02358*, 2025a.

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

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-2: Faster inference of language models with dynamic draft trees. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024b.

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

Xiaoxuan Liu, Lanxiang Hu, Peter Bailis, Alvin Cheung, Zhijie Deng, Ion Stoica, and Hao Zhang. Online speculative decoding. In *International Conference on Machine Learning (ICML)*, 2024.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Anselm Levskaya, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. In *Proceedings of Machine Learning and Systems (MLSys)*, 2023.

Yu-Yang Qian, Hao-Cong Wu, Chen Chen, Jiacheng Sun, Zhenhua Dong, Peng Zhao, and Zhi-Hua Zhou. AdaFlash: Adaptive speculative decoding via on-policy distilled diffusion drafters. *arXiv preprint arXiv:2607.19223*, 2026.

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

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Jameson Sandler, Jacob K. Christopher, Thomas Hartvigsen, and Ferdinando Fioretto. SpecDiff-2: Scaling diffusion drafter alignment for faster speculative decoding. In *Proceedings of Machine Learning and Systems (MLSys)*, 2026. arXiv:2511.00606.

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

Hanshi Sun, Zhuoming Chen, Xinyu Yang, Yuandong Tian, and Beidi Chen. TriForce: Lossless acceleration of long sequence generation with hierarchical speculative decoding. In *Conference on Language Modeling (COLM)*, 2024.

Algorithm 1 One training step of Depth-Aware Top-K Margin Distillation (DTopK)

Require: Drafter M_s with parameters θ; frozen verifier M_t; mini-batch $\{(\boldsymbol{x}_b, y_{b,1:\Gamma}, m_{b,1:\Gamma})\}_{b=1}^{B}$; draft length Γ; Top-K size K; temperature τ_{KD}; depth decay γ_{depth}; weights $\lambda_{\mathrm{TopK}}, \lambda_{\mathrm{Margin}}$; learning rate η

1: $\alpha_d \leftarrow \exp\big(-(d-1)/\gamma_{\mathrm{depth}}\big)$ for $d = 1, \dots, \Gamma$ ▷ depth weights, Eq. (4)
2: $\{\mathbf{z}^t_{b,d}\} \leftarrow M_t(\boldsymbol{x}_b)$ with no gradient ▷ teacher-forced verifier pass, or load cached Top-K
3: $\{\mathbf{z}^s_{b,d}\} \leftarrow M_s(\boldsymbol{x}_b; \theta)$ ▷ one parallel drafter pass over all Γ depths
4: $Z \leftarrow \sum_{b,d} m_{b,d}\, \alpha_d$
5: **for** all (b, d) with $m_{b,d} = 1$ **in parallel do**
6: $\quad \mathcal{I}^t_{b,d} = (i_1, \dots, i_K) \leftarrow \mathrm{TopK}(\mathbf{z}^t_{b,d})$ ▷ verifier's ranked candidates
7: $\quad \mathbf{q} \leftarrow \mathrm{softmax}\big(\mathbf{z}^t_{b,d}[\mathcal{I}^t_{b,d}]/\tau_{\mathrm{KD}}\big),\ \mathbf{p} \leftarrow \mathrm{softmax}\big(\mathbf{z}^s_{b,d}[\mathcal{I}^t_{b,d}]/\tau_{\mathrm{KD}}\big)$ ▷ Eq. (8)
8: $\quad \Delta^t_k \leftarrow z^t_{b,d,i_k} - z^t_{b,d,i_{k+1}},\ \Delta^s_k \leftarrow z^s_{b,d,i_k} - z^s_{b,d,i_{k+1}}$ for $k = 1, \dots, K-1$
9: $\quad \ell^{\mathrm{CE}}_{b,d} \leftarrow -\log \mathrm{softmax}(\mathbf{z}^s_{b,d})_{y_{b,d}}$
10: $\quad \ell^{\mathrm{TopK}}_{b,d} \leftarrow -\tau^2_{\mathrm{KD}} \sum_k q_k \log p_k$
11: $\quad \ell^{\mathrm{Margin}}_{b,d} \leftarrow \frac{1}{K-1} \sum_k \big(\Delta^s_k - \Delta^t_k\big)^2$
12: $\mathcal{L}_{\mathrm{DTopK}} \leftarrow \frac{1}{Z} \sum_{b,d} m_{b,d}\, \alpha_d \big(\ell^{\mathrm{CE}}_{b,d} + \lambda_{\mathrm{TopK}} \ell^{\mathrm{TopK}}_{b,d} + \lambda_{\mathrm{Margin}} \ell^{\mathrm{Margin}}_{b,d}\big)$ ▷ Eq. (13)
13: $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_{\mathrm{DTopK}}$

Gemma Team. Gemma 4 technical report, 2026. URL https://arxiv.org/abs/2607.02770.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Chien Van Nguyen, Chaitra Hegde, Van Cuong Pham, Ryan A Rossi, Franck Dernoncourt, and Thien Huu Nguyen. Orthrus: Memory-efficient parallel token generation via dual-view diffusion. *arXiv preprint arXiv:2605.12825*, 2026.

Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. In *Findings of the Association for Computational Linguistics (ACL)*, 2024.

Lefan Zhang, Xiaodan Wang, Yanhua Huang, and Ruiwen Xu. Learning harmonized representations for speculative sampling. In *International Conference on Learning Representations (ICLR)*, 2025.

Yongchao Zhou, Kaifeng Lyu, Ankit Singh Rawat, Aditya Krishna Menon, Afshin Rostamizadeh, Sanjiv Kumar, Jean-François Kagy, and Rishabh Agarwal. DistillSpec: Improving speculative decoding via knowledge distillation. In *International Conference on Learning Representations (ICLR)*, 2024.

A Training algorithm and analysis of the objective

Walking through Algorithm 1. Line 1 computes the depth weights once. They depend only on d, so they broadcast over the batch at no cost. Line 2 obtains the verifier's logits for all Γ depths in a single teacher-forced pass. Only the Top-K values and indices are ever used, so they can be *precomputed and cached offline*. This reduces teacher storage from $B\Gamma V$ to $B\Gamma K$ numbers and removes the verifier from the training loop entirely. Line 3 runs the drafter exactly as at inference time, in one parallel pass (or its usual denoising schedule), which is why DTopK requires no architectural change. Lines 5–12 are fully vectorized over (b, d). Line 6 selects the verifier's ranked candidates. Line 7 gathers both models' logits at those *same* indices, so the drafter is always evaluated on the verifier's preferred alternatives. Line 8 forms the adjacent-rank margins, and Lines 9–11 compute the three per-position losses. Line 13 combines them with the shared depth weight α_d and normalizer Z. The extra work over standard training is one TopK and two gathers of size K per position, negligible next to the drafter's forward pass. At inference nothing changes. The trained drafter is plugged into the unchanged drafting and verification procedure, so acceptance gains can translate into higher speedup without adding inference overhead.

A.1 Analysis of the objective

Margins pin down the Top-K distribution at every temperature. Margin matching is invariant to adding a constant $c_{b,d}$ to all drafter logits at a position, since such shifts cancel in every difference. This invariance matters because a softmax is itself shift-invariant, and it is where margin matching differs from raw logit regression (Ba & Caruana, 2014). Nevertheless, margins carry all the information needed for verification within the Top-K set.

Proposition 2 (Margin sufficiency)**.** *If $\Delta^s_{b,d,k} = \Delta^t_{b,d,k}$ for all $k = 1, \dots, K-1$, then $p^s_{b,d,k} = q^t_{b,d,k}$ for all k and* every *temperature $\tau_{\mathrm{KD}} > 0$, and the drafter reproduces the verifier's ranking of $\mathcal{I}^t_{b,d}$.*

Proof. Telescoping the equal margins gives $z^s_{b,d,i_k} = z^t_{b,d,i_k} + c_{b,d}$ for all k, with $c_{b,d} = z^s_{b,d,i_1} - z^t_{b,d,i_1}$. The restricted softmax in Eq. (8) is invariant to the shift $c_{b,d}$ for any temperature, so $p^s = q^t$. Since $\Delta^s_{b,d,k} = \Delta^t_{b,d,k} \geq 0$, the ordering is preserved. □

Same minimizers, different gradients. Conversely, $p^s = q^t$ at a fixed temperature also implies equal margins, so $\mathcal{L}_{\mathrm{TopK}}$ and $\mathcal{L}_{\mathrm{Margin}}$ share their minimizers at each position. They differ in *how* they get there. The gradient of Eq. (9) with respect to a drafter logit is $\tau_{\mathrm{KD}}\,(p^s_{b,d,k} - q^t_{b,d,k})$, which is scaled by probability mass. It is dominated by the top one or two candidates and vanishes for low-probability alternatives, and it saturates once the drafter is confidently correct. The margin gradient is linear in the gap error at every rank, independent of probability mass. It keeps correcting a drafter that is over-confident where the verifier sees a near-tie, and it keeps shaping ranks $3, \dots, K$ that KL largely ignores. In practice the two terms are complementary: KL aligns the high-mass region that dominates the stochastic acceptance $1 - \mathrm{TV}(p, q)$, while margin matching stabilizes the ranking that decides greedy acceptance.

Why keep cross-entropy. Both Top-K terms are blind to tokens outside $\mathcal{I}^t_{b,d}$. A drafter could satisfy them perfectly while placing a large logit on an out-of-set token, which would then win its argmax and be rejected by the verifier. The full-vocabulary cross-entropy of Eq. (6) rules this out, anchoring the drafter's global distribution while the two Top-K terms refine its local decision structure.

Scope. Nothing in Eqs. (6)–(13) depends on how the drafter produces $\mathbf{z}^s_{b,d}$. The objective therefore applies to autoregressive, multi-head, block-parallel, and diffusion drafters alike. We expect the largest gains for parallel and diffusion drafters. They predict all depths without conditioning on their own earlier drafts, which makes their acceptance decay steepest and puts the most value on depth-aware prioritization.