FAIRSPEC: ROBUST EXPERT SELECTION FOR EFFICIENT MOE SPECULATIVE DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Speculative decoding accelerates large language-model (LLM) inference by using a lightweight drafter model to propose several tokens and verifying them simultaneously in one target-model pass. For mixture-of-experts (MoE) models, however, the speculative tokens may route to different experts, forcing each MoE layer to read more distinct expert parameters from GPU high-bandwidth memory and potentially making verification slower than autoregressive decoding. Existing methods cap this cost by retaining the experts with the greatest aggregate routing demand from speculative tokens, but aggregate selection can neglect individual tokens, shortening the accepted draft prefix and/or degrading model inference quality. We therefore propose FAIRSPEC, a verifier-side expert routing method that allocates an expert budget that protects each speculated token's highest-ranked experts by exploiting expert overlap across tokens. FAIRSPEC requires no model retraining or changes to the downstream MoE operator and is implemented as a fused, CUDA-graph-compatible routing kernel. We evaluate FAIRSPEC against state-of-the-art speculative decoding methods that perform expert-subset selection. Across 4 benchmarks and 3 sparse MoE architectures, at comparable throughput, FAIRSPEC achieves 10–50pp higher success rates under low and moderate expert budgets, with larger gains at lower budgets.

1 INTRODUCTION

Speculative Decoding (SD) accelerates autoregressive large language-model (LLM) inference by asking a lightweight *drafter* model to propose a short continuation of tokens and then verifying all proposed tokens using some *target* model (i.e. the verifier) of interest (Leviathan et al., 2023; Chen et al., 2023). The verifier runs a forward pass for all draft tokens in parallel, and then accepts draft tokens from left to right until the first disagreement, after which generation resumes from the corrected prefix. By replacing several sequential target-model calls with one batched verification pass, speculative decoding amortizes the target's compute cost across multiple generated tokens and can substantially improve inference throughput without changing the target model's output distribution using rejection sampling.

However, this argument assumes that verification cost grows modestly with the number of tokens verified, which need not hold for sparse mixture-of-experts (MoE) models. Although each token activates only k of N experts, verifying $T = \gamma + 1$ tokens together requires loading the union of their selected experts. In Qwen3-30B-A3B, (Team, 2025) each token routes to 8 of 128 experts; with $\gamma = 7$, this union grows from 8 experts during autoregressive decoding to approximately 20 per layer during verification. Consequently, the MoE layers must fetch roughly $2.5\times$ more expert weights from GPU high-bandwidth memory (HBM). Because these weights exceed the capacity of on-chip caches and low-batch decoding is bandwidth-bound, verification can therefore become limited by expert-weight movement rather than computation, consistent with reports that MoE verification takes 2–3$\times$ longer and can turn speculative decoding into a slowdown of up to $1.5\times$ (Saxena et al., 2025).

Recent methods address this bottleneck by imposing a per-layer expert budget during verification. Speculatively drafted tokens are consecutive continuations of one sequence, so their expert preferences overlap substantially (see Figure 4). And this overlap is what makes a shared expert budget feasible. In our Qwen3-30B-A3B traces, the eight tokens still activate roughly 20 experts per layer,

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.

$2.5\times$ the one-token working set rather than the 64 a disjoint block would need. This is what a selection policy must exploit, since an expert requested by several tokens costs the budget only once. Concretely, MoE-Spec(McDanel et al., 2026) sums routing probabilities across tokens, ranks experts by this aggregate mass, and retains the top M. AcceptMoE (Liang et al., 2026) refines this paradigm by weighting each token's demand by its probability of surviving to the output, always retaining the first verification token's original top-k experts, and adapting the expert budget to the effective rank of the demand distribution. These methods capture how much the block wants each expert, but an aggregate ranking does not guarantee how well *each token* is served. A moderately popular expert can displace the first-choice expert of an atypical token even when the union of all first choices fits within the same budget. This failure mode is amplified by prefix acceptance: harming one early token rejects not only that token but also every later proposal in the cycle, and can also lead to accepting tokens that do not agree with the target's original distribution, degrading model quality.

We propose FairSpec, a verifier-side expert-selection method built around a per-speculative token coverage guarantee. FairSpec admits experts in rounds: first every token's first choice, then every token's second choice, and so forth. This rule treats overlap as a benefit, i.e. an expert requested by several tokens is expended for once, while preventing common experts from consuming the budget. Equivalently, FairSpec solves a discrete max-min allocation problem over shared expert resources. Moreover, FairSpec is training-free and confined to the verification pass. At each MoE layer, it converts the batched router logits into a shared expert mask, leaving the model's standard top-k routing, expert computation, and mixing unchanged; prefill and ordinary autoregressive decoding are unaffected. A graded expert score reduces both round ordering and partial-round tie-breaking to a single top-M operation, enabling implementation as a fused kernel with fixed control flow under CUDA graph capture.

We summarize the paper's contributions as follows.

- We identify a mismatch in existing expert-subsetting methods: a globally selected subset can exclude the experts preferred by individual tokens, making generation quality brittle under tight expert budgets. We propose FairSpec, a robust expert-subsetting procedure that ensures every token retains its top-preferred expert, preserving routing quality under constrained expert budgets.
- We theoretically show that FairSpec maximizes worst-token routing-prefix coverage under a shared per-layer budget and provide instances showing that selecting based on aggregate mass can perform arbitrarily poorly.
- We run extensive evaluation experiments spanning 3 sparse MoE architectures across 4 benchmark eval datasets, measuring quality and throughput across multiple budgets and speculative depths, as well as diagnostics of expert traffic and per-token coverage. We find that at comparable throughput, FairSpec achieves 10–50pp higher success rates than those of the baselines under low and moderate expert budgets, with the largest gains at lower budgets.

1.1 Related Work

The full related work section can be found in Appendix A. The area of speculative decoding (Leviathan et al., 2023; Chen et al., 2023) has grown rapidly, with research exploring specialized drafters (Li et al., 2025), multiple drafters (Yang et al., 2025), and parallel drafting (Cai et al., 2024; Chen et al., 2026; Cheng et al., 2026). The work most closely related to ours focuses on sparse mixture-of-experts (MoE) models (Shazeer et al., 2017; Fedus et al., 2022), which reduce inference cost by activating only a subset of parameter blocks. Recent work addresses the interaction between sparse routing and speculative verification from two sides. On the draft side, utility-driven speculative decoding disables speculation or adapts its depth when the expected accepted-token gain does not justify MoE verification compute cost (Saxena et al., 2025). EVICT truncates a draft tree to a cost-effective prefix before target verification (Pan et al., 2026), while EcoSpec incorporates predicted marginal expert activation into draft-path selection and favors candidates that reuse experts already represented in the verification set (Xie et al., 2026). These methods choose *which draft tokens* to verify. On the verifier side, MoE-Spec caps the expert union by ranking experts according to routing mass aggregated over the verification block (McDanel et al., 2026); AcceptMoE weights that demand by commitment probability, anchors the root token's natural experts, and self-sizes the eligible set (Liang et al., 2026). FairSpec operates on the same verifier-side lever but replaces aggregate maximization with max-min token coverage. Draft-side pruning, parallel drafting, and

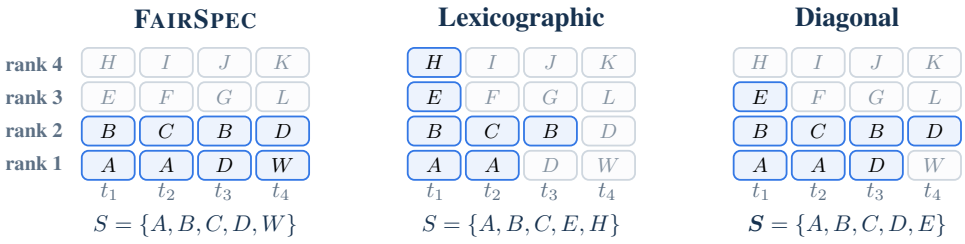

Figure 1: Illustrative example for 3 expert-selection policies applied with $T = 4$ speculative tokens $\{t_1, t_2, t_3, t_4\}$, total expert budget $M = 5$, and each token routing to $k = 4$ experts. Letters denote expert indices, and shaded cells correspond to experts in the selected set $\boldsymbol{S}$.

verifier-side expert selection are complementary: the former determine the verification structure, whereas FAIRSPEC bounds and allocates the target experts used for that structure.

2 PRELIMINARIES

Mixture-of-Experts (MoE) Language Models. Mixture-of-experts models increase parameter capacity while activating only a small subset of weights for each token (Shazeer et al., 2017; Fedus et al., 2022). In an MoE layer, the dense feed-forward network is replaced by N expert networks $\{E_1, \ldots, E_N\}$ and a learned router. Given the hidden representation $\mathbf{h}_t \in \mathbb{R}^d$ of token t, a softmax router assigns expert e the probability

$$p_{t,e} = \exp(z_{t,e}) / \sum_{j=1}^{N} \exp(z_{t,j}), \quad \text{where } z_{t,j} = \mathbf{w}_j^\top \mathbf{h}_t \tag{1}$$

where $\mathbf{w}_e \in \mathbb{R}^d$ is the router vector for expert e. The method below also applies to routers that use sigmoid gates; we use softmax notation for clarity.

Let

$$R_t(k) = \mathrm{TopK}_{e \in [N]}\, p_{t,e} \tag{2}$$

denote the indices of the k highest-probability experts for token t, using deterministic tie-breaking. More generally, $R_t(r)$ denotes the corresponding top-r set, with $R_t(0) = \varnothing$. A standard top-k MoE layer computes

$$\mathrm{MoE}(\mathbf{h}_t) = \sum_{e \in R_t(k)} \bar{p}_{t,e} E_e(\mathbf{h}_t), \tag{3}$$

where $\bar{p}_{t,e}$ denotes either the original gate or the gate renormalized over $R_t(k)$, depending on the architecture. Although one token executes only $k \ll N$ experts, processing several tokens jointly requires the activating the union of experts for each token.

Speculative Decoding. In autoregressive speculative decoding, a draft model proposes γ tokens. The target evaluates those tokens together with one additional token, producing a verification block of size $T = \gamma + 1$. Draft tokens are accepted from left to right until the first mismatch using rejection sampling. See more details in (Leviathan et al., 2023).

Verification-time expert subsetting. Expert subsetting selects, at each layer, a shared set $\boldsymbol{S} \subseteq [N]$ with $|\boldsymbol{S}| \leq M$ and masking all experts outside it before entering the expert probabiltiy computation in Eq (1)

$$\tilde{z}_{t,e} = \begin{cases} z_{t,e}, & e \in \boldsymbol{S}, \\ -\infty, & e \notin \boldsymbol{S}. \end{cases} \tag{4}$$

The model's standard top-k router then operates on the masked logits.

An example verification-time expert subsetting mechanism is MoE-Spec (McDanel et al., 2026), which selects the M experts with largest routing mass,

$$\boldsymbol{S}^m \in \arg\max_{|\boldsymbol{S}|\leq M} \sum_{e\in\boldsymbol{S}} m_e \quad \text{where} \quad m_e = \sum_{t=1}^{T} p_{t,e}, \tag{5}$$

Another example is AcceptMoE (Liang et al., 2026), which instead weights expert demand by each token's commitment probability, preserves the root token's natural top-k experts, and adaptively sizes the shortlist.

Note that both MoE-Spec and AcceptMoE rank experts primarily by expert demand aggregated across all speculative tokens. As a consequence, an expert receiving moderate weight from many tokens can therefore displace the preferred expert of an atypical token, even when a more balanced allocation is possible under the same budget. In the next Section 3 we formalize this limitation and introduce a mitigation policy.

3 FairSpec

In this section we propose FairSpec, a new verification-time expert-subsetting algorithm motivated by two complementary observations. First, speculative tokens exhibit substantial expert overlap, e.g. our analysis in Appendix B.1 for Qwen3-30B-A3B shows that 8 tokens demand only about 20 distinct experts rather than the worst-case 64 (8 speculative token $\times$ 8 experts per token). Second, the shared expert budget should retain the highest-ranked experts of every token, rather than merely maximize aggregate routing mass. This is because an aggregate policy can exclude the preferred experts of a token whose routing choices differ from the rest of the verification block. which is particularly harmful in speculative decoding because rejection at one token discards the remainder of the proposed suffix.

Motivated by these observations, FairSpec adopts a *round-based selection* approach. Each round considers the next-highest-ranked expert of every token, allowing shared experts to reduce budget usage while ensuring that no token is neglected. The following Section 3.1 formalizes this procedure.

3.1 Round-based selection

Recall from Equation 8 that $U_r = \bigcup_{t=1}^{T} R_t(r)$ is the set of experts needed to give every token its top r choices. FairSpec admits experts in *rounds*: first every token's first choice, then every token's second choice, and so on, taking every complete round that fits within the budget. Formally, given budget M it sets

$$r^*(M) = \max\{r : |U_r| \leq M\}, \qquad S^* = U_{r^*(M)}. \tag{6}$$

Any capacity left over cannot fund a further complete round, so it is spent on a *partial* round: among the experts of round $r^* + 1$, those of greatest aggregate mass m_e are admitted until the budget is exhausted. The partial round never displaces a complete one, so it cannot reduce the guarantee established below; it converts unused budget into extra coverage for the tokens that happen to share the heaviest next-choice experts.

As an illustrative example, Figure 1 considers $T = 4$ speculative tokens, each routed to $k = 4$ experts, with an expert budget of $M = 5$. In the first round, FairSpec selects the tokens' first-choice experts $\{A, D, W\}$, counting the shared expert A only once. In the second round, it adds $\{B, C\}$, while D is already included. The resulting subset $S = \{A, B, C, D, W\}$ therefore preserves the top two choices of every token using only five distinct experts.

3.2 Robustness guarantees

The purpose of FairSpec's round-based selection is to prevent any token from being sacrificed for small gains distributed across the rest of the verification block. Suppose that r^* complete rounds fit within the budget. Then every token retains its top r^* experts. Guaranteeing one additional expert to every token would require selecting the entire union of their top-$r^* + 1$ experts, which exceeds the budget by definition. Thus, no selection policy can provide a stronger uniform guarantee under the same budget.

We formalize this property in the below Theorem 1. For a selected expert set S, define

$$q_t(S) = \max\{r : R_t(r) \subseteq S\}, \qquad q(S) = \min_{1 \le t \le T} q_t(S).$$

Here, $q_t(S)$ is the number of consecutive highest-ranked experts retained for token t, while $q(S)$ measures the coverage of the worst-served token.

Theorem 1 (Worst-token optimality)**.** *Among all expert sets satisfying $|S| \le M$, S^* maximizes $q(S)$.*

Remark on relation to max–min allocation. Theorem 1 places FAIRSPEC in the tradition of max-min resource allocation (Bertsekas & Gallager, 1992; Radunovic & Le Boudec, 2007): each verification token is a participant, its prefix coverage $q_t(S)$ is its service level, and the budget limits the number of distinct shared experts. The distinctive feature of this setting is sharing, i.e. one selected expert can raise the service level of several tokens while counting once against the budget, so FAIRSPEC is a discrete max-min allocation rule with shared resources.

Although Theorem 1concerns expert ranks, it also provides a lower bound on retained routing probability. Because experts are ordered by decreasing router probability, the top r^* experts account for at least an r^*/k fraction of the total probability assigned to the original top-k experts. Define the routing mass available to token t from a subset S as

$$g_t(k, S) = \max_{\substack{A \subseteq S \\ |A| \le k}} \sum_{e \in A} p_{t,e}.$$

Theorem 2 (Routing-mass retention)**.** *Let S^* be the output of* FAIRSPEC *and $r^* \in [k]$ the coverage level it guarantees for every token. Then for any token t, $g_t(k, S^*) \ge \frac{r^*}{k}\, g_t(k, [N])$.*

Thus, whenever at least one complete round fits, every token retains a nonzero and quantifiable fraction of its unrestricted top-k routing mass. The guarantee applies separately to every token rather than only on average across the block.

This distinction is absent from aggregate-mass selection. Because aggregate mass sums demand across tokens, an expert requested by several tokens can displace the preferred expert of a token whose routing choices are less widely shared. Consequently, high aggregate retention does not rule out a token receiving none of its preferred routing mass.

Theorem 3 (Failure of aggregate selection)**.** *Let $\boldsymbol{S}^m = \arg\max_{|\boldsymbol{S}| \le M} \sum_{e \in \boldsymbol{S}} m_e$ and $r^m = \min_{t \in [T]} \max\{r : R_t(r) \subseteq \boldsymbol{S}^m\}$. There exists an instance with $0 = r^m < r^*$ in which some token t has $g_t(k, \boldsymbol{S}^m) = 0$.*

Theorem 3 does not imply that aggregate selection always neglects a token; rather, it shows that its objective provides no protection against this failure. In speculative decoding, such worst-token behavior matters because rejection at one token terminates acceptance for the remainder of the proposed suffix. FAIRSPEC avoids this failure while achieving the strongest possible uniform coverage under the expert budget. The proofs for the above are deferred to Appendix C.

3.3 A SINGLE-PASS IMPLEMENTATION FOR FAIRSPEC

Note that although FAIRSPEC's round-based selection mechanism in Equation 6 is stated as a search over coverage levels, it can be implemented in a single pass. Assigning each expert the earliest rank at which any token requests it,

$$d_e = \min\big(\{r : e \in R_t(r) \text{ for some } t,\ 1 \le r \le k\} \cup \{k+1\}\big),$$

turns the whole policy into one top-M selection.

Proposition 1. *Let $B > \max_e m_e$ and $g_e = m_e + B\,(k + 1 - d_e)$. Then the M experts of greatest g_e are exactly S^*, together with the mass-ordered partial round.*

Proposition 1 is what makes the policy cheap: no explicit round loop, no data-dependent branch, and one ranking operation per layer. Algorithm 1 states the resulting procedure. It runs inside the forward pass of each MoE layer, after the router has scored all verification tokens in one batch and before expert dispatch, so it requires no additional model forward pass. The two early exits ($M \ge N$ and $M \ge Tk$) are decidable from shapes alone: above either threshold the union cannot exceed the budget and the mask is provably a no-op.

Algorithm 1 FAIRSPEC

Require: Batched router logits $Z \in \mathbb{R}^{T \times N}$, router top-$k$, budget M
Ensure: Masked router logits $\widetilde{Z}$
$M \leftarrow \max\{k, \min\{M, N\}\}$
if $M \geq N$ or $M \geq Tk$ **then**
  return Z
end if
$P \leftarrow \text{SOFTMAX}(Z)$ ▷ applied row-wise to all tokens
for $e = 1, \ldots, N$ **do**
  $m_e \leftarrow \sum_{t=1}^{T} p_{t,e}$ ▷ aggregate routing mass
  $d_e \leftarrow \min\big(\{r : e \in R_t(r) \text{ for some } t,\ 1 \leq r \leq k\} \cup \{k+1\}\big)$ ▷ earliest rank
end for
$B \leftarrow 1 + \max_e m_e$
$g_e \leftarrow m_e + B(k + 1 - d_e)$ for every expert e
$S \leftarrow$ indices of the M largest values of g_e
$\widetilde{z}_{t,e} \leftarrow \begin{cases} z_{t,e}, & e \in S, \\ -\infty, & e \notin S \end{cases}$
return $\widetilde{Z}$

Table 1: MoE target models and their drafters. Total parameters are summed over each model's published `safetensors` index.

	TARGET				DRAFTER	
MODEL	**PARAMS**	**ACTIVE**	**EXPERTS**	**TOP-k**	**METHOD**	**PARAMS**
Qwen3-30B-A3B (Team, 2025)	30.5B	3.3B	128	8	EAGLE-3	0.52B
gpt-oss-20b (OpenAI, 2025)	20.9B	3.6B	32	4	EAGLE-3	0.85B
Gemma-4-26B-A4B (Team et al., 2026)	26.5B	4B	128	8	MTP	0.42B

4 EXPERIMENTS

We run experiments with state-of-the art benchmark eval data to address three questions: (i) how task quality changes as the verifier's expert budget is reduced; (ii) whether reduced expert traffic produces end-to-end throughput gains after accounting for draft acceptance; and (iii) whether worst-token expert coverage explains differences between selection policies better than aggregate retained mass.

4.1 EXPERIMENTAL SETUP

Target models and drafters. We evaluate our methods using 3 different sparse MoE target models with model-matched drafter which are either an EAGLE-3 drafter (Li et al., 2025) or an MTP drafter. The full list of used models and drafters is presented in Table 1. Unless otherwise stated, we use batch size one, tensor parallelism one, and greedy single-path speculative decoding.[1]

Baselines for expert selection. We compare our proposed FAIRSPEC verifier-side decoding algorithm to the following baselines:

1. **Autoregressive (AR):** target-only greedy decoding without speculation.
2. **Unbudgeted speculative decoding (SD):** the same drafter and verifier with the target's natural expert routing (no expert budgeting).
3. **MoE-Spec w/ top-M:** MoE-Spec's router-mass ranking in Equation equation 11, followed by the same substitution mask as FairSpec.
4. **Lexicographic.** The experts subset S is built by first adding the top-k experts of the first token in the sequence, followed by the top-k experts of the second token in the sequence, and so forth. If not all experts of a given token fit in the set S without exceeding M, the better ranked experts are prioritized. This heuristic is motivated by the findings in Liang et al. (2026), where higher priority is given to tokens earlier in the sequence since they are more likely to be accepted.

[1] Running a single path allows the use of FlashAttention (Dao et al., 2022).

5. **Diagonal.** This algorithm traverses the ranking in diagonal. That is, it first includes the top expert for token 1 in S. It then includes expert 1 of token 2 and expert 2 of token 1. So on and so forth, including experts r for token t whenever $t + r$ sum up to the diagonal being processed. In case of ties, priority is given to experts with higher ranking for their respective token. This policy can be seen as a midpoint between FAIRSPEC and Lexicographic, trying to accommodate the best choices of all tokens, while giving priority to tokens that appear earlier in the sequence.

Figure 1 depicts how baselines 4. Lexicographic and 5. Diagonal work, contrasting them to FAIRSPEC. The experiments hold the downstream masking and top-k computation fixed, isolating the shortlist-selection rule.

Benchmark eval data. We use 4 evaluation sets in our experiments: GSM8K (Cobbe et al., 2021), HumanEval (Chen et al., 2021), Math500 Lightman et al. (2024), and MBPP (Austin et al., 2021)

Implementation details Every experiment run uses one NVIDIA A100 GPU. The expert selection procedure is implemented on top of vLLM (Kwon et al., 2023) as a single fused Triton kernel invoked once per MoE layer during each verification step. Given the batched router logits, the kernel computes aggregate routing mass and earliest rank, forms the combined expert score, selects the top-M experts, and writes the masked logits in one launch. The implementation uses fixed control flow and does not synchronize with the host, making it compatible with CUDA graph capture.

Reference experimental configurations and variations. We study FAIRSPEC across multiple model and benchmark pairs using a common reference configuration. This configuration generates $\gamma = 7$ speculative tokens, greedy decoding with sampling temperature $T_{\text{temp}} = 0$, and a model-dependent range of expert budgets The reference configuration has an expert budget of $2 \times k$, where k is the number of experts to which a single token would be routed. This normalization allows us to compare models with different routing widths using the same relative budget scale. In addition to the reference configuration, we study the following variations:

- **Expert budget.** We vary the budget over $M \in \{k, 1.5k, 2k, 2.5k\}$ while fixing the number of speculative tokens to seven and the sampling temperature to $T_{\text{temp}} = 0$.
- **Number of speculative tokens.** We compare $\gamma \in \{3, 7\}$ while retaining the model-dependent expert-budget of $2k$ and fixing the sampling temperature to $T_{\text{temp}} = 0$.
- **Sampling temperature.** We compare $T_{\text{temp}} \in \{0, 1\}$ while retaining the model-dependent expert-budget of $2k$ and using seven speculative tokens.

Performance evaluation metrics. Finally, for each aforementioned reference and variant configuration, we evaluate the performance for our proposed FAIRSPEC algorithm as well as other baselines using the following metrics.

- **Success rate (SR)**: We define success differently for each eval. For GSM8K, we use exact numeric match up to 1e-6 tolerance. For MATH-500, we extract the final boxed expression and accept it when SymPy determines that it is mathematically equivalent to the reference. For HumanEval and MBPP, we report pass@1.
- **Throughput (λ)**: We define throughput λ as

$$\lambda = \frac{\text{mean accepted tokens per speculative cycle}}{\text{mean cycle time}} \tag{7}$$

- **Mean accepted length**: The mean number of tokens accepted per speculative-decoding cycle.
- **Average minimum mass ratio ($\overline{\rho}_{\min}$)**: This metric measures the routing mass of the worst-served token. Formally, for token t at layer ℓ, we define the retained routing-mass ratio as

$$\rho_{\ell,t}(\boldsymbol{S}) = \frac{\sum_{e \in \boldsymbol{S}_\ell \cap R_{\ell,t}(k)} p_{\ell,t,e}}{\sum_{e \in R_{\ell,t}(k)} p_{\ell,t,e}}.$$

Here, $\boldsymbol{S}_\ell$ denotes the shared set of experts retained at layer ℓ, and recall $R_{\ell,t}(k)$ denotes the top-k experts selected for token t at layer ℓ before expert masking. Therefore, $\boldsymbol{S}_\ell \cap R_{\ell,t}(k)$ is the set of token t's top-k preferred experts that remain available after masking. Finally, define average minimum mass ratio as the minima across all L layers:

$$\overline{\rho}_{\min} = \frac{1}{L} \sum_{\ell} \min_{t} \rho_{\ell,t}(\boldsymbol{S}).$$

For each configuration, model, and eval, we run different experiments to obtain different metrics. All of these use batch size = 1 (no continuous batching) and no parallelism (a single gpu). For the first three, we run the complete eval and assess the outputs and decode timing profiling including the number of tokens produced and the total decoding time. For the average minimum mass ratio, we run 10 prompts and capture the routing probabilities for each token and each layer.

4.2 RESULTS

Topline metrics for reference configuration . The summary of our results for the reference configuration is presented in Table 2. We can observe that FAIRSPEC consistently obtains a better or on-par success rate than all of the expert subsetting baselines. This is achieved while remaining nearly the fastest algorithm. We observe that the worst success rate is obtained by the Lexicographic baseline, showcasing that greedily giving preference to the first tokens in the sequence severely degrades model quality. FAIRSPEC also outperforms Diagonal in success rate. Diagonal can be thought as having some degree of balance between robustness and giving priority to earlier tokens, meaning that the optimal balance is leaning more towards FAIRSPEC than the evenly designed Diagonal. Finally, FAIRSPEC obtains the highest $\overline{\rho}_{\min}$ in all cells.

We observe that FAIRSPEC can outperform MoE-Spec in terms of throughput, even if it has strictly more overhead. This can be explained by the higher acceptance rate, ultimately requiring less cycles to produce the same number of tokens.

Table 2: Reference configuration: $\gamma = 7$, greedy decoding, and budget $M = 2 \times k$ (16 of 128 experts for Qwen3 and Gemma 4; 8 of 32 for gpt-oss). S: success rate; $\times$: decode speedup over AR; τ: mean accepted length; $\overline{\rho}_{\min}$: average minimum mass ratio. Highlighted and bold entries denote the best and runner-up policies, respectively. Note that under greedy decoding with temperature 0, unbudgeted speculative decoding is expected to reproduce autoregressive decoding, but this may not be the case due to numerical or kernel nondeterminism.

		GSM8K				**HumanEval**				**MATH-500**				**MBPP**			
MODEL	**ALG**	S	$\times$	τ	$\overline{\rho}_{\min}$	S	$\times$	τ	$\overline{\rho}_{\min}$	S	$\times$	τ	$\overline{\rho}_{\min}$	S	$\times$	τ	$\overline{\rho}_{\min}$
Qwen3-30B-A3B	AR	96.0	1.00	—	—	91.5	1.00	—	—	82.8	1.00	—	—	87.1	1.00	—	—
	SD	96.2	1.75	4.07	—	91.5	2.01	4.49	—	80.6	1.73	4.33	—	87.6	1.67	3.79	—
	MoE-Spec	87.3	1.65	3.54	**0.42**	75.6	**2.04**	4.38	0.45	70.8	1.80	4.08	0.39	72.8	1.77	3.69	**0.50**
	FAIRSPEC	96.1	1.81	3.95	0.59	93.3	2.03	4.48	0.62	81.6	1.80	4.26	0.58	87.1	1.66	**3.68**	0.65
	Lex	80.7	1.56	3.58	0.25	28.0	1.70	3.87	0.32	39.8	**1.64**	4.04	0.24	44.3	1.50	3.38	0.35
	Diag	**89.5**	**1.69**	**3.79**	0.41	**87.2**	2.06	**4.43**	**0.47**	**73.0**	1.80	**4.22**	**0.40**	**82.9**	**1.68**	3.69	**0.50**
gpt-oss-20b	AR	94.8	1.00	—	—	87.8	1.00	—	—	64.4	1.00	—	—	89.9	1.00	—	—
	SD	95.0	1.50	3.79	—	90.9	1.33	3.31	—	66.2	1.58	3.76	—	86.9	1.31	3.18	—
	MoE-Spec	**94.5**	1.78	3.74	0.36	85.4	1.57	**3.28**	0.33	67.6	1.82	3.83	0.43	87.6	1.52	**3.17**	**0.35**
	FAIRSPEC	95.1	**1.72**	3.74	0.59	85.4	**1.52**	3.32	0.55	65.6	**1.75**	**3.79**	0.67	**86.9**	**1.49**	3.20	0.55
	Lex	93.6	1.68	3.69	0.32	66.5	1.48	3.26	0.28	63.2	1.70	3.70	0.43	75.2	1.47	3.12	0.26
	Diag	94.2	1.69	**3.70**	**0.38**	**79.3**	1.48	3.24	**0.36**	**67.2**	1.71	3.71	**0.50**	80.6	1.46	3.11	0.34
Gemma 4 26B-A4B	AR	96.7	1.00	—	—	97.6	1.00	—	—	85.2	1.00	—	—	91.6	1.00	—	—
	SD	96.5	2.07	5.94	—	97.6	2.33	6.73	—	84.4	2.18	6.14	—	91.3	1.98	5.64	—
	MoE-Spec	**96.7**	2.07	**5.87**	**0.53**	**95.7**	2.44	6.70	0.57	84.4	2.16	6.08	**0.59**	92.0	1.96	**5.59**	**0.61**
	FAIRSPEC	96.8	**2.06**	5.88	0.67	96.3	**2.31**	**6.60**	0.70	84.4	**2.13**	6.08	0.71	**91.1**	**1.94**	5.62	0.73
	Lex	89.5	1.90	5.57	0.34	59.8	2.14	6.01	0.42	76.2	2.06	5.85	0.40	70.3	1.81	5.40	0.44
	Diag	94.8	2.00	5.80	0.51	90.2	2.26	6.43	**0.58**	**82.6**	2.10	**6.00**	0.57	88.5	**1.94**	5.53	0.60

Varying expert budgets. Figure 2 shows FairSpec generally achieves the highest success rate compared to all baseline methods, with most significant advantage under tight expert budgets; by $1.5 \times k$, it is within three percentage points of unbudgeted decoding on every benchmark, and at $2 \times k$ it is effectively lossless. Figure 3 shows increasing the budget improves worst-token routing-mass retention and success rate, but the gains largely plateau near $2 \times k$. Throughput is non-monotonic because loading additional experts must be balanced against the longer accepted prefixes they enable, with several workloads attaining their highest speedup at $1.5 \times k$. Thus, the best budget depends on whether a workload prioritizes maximum throughput or near-lossless quality.

Varying the number of speculative tokens. Figure 5 in Appendix compares $\gamma = 3$ and $\gamma = 7$ at a fixed $2 \times k$ budget. Increasing the speculative depth consistently raises the mean accepted length and speedup while leaving success rate nearly unchanged. This benefit comes at the cost of lower worst-token routing-mass retention – because more tokens share the same expert budget, $\mathbb{E}[\min_t \rho_t]$

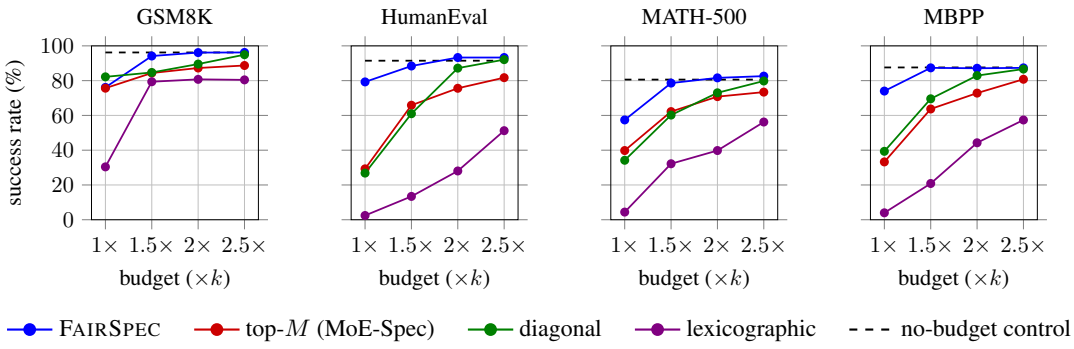

Figure 2: Success rate versus per-layer expert budget for four expert-selection policies on Qwen3-30B-A3B across four benchmarks ($k = 8$, $\gamma = 7$, temperature 0).

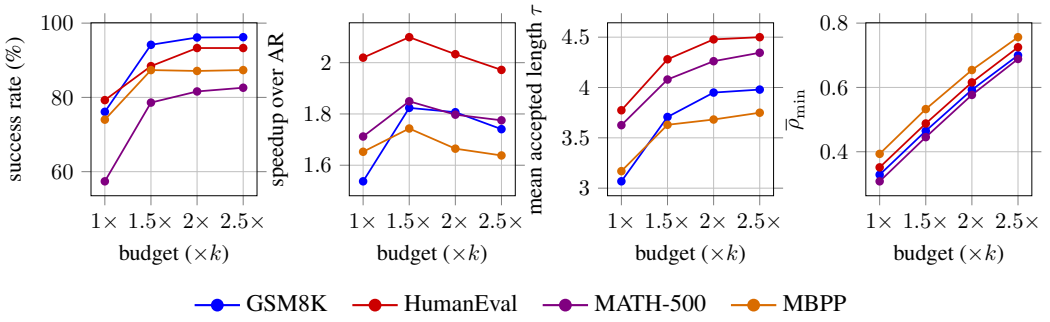

Figure 3: Effect of the per-layer expert budget on FairSpec's success rate, decode speedup, mean accepted length, and worst-token routing-mass retention ($\gamma = 7$, temperature 0). Speedup is relative to the matched autoregressive run.

decreases from approximately 0.83 to 0.62. These results indicate that FAIRSPEC can exploit longer drafts efficiently at this budget, although the reduced coverage may make deeper speculation more sensitive under tighter budgets.

Varying sampling temperature. Figure 6 in Appendix shows that FAIRSPEC remains effectively lossless under stochastic decoding at temperature 1 – relative to the corresponding unbudgeted control, every success-rate difference is within approximately one seed standard deviation, and the direction of the difference is inconsistent across benchmarks. This suggests that expert budgeting does not introduce a systematic quality penalty when sampling replaces greedy decoding. Mean accepted lengths also remain broadly comparable.

5 CONCLUSION

We introduced FAIRSPEC, a training-free method for verification-time expert subsetting for MoE speculative decoding. FAIRSPEC prioritizes experts by their token-wise rank and uses aggregate routing mass to resolve ties, thereby protecting every verification token while exploiting shared expert choices. We established its worst-token coverage guarantee and implemented selection and masking in a single fused Triton kernel. At comparable throughput, FAIRSPEC achieves success rates of 10–50pp higher than those of the baselines under low and moderate expert budgets, with larger gains at lower budgets. The results also show that appropriate budgets depend on the model and workload. Future work includes adaptive per-layer and per-request budgets and evaluation at larger batch sizes, speculation depths, and supporting tree drafters.

AI USE STATEMENT

In this work, we used generative AI tools to formulate mathematical claims, provide feedback on research methodology or experiments, and assist in the writing of proofs. We have not used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, design research methodology or experiments, interpret results, or propose or refine hypotheses, and cleaning and reformatting datasets, implementing methods, supporting qualitative and thematic data analysis, or assisting with translation are not applicable to this work. Additionally, we used generative AI tools to create or modify scientific figures, create or edit software code, summarize or analyze existing literature, source/search for information, edit our paper to improve readability, and identify relevant literature. We have reviewed all AI-assisted work. LLM-assisted proofs were carefully checked and refined by a human. LLMs were used as a search tool to surface related work for authors to read. Agentic coding assistants generated code which was reviewed by human authors.

REFERENCES

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Dimitri P. Bertsekas and Robert G. Gallager. *Data Networks*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2nd edition, 1992. ISBN 0132009161.

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 *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 5209–5235. PMLR, 2024. URL https://proceedings.mlr.press/v235/cai24b.html.

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. URL https://arxiv.org/abs/2302.01318.

Jian Chen, Yesheng Liang, and Zhijian Liu. DFlash: Block diffusion for flash speculative decoding. *arXiv preprint arXiv:2602.06036*, 2026. URL https://arxiv.org/abs/2602.06036.

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

Xin Cheng, Xingkai Yu, Chenze Shao, Jiashi Li, Yunfan Xiong, Yi Qian, Jiaqi Zhu, Shirong Ma, Xiaokang Zhang, Jiasheng Ye, Qinyu Chen, Chengqi Deng, Jiping Yu, Damai Dai, Zhengyan Zhang, Yixuan Wei, Yixuan Tan, Wenkai Yang, Runxin Xu, Yu Wu, Zhean Xu, Xuanyu Wang, Muyang Chen, Rui Tian, Xiao Bi, Zhewen Hao, Shaoyuan Chen, Huanqi Cao, Wentao Zhang, Anyi Xu, Huishuai Zhang, Dongyan Zhao, and Wenfeng Liang. DSpark: Confidence-scheduled speculative decoding with semi-autoregressive generation. *arXiv preprint arXiv:2607.05147*, 2026. URL https://arxiv.org/abs/2607.05147.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 16344–16359, 2022.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL https://www.jmlr.org/papers/v23/21-0998.html.

Zhengmian Hu, Tong Zheng, Vignesh Viswanathan, Ziyi Chen, Ryan A. Rossi, Yihan Wu, Dinesh Manocha, and Heng Huang. Towards optimal multi-draft speculative decoding. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/0907335ecf28faf15be54485dbcbe70e-Abstract-Conference.html.

Taehyeon Kim, Hojung Jung, and Se-Young Yun. Multi-drafter speculative decoding with alignment feedback. *arXiv preprint arXiv:2604.05417*, 2026. URL https://arxiv.org/abs/2604.05417.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19274–19286. PMLR, 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-3: Scaling up inference acceleration of large language models via training-time test. *arXiv preprint arXiv:2503.01840*, 2025. URL https://arxiv.org/abs/2503.01840.

Shuang Liang, Hao (Mark) Chen, Zhiwen Mo, Qianzhou Wang, Guoyu Li, Lingxiao Ma, and Wayne Luk. AcceptMoE: Commitment-weighted self-sizing verifier expert sets for efficient MoE speculative decoding. *arXiv preprint arXiv:2608.02989*, 2026. URL https://arxiv.org/abs/2608.02989.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Bradley McDanel, Steven Li, Sruthikesh Surineni, and Harshit Khaitan. MoE-Spec: Expert budgeting for efficient speculative decoding. *arXiv preprint arXiv:2602.16052*, 2026. URL https://arxiv.org/abs/2602.16052.

OpenAI. gpt-oss-120b & gpt-oss-20b model card, 2025. URL https://arxiv.org/abs/2508.10925.

Lehan Pan, Ziyang Tao, Xiao Wang, Zhimeng Zhao, Meng Wang, and Yanyong Zhang. Making every verified token count: Adaptive verification for MoE speculative decoding. *arXiv preprint arXiv:2605.00342*, 2026. URL https://arxiv.org/abs/2605.00342.

Bozidar Radunovic and Jean-Yves Le Boudec. A unified framework for max-min and min-max fairness with applications. *IEEE/ACM Transactions on Networking*, 15(5):1073–1083, 2007. doi: 10.1109/TNET.2007.896231.

Anish Saxena, Po-An Tsai, Hritvik Taneja, Aamer Jaleel, and Moinuddin Qureshi. Utility-driven speculative decoding for mixture-of-experts. *arXiv preprint arXiv:2506.20675*, 2025. URL https://arxiv.org/abs/2506.20675.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=B1ckMDqlg.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, et al. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

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

Jincheng Xie, Runheng Liu, Heyan Huang, Yawen Ling, Hanbin Dai, Yu Zheng, and Wen Hu. Less experts, faster decoding: Cost-aware speculative decoding for mixture-of-experts. *arXiv preprint arXiv:2607.12696*, 2026. URL https://arxiv.org/abs/2607.12696.

Sen Yang, Shujian Huang, Xinyu Dai, and Jiajun Chen. Multi-candidate speculative decoding. In *Natural Language Processing and Chinese Computing*, Lecture Notes in Computer Science, pp. 335–348. Springer Nature Singapore, 2025. doi: 10.1007/978-981-95-3343-5_26. URL https://doi.org/10.1007/978-981-95-3343-5_26.

A Extended Related Work

In this section, we present an extended survey of related work.

Single-drafter speculative decoding. Canonical speculative decoding uses one lightweight autoregressive drafter to propose a short, single-path continuation and one target-model pass to score the proposed tokens in parallel (Leviathan et al., 2023; Chen et al., 2023). A rejection-sampling correction preserves the target distribution, so speed depends primarily on the compute cost of drafting, the compute cost of verification, and drafter-target agreement. Subsequent work improves the drafter rather than the acceptance rule. For example, EAGLE-3 fuses intermediate target-model features and directly predicts future tokens, improving alignment while retaining a single learned drafter (Li et al., 2025). Our experiments use this single-drafter, single-path setting. FairSpec is agnostic to how the drafter is trained: it acts only on the target model after the verification block has been formed.

Multiple candidates and multiple drafters. The phrase *multi-draft* is used for two related but distinct ideas. Multi-candidate speculative decoding samples several candidate continuations from one drafter and verifies their shared prefix tree in a batch, increasing the probability that at least one branch agrees with the target (Yang et al., 2025). Theoretical work studies the best achievable acceptance rate for such multiple proposals and connects verification to optimal transport, showing that the proposal-generation rule itself is consequential (Hu et al., 2025). A genuinely multi-drafter system instead maintains several heterogeneous draft models or policies. MetaSD, for example, uses online alignment feedback to select among specialized drafters as the request distribution changes (Kim et al., 2026). Multiple candidates improve coverage, while multiple drafters improve robustness across domains; both may increase draft-side work, verification width, and state-management complexity. These costs are especially relevant for MoE targets because a wider or more diverse candidate set can activate a larger union of experts.

Parallel drafting. Parallel drafting removes or shortens the drafter's own autoregressive dependency. Medusa attaches multiple decoding heads to a shared backbone, predicts several future tokens concurrently, and organizes the predictions into a tree for joint verification (Cai et al., 2024). DFlash instead uses a lightweight block-diffusion model to generate an entire draft block in parallel, conditioning it on features from the target model to improve acceptance (Chen et al., 2026). DSpark combines a parallel backbone with a lightweight sequential module, thereby restoring some intra-block dependence, and schedules the verified prefix according to confidence and serving load to avoid spending verification capacity on a low-survival suffix (Cheng et al., 2026). Parallel drafting is therefore orthogonal to the number of draft models: one drafter can emit many tokens or branches concurrently. It reduces draft latency, but it does not remove target-side verification compute cost. FairSpec complements these approaches by limiting the expert working set used to verify whatever block or tree the drafter produces.

Mixture-of-experts models. Sparse MoE models scale parameter capacity by replacing dense feed-forward blocks with routed expert banks and evaluating only a small number of experts per token (Shazeer et al., 2017; Fedus et al., 2022). This conditional computation reduces per-token arithmetic, but introduces routing imbalance, expert communication, and weight-movement costs. The distinction between token-expert assignments and distinct active experts is central at inference time: several tokens may execute the same number of experts yet require very different amounts of parameter traffic depending on the overlap of their routes. FairSpec targets this union-level cost without changing the trained router or expert parameters.

MoE-aware speculative decoding. Recent work addresses the interaction between sparse routing and speculative verification from two sides. On the draft side, utility-driven speculative decoding disables speculation or adapts its depth when the expected accepted-token gain does not justify MoE verification compute cost (Saxena et al., 2025). EVICT truncates a draft tree to a cost-effective prefix before target verification (Pan et al., 2026), while EcoSpec incorporates predicted marginal expert activation into draft-path selection and favors candidates that reuse experts already represented in the verification set (Xie et al., 2026). These methods choose *which draft tokens* to verify. On the verifier side, MoE-Spec caps the expert union by ranking experts according to routing mass aggregated over

the verification block (McDanel et al., 2026); AcceptMoE weights that demand by commitment probability, anchors the root token's natural experts, and self-sizes the eligible set (Liang et al., 2026). FairSpec operates on the same verifier-side lever but replaces aggregate maximization with max-min positional coverage. Draft-side pruning, parallel drafting, and verifier-side expert selection are complementary: the former determine the verification structure, whereas FairSpec bounds and allocates the target experts used for that structure.

B Extended Motivation for FairSpec

In this section, we elaborate on the motivation behind FairSpec by looking at the overlaps between the experts at nearby tokens.

B.1 Verification tokens request overlapping experts

Since weight traffic is set by the number of distinct experts a block loads, the quantity that matters during verification is the size of their union. Writing U_r for the experts needed to give every position its top r choices,

$$U_r = \bigcup_{t=1}^{T} R_t(r), \qquad r \le |U_r| \le \min\{N, Tr\}, \tag{8}$$

the upper bound is attained only when no two tokens agree on any expert, and the lower bound only when they all agree completely. Where a real verification block falls between these is an empirical question, and the answer governs how expensive verification is.

It falls close to the bottom. For Qwen3-30B-A3B with $\gamma = 7$ and $k = 8$, a block of $T = 8$ tokens could in principle require 64 distinct experts per layer; in our traces it requires roughly 20. The reason is that the tokens of a verification block are consecutive continuations of one sequence and tend to want the same experts.

We quantify this with the *Jaccard overlap* of two tokens' top-k sets ,defined as the size of their intersection divided by the size of their union. It is 1 when two tokens route identically, and two tokens drawing k of N experts independently would average 0.032 at $k = 8$, $N = 128$. Figure 4 reports the pairwise overlap among eight tokens at one MoE layer of Qwen3-30B-A3B under unconstrained routing, in two regimes. On the left the eight tokens form a single verification block: seven drafted tokens and the bonus token of one sequence. Their mean off-diagonal overlap is 0.27, eight times chance, and it decays with distance. Adjacent tokens agree at 0.34, the most separated pair at 0.22. On the right the eight tokens are eight unrelated sequences decoded together under continuous batching. The positional structure is gone and mean overlap falls to 0.04, indistinguishable from independent draws.

The contrast localizes the effect. Concentration of this kind is a property of speculative verification rather than of batched inference generally: a block's experts overlap because its tokens continue one context, and a serving system batching unrelated requests has no equivalent structure. What remains on the right is domain rather than position. The three biomedical sequences agree at 0.098 and the mathematics and code sequences at 0.069, against 0.028 between those groups. The residual similarity among unrelated requests tracks what they are about, not where they sit in the batch.

B.2 Verification-time expert subsetting

Overlap therefore does much of the work by itself, but more can be done. Twenty experts per layer is far below the worst case of 64, yet still $2.5\times$ what a single decoded token needs, and it is paid at every one of the model's 48 MoE layers on every verification pass. The remaining gap is what an expert-subsetting policy closes, by choosing in advance which experts the block is allowed to use.

Consider one MoE layer during a verification pass, with router logits $Z = [z_{t,e}] \in \mathbb{R}^{T\times N}$ and their row-wise softmax $P = [p_{t,e}]$. A policy selects one shared set $\boldsymbol{S} \subseteq [N]$ with $|\boldsymbol{S}| \le M$, for a per-layer expert budget M, and masks everything outside it,

$$\tilde{z}_{t,e} = \begin{cases} z_{t,e}, & e \in \boldsymbol{S}, \\ -\infty, & e \notin \boldsymbol{S}, \end{cases} \tag{9}$$

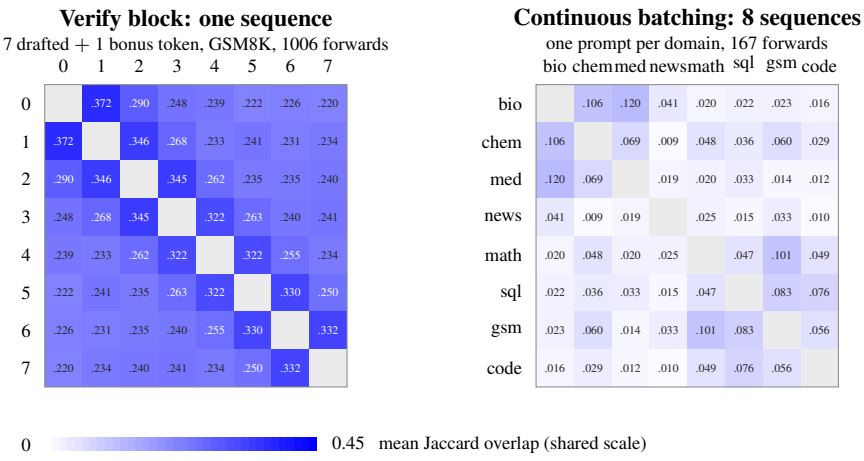

Figure 4: Expert-set overlap between the eight rows of a single MoE layer (layer 6, Qwen3-30B-A3B), measured as the mean Jaccard overlap of the two rows' top-k expert sets under unconstrained routing. **Left:** the rows are the eight positions of one verification block, so neighbouring rows are neighbouring positions in the same text; overlap is strong and decays away from the diagonal. **Right:** the rows are eight unrelated sequences decoding in parallel, one per domain. the positional structure disappears and the mean off-diagonal overlap falls to the level expected of independent draws (0.032). What remains on the right is domain, not position.

before handing $\tilde{Z}$ to the model's unmodified top-k router. The set is chosen afresh at every MoE layer and every verification pass rather than fixed globally. That one set is shared across tokens is what caps the cost: the block loads the union of what its tokens ask for, so masking each position separately would leave that union untouched.

The difficulty is that $\boldsymbol{S}$ must be small enough to cut weight traffic and still close enough to the model's own routing that neither draft acceptance nor model quality pays for the saving.

MoE-Spec (McDanel et al., 2026) resolves this by keeping the experts of greatest aggregate routing mass. Writing

$$m_e = \sum_{t=1}^{T} p_{t,e}, \tag{10}$$

its selection is

$$\boldsymbol{S}^m \in \arg\max_{|\boldsymbol{S}| \le M} \sum_{e \in \boldsymbol{S}} m_e, \tag{11}$$

which is obtained by retaining the M largest m_e. AcceptMoE (Liang et al., 2026) replaces m_e with demand weighted by each position's probability of surviving to the output, protects the root position's natural top-k, and sizes the shortlist from the effective rank of the demand distribution.

Both rank experts by a quantity summed over the block. Aside from AcceptMoE's root anchor, neither guarantees anything to an individual position: an expert with moderate weight at many tokens outranks the first choice of an atypical one, even when admitting both would fit the same budget. Section 3 makes that failure precise and gives a policy without it.

C FairSpec Theoretical Results

In this section, we present the proofs of theoretical results in the main body of the paper.

C.1 Proof for Theorem 1

Proof. Any set satisfying $q(S) \geq r$ must contain $R_t(r)$ for every position and therefore must contain U_r. Consequently coverage level r is feasible only if $|U_r| \leq M$. By definition $r^*(M)$ is the largest feasible level, and $S^* = U_{r^*(M)}$ attains it. □

C.2 Proof for Theorem 2

Proof. Fix a position t and set $A = \sum_{e \in R_t(r^*)} p_{t,e}$. Since $r^* \leq k$ and $R_t(r^*) \subseteq S^*$, the set $R_t(r^*)$ is feasible in the maximisation defining $g_t(k, S^*)$, so $g_t(k, S^*) \geq A$. For the other side, $g_t(k, [N]) = \sum_{e \in R_t(k)} p_{t,e}$; by monotonicity of the ranking every expert in $R_t(k) \setminus R_t(r^*)$ has probability at most $\min_{e \in R_t(r^*)} p_{t,e}$, which is at most the average A/r^*. There are $k - r^*$ such experts, hence

$$g_t(k, [N]) \;\leq\; A + (k - r^*)\frac{A}{r^*} \;=\; \frac{k}{r^*} A \;\leq\; \frac{k}{r^*}\, g_t(k, S^*).$$ □

C.3 Proof for Theorem 3

Proof. Take $T = 4$, $N = 3$, $k = 2$, $M = 2$, and experts $\{A, B, C\}$. For $t \in \{1, 2, 3\}$ let $(p_{t,A}, p_{t,B}, p_{t,C}) = (0.6, 0, 0.4)$, and let $(p_{4,A}, p_{4,B}, p_{4,C}) = (0, 1, 0)$. The aggregate masses of A, B and C are 1.8, 1.0 and 1.2, so top-mass selection chooses $\boldsymbol{S}^m = \{A, C\}$ and discards the fourth position's first choice: $R_4(1) = \{B\} \not\subseteq \boldsymbol{S}^m$, hence $r^m = 0$. By contrast $U_1 = \{A, B\}$ has $|U_1| = 2 \leq M$, so $r^* \geq 1 > r^m$. Moreover $g_4(k, \boldsymbol{S}^m) = 0$, whereas $g_t(k, [N]) = 1$ for every $t \in [T]$. □

C.4 Proof for Proposition 1

Proof. Because B exceeds every aggregate mass, g is ordered lexicographically by (d_e, m_e): any expert with a smaller earliest rank outranks every expert with a larger one, and ties in d_e are broken by m_e. Selecting the M largest therefore admits all of round 1, then all of round 2, and so on while each complete round fits, and spends the remainder on the highest-mass experts of the first round that does not. This is Definition 6 and its partial round. □

D Additional Experiments

This section contains additional experimental results based on varying the number of speculative tokens and temperature.

Varying number of speculative tokens. We observe that the accuracy is higher on $\gamma = 3$ than $\gamma = 7$. This can be explained by the fact that $\gamma = 3$ means $T = 4$ tokens in parallel, so less expert budget is needed to recover all the top-k original experts. This is observed in $\overline{\rho}_{\min}$ which is much greater for $\gamma = 3$ than $\gamma = 7$. We observe that throughput is higher in $\gamma = 7$, since most tokens are always accepted.

Varying sampling temperature. We observe that the accuracy achieves is lower with $T_{\text{temp}} = 1$ than with $T_{\text{temp}} = 0$. We interpret this result as $T_{\text{temp}} = 1$ being more sensitive to changes in distibution, as $T_{\text{temp}} = 0$ needs only that the token with highest logic remains being the same.

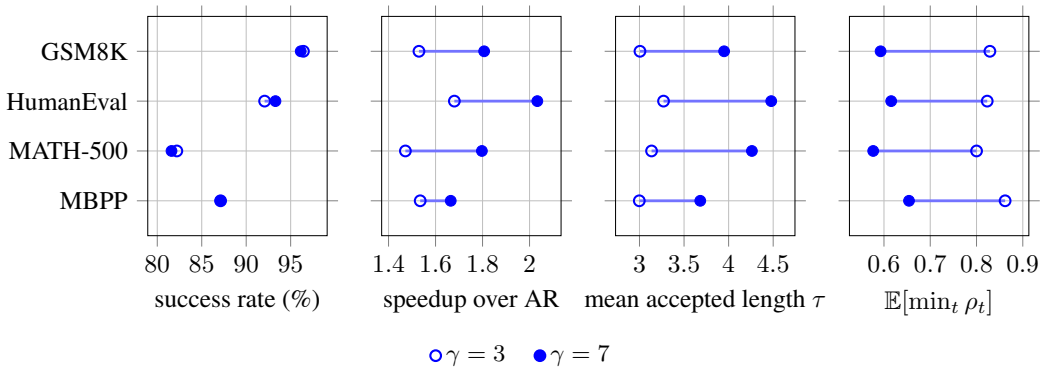

Figure 5: Effect of the number of speculative tokens on Qwen3-30B-A3B (FAIRSPEC, budget $2\times k$, temperature 0, full benchmarks). A longer draft raises τ and the speedup, but spreads one shared expert budget over more draft positions, so the worst-served position keeps less of its routing mass and $\mathbb{E}[\min_t \rho_t]$ falls from about 0.83 to about 0.61. Success rate is nearly unchanged, so at this budget the longer draft is bought cheaply. The two ends of each speedup dumbbell were measured on different machines, each against its own autoregressive reference; accuracy, τ and ρ are machine-independent.

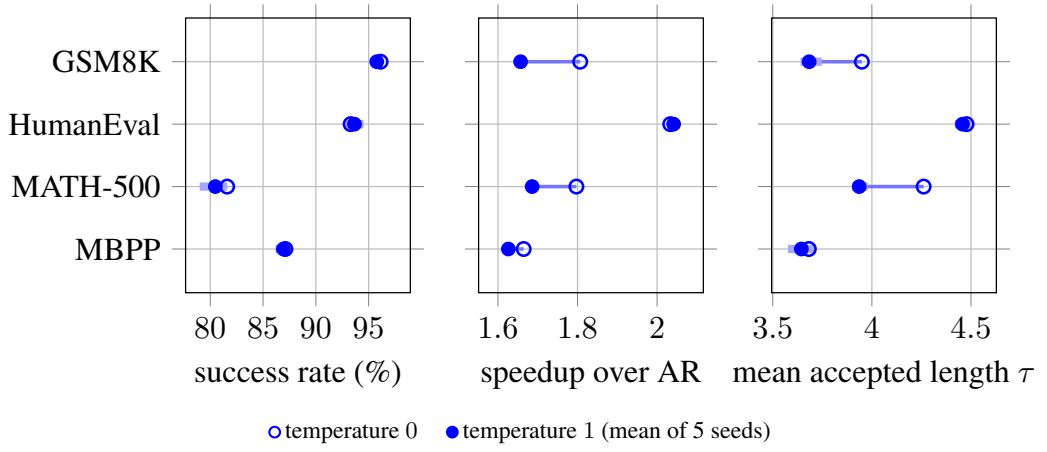

Figure 6: Effect of sampling temperature on Qwen3-30B-A3B (FAIRSPEC, budget $2\times k$, $\gamma = 7$, full benchmarks). Open marks are greedy decoding; filled marks are the mean over five seeds at temperature 1, and the thick bar is the seed min–max range. The budget stays effectively lossless once decoding is stochastic: every gap to the unbudgeted control at temperature 1 is within roughly one seed standard deviation and the sign is not consistent across benchmarks.