

Appendix

Table of Contents

298	– Cost Model	10
299	• Max Cost Model v.s. Additive Cost Model	10
300	• Details about Sparse Attention Cost Model	12
301	– Dense Scaling Law	12
302	• Additional Benchmarks	12
303	• Additional Reasoning Models	14
304	– Sparse Scaling Law	14
305	• Additional Benchmarks	14
306	• Additional Analysis	15
307	– Experimental Details	15
308	• Estimate Cost, Accuracy and Solving Rate	16
309	• Greedy Algorithm for Optimal Resource Allocation	17
310	• Top- K Attention and Block Top- K Attention	17
311	– Extended Related Work	19
312	– Limitations, Future Scope, and Broader Impact	19

A Cost Model

In this section, we delve into the cost models used in the Kinetics Scaling Law. We show empirically that adopting a max cost model does not alter the scaling behavior and outline methods for calculating the cost of sparse attention models.

A.1 Max Cost Model v.s. Additive Cost Model

Max cost model is widely used in performance modeling [97]. It assumes that computation and memory operations can be fully overlapped with each other and only considers the bottleneck operation for cost measurement.

$$C_{\text{max-cost}} = \max(C_{\text{comp}}, C_{\text{mem}} \times I)$$

where C_{comp} denotes the compute cost, C_{mem} the memory cost per access, and I the memory intensity.

In this section, we analyze the Kinetics Scaling Law using the max cost model. For clarity, we refer to the cost model $C_{\text{comp}} + C_{\text{mem}} \times I$, which is used in the main paper, as **the additive cost model**.

We draw two conclusions from empirical results **under the max cost model**:

- **Kinetics scaling law for dense models still holds.** We re-plot Figure 4(a)(b) and Figure 6a under the measurement of max cost models in Figures 12 and 13. We find except that in Long-CoTs scenarios, large models become slightly more effective in low-cost regime (with accuracy ~ 0.3), the overall trends are very close to the plots with additive cost models.
- **Sparse attention solves problems more cost-effectively.** We re-plot Figures 8a and 8d in Figures 14a and 14b. Under the max cost models, in Long-CoTs, the accuracy and efficiency gaps increase from 47.5 points and $11.21\times$ to 52.8 points and $15.71\times$, respectively. In Best-of- N , the gaps widen from 65 points and $10.67\times$ to 69.4 points and $19.64\times$. These results indicate that under the max cost model, our claim that sparse attention can enhance problem-solving performance is strengthened. Compared to dense attention models, sparse attention models tend to have more balanced memory and compute costs. Thus omitting one of them via a max cost model will favor sparse attention models.

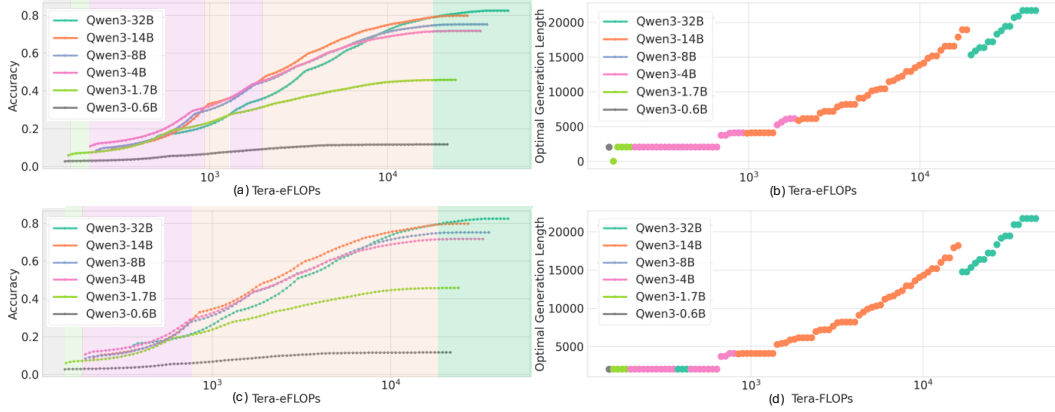


Figure 12: AIME Pareto Frontier (Long-CoTs) with Max Cost Models. (a)(b) is the original plot with the additive cost model. (c)(d) is the corresponding plot using max cost models. Compared to the original plots, the overall trend is similar except that larger models span a slightly broader region on the Pareto frontier. For example, the 14B model now consistently outperforms the 4B model with a noticeable gap around accuracy 0.3 and maintains dominance thereafter. In contrast, under the additive cost model in Figure 4(a), the two models alternate in performance until accuracy exceeds 0.4. This suggests that, when evaluated using a max cost model, larger models appear slightly more efficient relative to their performance under additive cost models.

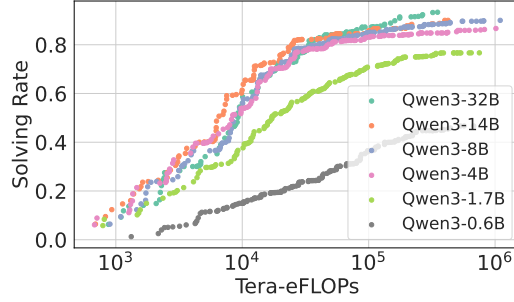


Figure 13: AIME Pareto Frontier (Best-of-N) with Max Cost Models. We re-plot Figure 6a using max cost models. The Pareto Frontier is very similar under different cost models.

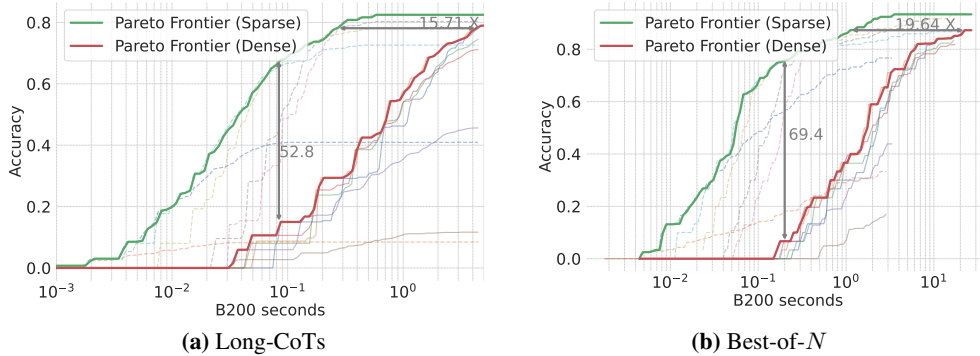


Figure 14: Sparse attention scales significantly better under max cost models. We re-plot Figures 8a and 8d using max cost models. Compared to the original plots, the performance and efficiency gaps between sparse attention models and dense models become more pronounced. In Long-CoTs, the accuracy and efficiency gaps increase from 47.5 points and $11.21\times$ to 52.8 points and $15.71\times$, respectively. In Best-of-N, the gaps widen from 65 points and $10.67\times$ to 69.4 points and $19.64\times$.

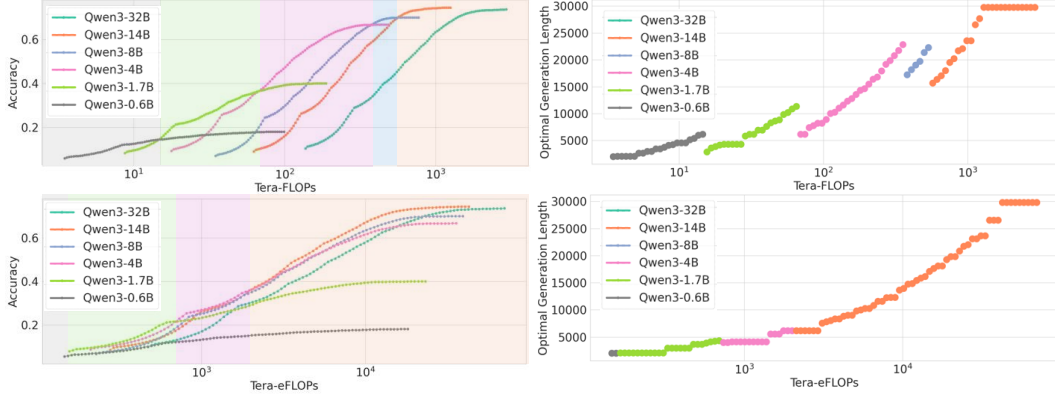


Figure 15: AIME25 Pareto Frontier (Long-CoTs). We conduct the same experiments as Figure 4.

A.2 Details about Sparse Attention Cost Model

Sparse attention models follow different cost functions due to the sparsification of KV memory access. In this paper, we focus on algorithms that impose a uniform KV budget (denoted as B) per attention head for each decoded token. We consider $L_{in} \geq B$ for the sake of simplicity. Under this setting, the cost model for sparse attention is given by:

$$C_{\text{sparse}} = \underbrace{2NPL_{\text{out}} + 2rNDBL_{\text{out}}}_{\text{compute}} + \underbrace{2INDBL_{\text{out}}}_{\text{memory}}. \quad (8)$$

In practical implementations, we must also account for the overhead associated with retrieving or searching KV memory, denoted as C_{search} , which depends on the specific sparse attention algorithm \mathcal{A} . For example, in block top- k selection, the search cost is:

$$C_{\text{search}} = \underbrace{\frac{2NL_{\text{in}}DL_{\text{out}} + rNDL_{\text{out}}^2}{2\text{Block-Size}}}_{\text{compute}} + \underbrace{\frac{2IL_{\text{in}}DL_{\text{out}} + INDL_{\text{out}}^2}{2\text{Block-Size}}}_{\text{memory}}. \quad (9)$$

In our work, we choose the Block-Size in such a way that C_{sparse} and C_{search} are roughly balanced, so that the sparse attention cost increases sub-linearly with generation length.

For local attention and oracle top- k attention, we assume no search overhead, i.e., $C_{\text{search}} = 0$.

Many sparse attention algorithms skip the first layer [79, 10, 101], resulting in only a minor increase in total cost. For the Qwen3 series, this additional overhead is bounded by 3.57% for the 0.6B model and by 1.56% for the 32B model.

B Dense Scaling Law

In this section, we further verify Kinetics Scaling Law for dense models proposed in Section 3 with extended experimental results of different benchmarks and model series.

B.1 Additional Benchmarks

We evaluate on AIME25 in Figures 15 and 16a to 16c and LiveCodeBench⁵ in Figures 17 and 18a to 18c (excluding the 0.6B model), following the setting described in Section 3. The empirical results support the Kinetics Scaling Law: across both benchmarks, the 0.6B and 1.7B models are consistently less effective, and the Pareto frontier is almost always dominated by the 14B models.

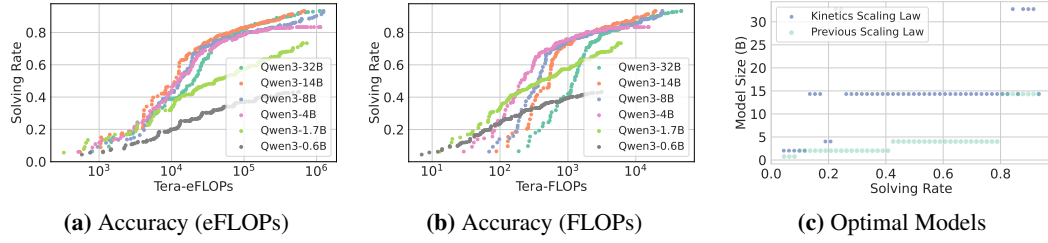


Figure 16: AIME25 Score Curve (Best-of- N). We conduct the same experiments as Figures 6a to 6c.

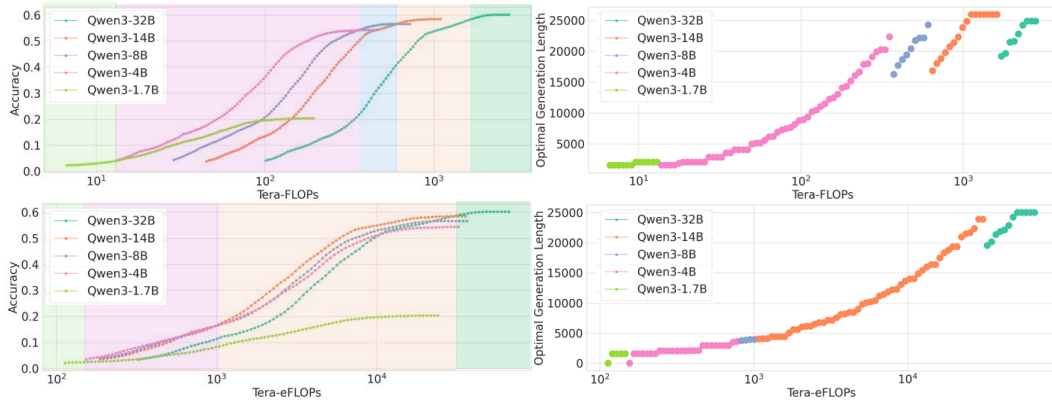


Figure 17: LiveCodeBench Pareto Frontier (Long-CoTs). We conduct the same experiments as Figure 4.

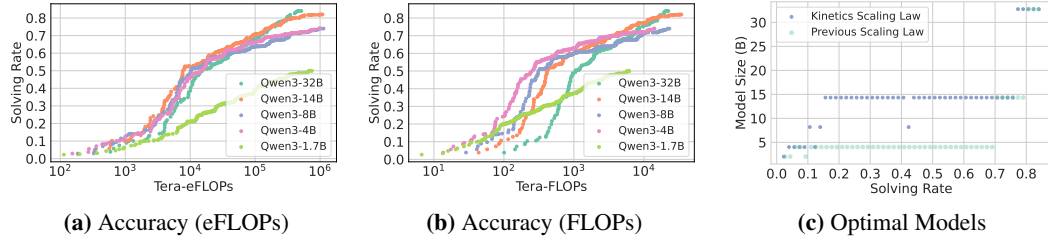


Figure 18: LiveCodeBench Score Curve (Best-of- N). We conduct the same experiments as Figures 6a to 6c.

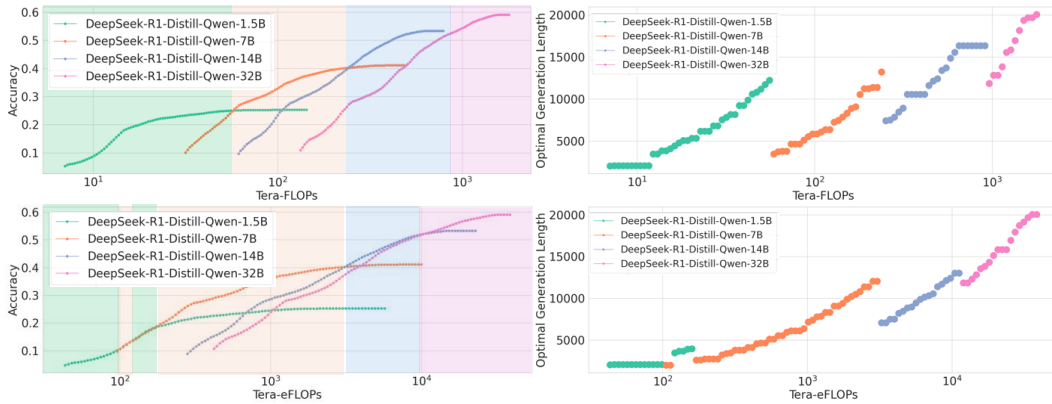


Figure 19: AIME24 Pareto Frontier (Long-CoTs). We conduct the same experiments as Figure 4 on DeepSeek Distilled Qwen series.

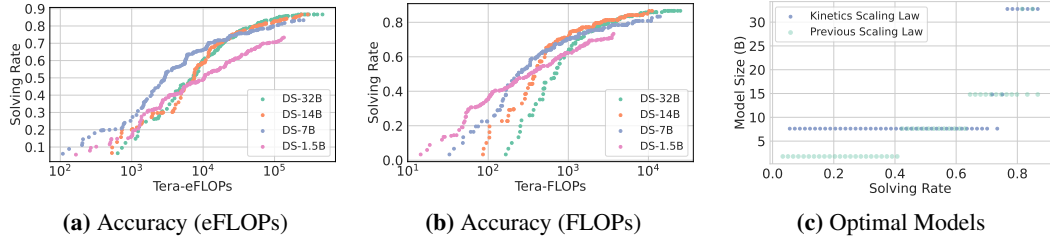


Figure 20: AIME24 Score Curve Envelope (Best-of- N). We conduct the same experiments as Figures 6a to 6c on DeepSeek Distilled Qwen series.

B.2 Additional Reasoning Models

In Figures 19 and 20a to 20c, we evaluate DeepSeek-R1 Distilled Qwen models (abbreviated as DS models) [30] on AIME24. The DeepSeek series models further demonstrate that previous scaling laws—those based on FLOPs—significantly overestimate the effectiveness of the 1.5B model. As predicted by the Kinetics Scaling Law, increasing the number of generated tokens for the 1.5B model is less effective than scaling up the model size, such as using the 7B or larger variants.

Interestingly, we observe a shift in the emerging model size: unlike Qwen3, where the 14B model dominates, the 7B model becomes the dominant choice in the DeepSeek series. In Figures 19, 20a and 20c, the 7B model spans most of the Pareto frontier, and Figure 19 shows that 7B models with long CoTs are more efficient and effective than 14B models with short generations. We attribute this to an architectural outlier in the DeepSeek-R1 (Qwen2.5) model series. As shown in Table 2, the DeepSeek-R1 7B model is significantly more KV memory-efficient than the Qwen3-8B model. Unlike most model series illustrated in Figure 5a, where KV cache size typically grows sublinearly with respect to model parameters, DeepSeek-R1 shows a deviation from this trend: the 14B model has approximately $3.4\times$ more KV memory than the 7B model, while having only $2\times$ more parameters.

Table 2: KV memory Size for Qwen3 and DeepSeek-R1 Distilled models (per 32K tokens, unit: GB).

Qwen3	Qwen3-1.7B	Qwen3-8B	Qwen3-14B	Qwen3-32B
	3.5	4.5	6	8
DeepSeek	DS-1.5B	DS-7B	DS-14B	DS-32B
	0.875	1.75	6	8

This finding highlights the importance of concrete model architecture design, rather than focusing solely on the number of model parameters. Whether KV memory size is directly related to reasoning performance remains an open question, which we leave for future investigation.

C Sparse Scaling Law

We present additional results supporting the kinetics sparse scaling law across multiple tasks and demonstrate how these insights enable scalable test-time scaling with sparse attention.

C.1 Additional Benchmarks

Beyond AIME24, we evaluate our approach on LiveCodeBench [39] and AIME25 [60]. LiveCodeBench features complex programming problems from recent coding contests, while AIME25 consists of challenging math problems. In both cases, sparse attention—particularly oracle top- k —consistently outperforms dense attention. Block top- k attention, a tractable alternative, closely matches the performance of the oracle.

⁵For LiveCodeBench dataset, we have sampled 50 examples from the v5 subset consisting 167 examples. Our subset comprises 24 hard, 16 medium and 10 easy examples respectively.

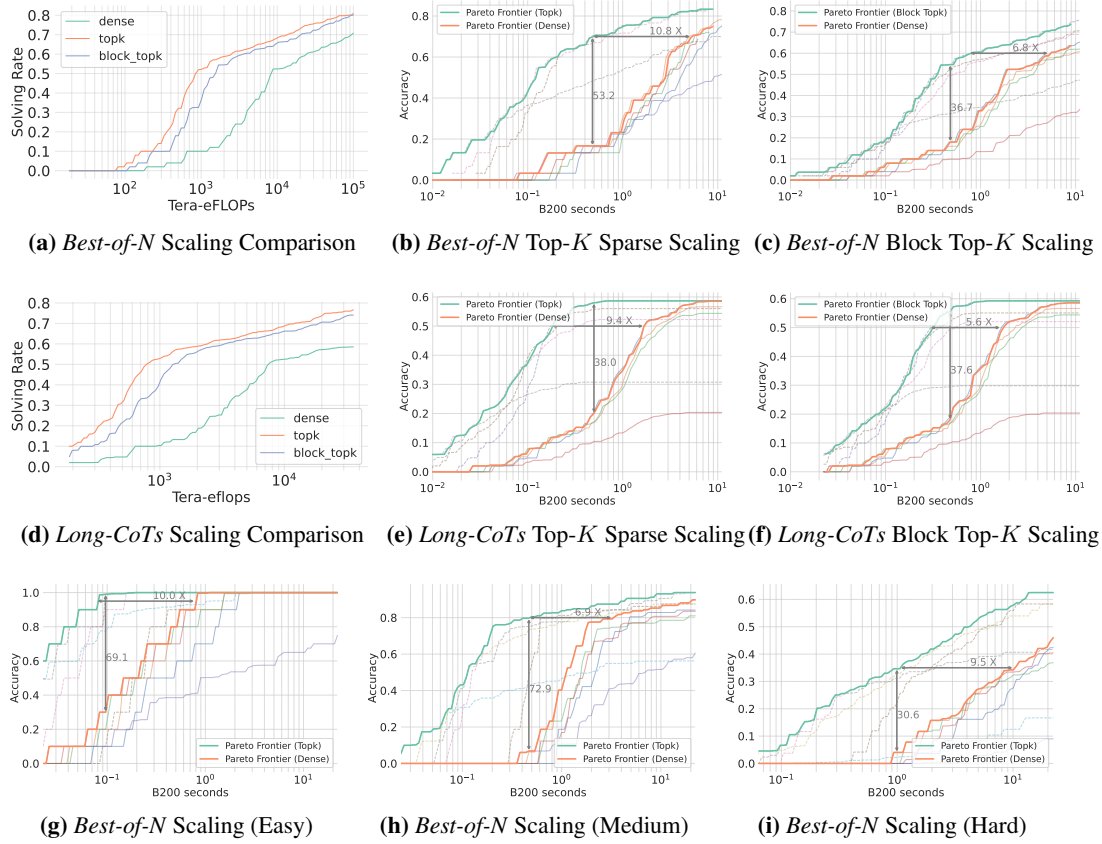


Figure 21: LiveCodeBench Sparse Scaling. We evaluate sparse scaling laws for Qwen3-14B model using oracle top- k and block-top- k attention on the LiveCodeBench dataset. (a)(d) compare block-top- k and oracle top- k with dense scaling under *Best-of-N* and *long-CoT* TTS settings. (b)(e) show cost-accuracy trade-offs for top- k attention. (c)(f) show trade-offs for block-top- k attention. (g)(h)(i) compare the oracle top- k scaling for easy, medium and hard difficulty questions.

For LiveCodeBench, we sample 50 problems from the v5 subset (24 hard, 16 medium, 10 easy). As shown in Figure 21, oracle top- k attention can achieve $\sim 10\times$ speedup in high-accuracy regimes and improves coverage by 40–50% in low-cost regimes. Conversely, the tractable alternative, Block top- k yields 5–6 \times speedup and 30–40% coverage gains. We further show how the benefits of sparse attention scale with problem difficulty (Figures 21g to 21i).

Figure 22 confirms similar trends for AIME25, with substantial gains in both accuracy and efficiency under sparse attention.

C.2 Additional Analysis

Fixing a model (e.g., Qwen3-8B), we investigate the tradeoff between generating more tokens through Best-of- N and increasing the KV budget in Figures 23a to 23d. As the figures suggest, on AIME25, each doubling of total compute cost increases the optimal KV budget by 1.13 \times , while generated tokens grow by 1.67 \times ; on LiveCodeBench, these factors are 1.14 \times and 1.89 \times , respectively. We find that although the concrete numbers depend on the types of tasks, the overall results confirm our suggestions in the main paper that allocating compute toward generating more responses is generally more effective than expanding KV budget, highlighting the scalability of sparse attention.

D Experimental Details

In this section, we explain the details about our experiments.

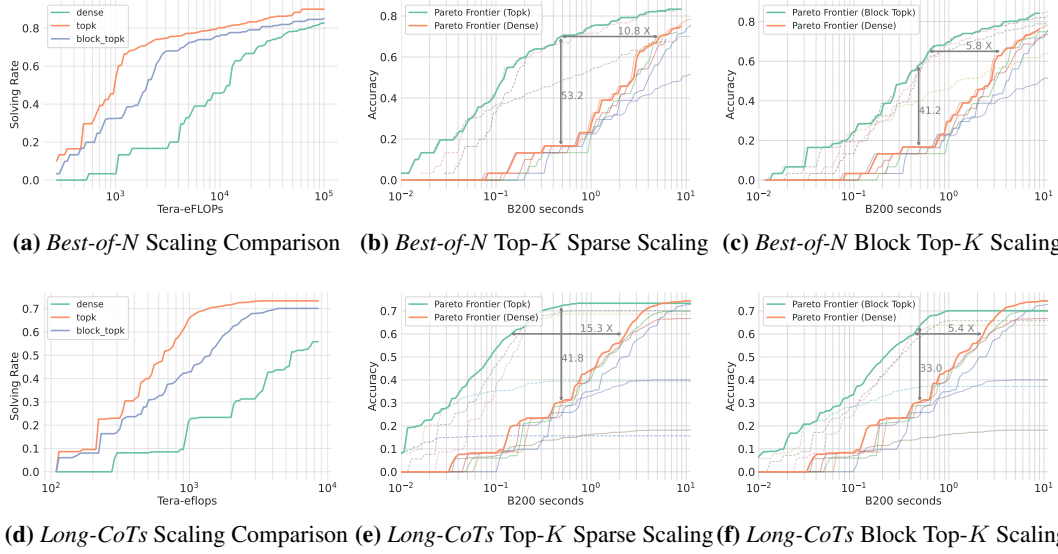


Figure 22: AIME25 Sparse Scaling. We evaluate sparse scaling laws for Qwen3-14B model using oracle top- k and block-top- k attention on the AIME25 dataset. (a)(d) compare block-top- k and oracle top- k with dense scaling under *Best-of-N* and *long-CoT* settings. (b)(e) show cost-accuracy trade-offs for oracle top- k attention. (c)(f) show trade-offs for block-top- k attention.

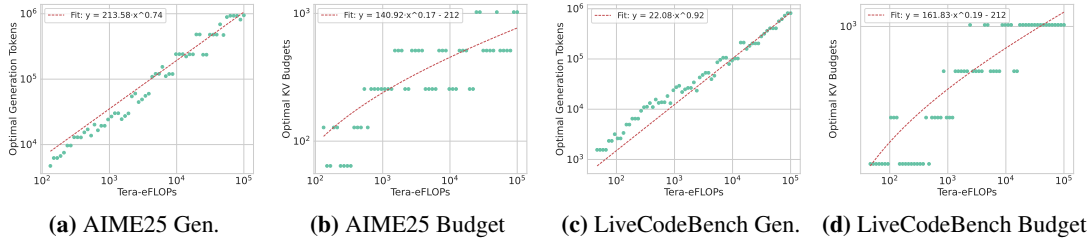


Figure 23: Tradeoff Between Generated Tokens and KV Budget. We empirically characterize the tradeoff between increasing generation length and allocating a larger KV cache budget using Qwen3-8B. For AIME25 ((a)(b)) and LiveCodeBench ((c)(d)), we identify the optimal KV budget and generated tokens (defined as number of reasoning trials times the average generated tokens per trial) to achieve the highest problem-solving rate under every cost constraint C .

403 D.1 Estimate Cost, Accuracy and Solving Rate

404 When empirically measuring cost, one major challenge is the difficulty of controlling the actual
 405 generation length. Although it is possible to set an upper bound on the number of generated tokens,
 406 there is no guarantee that the model will utilize the full budget. For instance, in our *Best-of-N*
 407 experiments, we set the maximum number of generated tokens to 32,768, yet the average generation
 408 length was only 14K–16K tokens.

409 Furthermore, it is important to model the relationship between actual inference cost and performance
 410 metrics, such as accuracy in *Long-CoTs* or solving rate in *Best-of-N*. Relying solely on the maximum
 411 allowed generation length to estimate cost can substantially underestimate the efficiency of models
 412 that solve problems with much shorter responses—an ability that **may** reflect higher capability.

413 To address this challenge, we first sample S independent reasoning traces r_1, r_2, \dots, r_S from model
 414 M on task T , with the maximum allowed number of tokens set to n . We slightly generalize Equa-

tion (4) as:

$$\begin{aligned}
C_{\text{TTS}} &= 2NP\mathbb{E}[L_{\text{out}}] + 2rNL_{\text{in}}D\mathbb{E}[L_{\text{out}}] + rND\mathbb{E}[L_{\text{out}}^2] \\
&\quad + 2IL_{\text{in}}D\mathbb{E}[L_{\text{out}}] + IND\mathbb{E}[L_{\text{out}}^2] \\
&= a\mathbb{E}[L_{\text{out}}] + b\mathbb{E}[L_{\text{out}}^2] + c,
\end{aligned} \tag{10}$$

where a , b , and c are constants determined by the model architecture and test-time strategies (e.g., the value of n). The expectations are estimated from the sampled traces, whose distribution is influenced by the model M , the token limit n , and the task T .

For Long-CoTs, we fix $N = 1$ in Equation (10) and vary n . From the sampled traces, we estimate the accuracy (Pass@1), and compute the corresponding cost by substituting the empirical values of $\mathbb{E}[L_{\text{out}}]$ and $\mathbb{E}[L_{\text{out}}^2]$ measured under each n .

For Best-of- N , we fix $n = 32,768$, and estimate the solving rate (Pass@ K) following the methodology of Brown et al. [4]. The corresponding cost is then computed by substituting $N = K$ into Equation (10).

Similarly, we can estimate the cost for sparse attention models using Equations (8) and (9).

Advanced control of generation lengths is an active area of research [91, 65, 57], but it is beyond the scope of this paper.

D.2 Greedy Algorithm for Optimal Resource Allocation

We describe the procedure for identifying optimal resource allocations and establishing the Pareto frontier for sparse attention models in Algorithms 1 and 2, as a supplement to Section 4.1. Given a fixed cost constraint C , we perform a grid search over key parameters: KV budgets and either reasoning trials or maximum generation lengths.

Empirically, we sweep over KV budgets {32, 64, 128, 256, 512, 1024}; reasoning trials {1, 2, 4, 8, 16, 32} (with a reduced upper limit for the 14B and 32B models to save computation time); and generation lengths {2k, 4k, 6k, 8k, 10k, 12k, 14k, 16k, 18k, 20k, 22k, 24k, 26k, 28k, 30k, 32k}.

By varying the cost constraint C in Algorithms 1 and 2, we obtain the performance of sparse attention models under optimal resource allocation, as shown in Figures 8a to 8f and 10a to 10c.

It is important to note that we do not consider inter-request resource scheduling strategies, such as early stopping or dynamic reallocation across requests [26], since we aim to ensure fairness across all inputs. Instead, the cost constraint C is interpreted as the maximum allowable cost per request (not the average), even if some requests achieve saturated accuracy below that threshold.

D.3 Top- K Attention and Block Top- K Attention

In this section, we explain the sparse attention algorithms discussed in the main paper, namely *Top- K Attention* and *Block Top- K Attention*.

During the decoding phase of a large language model (LLM), the self-attention mechanism computes a weighted average of past values as follows:

$$o = \text{Softmax}\left(\frac{qK^\top}{\sqrt{d}}\right)V = wV, \quad q \in \mathbb{R}^{1 \times d}, \quad K, V \in \mathbb{R}^{n \times d}, \quad w \in \mathbb{R}^{1 \times n}, \tag{11}$$

where d is the head dimension and n is the context length. The key and value matrices are given by $K = [k_1, k_2, \dots, k_n]$, $V = [v_1, v_2, \dots, v_n]$, where each $k_i, v_i \in \mathbb{R}^{1 \times d}$ are cached from previous decoding steps.

Top- K Attention. Top- K Attention is a sparsification method where only the K most relevant tokens (i.e., those with the highest attention scores) are selected to compute the output. Formally, instead of computing the full softmax, we define a sparse attention weight vector:

$$w_i = \begin{cases} \frac{\exp(s_i)}{\sum_{j \in \mathcal{I}_K} \exp(s_j)} & \text{if } i \in \mathcal{I}_K, \\ 0 & \text{otherwise,} \end{cases} \quad \text{where } s_i = \frac{qk_i^\top}{\sqrt{d}}, \quad \mathcal{I}_K = \text{TopK}_K(s), \tag{12}$$

Here, \mathcal{I}_K denotes the indices of the top K attention scores s_i . By masking out the less important positions, this approach reduces the computational and memory cost of attention from $\mathcal{O}(n)$ to $\mathcal{O}(K)$, where $K \ll n$.

Algorithm 1: Best-of- N optimal resource allocation under cost C

Data: Tasks \mathcal{T} , KV budgets $\{B_1, \dots, B_j\}$, trial counts $\{N_1, \dots, N_i\}$, cost limit C

Result: Average of maximum accuracy per task under cost C

```
1 AccumBestAcc  $\leftarrow$  0 Count  $\leftarrow$  0;
2 for task  $T$  in  $\mathcal{T}$  do
3   for KV budget  $B_b$  do
4     Generate  $S \geq \max\{N_1, \dots, N_i\}$  responses using  $B_b$  for task  $T$ ;
5     for trial count  $N_a$  do
6       compute cost  $c_{b,a}^{(T)}$ ;
7       if  $c_{b,a}^{(T)} \leq C$  then
8         Compute accuracy  $\text{Acc}_{b,a}^{(T)} = \text{Pass}@N_a$ ;
9         if  $\text{Acc}_{b,a}^{(T)} > \text{BestAcc}$  then
10          BestAcc  $\leftarrow$   $\text{Acc}_{b,a}^{(T)}$ ;
11        end if
12      end if
13    end for
14  end for
15  AccumBestAcc  $+=$  BestAcc; Count  $+=$  1;
16 end for
17 AvgBestAcc = AccumBestAcc/Count;
18 return AvgBestAcc;
```

Algorithm 2: Long-CoTs optimal resource allocation under cost C

Data: Tasks \mathcal{T} , KV budgets $\{B_1, \dots, B_j\}$, gen. lengths $\{n_1, \dots, n_i\}$, samples S , cost limit C

Result: Average of maximum accuracy per task under cost C

```
1 AccumBestAcc  $\leftarrow$  0 Count  $\leftarrow$  0;
2 for task  $T$  in  $\mathcal{T}$  do
3   BestAcc  $\leftarrow$  0;
4   for gen. length  $n_a$  do
5     for KV budget  $B_b$  do
6       Generate  $S$  responses using  $(B_b, n_a)$ ; compute cost  $c_{b,a}^{(T)}$ ;
7       if  $c_{b,a}^{(T)} \leq C$  then
8         Compute accuracy  $\text{Acc}_{b,a}^{(T)} = \text{Pass}@1$ ;
9         if  $\text{Acc}_{b,a}^{(T)} > \text{BestAcc}$  then
10          BestAcc  $\leftarrow$   $\text{Acc}_{b,a}^{(T)}$ ;
11        end if
12      end if
13    end for
14  end for
15  AccumBestAcc  $+=$  BestAcc; Count  $+=$  1;
16 end for
17 AvgBestAcc = AccumBestAcc/Count;
18 return AvgBestAcc;
```

456 **Block Top- K .** Block Top- K Attention is a block-level sparse attention mechanism. Instead of
 457 selecting individual tokens based on attention scores, this method selects entire blocks of tokens,
 458 thereby reducing the number of attention computations.

459 Specifically, assume the full sequence of n keys is divided into $m = \frac{n}{\text{BLOCK_SIZE}}$ consecutive blocks,
 460 each of size BLOCK_SIZE:

$$K = [k_1, \dots, k_n] \rightarrow \{K_1, K_2, \dots, K_m\}, \quad K_i \in \mathbb{R}^{\text{BLOCK_SIZE} \times d}$$

461 For each block K_i , we first compute the average key vector:

$$\bar{k}_i = \frac{1}{\text{BLOCK_SIZE}} \sum_{j=1}^{\text{BLOCK_SIZE}} k_{i,j}$$

462 Next, we compute the attention score between the query q and each block’s average key:

$$s_i = \frac{q\bar{k}_i^\top}{\sqrt{d}}, \quad \text{for } i = 1, 2, \dots, m$$

463 We then select the top $K' = \frac{K}{\text{BLOCK_SIZE}}$ blocks based on the scores s_i , denoted by the index set
 464 $\mathcal{J}_{K'} = \text{TopK}_{K'}(s)$. Attention is computed only over the tokens within the selected blocks. The
 465 sparse attention weights are defined as:

$$w_i = \begin{cases} \frac{\exp(s_i)}{\sum_{j \in \mathcal{I}_K} \exp(s_j)} & \text{if } i \in \mathcal{I}_K \subseteq \text{tokens in selected blocks,} \\ 0 & \text{otherwise} \end{cases}$$

466 For both algorithms, K is the KV budget. For GQA, we conduct an average pooling across all the
 467 query heads in a group, ensuring that the total number of retrieved key-value vectors does not exceed
 468 the allocated KV budget.

469 E Extended Related Work

470 **Efficient Attention.** Sparse attention [44, 99, 6, 10, 101, 88, 96, 67, 11, 48, 5] has been comprehen-
 471 sively studied to reduce the attention cost when processing long sequences. In parallel, approaches
 472 like FlashAttention [16, 14] accelerate attention by maximizing hardware efficiency. To address the
 473 quadratic complexity of standard attention, researchers have also explored linear attention architec-
 474 tures [28, 29, 43, 12]. Additionally, quantization and low-precision methods [56, 34, 52] have been
 475 broadly applied for improving inference efficiency.

476 **Efficient Inference.** Orca [95], vLLM [46], and SGLang [102] are widely adopted to enhance
 477 the efficiency of LLM serving. Our analysis builds on the practical designs and implementations
 478 of these systems. In parallel, speculative decoding [47, 7, 61, 71] has been proposed to mitigate
 479 the memory-bandwidth bottleneck during LLM decoding. Additionally, model compression and
 480 offloading [19, 51, 77, 73, 25] techniques are playing a crucial role in democratizing LLM deployment.

481 **Efficient Test-time Strategies.** Optimizing reasoning models to generate fewer tokens has been
 482 shown to directly reduce inference-time cost [80, 2, 58]. Recent work such as CoCoNut [31] and
 483 CoCoMix [78] explores conducting reasoning in a latent space, thereby reducing decoding time.
 484 Methods like ParScale [9], Tree-of-Thoughts [93], and Skeleton-of-Thoughts [68] aim to improve
 485 efficiency by enabling parallel reasoning. Architectural innovations such as CoTFormer [63] further
 486 enhance efficiency by adaptively allocating computational resources across tokens. Efficient reward-
 487 model-based [87, 74, 76] test-time scaling algorithms are also comprehensively studied.

488 F Limitations, Future Scope, and Broader Impact

489 **Limitations.** Our experiments primarily focus on **Qwen3** [91] and **DeepSeek-R1-Distilled-**
 490 **Qwen** [30], two state-of-the-art pretrained reasoning model series, evaluated from the inference
 491 perspective. However, the effects of training and post-training strategies are not fully explored and
 492 may influence the performance gaps and robustness to sparse attention mechanisms. In addition,

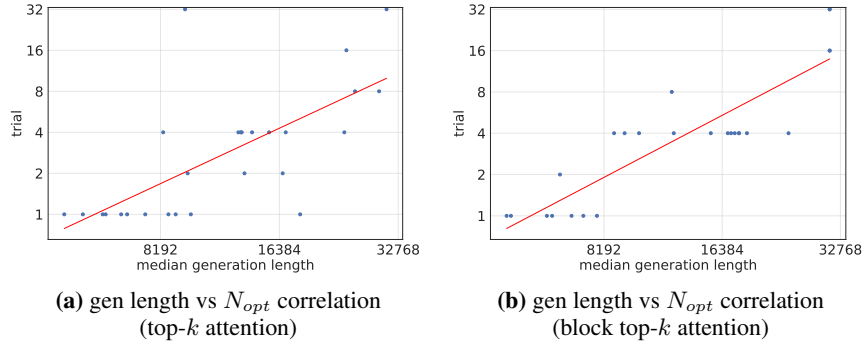


Figure 24: Correlation between Generation Length and Number of Trials. Longer generations correlate strongly with the optimal number of trials (N_{opt}), serving as a proxy for problem difficulty. (a) shows this trend for top- k and block top- k attention on the AIME24 dataset using the Qwen3-8B model.

our cost analysis assumes a cloud-based serving environment, where computational resources are typically sufficient and large batch sizes are feasible. In contrast, local deployment scenarios, such as those using Ollama⁶, often face limited VRAM where access to model parameters can dominate inference costs. Smaller models may be more appropriate in such settings, and our findings may not fully extend to these use cases.

Future Scope. Our sparse scaling law offers valuable insights for enriching the applications of sparse attention algorithms and the design space of test-time scaling strategies. On one hand, except for top- k , currently we only discuss a simple variant, i.e., block top- k , and have already demonstrated strong scalability. More advanced sparse attention algorithms [79, 10, 96, 50] are emerging these days. We do believe they can eventually push the scalability of test-time scaling to a much higher boundary. On the other hand, test-time scaling algorithms are proposed to adaptively allocate computation to tasks, or even to tokens [2, 63, 58, 57]. Extending them towards to new resource allocation problems in sparse attention is critical to reach the limit of Kinetics sparse scaling law. For instance, since generation length strongly correlates with the optimal number of trials under sparse attention (as shown in Figure 24), it can be used as a dynamic signal to adjust the number of trials and KV budget. Moreover, sparse attention drastically reduces inference cost, enabling more reasoning trials and longer generations. This unlocks greater flexibility in configuring TTS strategies within a fixed resource budget.

Broader Impact. This work aims to contribute to the understanding of efficiency and scalability challenges in the test-time scaling era, spanning model architecture, system-level implementation, and hardware design. We highlight the central role of sparsity in addressing these challenges. Our study is algorithmic in nature and does not target specific applications. While large language models can be misused in harmful ways, this work does not introduce new capabilities or risks beyond those already present in existing systems. Test-time scaling can consume a substantial amount of energy, raising concerns about the environmental sustainability of widespread deployment. By promoting sparse attention, our work hopes to help to reduce the carbon footprint and energy consumption of inference systems and support the broader goal of sustainable AI.

References

- [1] AI@Meta. Llama 4 model card. 2025. URL https://github.com/meta-llama/llama-models/blob/main/models/llama4/MODEL_CARD.md.
- [2] Daman Arora and Andrea Zanette. Training language models to reason efficiently, 2025. URL <https://arxiv.org/abs/2502.04463>.
- [3] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

⁶<https://github.com/ollama/ollama>

[4] Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

[5] Ruisi Cai, Yuandong Tian, Zhangyang Wang, and Beidi Chen. Lococo: Dropping in convolutions for long context compression. *arXiv preprint arXiv:2406.05317*, 2024.

[6] Beidi Chen, Tri Dao, Eric Winsor, Zhao Song, Atri Rudra, and Christopher Ré. Scatterbrain: Unifying sparse and low-rank attention. *Advances in Neural Information Processing Systems*, 34:17413–17426, 2021.

[7] 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.

[8] Kevin Chen, Marco Cusumano-Towner, Brody Huval, Aleksei Petrenko, Jackson Hamburger, Vladlen Koltun, and Philipp Krähenbühl. Reinforcement learning for long-horizon interactive llm agents. *arXiv preprint arXiv:2502.01600*, 2025.

[9] Mouxian Chen, Binyuan Hui, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Jianling Sun, Junyang Lin, and Zhongxin Liu. Parallel scaling law for language models. *arXiv preprint arXiv:2505.10475*, 2025. URL <https://arxiv.org/abs/2505.10475>.

[10] Zhuoming Chen, Ranajoy Sadhukhan, Zihao Ye, Yang Zhou, Jianyu Zhang, Niklas Nolte, Yuandong Tian, Matthijs Douze, Leon Bottou, Zhihao Jia, et al. Magicpig: Lsh sampling for efficient llm generation. *arXiv preprint arXiv:2410.16179*, 2024.

[11] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[12] Krzysztof Choromanski, Valerii Likhoshesterov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

[13] Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.

[14] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *CoRR*, abs/2307.08691, 2023. doi: 10.48550/ARXIV.2307.08691. URL <https://doi.org/10.48550/arXiv.2307.08691>.

[15] Tri Dao, Beidi Chen, Kaizhao Liang, Jiaming Yang, Zhao Song, Atri Rudra, and Christopher Ré. Pixelated butterfly: Simple and efficient sparse training for neural network models. In *International Conference on Learning Representations (ICLR)*, 2021. URL <https://arxiv.org/abs/2112.00029>.

[16] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[17] Giannis Daras, Nikita Kitaev, Augustus Odena, and Alexandros G Dimakis. Smyrf: Efficient attention using asymmetric clustering. *arXiv preprint arXiv:2010.05315*, 2020.

[18] DeepSeek-AI. Deepseek open infra index. 2025. URL https://github.com/deepseek-ai/open-infra-index/blob/main/2025020openSourceWeek/day_6_one_more_thing_deepseekV3R1_inference_system_overview.md.

[19] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35:30318–30332, 2022.

- [20] Danny Driess, Minh Nguyen, Fei Xia, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.
- [21] Nan Du, Yanping Huang, Andrew M. Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Dehao Chen, Yonghui Wu, and Jeff Dean. Glam: Efficient scaling of language models with mixture-of-experts. *arXiv preprint arXiv:2112.06905*, 2021.
- [22] 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(1):5232–5270, 2022.
- [23] Xidong Feng, Ziyu Wan, Muning Wen, Stephen Marcus McAleer, Ying Wen, Weinan Zhang, and Jun Wang. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*, 2023.
- [24] Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR, 2023.
- [25] Elias Frantar, Saleh Ashkboos, Torsten Hoeffler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.
- [26] Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Aurick Qiao, and Hao Zhang. Efficiently serving llm reasoning programs with certaintindex. *arXiv preprint arXiv:2412.20993*, 2024.
- [27] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.
- [28] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *CoRR*, abs/2312.00752, 2023. doi: 10.48550/ARXIV.2312.00752. URL <https://doi.org/10.48550/arXiv.2312.00752>.
- [29] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL <https://openreview.net/forum?id=uYLFoz1v1AC>.
- [30] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.
- [31] Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.
- [32] Torsten Hoeffler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *Journal of Machine Learning Research*, 22(241):1–124, 2021.
- [33] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.
- [34] Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. Kvquant: Towards 10 million context length llm inference with kv cache quantization. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 1270–1303. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/028fcbcf85435d39a40c4d61b42c99a4-Paper-Conference.pdf.

- [35] Junhao Hu, Wenrui Huang, Weidong Wang, Zhenwen Li, Tiancheng Hu, Zhixia Liu, Xusheng Chen, Tao Xie, and Yizhou Shan. Efficient long-decoding inference with reasoning-aware attention sparsity. *arXiv preprint arXiv:2502.11147*, 2025.
- [36] Wenlong Huang, Fei Fei, and Chelsea Finn. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *arXiv preprint arXiv:2201.07207*, 2022.
- [37] Sutskever Ilya. Ilya sutskever: “sequence to sequence learning with neural networks: what a decade”. URL <https://www.youtube.com/watch?v=1yvBqasHLZs>.
- [38] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.
- [39] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.
- [40] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.
- [41] Jordan Juravsky, Bradley Brown, Ryan Ehrlich, Daniel Y Fu, Christopher Ré, and Azalia Mirhoseini. Hydragen: High-throughput llm inference with shared prefixes. *arXiv preprint arXiv:2402.05099*, 2024.
- [42] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.
- [43] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5156–5165. PMLR, 2020. URL <http://proceedings.mlr.press/v119/katharopoulos20a.html>.
- [44] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *The International Conference on Machine Learning (ICML)*, 2020.
- [45] Tanishq Kumar, Zachary Ankner, Benjamin F Spector, Blake Bordelon, Niklas Muennighoff, Mansheej Paul, Cengiz Pehlevan, Christopher Ré, and Aditi Raghunathan. Scaling laws for precision. *arXiv preprint arXiv:2411.04330*, 2024.
- [46] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention, 2023. URL <https://arxiv.org/abs/2309.06180>.
- [47] Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR, 2023.
- [48] Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation, 2024. URL <https://arxiv.org/abs/2404.14469>.
- [49] Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, Omri Abend, Raz Alon, Tomer Asida, Amir Bergman, Roman Glozman, Michael Gokhman, Avashalom Manevich, Nir Ratner, Noam Rozen, Erez Shwartz, Mor Zusman, and Yoav Shoham. Jamba: A hybrid transformer-mamba language model, 2024. URL <https://arxiv.org/abs/2403.19887>.

- [50] Chaofan Lin, Jiaming Tang, Shuo Yang, Hanshuo Wang, Tian Tang, Boyu Tian, Ion Stoica, Song Han, and Mingyu Gao. Twilight: Adaptive attention sparsity with hierarchical top- p pruning. *arXiv preprint arXiv:2502.02770*, 2025.
- [51] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100, 2024.
- [52] Yujun Lin, Haotian Tang, Shang Yang, Zhekai Zhang, Guangxuan Xiao, Chuang Gan, and Song Han. Qserve: W4a8kv4 quantization and system co-design for efficient llm serving. *arXiv preprint arXiv:2405.04532*, 2024.
- [53] Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Daya Guo, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024.
- [54] Di Liu, Meng Chen, Baotong Lu, Huiqiang Jiang, Zhenhua Han, Qianxi Zhang, Qi Chen, Chengruidong Zhang, Bailu Ding, Kai Zhang, et al. Retrievalattention: Accelerating long-context llm inference via vector retrieval. *arXiv preprint arXiv:2409.10516*, 2024.
- [55] Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, et al. Deja vu: Contextual sparsity for efficient llms at inference time. In *International Conference on Machine Learning*, pages 22137–22176. PMLR, 2023.
- [56] Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750*, 2024.
- [57] Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. Reasoning models can be effective without thinking. *arXiv preprint arXiv:2504.09858*, 2025.
- [58] Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. Cot-valve: Length-compressible chain-of-thought tuning. *arXiv preprint arXiv:2502.09601*, 2025.
- [59] MAA. American invitational mathematics examination 2024, 2024. URL https://artofproblemsolving.com/wiki/index.php/American_Invitational_Mathematics_Examination?srltid=AfmB0oqiDCiaGTLQrsRTKsZui8RFnj0ZqM4qIqY3yGB3sBaq0axwf_Xt.
- [60] MAA. American invitational mathematics examination 2025, 2025. URL https://artofproblemsolving.com/wiki/index.php/American_Invitational_Mathematics_Examination?srltid=AfmB0oqiDCiaGTLQrsRTKsZui8RFnj0ZqM4qIqY3yGB3sBaq0axwf_Xt.
- [61] Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, et al. Specinfer: Accelerating generative large language model serving with tree-based speculative inference and verification. *arXiv preprint arXiv:2305.09781*, 2023.
- [62] Asit Mishra, Jorge Albericio Latorre, Jeff Pool, Darko Stosic, Dusan Stosic, Ganesh Venkatesh, Chong Yu, and Paulius Micikevicius. Accelerating sparse deep neural networks. *arXiv preprint arXiv:2104.08378*, 2021.
- [63] Amirkeivan Mohtashami, Matteo Pagliardini, and Martin Jaggi. Cotformer: A chain-of-thought driven architecture with budget-adaptive computation cost at inference. *arXiv preprint arXiv:2310.10845*, 2023.
- [64] Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variational dropout sparsifies deep neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2498–2507. PMLR, 2017.

- [65] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.
- [66] Reiichiro Nakano, Jacob Hilton, Jeffrey Wu, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.
- [67] Piotr Nawrot, Robert Li, Renjie Huang, Sebastian Ruder, Kelly Marchisio, and Edoardo M Ponti. The sparse frontier: Sparse attention trade-offs in transformer llms. *arXiv preprint arXiv:2504.17768*, 2025.
- [68] Xuefei Ning, Zinan Lin, Zixuan Zhou, Zifu Wang, Huazhong Yang, and Yu Wang. Skeleton-of-thought: Prompting llms for efficient parallel generation. *arXiv preprint arXiv:2307.15337*, 2023.
- [69] Daniele Paliotta, Junxiong Wang, Matteo Pagliardini, Kevin Y. Li, Aviv Bick, J. Zico Kolter, Albert Gu, François Fleuret, and Tri Dao. Thinking slow, fast: Scaling inference compute with distilled reasoners, 2025. URL <https://arxiv.org/abs/2502.20339>.
- [70] Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. *Proceedings of Machine Learning and Systems*, 5:606–624, 2023.
- [71] 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. *arXiv preprint arXiv:2408.11049*, 2024.
- [72] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarsz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.
- [73] Ying Sheng, Lianmin Zheng, Binhang Yuan, Zhuohan Li, Max Ryabinin, Beidi Chen, Percy Liang, Christopher Ré, Ion Stoica, and Ce Zhang. Flexgen: High-throughput generative inference of large language models with a single gpu. In *International Conference on Machine Learning*, pages 31094–31116. PMLR, 2023.
- [74] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.
- [75] Hanshi Sun, Zhuoming Chen, Xinyu Yang, Yuandong Tian, and Beidi Chen. Triforce: Lossless acceleration of long sequence generation with hierarchical speculative decoding, 2024. URL <https://arxiv.org/abs/2404.11912>.
- [76] Hanshi Sun, Momin Haider, Ruiqi Zhang, Huitao Yang, Jiahao Qiu, Ming Yin, Mengdi Wang, Peter Bartlett, and Andrea Zanette. Fast best-of-n decoding via speculative rejection. *arXiv preprint arXiv:2410.20290*, 2024.
- [77] Ruslan Svirschevski, Avner May, Zhuoming Chen, Beidi Chen, Zhihao Jia, and Max Ryabinin. Specexec: Massively parallel speculative decoding for interactive llm inference on consumer devices. *Advances in Neural Information Processing Systems*, 37:16342–16368, 2024.
- [78] Jihoon Tack, Jack Lanchantin, Jane Yu, Andrew Cohen, Ilia Kulikov, Janice Lan, Shibo Hao, Yuandong Tian, Jason Weston, and Xian Li. Llm pretraining with continuous concepts. *arXiv preprint arXiv:2502.08524*, 2025.
- [79] Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context llm inference. *arXiv preprint arXiv:2406.10774*, 2024.
- [80] NovaSky Team. Sky-t1: Train your own o1 preview model within \$450. <https://novasky-ai.github.io/posts/sky-t1>, 2025. Accessed: 2025-01-09.

- [81] Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025. URL <https://qwenlm.github.io/blog/qwq-32b/>.
- [82] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.
- [83] Ajay Tirumala and Raymond Wong. Nvidia blackwell platform: Advancing generative ai and accelerated computing. In *2024 IEEE Hot Chips 36 Symposium (HCS)*, pages 1–33. IEEE Computer Society, 2024.
- [84] Junxiong Wang, Wen-Ding Li, Daniele Paliotta, Daniel Ritter, Alexander M. Rush, and Tri Dao. M1: Towards scalable test-time compute with mamba reasoning models, 2025. URL <https://arxiv.org/abs/2504.10449>.
- [85] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.
- [86] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.
- [87] Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*, 2024.
- [88] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks, 2024. URL <https://arxiv.org/abs/2309.17453>.
- [89] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.
- [90] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.
- [91] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL <https://arxiv.org/abs/2505.09388>.
- [92] Shinn Yao, Jiaming Zhao, Dian Yu, et al. React: Synergizing reasoning and acting in language models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
- [93] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

- [94] Zihao Ye, Lequn Chen, Ruihang Lai, Wuwei Lin, Yineng Zhang, Stephanie Wang, Tianqi Chen, Baris Kasikci, Vinod Grover, Arvind Krishnamurthy, and Luis Ceze. Flashinfer: Efficient and customizable attention engine for llm inference serving. *arXiv preprint arXiv:2501.01005*, 2025. URL <https://arxiv.org/abs/2501.01005>.
- [95] Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. Orca: A distributed serving system for Transformer-Based generative models. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, pages 521–538, Carlsbad, CA, July 2022. USENIX Association. ISBN 978-1-939133-28-1. URL <https://www.usenix.org/conference/osdi22/presentation/yu>.
- [96] Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, YX Wei, Lean Wang, Zhiping Xiao, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. *arXiv preprint arXiv:2502.11089*, 2025.
- [97] Zhihang Yuan, Yuzhang Shang, Yang Zhou, Zhen Dong, Zhe Zhou, Chenhao Xue, Bingzhe Wu, Zhikai Li, Qingyi Gu, Yong Jae Lee, et al. Llm inference unveiled: Survey and roofline model insights. *arXiv preprint arXiv:2402.16363*, 2024.
- [98] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33, 2020.
- [99] Amir Zandieh, Insu Han, Majid Daliri, and Amin Karbasi. Kdeformer: Accelerating transformers via kernel density estimation. In *International Conference on Machine Learning*, pages 40605–40623. PMLR, 2023.
- [100] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.
- [101] Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710, 2023.
- [102] Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Livia Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. Sglang: Efficient execution of structured language model programs. *Advances in Neural Information Processing Systems*, 37:62557–62583, 2024.
- [103] Kan Zhu, Yilong Zhao, Liangyu Zhao, Gefei Zuo, Yile Gu, Dedong Xie, Yufei Gao, Qinyu Xu, Tian Tang, Zihao Ye, et al. Nanoflow: Towards optimal large language model serving throughput. *arXiv preprint arXiv:2408.12757*, 2024.