BUDGET-AWARE LAYER PRUNING FOR LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Layer pruning is attractive for large language model deployment because it yields practical gains in latency and throughput. The key challenge is selecting which layers to remove under a pruning budget K, where K denotes the number of layers to remove. Many existing methods use a single, budget-independent ranking and select the top-K layers, imposing a nested-prefix constraint across pruning budgets. Through exhaustive search on LLaMA3.1-8B, we show that optimal pruning sets can be non-nested across budgets, exposing a limitation of fixed global rankings. Motivated by this finding, we shift layer scoring from intrinsic importance to a budget-aware continuation value and formulate layer pruning as a budget-conditioned finite-horizon Markov decision process. We approximate this value using Double Dueling DQN, enabling a single scorer to construct pruning sets conditioned on both the target budget and the current pruning context. Experiments demonstrate better language modeling quality and downstream accuracy than strong depth-pruning baselines at the same pruning budget, and greater robustness under aggressive pruning. Code is available at `https://anonymous.4open.science/r/Budget-Aware-Layer-Pruning-B2CD`.

1 INTRODUCTION

Layer pruning is attractive for LLM deployment because it gives real wall-clock speedup and throughput gains (Kim et al., 2024; Men et al., 2024; Song et al., 2024; Ding et al., 2026). The key bottleneck is deciding which layers to remove under a given pruning budget. Existing methods usually reduce this decision to a single global layer ranking (Men et al., 2024; Kim et al., 2024; Ding et al., 2026) or iterative one-step deletion (Song et al., 2024).

Here the pruning budget K means how many layers we remove. This matters because pruning at small K and large K requires different trade-offs. In many existing methods, one global ordering is computed without explicit conditioning on K (Men et al., 2024; Kim et al., 2024) and then reused for all budgets. Reusing one ordering π gives $\mathcal{P}_K = \{\pi_1, \ldots, \pi_K\}$, which enforces a hidden *nested constraint* $\mathcal{P}_{K_1} \subseteq \mathcal{P}_{K_2}$ for $K_1 \leq K_2$. At the same time, many methods select layers by immediate local effect (Men et al., 2024; Kim et al., 2024; Song et al., 2024), without explicitly modeling how current choices change later choices.

Figure 1 shows why this is a real bottleneck. Through exhaustive search on LLaMA3.1-8B for $K \in \{1, 2, 3, 4\}$, we find non-nested optima, $\mathcal{P}_3^* = \{10, 25, 26\}$ and $\mathcal{P}_4^* = \{11, 12, 25, 26\}$. No budget-independent ranking can represent both sets simultaneously. This motivates scoring each layer under both the current pruning context and the target budget. We quantify the constraint loss and the gap to the exhaustive-search reference in Section 4.5.

Based on this observation, we redefine layer scoring from intrinsic importance to a budget-aware continuation value. The score $q_\theta(s_t, a; K)$ should reflect not only immediate damage but also downstream pruning consequences under budget K. To give this continuation value a principled training target, we formulate pruning-set construction as a budget-conditioned finite-horizon MDP. Under a telescoping reward and $\gamma = 1$, the return equals the negative degradation of the final pruned model, so the optimal action-value function $Q^*(s_t, a; K)$ is exactly aligned with the pruning objective. This exact alignment, together with joint multi-budget training, allows one shared scorer to be learned and reused across pruning stages and budgets.

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.

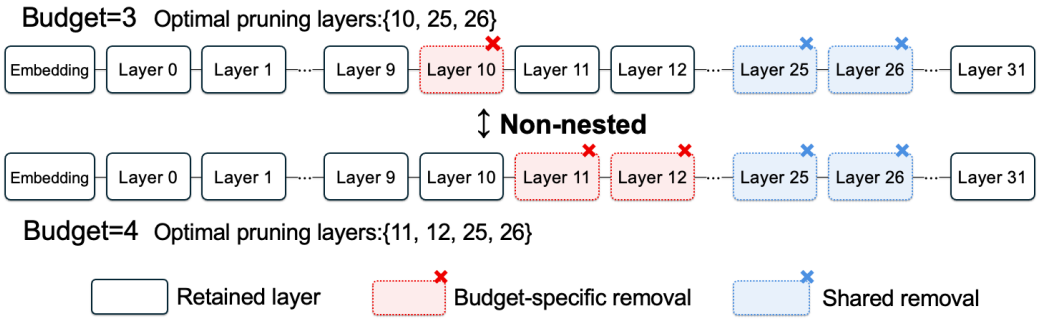

Figure 1: Optimal pruning sets on LLaMA3.1-8B obtained by exhaustive search using the BookCorpus calibration set. The optima for $K = 3$ and $K = 4$ are non-nested, showing that optimal pruning sets can violate the nested-prefix structure imposed by fixed-ranking methods.

This formulation also motivates how we optimize the scorer in practice. At each step, the action is discrete (choose one layer), and we need to rank all feasible layers repeatedly under different budgets. A value-based method naturally provides this ranking signal and supports replay-based reuse across mixed-budget trajectories. We therefore use Double Dueling DQN (D3QN) (van Hasselt et al., 2016; Wang et al., 2016) to approximate this continuation value and generate the pruning set by deterministic rollout until K layers are removed.

Experiments on LLaMA and Qwen models show that budget-aware selection improves language modeling quality most consistently at larger pruning budgets, including on LLaMA3.1-70B. Evaluations on nine zero-shot benchmarks, GSM8K, and HumanEval reveal task- and budget-dependent advantages. Comparisons with exhaustive-search references assess pruning-set quality, while inference-time feature perturbations quantify how the trained scorer's layer selections and resulting perplexity depend on budget and state information. The cost analysis further characterizes the trade-off between pruning quality and offline computation.

The main contributions of this work are as follows:

- We provide exhaustive-search evidence that optimal layer-pruning sets can be non-nested across budgets. This finding reveals a structural limitation of budget-independent rankings, whose nested-prefix form cannot represent all optimal pruning sets.
- We propose a budget-aware contextual layer scorer that estimates continuation value rather than intrinsic importance. We formulate layer pruning as a budget-conditioned finite-horizon MDP with exact objective alignment, and use joint multi-budget training to learn one shared scorer across budgets. In implementation, we use Double Dueling DQN to approximate this score.
- We design a practical training setup that improves scorer reuse and stability across budgets, including mixed-budget replay and safety-constrained value updates. We evaluate the full method on language modeling and downstream tasks, and use exhaustive-search references, inference-time feature perturbations, and offline-cost measurements to assess selection quality, sensitivity to state, and computational trade-offs.

2 Related Work

Existing layer-pruning approaches for LLMs mainly differ in how they define layer importance. Early methods often rely on local redundancy or sensitivity signals, such as hidden-state similarity in ShortGPT (Men et al., 2024), iterative redundancy estimation in SLEB (Song et al., 2024), and magnitude-, Taylor-, or perplexity-based criteria in Shortened LLaMA (Kim et al., 2024). Other methods estimate sensitivity from training data, such as influence-function-based LayerIF (Askari et al., 2025). More recent work considers broader layer interactions, including search over pruning masks with calibration metrics in EvoPress (Sieberling et al., 2025), neighboring-layer redundancy in LaCO (Yang et al., 2024) and MKA (Liu et al., 2025), and coalition-level contributions estimated with Shapley-style values (Ding et al., 2026).

Overall, prior scoring rules draw on a broad range of signals, including representation change, parameter magnitude, output-distribution shift, objective sensitivity, and coalition-level contribution. Most of these scores are ultimately used to produce a budget-independent ranking: once a ranking or pruning order is obtained, different pruning counts are handled by truncating the same order.

3 METHOD

3.1 PROBLEM FORMULATION

We formulate layer pruning (i.e., pruning whole Transformer layers) as budget-aware subset optimization and derive an exactly aligned sequential MDP as its learning surrogate. This formulation, combined with joint multi-budget optimization, supports learning one shared scorer $q_\theta(s, a; K)$ across pruning stages and budgets.

3.1.1 GENERAL FORMULATION

Let $\mathcal{M}_0$ be a pre-trained LLM with N Transformer layers $\{l_0, l_1, \ldots, l_{N-1}\}$ (0-based indexing, used for all reported layer sets). Let $\mathcal{D}(\mathcal{M}, \mathcal{M}_0)$ measure degradation relative to $\mathcal{M}_0$. Given budget K, we seek a layer set $\mathcal{P}$ with $|\mathcal{P}| = K$ that minimizes degradation:

$$\mathcal{P}^* = \arg \min_{\substack{\mathcal{P} \subseteq \{0,\ldots,N-1\} \\ |\mathcal{P}|=K}} \mathcal{D}(\mathcal{M}_0 \setminus \{l_i\}_{i\in\mathcal{P}},\, \mathcal{M}_0)\,. \tag{1}$$

Layer pruning is not a fixed-ranking problem. The best layer to remove depends on budget K and on jointly removed layers. Since the final model is determined by set $\mathcal{P}$ rather than removal order, pruning is a combinatorial subset-selection problem with search space $\binom{N}{K}$ (about 10^{10} for N=36, K=18). One-shot ranking is therefore structurally limited, because it assumes additive layer effects while LLM layers interact through residual and attention pathways.

The effect of removing a layer can depend on the layers already removed. Writing degradation as a function of the pruned set, the marginal degradation is

$$\Delta_i(S) = D(S \cup \{i\}) - D(S), \tag{2}$$

where S is the already-pruned set. If this marginal effect changes with S, a layer that is safe to remove at a small budget may become harmful when removed together with other layers at a larger budget. This dependence motivates evaluating a candidate under its current pruning context and target budget. Appendix E gives the corresponding pairwise-interaction expression.

To make this subset optimization learnable, we use a sequential computational form with horizon K. At step t, the policy observes state s_t built from current model $\mathcal{M}_t$ and pruning context (Section 3.2), selects one feasible layer,

$$\begin{aligned} \mathcal{A}_t^{\text{feasible}} &= \mathcal{A}_t \cap \{\, i \mid M_{\text{safe}}(i) = 1 \,\}, \\ \mathcal{A}_t &= \{0, \ldots, N-1\} \setminus \mathcal{P}_t, \end{aligned} \tag{3}$$

and transitions deterministically to $\mathcal{M}_{t+1}$ after removing l_{a_t}. Here M_{safe} is a static safety mask from one-pass pre-screening of catastrophic layers, shared by training and inference.

The reward uses telescoping degradation differences, $r_t = -\big[\mathcal{D}(\mathcal{M}_{t+1}, \mathcal{M}_0) - \mathcal{D}(\mathcal{M}_t, \mathcal{M}_0)\big]$, where $\mathcal{D}$ can be task-specific (e.g., log-perplexity, KL divergence, or accuracy loss). We set $\gamma = 1$ in this finite-horizon setting to preserve exact alignment with subset optimization.

Set–Sequence Equivalence. Under $\gamma = 1$ and the telescoping reward, the return collapses to

$$\begin{aligned} \sum_{t=0}^{K-1} r_t &= -\big[\mathcal{D}(\mathcal{M}_K, \mathcal{M}_0) - \mathcal{D}(\mathcal{M}_0, \mathcal{M}_0)\big] \\ &= -\mathcal{D}(\mathcal{M}_K, \mathcal{M}_0). \end{aligned} \tag{4}$$

Hence any two trajectories with the same final set $\mathcal{P}$ have identical total return. The sequential form is therefore an exact surrogate of the subset objective, not an order-biased heuristic. In contrast, $\gamma < 1$ down-weights late rewards and can favor specific orders even when the final subset is fixed.

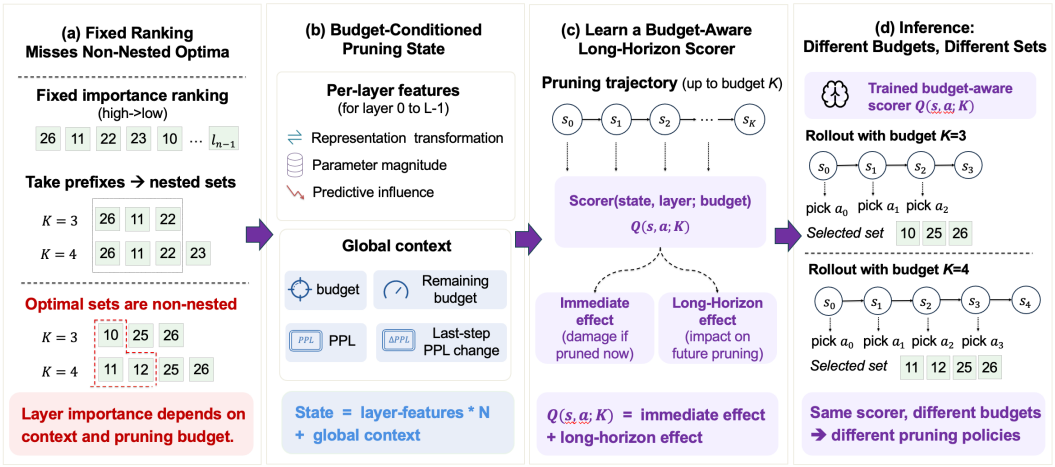

Figure 2: Overview of our budget-aware layer pruning framework. Unlike fixed-ranking methods, which take prefixes of a single pruning order and therefore impose nested pruning sets (a), our method builds a budget-conditioned pruning state (b) and learns a long-horizon scorer $Q(s, a; K)$ (c). The scorer evaluates each candidate layer under both the current pruning context and budget, enabling different, potentially non-nested pruning sets for different budgets at inference time (d).

3.1.2 Instantiation in This Work

We use $\mathcal{D} = \log \mathrm{PPL}_{\mathcal{D}_{\mathrm{cal}}}$ on held-out calibration data, which gives $r_t = -\log(\mathrm{PPL}_{t+1}/\mathrm{PPL}_t)$ and is invariant to absolute perplexity scale.

3.2 State Design

Our goal is a compact state that is both Markovian and budget-aware. In implementation, we maintain a fixed per-layer feature tensor of shape $N \times d$ at every step and never change its shape during pruning. For each layer, we use a masked feature $\tilde{\mathbf{f}}_{i,t} = [m_{i,t}\mathbf{f}_{i,t}; m_{i,t}]$ with $m_{i,t} \in \{0, 1\}$, where $m_{i,t} = 1$ means layer i is alive and $m_{i,t} = 0$ means layer i is pruned. The full state is then

$$\begin{aligned} s_t &= [\tilde{\mathbf{f}}_{1,t}; \ldots; \tilde{\mathbf{f}}_{N,t}; \mathbf{m}_t], \\ s_t &\in \mathbb{R}^{N(d+1)+d_{\mathrm{meta}}}. \end{aligned} \tag{5}$$

This slot-preserving design keeps input/output interfaces fixed, so one shared scorer can be trained across pruning steps and budgets rather than requiring budget-specific or step-specific networks.

(i) Per-layer structural features. For each layer l_i, we extract $\mathbf{f}_i \in \mathbb{R}^d$ with $d = 7$:

- **Representation transformation:** cosine similarity $\cos(\mathbf{h}_i^{\mathrm{in}}, \mathbf{h}_i^{\mathrm{out}})$, residual ratio $\|\mathbf{h}_i^{\mathrm{out}} - \mathbf{h}_i^{\mathrm{in}}\|/\|\mathbf{h}_i^{\mathrm{in}}\|$, and output–input norm ratio $\|\mathbf{h}_i^{\mathrm{out}}\|/\|\mathbf{h}_i^{\mathrm{in}}\|$.
- **Parameter magnitude:** L_1 and L_2 norms of layer weights.
- **Predictive influence:** KL divergence between model outputs with/without l_i on calibration tokens, plus entropy shift ΔH_i.

Because all features are computed from the current model $\mathcal{M}_t$, transitions depend only on $(\mathcal{M}_t, a_t)$; the Markov property follows directly.

(ii) Global context features. $\mathbf{m}_t \in \mathbb{R}^{d_{\mathrm{meta}}}$ with $d_{\mathrm{meta}} = 4$ summarizes trajectory and budget: normalized progress $p_t = t/K$, normalized current perplexity $\widetilde{\mathrm{PPL}}_t$, per-step perplexity change $\Delta\mathrm{PPL}_t$, and budget ratio K/N. Including K/N allows one policy to handle multiple budgets.

Per-layer features estimate each layer's current functional role, while global features disambiguate similar local patterns under different pruning stages and budgets. This combination is necessary for consistent decisions across multi-budget episodes, especially at larger budgets.

3.3 BUDGET-AWARE LAYER SCORER

Most ranking methods treat layer importance as intrinsic and budget-agnostic, then obtain different budgets by truncating one global ordering. We instead learn a budget-aware score under the current pruning state and directly use $q_\theta(s_t, a; K)$ as the score for feasible action $a \in \mathcal{A}_t^{\text{feasible}}$. Budget K is encoded in s_t through global context features, and we write K explicitly to emphasize budget-aware conditioning. We use lowercase q_θ for the learned scorer and uppercase Q^* for its optimal target. The score is therefore not intrinsic layer importance, but a budget-aware value under current state.

Multi-Budget Joint Training. Each training episode samples a budget K from $\mathcal{K}$, and a single network q_θ is optimized over transitions from all budgets (Section 3.4), so that its scores are comparable across budgets.

We use a value-based formulation because the pruning action is discrete (choose one layer index), and at each step we need scores for all feasible layers under the current state and budget. A single forward pass of $q_\theta(s_t, \cdot; K)$ provides exactly this ranking signal, and replay over mixed-budget trajectories yields high sample efficiency through repeated cross-budget reuse. In implementation, Double Dueling DQN (van Hasselt et al., 2016; Wang et al., 2016) approximates Q^* with $q_\theta : \mathcal{S} \times \mathcal{A} \times \mathcal{K} \to \mathbb{R}$ and $q_\theta(s_t, a; K) \approx Q^*(s_t, a; K)$, where $\mathcal{K} = \{K_{\min}, \ldots, K_{\max}\}$.

Double DQN reduces overestimation bias by decoupling action selection and target evaluation: the online network selects the best next feasible action, and the target network evaluates that action when forming the TD target.

Bellman View and Relation to Prior Methods. Each score combines the immediate degradation change with the best feasible continuation value under the same budget, so the greedy action optimizes both current and downstream pruning quality. This view also clarifies prior methods: static-score approaches (Men et al., 2024) compute one ranking at s_0 and ignore state evolution, while iterative greedy methods (Song et al., 2024) re-score each step but optimize only the immediate reward, effectively removing lookahead.

Following the dueling architecture, q_θ is represented by a context-dependent state-value term plus a centered action-advantage term. This realizes budget-aware scoring, where the score of each candidate layer depends jointly on current context and budget.

Inference. At inference, generating a pruning set for budget K is a deterministic rollout. We start from s_0, repeatedly remove the highest-scoring feasible layer under $q_\theta(s_t, \cdot; K)$, update state, and stop after K steps. This yields $\mathcal{P}_K$. Because the scorer is budget-aware, the same trained network outputs different pruning sets for different K without retraining.

3.4 PRACTICAL TRAINING DESIGN

To make the method practical to use, we design the training procedure as part of the method itself rather than as post-hoc heuristics. It supports shared optimization across budgets, robust state estimation with limited calibration compute, and stable value learning at high pruning ratios.

First, **Shared Multi-Budget Optimization.** All transitions share one replay buffer, so experience collected at different K updates the same parameters:

$$\mathcal{L}(\theta) = \mathbb{E}_K \, \mathbb{E}_{(s,a,r,s') \sim \mathcal{B}_K} \left[\left(y_t - q_\theta(s, a; K) \right)^2 \right], \tag{6}$$

where $\mathcal{B}_K$ denotes transitions collected under budget K, and all $\mathcal{B}_K$ are merged into one shared replay buffer in implementation.

Second, **Robust State Estimation.** We use stochastic calibration probing to construct state features. Instead of fixing one deterministic probe, we sample calibration slices during training so that the scorer is exposed to mild state-estimation noise and learns decisions that are less sensitive to a

particular calibration draw. Let ξ_t denote a sampled calibration slice at step t with $\xi_t \sim p(\xi \mid \mathcal{D}_{\text{cal}})$ and $\mathbf{f}_{i,t} = \phi_i(\mathcal{M}_t; \xi_t)$, so the scorer is trained under a distribution of feature realizations rather than a single fixed estimate.

Third, **Safety-Constrained Value Learning.** We use safety-constrained value learning with feasible-action masking (the feasible-action definition in Section 3.1), perplexity clipping, and catastrophic-step early termination. Together these constraints suppress outlier TD targets from pathological pruning actions and keep optimization in a numerically stable regime. Concretely, we clip rewards as $\tilde{r}_t = \text{clip}(r_t, r_{\min}, r_{\max})$, set $a^\star_{t+1} = \arg\max_{a \in \mathcal{A}^{\text{feasible}}_{t+1}} q_\theta(s_{t+1}, a; K)$, and use

$$y_t = \tilde{r}_t + \gamma\, q_{\bar{\theta}}(s_{t+1}, a^\star_{t+1}; K). \tag{7}$$

We terminate an episode early if $\log \text{PPL}_{t+1} - \log \text{PPL}_t > \delta_{\text{stop}}$.

Additional implementation settings are reported in Appendix A.

4 Experiments

Model	Method	K=4		K=8		K=12		K=16	
		WT2	C4	WT2	C4	WT2	C4	WT2	C4
LLaMA3.1-8B	SLEB	9.37	14.34	15.08	22.77	133.02	57.28	241.03	174.07
	ShortGPT	13.14	19.68	3437.19	4853.45	93462.21	16309.41	2127.54	1956.60
	Shortened LLaMA	9.36	14.03	18.37	25.79	95.20	58.33	3173.21	1056.68
	Reassessing	9.74	15.34	19.04	27.05	184.94	136.59	338.16	260.22
	Prune&Comp	8.91	**13.94**	14.51	21.72	123.71	54.99	236.21	177.55
	BlockPruner	9.18	14.46	14.74	22.17	56.96	51.87	241.11	228.03
	Ours	**8.68**	14.03	**14.27**	**21.57**	**33.87**	**36.50**	**170.71**	**133.43**
Qwen3-8B	SLEB	10.40	17.50	18.09	**23.77**	35.82	47.47	84.56	119.05
	ShortGPT	29.43	31.32	494.82	232.54	1543.03	1373.41	10984.71	8457.11
	Shortened LLaMA	10.40	17.50	18.47	28.43	44.27	58.92	119.59	156.81
	Reassessing	11.02	18.20	22.08	28.10	61.63	74.91	139.77	182.38
	Prune&Comp	**10.19**	**16.96**	18.48	26.06	38.10	49.19	88.79	116.67
	BlockPruner	10.89	18.03	19.54	26.15	41.55	56.01	105.70	147.62
	Ours	10.83	17.40	**17.26**	25.55	**33.15**	**45.40**	**82.49**	**98.27**
Qwen3-14B	SLEB	8.92	15.19	12.93	**20.32**	23.26	34.47	55.91	71.02
	ShortGPT	14.94	21.32	55.35	65.89	97.98	112.58	1960.00	1541.00
	Shortened LLaMA	9.02	15.68	13.33	21.40	37.94	57.27	291.26	268.15
	Reassessing	9.28	15.95	16.22	23.76	41.21	54.36	91.01	113.04
	Prune&Comp	**8.79**	**14.89**	13.42	20.81	23.38	34.13	57.20	72.43
	BlockPruner	9.19	15.80	13.96	22.15	26.52	39.99	68.21	88.77
	Ours	8.92	15.19	**12.87**	20.41	**22.92**	**33.37**	**55.00**	**69.64**

Table 1: Perplexity on WikiText2 and C4 for the 8B/14B models. Results of Ours are averaged over 3 training seeds; standard deviations are reported in Table 8 (Appendix B).

4.1 Experimental Setup

The pruning budget K denotes the number of Transformer layers removed.

Language modeling. We measure perplexity on WikiText2 (WT2) (Merity et al., 2016) and C4 (Raffel et al., 2023) using LLaMA3.1-8B (Grattafiori et al., 2024), Qwen3-8B, and Qwen3-14B (Yang et al., 2025) at $K \in \{4, 8, 12, 16\}$, and LLaMA3.1-70B at $K \in \{10, 20, 30, 40\}$. The 8B/14B comparison includes SLEB (Song et al., 2024), Shortened LLaMA (Kim et al., 2024), ShortGPT (Men et al., 2024), Reassessing (Lu et al., 2026), Prune&Comp (Chen et al., 2026), and BlockPruner (Zhong et al., 2025); the 70B comparison includes the latter three methods.

Downstream tasks. We evaluate Qwen3-8B at $K \in \{4, 8, 12, 16\}$ on nine zero-shot benchmarks, reporting average accuracy against SLEB (Song et al., 2024), Shortened LLaMA (Kim et al., 2024), and ShortGPT (Men et al., 2024), with the averages of Reassessing, Prune&Comp, and Block-Pruner included in the same table. We also evaluate LLaMA3.1-8B on GSM8K using accuracy and

CodeQwen1.5-7B-Chat on HumanEval using Pass@1. For these two tasks, baseline results and pruning ratios follow Reassessing (Lu et al., 2026). Benchmark definitions, per-task zero-shot results, and evaluation notes are provided in Appendix C.

For our method and the SLEB, Shortened LLaMA, and ShortGPT implementations, we use a shared BookCorpus calibration set. We train one budget-conditioned scorer per base model and reuse it across the evaluated budgets. For the 8B/14B language-modeling results, we train the scorer with 3 random seeds and report the mean (Appendix B). Appendix A specifies the calibration protocol, architecture, and training configuration. Offline pruning cost, wall-clock inference acceleration, and compatibility with post-training quantization are reported in Appendices I, J, and K.

4.2 Language Modeling

Table 1 shows that our method achieves lower perplexity than all six baselines on both datasets for all three models at $K = 12$ and $K = 16$. For example, on LLaMA3.1-8B at $K = 12$, it reduces WT2/C4 perplexity to 33.87/36.50, compared with 56.96/51.87 for the strongest baseline, BlockPruner. At smaller budgets, the advantage is less uniform: at $K = 4$, Prune&Comp has lower perplexity on both datasets for both Qwen models and on C4 for LLaMA3.1-8B, SLEB and Shortened LLaMA have lower WT2 perplexity on Qwen3-8B, and SLEB ties our method on Qwen3-14B; at $K = 8$, SLEB has lower C4 perplexity on both Qwen models. The results therefore support the benefit of budget-aware selection most consistently under aggressive pruning.

4.3 Results on Larger Models

Model	Method	K=10		K=20		K=30		K=40	
		WT2	C4	WT2	C4	WT2	C4	WT2	C4
LLaMA3.1-70B	Ours	5.56	10.18	**10.93**	**19.07**	**41.62**	**67.74**	**132.85**	**188.39**
	Reassessing	6.31	11.58	20.36	35.71	131.64	192.37	512.76	744.21
	Prune&Comp	**5.39**	**9.91**	13.46	23.58	104.28	170.56	244.19	352.68
	BlockPruner	6.17	11.32	17.52	31.34	68.72	112.43	468.31	681.94

Table 2: Perplexity on WikiText2 and C4 for LLaMA3.1-70B.

The LLaMA3.1-70B results in Table 2 exhibit the same pattern. Our method achieves the lowest perplexity on both datasets at $K = 20, 30, 40$, while Prune&Comp performs best at $K = 10$. At $K = 40$, our method reaches 132.85/188.39 WT2/C4 perplexity, compared with 244.19/352.68 for Prune&Comp, the strongest baseline at this budget. These results extend the observed quality advantage at larger pruning budgets to a 70B model.

4.4 Downstream Evaluation

Zero-shot tasks. Table 3 reports average accuracy over nine benchmarks on Qwen3-8B, with per-task results deferred to Appendix C. Among all seven methods, our method has the highest average accuracy at $K = 8, 16$, nearly matches SLEB at $K = 12$ (0.4022 versus 0.4023), and ranks second to Reassessing at $K = 4$ (0.5714 versus 0.5858). At $K = 16$, it improves average accuracy by 1.91 percentage points over SLEB and 5.73 points over Shortened LLaMA.

Perplexity and downstream accuracy. Perplexity and downstream accuracy need not improve in parallel. On Qwen3-8B at $K = 12$, our method improves both WT2 and C4 perplexity over SLEB, yet their average zero-shot accuracies are nearly identical. Perplexity measures token-level predictive loss, whereas task accuracy measures correct task predictions; improvements in one do not imply proportional gains in the other. This observation is consistent with the discussion in Liu et al. (2023) that pre-training loss alone does not fully explain downstream performance.

Mathematical reasoning. On GSM8K, our method achieves 32.76% accuracy on LLaMA3.1-8B, improving over Reassessing by 3.57 percentage points at the same reported pruning rate of 25.00%

Method	K=4	K=8	K=12	K=16
SLEB	0.5595	0.4789	**0.4023**	0.3320
Shortened LLaMA	0.5595	0.4816	0.3912	0.2938
ShortGPT	0.5209	0.4001	0.3565	0.3196
Reassessing	**0.5858**	0.4780	0.3839	0.3268
Prune&Comp	0.5681	0.4727	0.3803	0.3183
BlockPruner	0.5455	0.4609	0.3723	0.2960
Ours	0.5714	**0.4826**	0.4022	**0.3511**

Table 3: Average accuracy over nine zero-shot benchmarks on Qwen3-8B across all seven methods.

(Table 4). It also exceeds the other listed results, whose pruning rates differ and are retained in the table.

Code generation. On HumanEval, our method achieves 44.92% Pass@1 with CodeQwen1.5-7B-Chat (Table 5). The baselines use the 20% pruning rate reported in Reassessing, whereas our method removes 7 of 32 layers (21.88%). It ranks second to Reassessing at 48.72% and exceeds the remaining baselines despite the higher pruning rate. Together, the downstream results show that the relative advantage depends on both the task and pruning budget.

Method	Pruning Rate	GSM8K Acc (%) ↑
ShortGPT	19.02%	0.60
LaCo	19.02%	0.40
SliceGPT	20.12%	0.00
2SSP	19.99%	4.30
COMPACT	20.00%	10.80
Reassessing	25.00%	29.19
Ours	25.00%	**32.76**

Table 4: GSM8K accuracy on LLaMA3.1-8B, with baselines and their pruning rates from Reassessing (Lu et al., 2026). Ours removes whole Transformer layers (8 of 32).

Method	HumanEval Pass@1 (%) ↑
ShortGPT	42.68
UIDL	0.00
Linearity	0.00
SLEB	20.73
LLM-Pruner	15.85
Reassessing	**48.72**
Ours	44.92

Table 5: HumanEval Pass@1 on CodeQwen1.5-7B-Chat, with baselines from Reassessing (Lu et al., 2026) at a 20% pruning rate. Ours removes 7 of 32 layers (21.88%).

4.5 Analysis of Budget-Aware Selection

To isolate the effect of the nested-prefix constraint, we compare independently optimized pruning sets with the best extension of a smaller-budget optimum. On LLaMA3.1-8B, exhaustive search yields $\{10, 25, 26\}$ at $K = 3$ and $\{11, 12, 25, 26\}$ at $K = 4$. A fixed ranking cannot represent both optima. If the $K = 4$ set must contain the $K = 3$ optimum, its best extension is $\{10, 19, 25, 26\}$.

Method / Set	WT2 PPL ↓	WT2 gap vs. reference
Exhaustive-search reference {11,12,25,26}	**8.6334**	–
Best nested alternative {10,19,25,26}	9.1850	+6.39%
Shortened LLaMA {10,11,25,26}	9.3649	+8.47%
Ours	8.6811	+0.55%

Table 6: LLaMA3.1-8B at $K = 4$: nested-constraint loss and method performance relative to the exhaustive-search reference.

Table 6 separates the cost of this constraint from the quality of the learned selection. Enforcing the nested extension increases WT2 perplexity from 8.6334 to 9.1850, a 6.39% relative increase. Our method reaches 8.6811, only 0.55% above the exhaustive-search reference, compared with 8.47% for Shortened LLaMA. Thus, preserving the optimum at one budget can restrict the choices available at another, while budget-conditioned selection can approach the reference quality in this case. Appendix E provides the LLaMA3.2-1B example; calibration ablations and budget ablations

are reported in Appendices G and H, and the inference-time sensitivity analysis is presented in Section 4.6.

4.6 Inference-Time Feature Sensitivity

We perturb one feature group at a time during deterministic rollout of the trained scorer, without retraining. Table 7 reports the resulting WT2 perplexity and the Jaccard overlap $|S_{\text{full}} \cap S_{\text{pert}}|/|S_{\text{full}} \cup S_{\text{pert}}|$ between the pruning sets selected with the full and perturbed states at $K = 8$ and $K = 16$.

	K=8			K=16		
Perturbed feature group	WT2 PPL	ΔWT2	Jaccard	WT2 PPL	ΔWT2	Jaccard
Full	17.26	0.00%	1.00	82.49	0.00%	1.00
Budget	18.31	6.08%	0.60	141.36	71.37%	0.33
Representation transformation	17.87	3.53%	0.78	100.56	21.91%	0.60
Parameter magnitude	17.46	1.16%	0.78	87.47	6.04%	0.78
Predictive influence	18.48	7.07%	0.60	123.13	49.27%	0.45
Global context	23.51	36.21%	0.33	260.61	215.93%	0.23

Table 7: Inference-time feature sensitivity on Qwen3-8B without retraining.

Among the feature groups, perturbing global context causes the largest degradation, increasing WT2 perplexity by 36.21% at $K = 8$ and 215.93% at $K = 16$, with the Jaccard overlap dropping to 0.33 and 0.23; predictive influence and budget conditioning also produce sizable degradations. Representation-transformation and parameter-magnitude features are comparatively stable. These observations support the use of all four feature groups together with budget conditioning in the final scorer.

5 Conclusion

In this work, we revisit layer pruning for large language models from the perspective of budget-conditioned subset selection. We show through exhaustive search that optimal pruning sets can be non-nested across different pruning budgets, revealing a structural limitation of budget-independent ranking methods. Motivated by this observation, we propose a budget-aware contextual layer scorer and formulate the pruning process as a finite-horizon MDP. Under a telescoping reward with $\gamma = 1$, the MDP return is aligned with the final pruning degradation, and the optimal action-value function provides a principled long-horizon scoring target. We approximate this scorer with D3QN and use it to construct pruning sets conditioned on both the current pruning context and the pruning budget. Experiments on LLaMA- and Qwen-family models, including LLaMA3.1-70B, show gains in language-modeling quality at larger pruning budgets. The downstream evaluations cover nine zero-shot benchmarks, mathematics, and code generation, with task- and budget-dependent advantages. The search comparisons and state-feature ablations provide evidence for the selection design, while the deployment experiments show wall-clock acceleration and compatibility with post-training quantization. These results suggest that explicitly modeling the pruning budget and context is important for effective layer pruning, and provide a practical framework for learning budget-aware pruning for LLM compression.

AI use statement

We used large language model assistants (GPT and Claude) as tools to support this work. Specifically, they were used for polishing English wording of the manuscript, checking grammar, and assisting with LaTeX formatting and the preparation of comparison tables. All technical content, including the research ideas, the formulation of budget-conditioned layer pruning, the experimental design, and the analysis of results, was conducted solely by the authors. All AI-assisted text was reviewed and revised by the authors, who take full responsibility for the content of this paper.

REFERENCES

Hadi Askari, Shivanshu Gupta, Fei Wang, Anshuman Chhabra, and Muhao Chen. LayerIF: Estimating layer quality for large language models using influence functions. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2505.23811.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language, 2019. URL https://arxiv.org/abs/1911.11641.

Xinrui Chen, Hongxing Zhang, Fanyi Zeng, Yongxian Wei, Yizhi Wang, Xitong Ling, Guanghao Li, and Chun Yuan. Prune&comp: Free lunch for layer-pruned LLMs via iterative pruning with magnitude compensation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, 2026. doi: 10.1609/aaai.v40i24.39120.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018. URL https://arxiv.org/abs/1803.05457.

Xuan Ding, Pengyu Tong, Ranjie Duan, Yunjian Zhang, Rui Sun, and Yao Zhu. Pruning as a cooperative game: Surrogate-assisted layer contribution estimation for large language models, 2026. URL https://arxiv.org/abs/2602.07804.

Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, Ahmed A. Aly, Beidi Chen, and Carole-Jean Wu. Layerskip: Enabling early exit inference and self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 12622–12642, 2024. doi: 10.18653/v1/2024.acl-long.681.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers, 2023. URL https://arxiv.org/abs/2210.17323.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie

Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz,

Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Manas Gupta, Siddharth Aravindan, Aleksandra Kalisz, Vijay Chandrasekhar, and Jie Lin. Learning to prune deep neural networks via reinforcement learning. *arXiv preprint arXiv:2007.04756*, 2020. URL https://arxiv.org/abs/2007.04756.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL https://arxiv.org/abs/2009.03300.

Bo-Kyeong Kim, Geonmin Kim, Tae-Ho Kim, Thibault Castells, Shinkook Choi, Junho Shin, and Hyoung-Kyu Song. Shortened LLaMA: Depth pruning for large language models with comparison of retraining methods. *arXiv preprint arXiv:2402.02834*, 2024. URL https://arxiv.org/abs/2402.02834.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations, 2017. URL https://arxiv.org/abs/1704.04683.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, 2012. URL https://www.aaai.org/ocs/index.php/KR/KR12/paper/view/4492.

Deyuan Liu, Zhanyue Qin, Hairu Wang, Zhao Yang, Zecheng Wang, Fangying Rong, Qingbin Liu, Yanchao Hao, Xi Chen, Cunhang Fan, Zhao Lv, Zhiying Tu, Dianhui Chu, Bo Li, and Dianbo Sui. Pruning via merging: Compressing llms via manifold alignment based layer merging, 2025. URL https://arxiv.org/abs/2406.16330.

Hong Liu, Sang Michael Xie, Zhiyuan Li, and Tengyu Ma. Same pre-training loss, better downstream: Implicit bias matters for language models. In *International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pp. 22188–22214, 2023.

Yao Lu, Hao Cheng, Yujie Fang, Zeyu Wang, Jiaheng Wei, Dongwei Xu, Qi Xuan, and Zhaowei Zhu. Reassessing layer pruning in LLMs: New insights and methods. In *International Conference on Learning Representations (ICLR)*, 2026.

Xuan Luo, Weizhi Wang, and Xifeng Yan. DiffSkip: Differential layer skipping in large language models. In *Findings of the Association for Computational Linguistics: ACL*, pp. 7221–7231, 2025. doi: 10.18653/v1/2025.findings-acl.377.

Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. Shortgpt: Layers in large language models are more redundant than you expect. *arXiv preprint arXiv:2403.03853*, 2024. URL https://arxiv.org/abs/2403.03853.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016. URL https://arxiv.org/abs/1609.07843.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering, 2018. URL https://arxiv.org/abs/1809.02789.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context, 2016. URL https://arxiv.org/abs/1606.06031.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2023. URL https://arxiv.org/abs/1910.10683.

Oliver Sieberling, Denis Kuznedelev, Eldar Kurtic, and Dan Alistarh. Evopress: Accurate dynamic model compression via evolutionary search, 2025. URL https://arxiv.org/abs/2410.14649.

Jiwon Song, Kyungseok Oh, Taesu Kim, Hyungjun Kim, Yulhwa Kim, and Jae-Joon Kim. SLEB: Streamlining LLMs through redundancy verification and elimination of transformer blocks. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024. URL https://arxiv.org/abs/2402.09025.

Yi-Lin Sung, Jaehong Yoon, and Mohit Bansal. ECoFLaP: Efficient coarse-to-fine layer-wise pruning for vision-language models. *arXiv preprint arXiv:2310.02998*, 2023. URL https://arxiv.org/abs/2310.02998. Revised 2024.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016. URL https://arxiv.org/abs/1509.06461.

Boyao Wang and Volodymyr Kindratenko. RL-Pruner: Structured pruning using reinforcement learning for CNN compression and acceleration. *arXiv preprint arXiv:2411.06463*, 2024. URL https://arxiv.org/abs/2411.06463.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, 2016. URL https://arxiv.org/abs/1511.06581.

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, Jiaxi 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.

Yifei Yang, Zouying Cao, and Hai Zhao. Laco: Large language model pruning via layer collapse, 2024. URL https://arxiv.org/abs/2402.11187.

Xin Yuan, Siqi Li, Jiateng Wei, Chengrui Zhu, Yanming Wu, Qingpeng Li, Jiajun Lv, Xiaoke Lan, Jun Chen, and Yong Liu. Fastforward pruning: Efficient LLM pruning via single-step reinforcement learning. *arXiv preprint arXiv:2511.18977*, 2025. URL https://arxiv.org/abs/2511.18977.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence?, 2019. URL https://arxiv.org/abs/1905.07830.

Longguang Zhong, Fanqi Wan, Ruijun Chen, Xiaojun Quan, and Liangzhi Li. Blockpruner: Fine-grained pruning for large language models. In *Findings of the Association for Computational Linguistics: ACL*, pp. 5065–5080, 2025. doi: 10.18653/v1/2025.findings-acl.262.

A Training and Calibration Details

Unless otherwise specified, we optimize the Q-network with Adam using a learning rate of 10^{-4} and a batch size of 64. The replay buffer capacity is set to 25K transitions for multi-budget training. The target network is synchronized every 100 gradient steps, and each training iteration performs 50 gradient updates. We use 4 parallel GPU workers, with each worker collecting one episode per iteration. Training runs for 800 iterations, taking approximately 2 GPU-h in total on 4×A100-80GB GPUs. Following common layer-pruning practice, we protect the first two layers and exclude layers $\{0, 1\}$ (0-based) from pruning.

B Seed Robustness

For the 8B/14B language-modeling comparison, we train the budget-conditioned scorer with 3 random seeds (0, 1, and 2) under the configuration in Appendix A. The seed controls Q-network initialization, ϵ-greedy exploration, replay sampling, and the order of calibration batches; the calibration set and evaluation data are fixed. Table 8 reports the mean and standard deviation, and the means are the values shown for Ours in Table 1. The variance is small at $K \leq 8$, where the selected pruning sets largely coincide across seeds, and grows with the budget as the continuation value becomes harder to estimate. On LLaMA3.1-8B at $K \geq 12$, the margin over the strongest baseline exceeds five standard deviations. On Qwen3-14B, the margins over SLEB and Prune&Comp at $K \geq 8$ lie within about one standard deviation, so these entries should be read as comparable rather than as clear improvements.

Model	K=4		K=8		K=12		K=16	
	WT2	C4	WT2	C4	WT2	C4	WT2	C4
LLaMA3.1-8B	8.68±0.04	14.03±0.06	14.27±0.11	21.57±0.09	33.87±1.62	36.50±1.28	170.71±9.84	133.43±6.71
Qwen3-8B	10.83±0.05	17.40±0.07	17.26±0.19	25.55±0.22	33.15±1.07	45.40±1.36	82.49±3.58	98.27±4.12
Qwen3-14B	8.92±0.03	15.19±0.05	12.87±0.08	20.41±0.11	22.92±0.61	33.37±0.84	55.00±1.93	69.64±2.27

Table 8: Perplexity of our method on WikiText2 and C4, reported as mean±standard deviation over 3 training seeds. The means are the values reported for Ours in Table 1.

C Additional Evaluation Details

Zero-shot benchmarks. The Qwen3-8B evaluation includes PIQA (Bisk et al., 2019), HellaSwag (Zellers et al., 2019), ARC-easy and ARC-challenge (Clark et al., 2018), OpenBookQA (Mihaylov et al., 2018), RACE (Lai et al., 2017), WSC273 (Levesque et al., 2012), LAMBADA (Paperno et al., 2016), and MMLU (Hendrycks et al., 2021). Table 9 reports all per-task accuracies and the nine-task averages summarized in Table 3.

Task-specific baseline results. The GSM8K and HumanEval comparisons in Tables 4 and 5 use the baseline results reported in Reassessing (Lu et al., 2026). GSM8K is evaluated on LLaMA3.1-8B with each method's reported pruning rate, while HumanEval uses CodeQwen1.5-7B-Chat with a 20% pruning rate for the baselines and 7 of 32 layers (21.88%) for our method. Appendix D provides the separate cross-study comparison with DiffSkip.

ShortGPT under aggressive pruning. ShortGPT produces very high perplexity at several large budgets. We checked the official scoring rule, pruning order, calibration set, and unpruned-model perplexity to reduce the possibility of implementation artifacts. Table 1 reports the raw perplexity values, and Table 9 provides the corresponding downstream evaluation on Qwen3-8B.

Relative WT2 differences against SLEB. Table 10 reports the relative WT2 perplexity reductions, computed as $(\mathrm{PPL}_{\mathrm{SLEB}} - \mathrm{PPL}_{\mathrm{ours}})/\mathrm{PPL}_{\mathrm{SLEB}}$. Positive values favor our method; the negative entry for Qwen3-8B at $K = 4$ indicates higher perplexity.

K	Method	PIQA	HeSw	ARC-e	ARC-c	OBQA	Race	WSC273	LAMBADA	MMLU	Avg acc
4	SLEB	0.7704	0.4849	0.7925	0.4898	0.2940	0.3923	0.7582	0.5030	0.5506	0.5595
	ShortGPT	0.6948	0.4737	0.6890	0.4326	0.2840	0.3703	0.7436	0.3194	0.6804	0.5209
	Shortened LLaMA	0.7704	0.4849	0.7925	0.4898	0.2940	0.3923	0.7582	0.5030	0.5506	0.5595
	Ours	0.7650	0.5054	0.7832	0.4829	0.2940	0.3971	0.7436	0.5542	0.6175	**0.5714**
8	SLEB	0.7285	0.4293	0.7235	0.3831	0.2640	0.3474	0.6996	0.3513	0.3838	0.4789
	ShortGPT	0.6208	0.4039	0.4861	0.3038	0.2180	0.3282	0.5568	0.1556	0.5281	0.4001
	Shortened LLaMA	0.7416	0.4494	0.7361	0.3951	0.2660	0.3531	0.6923	0.3450	0.3561	0.4816
	Ours	0.7296	0.4322	0.7226	0.4155	0.2760	0.3541	0.6703	0.3561	0.3873	**0.4826**
12	SLEB	0.6600	0.3653	0.6040	0.2969	0.2320	0.2957	0.6374	0.2435	0.2861	**0.4023**
	ShortGPT	0.6192	0.3724	0.4407	0.2782	0.2060	0.3120	0.5714	0.1003	0.3088	0.3565
	Shortened LLaMA	0.6621	0.3618	0.6094	0.2628	0.2060	0.3091	0.6630	0.1877	0.2585	0.3912
	Ours	0.6600	0.3758	0.6006	0.2978	0.2460	0.3043	0.6227	0.2399	0.2726	<u>0.4022</u>
16	SLEB	0.5958	0.3026	0.4588	0.2278	0.2060	0.2517	0.5568	0.1440	0.2444	0.3320
	ShortGPT	0.5876	0.2771	0.3519	0.2628	0.1800	0.2287	0.5971	0.0019	0.3894	0.3196
	Shortened LLaMA	0.5620	0.2917	0.3914	0.2056	0.1460	0.2488	0.5018	0.0508	0.2455	0.2938
	Ours	0.6164	0.3192	0.4634	0.2389	0.1880	0.2679	0.5934	0.2111	0.2617	**0.3511**

Table 9: Zero-shot accuracy on Qwen3-8B across nine benchmarks. Per-task accuracy and the nine-task average are shown at each budget.

Model	Layers	K=4	K=8	K=12	K=16
Qwen3-14B	40	+0.00%	+0.46%	+1.46%	+1.63%
Qwen3-8B	36	-4.13%	+4.59%	+7.45%	+2.45%
LLaMA3.1-8B	32	+7.36%	+5.37%	+74.54%	+29.17%

Table 10: Relative WT2 PPL reduction of our method compared with SLEB, together with each model's layer count.

D Additional Cross-Study Evaluation

Table 11 compares our method with DiffSkip (Luo et al., 2025) and the methods evaluated in its paper. The DiffSkip setting skips 8 FFN blocks on LLaMA3-8B-Instruct, while our setting removes $K = 4$ whole layers from the same model. Results for methods other than ours are taken from the DiffSkip paper. Adaptation costs are given in single-GPU A100-equivalent GPU hours when available. Because token-level FFN skipping and static whole-layer removal have different deployment mechanisms, this reference table is kept separate from the main pruning comparisons.

E Extended Non-Nested Results

LLaMA3.1-8B. The optimal sets are $\{10, 25, 26\}$ at $K = 3$ and $\{11, 12, 25, 26\}$ at $K = 4$. The best $K = 4$ set constrained to contain the $K = 3$ optimum is $\{10, 19, 25, 26\}$. The exhaustive search uses the BookCorpus calibration set (Figure 1), and Table 6 reports the WT2 perplexity of these sets alongside Shortened LLaMA and our method.

LLaMA3.2-1B. The exhaustive optima are $\{3, 4, 5, 6, 7, 8, 12\}$ at $K = 7$ and $\{3, 4, 5, 6, 7, 11, 12, 14\}$ at $K = 8$. The best nested alternative at $K = 8$ retains the $K = 7$ optimum and adds layer 14. The WT2 gap between this alternative and the exhaustive-search reference is 20.05%.

Pairwise interaction. In addition to the marginal degradation defined in the problem formulation, the pairwise interaction is

$$I_{ij} = D(\{i, j\}) - D(\{i\}) - D(\{j\}) + D(\emptyset). \tag{8}$$

A positive value means that jointly pruning the two layers is more harmful than the additive prediction from their individual effects; a negative value means it is less harmful. This interaction-based view motivates conditioning the scorer on the current pruning context and the target budget, so it estimates continuation value rather than assigning each layer a static score.

Method	MMLU	HellaSwag	WinoGrande	GSM8K	BBH	XSum	Retention	Adaptation cost
EarlyExit	66.5	52.4	64.5	0.0	2.9	2.8	48.0%	–
ShortGPT	65.6	52.2	66.5	0.0	2.8	0.3	44.8%	–
LaCo	65.7	63.0	60.4	6.6	22.7	8.6	65.3%	–
MindSkip	64.8	54.2	67.0	0.4	4.6	1.8	47.9%	–
DiffSkip	62.4	68.7	74.2	57.8	44.6	10.7	91.3%	56 GPU-h
Ours	63.9	64.1	71.2	51.4	31.9	7.2	79.3%	2 GPU-h

Table 11: Cross-study task results, retention, and adaptation cost. The two pruning settings and the source of the cost values are specified in Appendix D.

Constraint	WT2 PPL ↓	WT2 gap vs. reference
Exhaustive-search reference {3,4,5,6,7,11,12,14}	**861.16**	–
Best nested alternative {3,4,5,6,7,8,12,14}	1033.82	+20.05%

Table 12: LLaMA3.2-1B exhaustive-search results at $K = 8$.

F Additional Search Discussion

Budget-conditioned layer selection can be expressed as a sequential decision problem because the value of removing a layer depends on the current pruning set and the target budget. Greedy search selects the action with the best immediate validation effect. When one greedy trajectory is reused across budgets, the resulting sets are nested prefixes. Beam search maintains multiple partial sets and provides limited lookahead; its cost grows with beam width and pruning budget, and rerunning it for each target budget adds separate searches. Random and evolutionary search can find non-nested subsets by treating each budget as a black-box optimization problem.

The learned scorer $Q(s, a; K)$ models budget-conditioned continuation value. The budget is included in the state, and Bellman backup propagates final pruning quality to earlier decisions. Joint training across budgets allows the same scorer to be reused. These properties motivate the chosen RL formulation.

Comparison with Validation-Guided Search. We compare the learned scorer with direct validation-guided selection using greedy search (SLEB) and beam search with width 16. Table 13 reports the resulting perplexities at four budgets.

Our method matches beam search at $K = 4$ to the reported precision and achieves lower perplexity than both search baselines on both datasets at $K = 8, 12, 16$. At $K = 12$, for example, it reduces WT2 perplexity from 51.29 with beam search to 33.87. These results support the quality of the learned continuation-value scorer in the evaluated settings. The comparison does not include matched search times or evaluation counts; the available pruning-cost measurements are reported in Appendix I.

G Calibration Configuration Ablation

State features are recomputed from calibration samples at each pruning step, so the calibration configuration affects both feature quality and training cost. We vary the number of sampled sequences per episode, calibration pool size, and sequence length in Figures 3–5. The default configuration uses a pool of 64 sequences, four samples per episode, and 512 tokens per sequence.

Performance is relatively stable under moderate changes, especially at mild pruning budgets. At larger budgets, increasing the pool size, sample count, or sequence length can improve quality, but the gains vary across datasets and budgets. We use the default configuration throughout the main experiments unless stated otherwise.

	K=4		K=8		K=12		K=16	
	WT2	C4	WT2	C4	WT2	C4	WT2	C4
Greedy search (SLEB)	9.37	14.34	15.08	22.77	133.02	57.28	241.03	174.07
Beam Search	**8.68**	**14.03**	15.67	22.63	51.29	56.38	221.01	192.74
Ours	**8.68**	**14.03**	**14.27**	**21.57**	**33.87**	**36.50**	**170.71**	**133.43**

Table 13: Validation-guided search comparison on LLaMA3.1-8B with beam width 16 (WT2/C4 perplexity).

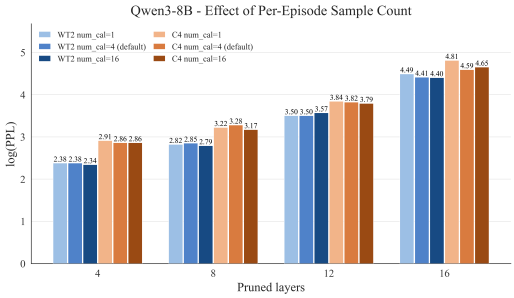

Figure 3: Effect of the number of calibration sequences sampled per episode on log perplexity.

H Budget and State-Feature Ablations

Retraining-based ablations. We retrain the Qwen3-8B scorer after removing budget information or one of four state-feature groups: representation transformation, parameter magnitude, predictive influence, and global context. Table 14 reports WT2 perplexity for each variant.

The full scorer has lower perplexity than every ablated variant at all four budgets. Removing global context causes the largest degradation, reaching 2548.17 at $K = 16$ compared with 82.49 for the full scorer. Removing budget conditioning yields 292.94, and removing predictive influence yields 134.22. Representation-transformation and parameter-magnitude features produce smaller but consistent gains in this comparison. Figure 6 gives the budget-conditioning results on both WT2 and C4.

I Offline Pruning Cost

Table 15 compares offline pruning time and language-modeling quality on LLaMA3.1-8B, with costs in A100-80GB GPU-hours. Quality is measured by perplexity averaged over WT2 and C4; the final column indicates whether a method supports multiple target budgets.

Our method achieves the lowest average perplexity at all four budgets with an offline cost of 2 GPU-h. This exceeds the 15 min required by SLEB and Prune&Comp and the 1 h required by BlockPruner, but is below the 16.5 h reported for Reassessing. The results show a trade-off between pruning quality and offline computation. A trained scorer can be reused across target budgets, and it is no longer needed once the pruning set is fixed.

Table 16 summarizes the average PPL reduction of our method over ShortGPT, Shortened LLaMA, and SLEB, computed from the per-budget values in Table 15.

Relative to SLEB, our method reduces average PPL by 24.82% while pruning cost increases from 15 min to 2 GPU-h. For the nine-task comparison, pruning times in A100-80GB GPU-hours are 2 h for our method, 18.8 h for Reassessing, 15 min for Prune&Comp, and 1 h for BlockPruner. The Reassessing time in the PPL/cost comparison is 16.5 h. The two Reassessing times come from different settings and are reported separately.

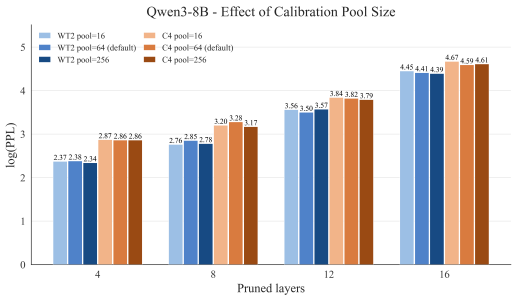

Figure 4: Effect of calibration pool size on log perplexity.

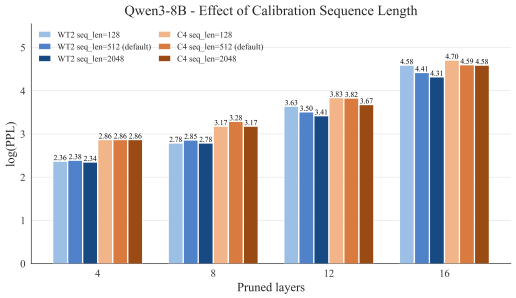

Figure 5: Effect of calibration sequence length on log perplexity.

J Wall-Clock Inference Acceleration

We compare the full BF16 LLaMA3.1-8B model ($N = 32$) with our model pruned at $K = 12$, which retains 20 Transformer layers. Figure 7 shows a single-stream generation speedup of approximately $1.53\times$ across output lengths of 32–512 tokens, close to the idealized depth ratio of $32/20 = 1.60$. Figure 8 shows comparable throughput gains for batch sizes from 1 to 16. These measurements show that removing layers yields practical inference acceleration across the evaluated generation lengths and batch sizes.

K Compatibility with Post-Training Quantization

Layer pruning and weight quantization reduce computation and storage at different granularities. Following Frantar et al. (2023), we apply GPTQ-INT4 to our pruned LLaMA3.1-8B models at $K \in \{4, 8, 12, 16\}$ and evaluate language-modeling quality and decoding throughput.

Figure 9 shows a small perplexity increase relative to the corresponding BF16 pruned models across the evaluated budgets. Figure 10 shows higher decoding throughput with GPTQ-INT4 at the same pruning budget. The results demonstrate compatibility between our layer selection and post-training quantization in this setting, allowing the throughput gains from reduced depth and low-bit weights to be combined.

L Extended Related-Work Comparison

RL-based pruning. Table 17 compares pruning targets and budget handling for three prior RL-based pruning methods.

LayerSkip. LayerSkip (Elhoushi et al., 2024) accelerates generation through early exit and self-speculative decoding: draft tokens are verified by the full model. Its training-time modifications include layer dropout and early-exit losses. The reported continual pretraining for LLaMA3-8B uses 419B tokens on 128 GPUs. Deployment also requires a draft-and-verify inference pipeline and

Variant	K=4	K=8	K=12	K=16
Ours	**10.83**	**17.26**	**33.15**	**82.49**
w/o budget	11.13	18.17	46.52	292.94
w/o Representation transformation	10.97	17.94	39.78	98.61
w/o Parameter magnitude	10.91	17.43	34.19	85.26
w/o Predictive influence	11.42	18.01	43.68	134.22
w/o Global context features	18.29	35.68	328.27	2548.17

Table 14: Retraining-based state-feature ablations on Qwen3-8B, measured by WT2 perplexity.

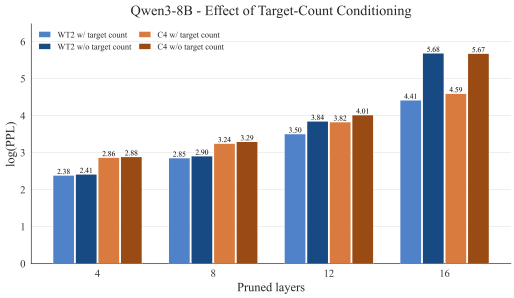

Figure 6: Effect of budget conditioning on Qwen3-8B layer pruning. The variants with and without budget information are compared on WikiText2 and C4 at each pruning count. Budget conditioning reduces log perplexity, especially under aggressive pruning.

Method	Avg PPL ↓				Pruning-time cost	Supports multiple K?
	K=4	K=8	K=12	K=16		
ShortGPT	16.41	4145.32	54885.81	2042.07	seconds	yes
Shortened LLaMA	11.70	22.08	76.77	2114.95	7 min	yes
SLEB	11.86	18.93	95.15	207.55	15 min	yes
Prune&Comp	11.43	18.12	89.35	206.88	15 min	yes
BlockPruner	11.82	18.46	54.42	234.57	1 h	yes
Reassessing	12.54	23.05	160.77	299.19	16.5 h	no
Ours	**11.36**	**17.92**	**35.19**	**152.07**	2 h	yes

Table 15: LLaMA3.1-8B pruning quality and offline cost. PPL is averaged over WT2 and C4; pruning costs are in A100-80GB GPU-hours.

KV-cache management, with speedup depending on the token acceptance rate. Our static pruning produces a smaller dense model without early-exit heads, draft verification, or runtime routing.

DiffSkip. DiffSkip (Luo et al., 2025) performs token-level dynamic FFN skipping using trained router/adaptor modules. The reported LLaMA3-8B-Instruct setting skips 8 FFN blocks. Its reported inference behavior includes no decoding-stage speedup on the evaluated GPU hardware because routing and adaptor execution introduce I/O and scheduling overhead; continuous generation can also be slower. Table 11 provides the available task results and adaptation costs as a separate cross-study reference.

ECoFLaP. ECoFLaP (Sung et al., 2023) performs coarse-to-fine unstructured weight pruning, mainly for vision-language models. It removes weights at a finer granularity than whole-layer pruning, and acceleration depends on sparse kernels and hardware support. Its reported evaluation emphasizes pruning quality and pruning-time memory usage. Our approach instead removes whole layers and evaluates the resulting dense model's wall-clock latency and throughput.

Method	Avg PPL reduction by Ours
ShortGPT	+80.71%
Shortened LLaMA	+42.18%
SLEB	+24.82%
Ours	–

Table 16: Average PPL reduction of our method relative to each baseline, averaged over the four budgets in Table 15.

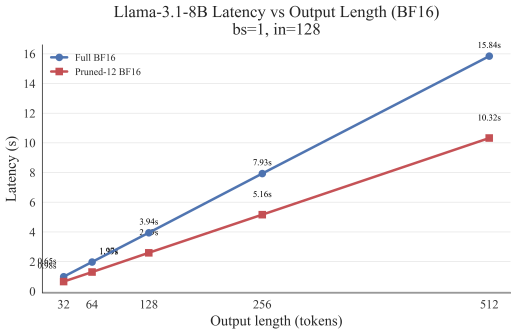

Figure 7: Generation latency of the full BF16 LLaMA3.1-8B model and the model pruned by 12 layers across output lengths.

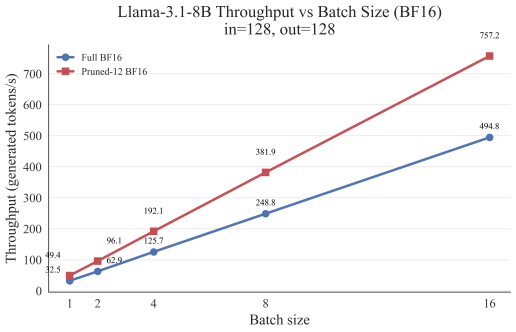

Figure 8: Decoding throughput of the full BF16 LLaMA3.1-8B model and the model pruned by 12 layers across batch sizes.

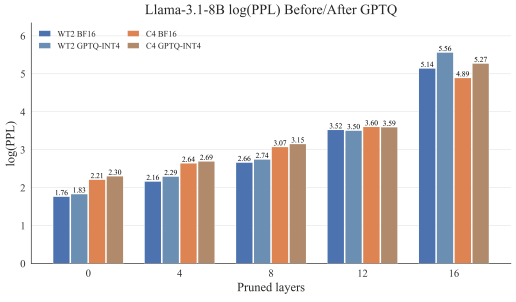

Figure 9: Log perplexity of pruned LLaMA3.1-8B models in BF16 and with GPTQ-INT4 across pruning budgets.

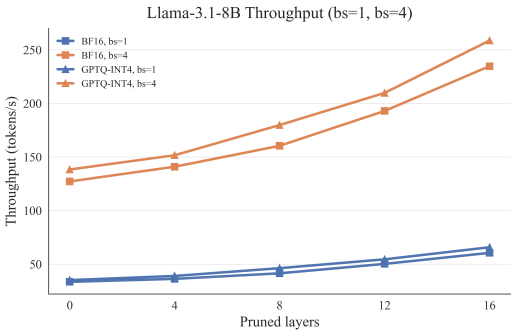

Figure 10: Decoding throughput of pruned LLaMA3.1-8B models in BF16 and with GPTQ-INT4 across pruning budgets.

Method	Target	Budget handling
Learning to Prune DNNs (RL) (Gupta et al., 2020)	CNN pruning	fixed target
RL-Pruner (Wang & Kindratenko, 2024)	CNN structured pruning	fixed target
FastForward Pruning (Yuan et al., 2025)	LLM pruning	fixed / one-step
Ours	LLM layer pruning	budget-conditioned

Table 17: Comparison of RL-based pruning methods with ours.