Task-aware Block Pruning with Output Distribution Signals for Large Language Models

Song-ha Jo[1] **Youngrok Ko**[1] **Sang-goo Lee**[1,2] **Jinseok Seol**[3,*]

[1]Seoul National University
[2]IntelliSys Co., Ltd.
[3]Dankook University

`{jos02, yrko1, sglee}@europa.snu.ac.kr`
`sglee@intellisys.co.kr`, `jinseok.seol@dankook.ac.kr`

Abstract

Large language models (LLMs) provide excellent performance, but their practical deployment is limited by the substantial compute and memory demands of large models and the latency of auto-regressive decoding. To mitigate these inefficiencies, block pruning reduces the number of executed transformer blocks, effectively lowering latency while preserving architectural coherence. However, existing methods typically rely on representation similarity or computationally expensive sensitivity analyses to estimate block importance, thereby neglecting task-aware model behavior. To address this limitation, we introduce Task-aware Block Pruning (TaBP), a novel approach that directly captures task-specific inference dynamics by quantifying block-level uncertainty from the statistics of each block's early-exited output distribution on a calibration dataset. Since output distributions reflect the model's confidence and decision uncertainty conditioned on downstream tasks, these statistics provide a principled signal for identifying blocks that are less critical for task performance. Extensive experiments demonstrate that TaBP preserves downstream task performance while substantially reducing inference latency and computational cost, without relying on cost-heavy sensitivity analyses. To facilitate reproducibility and further research, we release our implementation of TaBP.[1]

1 Introduction

Recent advances in LLMs have demonstrated remarkable performance across diverse natural language tasks (Dubey et al., 2024; Jiang et al., 2023; Yang et al., 2025a). Nevertheless, their growing size and inference cost pose significant challenges for practical deployment, particularly in resource-constrained environments. In practice, improving inference efficiency typically entails a trade-off, as aggressive compression often degrades downstream task performance. Model compression techniques, including pruning, quantization, and knowledge distillation (KD), have therefore emerged as essential tools to navigate this efficiency-performance trade-off. Furthermore, these methods are often compatible with one another to yield additive benefits (Han et al., 2016; Mirashi et al., 2024; Zafrir et al., 2021; Kurtic et al., 2022; Zeng et al., 2024a; Muralidharan et al., 2024; Song et al., 2024).

Among them, pruning is especially attractive when combined with post-pruning recovery strategies, such as fine-tuning or KD (*e.g.*, LLaMA 3.2-1B (Dubey et al., 2024)), as the pruned models inherit the strengths of larger models, avoiding costly neural architecture search (Frankle and Carbin, 2019; Chen et al., 2020; Zhang et al., 2021; Sarah et al., 2024; Bercovich et al., 2025). While pruning has emerged as a powerful tool to improve efficiency, its practical effectiveness depends heavily on its structural pattern. Unstructured pruning is difficult to exploit efficiently without specialized hardware architectures (Kim et al., 2018; Chen et al., 2019), while layer-wise width pruning and layer-level depth pruning often suffer from structural imbalance or instability, resulting in limited efficiency gains or even severe performance degradation (Kim et al., 2024; Lele et al., 2025; Xia et al., 2024; Zhang et al., 2024; He et al., 2024; Park et al., 2025).

Thus, in this work, we focus on block pruning,

[*]Corresponding author
[1]`https://github.com/Song-haJo/TaBP`

which has emerged as a promising solution due to its ability to achieve latency reductions proportional to the compression ratio (Kim et al., 2024; Song et al., 2024; Zhong et al., 2025). By shortening the computation path through the removal of certain transformer blocks, block pruning reduces inference costs while maintaining the internal structural integrity of each block. Unlike other methods, it offers structurally coherent reductions that translate directly to real-world latency improvements with stable execution (Zhong et al., 2025).

Existing pruning methods rely either on representational similarity or on computationally expensive sensitivity analyses, in which block importance is assessed by iteratively removing individual blocks and measuring performance degradation. However, relying on representational similarity or language modeling loss as indirect proxies, these methods often fail to capture task-relevant decision signals, as they do not directly indicate whether a block meaningfully contributes to task-level reasoning accuracy or decision confidence. For example, pruning decisions guided by perplexity do not necessarily correlate with reasoning accuracy of downstream tasks (Kim et al., 2024; Song et al., 2024). This limitation motivates an explicit output-driven perspective that captures task-relevant decision signals by analyzing how each block shapes the model's output distribution, including its confidence and uncertainty. In this study, we explore this perspective to identify critical blocks more effectively.

We specifically target downstream tasks (*e.g.*, multiple-choice question answering), which place stronger demands on task-level decision making and are less tolerant of superficial language modeling shortcuts. This setting differs fundamentally from next-token prediction, which focuses on approximating the underlying language distribution and local contextual coherence, whereas reasoning tasks require integrating global context and discriminating between competing answer candidates. Therefore, the impact of pruning cannot be inferred directly from perplexity-based metrics (Bachmann and Nagarajan, 2024; Hu et al., 2024).

This raises a key question: **which transformer blocks truly matter for reasoning-oriented tasks beyond surface-level fluency?** We hypothesize that block importance should be inferred from how each block alters the model's output distribution, since such signals capture task-level decision certainty beyond generic language modeling. Accordingly, we posit that pruning guided by output-distribution signals aligns more effectively with the demands of reasoning-oriented tasks than perplexity-based heuristics.

Most existing block pruning methods, however, evaluate importance by removing one block at a time and measuring perplexity increases (Kim et al., 2024; Song et al., 2024), which is known as the sensitivity analysis. While effective for LM loss, this criterion correlates poorly with downstream accuracy (Liu et al., 2023; Hu et al., 2024; Zeng et al., 2025). Our method departs from both sensitivity and similarity baselines by ranking blocks using output-distribution measures. To this end, Thus, drawing inspiration from the Early Exit (EE) mechanism (Liu et al., 2020; Hu et al., 2023), which effectively captures internal model behavior, we introduce **Task-aware Block Pruning (TaBP)**, a framework that leverages output distributions to estimate block importance. The main contributions are as follows:

- We introduce an output distribution-based block pruning framework, **TaBP**, which estimates block importance from early-exited output distributions.
- We propose two block-importance scoring metrics, **Directional Desirable Frequency (DDF)** and **Statistical Shift Norm (SSN)**, which aggregate sample-wise distributional shifts into block-level importance signals.
- With extensive experiments across diverse datasets, metrics, and measures, we demonstrate that analyzing the output distribution rather than hidden-representation similarity or costly sensitivity analysis better captures task-relevant behavior.
- We empirically show that task-specific calibration on MCQA yields more stable and interpretable pruning decisions than task-agnostic observation (*e.g.*, general text generation).

2 Related Work

2.1 Depth Pruning for LLMs

Depth pruning reduces model depth by removing components at varying levels of granularity, ranging from individual layers to entire blocks. Layer pruning refers to the removal of individual components within a block, mainly targeting multi-headed self-attention (MHSA) modules or feed-forward networks (FFNs), thereby skipping specific operations at a finer granularity (He et al., 2024). In contrast, block pruning drops an entire transformer block, including MHSA, FFN, and their associated normalization and residual paths (Zhong et al., 2025; Song et al., 2024; Yang et al., 2025c). In this work, we focus on block-level pruning, as removing individual layers (*i.e.*, subcomponents of blocks) disrupts the architectural consistency, thereby compromising the functionality of transformer blocks and the overall computational pipeline (see Appendix A.1 for more details).

Block pruning methods can be broadly categorized into sensitivity-based and similarity-based approaches. Sensitivity-based methods evaluate the impact of removing each block on model output, which can be the final representation or model performance, such as perplexity (PPL) (Kim et al., 2024; Song et al., 2024). However, this approach is computationally expensive, similar to grid search at the point that it finds the optimal set of blocks to prune, especially when it prunes in an iterative manner (see Appendix A.2 for detailed complexity analysis).

In contrast, similarity-based methods assess the impact of each block without explicitly dropping it, by measuring changes in intermediate representations. This approach estimates the importance of each block in a single experiment, whereas sensitivity analysis requires sequential experiments proportional to the number of blocks. Here, the representations before and after each block are compared using cosine similarity or entropy differences (He et al., 2024; Gromov et al., 2025; Yang et al., 2025c). In the case of cosine similarity, a higher value indicates a lower contribution of the block, while in the case of entropy, a smaller increase suggests that the block contributes less to representation enrichment. However, a drawback of this approach is the lack of a straightforward mapping between variations in intermediate representations and the final model's performance In other words, quantitative changes in representations do not strictly correlate with the functional behavior of the model, making their interpretation ambiguous in terms of actual performance.

To address these limitations, we propose TaBP, a novel framework designed to improve upon existing paradigms by mitigating the computationally prohibitive costs of sensitivity-based methods and the semantic ambiguity of similarity-based approaches. By effectively leveraging the strengths of both strategies, TaBP achieves robust performance preservation, ensuring the pruned model retains its capabilities without incurring excessive computational overhead.

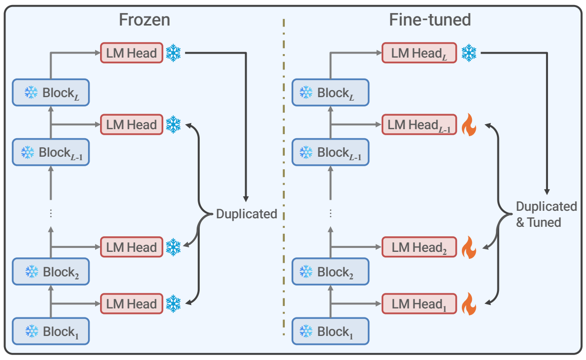

Figure 1: Illustration of the block-wise LM head fine-tuning procedure. In the original model, only the final block output is connected to the LM head. For the frozen probing, LM heads are duplicated from the final LM head and attached to each block. In the fine-tuned probing, these duplicated LM heads are further optimized, enabling block-wise token probability distributions to be directly probed.

2.2 Early Exit and Uncertainty Quantification

Early Exit (EE) is a dynamic acceleration technique that terminates the inference once the model attains a sufficiently high level of confidence in its prediction. In this context, confidence denotes the complement of uncertainty, referring to the degree to which the model is assured of its output. Accordingly, confidence estimation is therefore often used interchangeably with uncertainty quantification. Thus, by quantifying uncertainty, the model can achieve acceleration during the inference. Typical metrics include entropy and top-1 probability, the latter often referred to as the confidence score (Schuster et al., 2022; Bae et al., 2023; Zheng et al., 2025).

Uncertainty Quantification (UQ) for LLMs concerns estimating the reliability of generated responses, typically by analyzing the probability distributions over the vocabulary produced by the LM head. Output distributions obtained via probing provide the model's probabilistic semantics, enabling direct analysis of model behavior as well as the application of calibration techniques. Classical uncertainty measures, namely entropy and confidence score, are directly defined on these distributions and have been widely applied to LLM calibration and early-exit strategies (Jiang et al., 2020; Cao et al., 2025; Kao et al., 2021). In contrast, hidden representations, which encode rich semantic information in a high-dimensional space, lack an immediately interpretable probabilistic meaning, thereby necessitating probing to elicit output distributions for analysis (Nostalgebraist, 2020; Belrose et al., 2025; Yom Din et al., 2024) (see Appendix A.3 for details).

Instead of utilizing embeddings directly, EE and UQ methods typically rely on probing output distributions generated by LM heads, with metrics mentioned above that have been shown to provide effective measures of uncertainty in practice (Schuster et al., 2022; Bae et al., 2023; Zheng et al., 2025). This design works well because probability distributions naturally encode the model's degree of belief over the vocabulary, transforming abstract and opaque latent features into explicit probabilistic semantics, yielding quantities, namely entropy and confidence score, that are directly interpretable as measures of certainty. In contrast, hidden representations live in a high-dimensional feature space whose geometry is not calibrated to probabilities, making it difficult to distinguish between confident and uncertain predictions without additional mapping (Yom Din et al., 2024). By projecting into the output distribution space, EE and UQ methods align uncertainty estimation with the model's native probabilistic semantics, ensuring that uncertainty measures are both theoretically grounded and empirically reliable (Schuster et al., 2022; Belrose et al., 2025).

Inspired by the empirical success of EE in leveraging uncertainty for dynamic acceleration, we propose to distill this principle into a static block pruning framework. The key insight from EE is that the uncertainty metrics used to trigger early exits effectively measure the informational contribution of each layer. By aggregating these probabilistic signals across a calibration dataset, we incorporate task-awareness, transitioning from making instance-specific exit decisions to identifying globally redundant blocks tailored to the specific downstream task. Consequently, our method utilizes probing-based uncertainty quantification not for runtime termination, but as a criterion to permanently excise layers that contribute minimally to confidence accumulation, thereby integrating the efficiency of dynamic methods into a fixed, compact architecture.

3 Methodology

We propose a block pruning framework, **Task-aware Block Pruning (TaBP)**, inspired by early-exiting (EE). TaBP proceeds in three stages: (1) attach block-wise LM heads to expose intermediate output distributions (Fig. 1), (2) compute distributional shifts on calibration samples and aggregate them into block-level importance scores (Fig. 2), and (3) prune the k lowest-scoring blocks. Specifically, for each block, we duplicate the final LM head and optionally fine-tune it to better decode each block's representation. Using these LM heads, we measure how each block changes the output distribution, before and after passing through the block, on calibration samples, and aggregate these shifts to quantify importance. Unlike similarity-based approaches that compare hidden representations, TaBP analyzes output distributions, capturing task-level behavior without the computational overhead of iterative sensitivity analysis.

In this section, block importance refers to how strongly each transformer block contributes to shaping the model's output distribution. To quantify this, we propose two metrics, **Directional Desirable Frequency (DDF)** and **Statistical Shift Norm (SSN)**, which represent distinct aggregation schemes. The underlying statistical measures, such as entropy or KL-divergence, are applied to each block-wise output distribution to compute scalar values s_i. These values serve as interpretable signals of the model's probabilistic behavior and form the basis for both direction- and magnitude-based importance estimation.

3.1 Block-wise LM Head

As a pre-stage, to probe the output distribution at each block, LM heads should be attached to all blocks, as illustrated in Fig. 1. For more precise probing, the LM heads duplicated from the original LM head can be further fine-tuned (although empirically, not necessarily). In this way, LM heads are optimized to decode the representations of each block, enabling more accurate estimation of the output distribution according to the practice of EE and UQ (Wei et al., 2024; Schuster et al., 2022; Zeng et al., 2024b; Elhoushi et al., 2024). Once the LM heads are fine-tuned, the model is ready for block-wise importance estimation.

3.2 Block-wise Importance Estimation

Using the calibration samples, we estimate the block-wise importance via aggregating statistical shift of samples. The statistical shift of a sample can be derived from various criteria, listed in Table 4 provided in Appendix D, which serve as a sample-

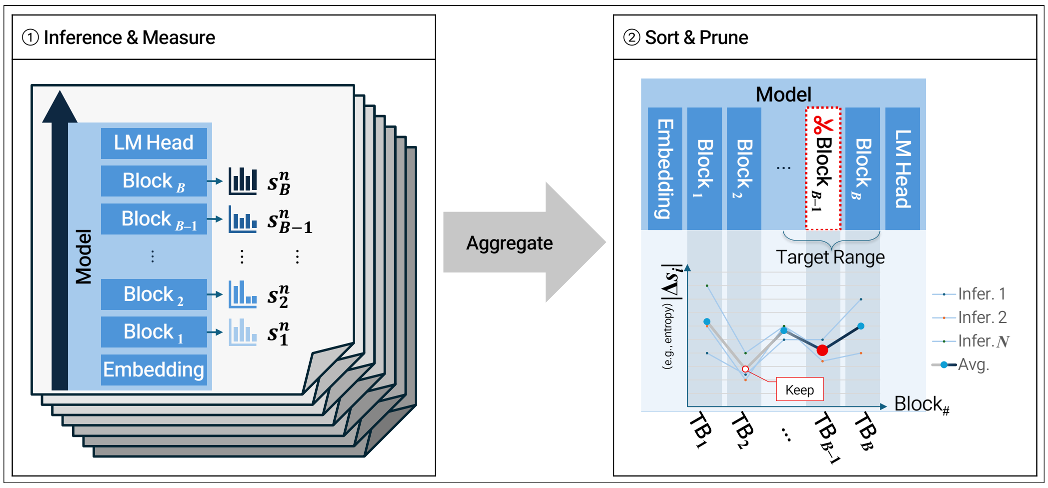

Figure 2: To estimate the importance of each block, we leverage the token probability distribution output from every block's attached LM head on calibration samples, as in EE (Kao et al., 2021) and LogitLens (Nostalgebraist, 2020). Then, they are compared to the nearest previous blocks and aggregated block-wisely via Directional Desirable Frequency (DDF) and Statistical Shift Norm (SSN). The graph on the right depicts SSN with $p = 1$. By sorting blocks with the scores, relatively less important blocks are pruned.

wise scalar value that quantify the contribution or importance of the model's internal behavior. For instance, if the entropy of the output distribution for a input calibration sample does not significantly change between before and after a block, it might indicate that the block is less influential for the specific sample. On the other hand, if the probability of the ground-truth alters drastically in the block, we can conclude that the block is essential for the task. Accordingly, we suggest two aggregation methods, namely DDF and SSN, which aggregates sample-wise statistical shift into a block-wise importance.

For each of the B transformer blocks $\{b_i\}_{i=1}^{B}$ and N calibration samples, let $s_i^{(n)}$ denote the *statistical value* of a sample, measured at block b_i for sample $n = 1, 2, \cdots, N$, computed via using one of the criteria from Table 4. We define the *statistical shift* of a sample as

$$\Delta s_i^{(n)} = s_i^{(n)} - s_{i-1}^{(n)}. \tag{1}$$

Directional Desirable Frequency One strategy to identify pruning candidates is to penalize blocks that *negatively* affect the reasoning process. This is done by checking the *direction* of change in each measured values, whether it increases or decreases compared to the previous block. For example, entropy over the probability distribution is preferred to be low, meaning that the model is more certain or decisive on its output. Here we introduce a signature coefficient α that standardizes the directionality of changes depending on whether higher values are considered undesirable or desirable for a given measure: $\alpha = +1$ for desirability of $\uparrow$ and -1 otherwise (see Table 4). Using this, DDF measures how often the block exhibits desirable directional shifts across the N samples:

$$\mathrm{DDF}(b_i) = \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}\big[\alpha \cdot \Delta s_i^{(n)} > 0\big]. \tag{2}$$

The pruning set is then obtained by selecting k blocks with the lowest DDF scores.

Statistical Shift Norm Alternatively, even an undesirable change (*e.g.*, an entropy increase) may indicate that a block meaningfully affects the model's predictive dynamics. From this macroscopic perspective, Statistical Shift Norm (SSN) quantifies how much a block changes the statistical value on average, regardless of direction. For each block, SSN computes the mean of the changes across all N samples, similar to L_p-norm:

$$\mathrm{SSN}(b_i) = \frac{1}{N}\left(\sum_{n=1}^{N}\left(\Delta s_i^{(n)}\right)^p\right)^{\frac{1}{p}}. \tag{3}$$

Blocks with smaller SSN scores contribute only marginally to overall variation, and are thus pruned.

Note that DDF captures the *frequency* of desirable directional shifts, while SSN measures the *magnitude* of the shift, serving as complementary signals of the block importance. DDF thus reflects consistency in certainty evolution, favoring blocks that consistently drive the model's confidence in the desired direction, whereas SSN captures contribution strength by preserving blocks that substantially modulate predictive dynamics even when their effects are occasionally undesirable. In SSN, since $p \geq 2$ shows a similar trend to $p = 1$, we omit the experimental results for brevity.

3.3 Calibration Samples

To clearly capture task-relevant behavior, we adopt multiple-choice question answering (MCQA) as the calibration dataset for the block-wise importance estimation. Comparing to the general text generation (GTG), MCQA differs significantly in their cognitive and behavioral demands placed on models. Particularly, MCQA is a setting focused on restricted options, while in GTG, entropy is computed over the full vocabulary space (*e.g.*, LLaMA-3-8B has vocab size of 128,256), which introduces considerable noise and interpretability challenges. In other words, high entropy may not reliably indicate genuine uncertainty, as it can be inflated by semantically insignificant tokens like function words or punctuation. A more detailed comparison between MCQA and GTG settings is discussed in Section 5.1.

Constraining the output space to discrete options (*e.g.*, A/B/C/D) promotes transparency and traceability of distributional changes to task-relevant decision points, eliminating spurious values. This makes distributional changes interpretable as genuine uncertainty about task-relevant alternatives, improving the assessment of block-level influence. This task-constrained setting not only improves signal clarity but also strengthens the validity of distributional uncertainty signal observation regarding block-wise importance.

4 Experiments

Our experiments address three main aspects of TaBP. First (**RQ1**), we evaluate whether TaBP more effectively preserves reasoning accuracy than sensitivity- or similarity-based baselines under equal sparsity. Second (**RQ2**), we examine how task-specific calibration using MCQA improves the reliability of block-importance estimation compared to task-agnostic GTG observation. Third (**RQ3**), we analyze which statistical measures and output-level signals contribute most to TaBP's performance and explain its strong alignment with reasoning behavior.

To answer these questions, we conducted experiments on pretrained open-source LLMs with 32 transformer blocks, namely LLaMA-3-8B-base and Mistral-7B-base (Grattafiori et al., 2024; Jiang et al., 2023). We used a subset of MCQA with 1,024 samples from ARC Easy training set (Clark et al., 2018) as calibration samples, and the inference was performed with logits processors from `transformers` library to strictly force a model to decode only the provided answer key options (Hugging Face, 2025). On the other hand, inference in GTG setting was performed for 1,024 tokens on Wikitext-2 while retaining the original vocabulary space (Merity et al., 2016). For LM head training in the pre-stage, multiple datasets were mixed to enhance generalization (further details are provided in Appendix B.1). Block-level pruning was applied with two importance metrics, namely DDF and SSN, setting k at most 8, *i.e.*, pruning ratio = 25%. Evaluation was implemented on the following datasets: ARC Easy & Challenge (ARC-E & ARC-C) (Clark et al., 2018), BoolQ (Clark et al., 2019), COPA (Gordon et al., 2012), HellaSwag (HeSw.) (Zellers et al., 2019), PIQA (Bisk et al., 2020), and WinoGrande (Wino.) (Sakaguchi et al., 2021). As baselines, ShortenedLLaMA (Kim et al., 2024), SLEB (Song et al., 2024), and EntroDrop (Yang et al., 2025c) implement block pruning, whereas JointLayerDrop (He et al., 2024) conducts layer-wise pruning.

Among them, ShortenedLLaMA and SLEB adopt a sensitivity-based criterion, evaluating block importance by the increase in PPL when a block is removed. A distinguishing aspect of SLEB is the iterative pruning strategy that allows precise control and potentially better trade-offs between efficiency and performance (Chen et al., 2025). Meanwhile, others are one-shot pruning methods, where pruning decisions are made once according to the chosen criterion, without subsequent refinement.

In contrast to the exhaustive yet straightforward performance-oriented approach, EntroDrop and JointLayerDrop rely on similarity-based criteria. JointLayerDrop computes cosine similarity between hidden representations, whereas EntroDrop leverages the entropy of hidden activations as a measure of information content, pruning blocks that contribute only marginal entropy gains (indicating limited information enrichment). For finer performance maintenance, JointLayerDrop executes depth pruning by dropping layers in a block. To show the effect of different granularity, we included

Method	**ARC-E ↑**	**ARC-C ↑**	**BoolQ ↑**	**COPA ↑**	**HeSw. ↑**	**PIQA ↑**	**Wino. ↑**	**Avg. ↑**
Original Model	80.09	53.33	81.44	89.00	79.17	79.71	72.85	76.51
Prune 2 blocks (pruning ratio = 6.25%)								
SLEB	**76.60**	48.29	74.80	87.00	74.33	77.80	70.48	72.76
ShortenedLLaMA	76.05	49.23	75.66	**90.00**	74.97	**77.86**	70.88	73.52
EntroDrop	67.55	44.45	63.09	84.00	68.58	75.57	71.43	67.81
JointLayerDrop	62.58	42.75	62.45	78.00	57.89	70.89	62.04	62.37
TaBP (ours)	76.01	**50.17**	**80.76**	87.00	**76.24**	78.07	**73.88**	**74.59**
Prune 4 blocks (pruning ratio = 12.5%)								
SLEB	**72.47**	43.26	66.12	**87.00**	70.69	**76.12**	69.46	69.30
ShortendLLaMA	70.16	38.65	57.25	**87.00**	66.83	75.63	61.72	65.32
EntroDrop	40.78	33.53	58.59	67.00	37.43	61.86	58.48	51.10
JointLayerDrop	56.73	41.13	62.39	74.00	51.71	68.88	61.96	59.54
TaBP (ours)	71.46	**47.44**	**73.21**	81.00	**72.05**	75.95	**72.53**	**70.52**
Prune 8 blocks (pruning ratio = 25%)								
SLEB	61.78	31.74	42.11	**77.00**	57.98	71.82	53.59	56.57
ShortendLLaMA	**61.83**	31.66	42.05	**77.00**	57.94	**71.87**	53.83	56.60
EntroDrop	33.33	29.78	62.11	66.00	26.97	57.67	56.59	47.49
JointLayerDrop	50.51	37.97	62.54	68.00	46.72	65.23	61.40	56.05
TaBP (ours)	53.83	**40.61**	**64.34**	72.00	**59.35**	68.61	**67.79**	**60.93**

Table 1: Zero-shot accuracy (%) of Llama-3-8B on each task after pruning. Best and the second best results are indicated as bold and underline, respectively.

JointLayerDrop as a baseline for direct comparison. For fairness, all methods prune the same number of blocks ($k = 2, 4, 8$), without additional fine-tuning, and are evaluated under identical decoding settings.

5 Results

Table 1 and 2 report the results for **RQ1**, which compares baselines to our suggesting method, **T**ask-aware **B**lock **P**runing (TaBP), using either of DDF or SSN with the best criteria for the statistical value measurement. Across multiple benchmarks, four baselines and the proposed methods with all configuration options are evaluated under equal pruning ratios, achieved by dropping 2, 4, and 8 blocks (corresponding to 6.25%, 12.5%, and 25%, respectively). In detail, criterion of Entropy performed best with TaBP on LLaMA-3-8B and Mistral-7B with DDF and SSN, respectively. As will be elaborated later, these measures operate at the distribution level, offering a relatively broad lens through which to interpret model behavior.

When pruning 2 blocks from LLaMA-3-8B, TaBP achieves the highest average accuracy 74.59%, outperforming other block-based methods overall, although SLEB and ShortendLLaMA remain competitive on individual tasks. At a pruning ratio of 12.5%, TaBP maintains the lead with an average accuracy of 70.52%, ahead of current state of the art, SLEB and considerably above ShortenedLLaMA and JointLayerDrop, showing particular strength on ARC Challenge and BoolQ. When pruning one fourth of the total 32 blocks, TaBP demonstrates strong robustness, achieving an average accuracy of 60.93%. High accuracy is retained on BoolQ (64.34%) and WinoGrande (67.79%), whereas competing methods degrade sharply.

When pruning 2 blocks from Mistral-7B, TaBP with DDF aggregation and KLD criterion attains the highest average accuracy 74.47%, with consistently strong results on ARC Challenge and BoolQ, and maintains its strength under further pruning. At 12.5% pruning, TaBP continues to lead with an average of 71.01%, slightly ahead of SLEB (70.04%) and substantially better than the others. At a pruning ratio of 25%, TaBP demonstrates notable robustness, achieving 64.24% on average. High accuracy is preserved on BoolQ (76.85%) and WinoGrande (66.69%), whereas competing methods degrade considerably. These results demonstrate the effectiveness of output distribution-based pruning in preserving accuracy across diverse tasks, especially under high pruning ratios.

Discussion on Granularity of Depth Pruning When examining results across both models, we find that JointLayerDrop appears more robust than anticipated, often exceeding EntroDrop and occasionally surpassing other methods as the pruning ratio increases. This may seem difficult to reconcile with our earlier observation about the instability of layer pruning. The discrepancy arises because

Method	ARC-E ↑	ARC-C ↑	BoolQ ↑	COPA ↑	HeSw. ↑	PIQA ↑	Wino. ↑	Avg. ↑
Original Model	80.09	53.50	81.31	89.00	79.16	79.71	72.61	76.48
Prune 2 blocks (pruning ratio = 6.25%)								
SLEB	76.22	46.67	77.71	88.00	77.30	**80.03**	67.96	73.41
ShortenedLLaMA	74.87	44.62	74.53	88.00	73.88	42.20	**78.62**	72.02
EntroDrop	67.55	44.45	63.09	84.00	68.58	75.57	71.43	67.81
JointLayerDrop	56.73	41.97	62.23	76.00	56.01	67.90	66.22	61.01
TaBP (ours)	**78.20**	**51.37**	**78.72**	**91.00**	**78.09**	79.65	70.40	**75.35**
Prune 4 blocks (pruning ratio = 12.5%)								
SLEB	71.68	41.98	**74.22**	87.00	72.83	**77.58**	64.96	70.04
ShortendLLaMA	66.84	34.30	59.51	**88.00**	60.68	75.41	56.43	63.02
EntroDrop	29.97	29.35	32.94	71.00	33.49	56.20	61.09	44.86
JointLayerDrop	49.83	36.09	62.51	66.00	46.18	64.47	62.75	55.40
TaBP (ours)	**73.02**	**47.44**	73.82	86.00	**74.97**	76.99	**68.67**	**71.56**
Prune 8 blocks (pruning ratio = 25%)								
SLEB	**63.59**	35.75	59.54	**79.00**	**62.95**	**72.47**	60.85	62.02
ShortendLLaMA	51.73	26.28	54.89	70.00	48.28	69.26	52.17	53.23
EntroDrop	26.77	29.18	42.48	72.00	28.25	56.15	55.80	44.38
JointLayerDrop	49.20	36.77	62.45	66.00	45.37	63.93	62.51	55.18
TaBP (ours)	55.89	**38.74**	**77.80**	74.00	61.59	68.50	**68.75**	**63.61**

Table 2: Zero-shot accuracy (%) of Mistral-7B on each task after pruning. Best results are in bold; second-best results are underlined.

block pruning is coarser, and thus its impact on performance can be more pronounced. Nevertheless, as the pruning ratio increases, instability emerges, as also illustrated in Fig. 3 (in Appendix), highlighting the inherent limitations of layer pruning. Further discussion is provided in Appendix C.

5.1 Ablation Study

Importance of task setting To highlight the role of task-specific calibration and verify the hypothesis of **RQ2**, we compare MCQA and GTG, which differ fundamentally in their output characteristics: MCQA involves a constrained, discrete choice space with known answers, while GTG entails open-ended generation with higher entropy and variance. As shown in Table 3, the best-performing statistical measure varies by task, yet those listed outperform all baselines. This finding supports both the use of task-specific observation datasets for criterion search, rather than relying solely on general language modeling, and the importance of leveraging output distributions over internal representations, to avoid overlooking task-dependent redundancy or reasoning signals.

Criteria candidates To identify a reliable criterion for **RQ3**, various options for a statistical measure were considered as listed in Table 4 (Appendix D). These measures differ in scope: a single, a pair, or a full set of probabilities in a distribution, or a pair of distributions. Table 5 (in Appendix D) reports performance in terms of the average accuracy across tasks and the AUC of the accuracy curve over pruning ratios from 0% to 25%. Among them, entropy performed best, as they capture global changes in the output distribution rather than local confidence variations. By capturing how the uncertainty evolves across blocks, these distribution-level measures provide a stable and interpretable signal for reasoning-oriented pruning decisions with DDF, which utilizes the intrinsic meaning of each measure.

Effect of Fine-tuning on LM Heads Table 5 (in Appendix D) compares the performance of models with and without fine-tuning of duplicated LM heads. Interestingly, the results show that fine-tuning is not strictly necessary for the accurate

Method		Avg. Acc.@8 ↑	AUC@8 ↑
DDF (ours)	MCQA	**62.87**	**.7022**
	GTG	58.02	.6906
SSN (ours)	MCQA	58.92	.6711
	GTG	59.59	.6902
SLEB		56.57	.6703
Short'dLLaMA		56.60	.6633
EntroDrop		45.68	.4821
JointLayerDrop		56.05	.5285

Table 3: Average zero-shot accuracy (%) of Llama-3-8B after pruning eight blocks and AUC over pruning ratios from 0% to 25%, across different option settings under the *frozen* LM head configuration. As a statistical measure, entropy was most effective for MCQA, while gap performed best for GTG.

block importance assessment. Even without additional training, the duplicated LM heads exhibit competitive or occasionally superior performance. This observation suggests that the original LM head already possesses a substantial ability to act as a token classifier across intermediate blocks, demonstrating the inherent language modeling competence embedded throughout the network. Similar findings have been reported in prior studies on depth pruning, early exiting, and uncertainty quantification (Shan et al., 2024; Nostalgebraist, 2020), which observed that intermediate representations can be decoded into meaningful token distributions using the final projection matrix alone. In summary, while fine-tuning improves block-wise alignment and yields smoother transitions of values derived from statistical measures, the marginal gain implies that frozen heads already suffice for block importance estimation.

6 Conclusion

This work introduces a simple yet effective method for block pruning in LLMs using output distribution signals, such as entropy and probability gaps. The approach eliminates reliance on language modeling loss and exhaustive sensitivity analysis and enables pruning aligned with task-relevant internal behavior. Extensive experiments demonstrate improved performance preservation over baselines and robustness under high sparsity. Our findings highlight that output-level uncertainty signals, particularly entropy and divergence, provide stable, interpretable, and task-aligned criteria for structural pruning.

Limitations

The generalizability of the proposed method across model architectures, scales, and datasets requires further validation. First, the estimation relies on a small calibration set (*e.g.*, 1k MCQA samples from a single dataset), which may not fully capture task diversity or linguistic variation. Calibration via ensembling over diverse domain-specific datasets could yield more robust importance rankings. Second, although distribution-level statistics such as entropy or KL-divergence are interpretable, they may not perfectly reflect the causal contribution of a block to reasoning success. Future work will explore cross-task generalization of distribution-based pruning signals.

Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2025-00562784). This research was also supported by the ICT at Seoul National University, which provided the research facilities for this study. We thank Hyuhng Joon Kim for careful review and feedback on the manuscript.

References

Gregor Bachmann and Vaishnavh Nagarajan. 2024. The pitfalls of next-token prediction. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 2296–2318. PMLR.

Sangmin Bae, Jongwoo Ko, Hwanjun Song, and Se-Young Yun. 2023. Fast and robust early-exiting framework for autoregressive language models with synchronized parallel decoding. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5910–5924, Singapore. Association for Computational Linguistics.

Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. 2025. Eliciting latent predictions from transformers with the tuned lens.

Akhiad Bercovich, Tomer Ronen, Talor Abramovich, Nir Ailon, Nave Assaf, Mohammad Dabbah, Ido Galil, Amnon Geifman, Yonatan Geifman, Izhak Golan, Netanel Haber, Ehud Karpas, Roi Koren, Itay Levy, Pavlo Molchanov, Shahar Mor, Zach Moshe, Najeeb Nabwani, Omri Puny, and 7 others. 2025. Puzzle: Distillation-based nas for inference-optimized llms. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 3806–3830. PMLR.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press.

Łukasz Borchmann. 2025. Arc 'challenge' is not that challenging. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 2797–2804.

Steven Cao, Gregory Valiant, and Percy Liang. 2025. On the entropy calibration of language models.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pre-trained BERT networks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Xinrui Chen, Hongxing Zhang, Fanyi Zeng, Yongxian Wei, Yizhi Wang, Xitong Ling, Guanghao Li, and Chun Yuan. 2025. Prune&comp: Free lunch for layer-pruned llms via iterative pruning with magnitude compensation. *Preprint*, arXiv:2507.18212.

Yu-Hsin Chen, Tien-Ju Yang, Joel Emer, and Vivienne Sze. 2019. Eyeriss v2: A flexible accelerator for emerging deep neural networks on mobile devices. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 9:292–308.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R. Glass, and Pengcheng He. 2024. Dola: Decoding by contrasting layers improves factuality in large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *Preprint*, arXiv:1803.05457.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 82 others. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, Ahmed Aly, Beidi Chen, and Carole-Jean Wu. 2024. LayerSkip: Enabling early exit inference and self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12622–12642, Bangkok, Thailand. Association for Computational Linguistics.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Andrew Gordon, Zornitsa Kozareva, and Melissa Roemmele. 2012. SemEval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 394–398, Montréal, Canada. Association for Computational Linguistics.

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

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Dan Roberts. 2025. The unreasonable ineffectiveness of the deeper layers. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Song Han, Huizi Mao, and William J. Dally. 2016. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Shwai He, Guoheng Sun, Zheyu Shen, and Ang Li. 2024. What matters in transformers? not all attention is needed. *Preprint*, arXiv:2406.15786.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701.

Boren Hu, Yun Zhu, Jiacheng Li, and Siliang Tang. 2023. Smartbert: A promotion of dynamic early exiting mechanism for accelerating BERT inference. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 5067–5075. ijcai.org.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*. Poster.

Yutong Hu, Quzhe Huang, Mingxu Tao, Chen Zhang, and Yansong Feng. 2024. Can perplexity reflect large language model's ability in long text understanding? In *The Second Tiny Papers Track at ICLR 2024*.

Hugging Face. 2025. Utilities for generation. `https://huggingface.co/docs/transformers/v4.53.2/en/internal/generation_utils#transformers.LogitsProcessor`. Accessed: 2025-07-16.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Wei-Tsung Kao, Tsung-Han Wu, Po-Han Chi, Chun-Cheng Hsieh, and Hung-Yi Lee. 2021. Bert's output layer recognizes all hidden layers? some intriguing phenomena and a simple way to boost bert. *Preprint*, arXiv:2001.09309.

Bo-Kyeong Kim, Geonmin Kim, Tae-Ho Kim, Thibault Castells, Shinkook Choi, Junho Shin, and Hyoung-Kyu Song. 2024. Shortened LLaMA: A simple depth pruning for large language models. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*.

Dongyoung Kim, Junwhan Ahn, and Sungjoo Yoo. 2018. Zena: Zero-aware neural network accelerator. *IEEE Design & Test*, 35:39–46.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *Preprint*, arXiv:1810.09305.

Eldar Kurtic, Daniel Campos, Tuan Nguyen, Elias Frantar, Mark Kurtz, Benjamin Fineran, Michael Goin, and Dan Alistarh. 2022. The optimal BERT surgeon: Scalable and accurate second-order pruning for large language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 4163–4181. Association for Computational Linguistics.

Nahush Lele, Arnav Chavan, Aryamaan Thakur, and Deepak Gupta. 2025. Rethinking the value of training-free structured pruning of LLMs. *Transactions on Machine Learning Research*.

Hong Liu, Sang Michael Xie, Zhiyuan Li, and Tengyu Ma. 2023. Same pre-training loss, better downstream: Implicit bias matters for language models. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 22188–22214. PMLR.

Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. 2020. Fastbert: a self-distilling BERT with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6035–6044. Association for Computational Linguistics.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *Preprint*, arXiv:1609.07843.

Aishwarya Mirashi, Purva Lingayat, Srushti Sonavane, Tejas Padhiyar, Raviraj Joshi, and Geetanjali Kale. 2024. On importance of pruning and distillation for efficient low resource NLP. *CoRR*, abs/2409.14162.

J. Pablo Muñoz, Jinjie Yuan, and Nilesh Jain. 2025. Multipruner: Balanced structure removal in foundation models. *CoRR*, abs/2501.09949.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. 2024. Compact language models via pruning and knowledge distillation. In *Advances in Neural Information Processing Systems*, volume 37, pages 41076–41102. Curran Associates, Inc.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *ArXiv*, abs/1808.08745.

Nostalgebraist. 2020. interpreting gpt: the logit lens. *LessWrong*.

Seungcheol Park, Sojin Lee, Jongjin Kim, Jinsik Lee, Hyunjik Jo, and U Kang. 2025. Accurate sublayer pruning for large language models by exploiting latency and tunability information. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pages 8213–8221. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106.

Anthony Sarah, Sharath Nittur Sridhar, Maciej Szankin, and Sairam Sundaresan. 2024. Llama-nas: Efficient neural architecture search for large language models. In *Computer Vision - ECCV 2024 Workshops - Milan, Italy, September 29-October 4, 2024, Proceedings, Part XI*, volume 15633 of *Lecture Notes in Computer Science*, pages 67–74. Springer.

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Tran, Yi Tay, and Donald Metzler. 2022. Confident adaptive language modeling. In *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*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Weiqiao Shan, Long Meng, Tong Zheng, Yingfeng Luo, Bei Li, junxin Wang, Tong Xiao, and Jingbo Zhu. 2024. Early exit is a natural capability in transformer-based models: An empirical study on early exit without joint optimization.

Jiwon Song, Kyungseok Oh, Taesu Kim, Hyungjun Kim, Yulhwa Kim, and Jae-Joon Kim. 2024. SLEB: Streamlining LLMs through redundancy verification and elimination of transformer blocks. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 46136–46155. PMLR.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Florian Valade. 2024. Accelerating large language model inference with self-supervised early exits. *Preprint*, arXiv:2407.21082.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Xinpeng Wang, Bolei Ma, Chengzhi Hu, Leon Weber-Genzel, Paul Röttger, Frauke Kreuter, Dirk Hovy, and Barbara Plank. 2024. "My Answer is C": First-Token Probabilities Do Not Match Text Answers in Instruction-Tuned Language Models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7407–7416, Bangkok, Thailand. Association for Computational Linguistics.

Zhepei Wei, Wei-Lin Chen, Xinyu Zhu, and Yu Meng. 2024. Fast and accurate language model decoding via parallel token processing. In *NeurIPS Workshop on Adaptive Foundation Models: Evolving AI for Personalized and Efficient Learning*. Oral presentation.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2024. Sheared LLaMA: Accelerating language model pre-training via structured pruning. In *The Twelfth International Conference on Learning Representations*.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. Deebert: Dynamic early exiting for accelerating BERT inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2246–2251. Association for Computational Linguistics.

Takateru Yamakoshi, James McClelland, Adele Goldberg, and Robert Hawkins. 2023. Causal interventions expose implicit situation models for commonsense language understanding. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13265–13293, Toronto, Canada. Association for Computational Linguistics.

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, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Zheng Lin, Li Cao, and Weiping Wang. 2025b. Dynamic early exit in reasoning models. *Preprint*, arXiv:2504.15895.

Liangwei Yang, Yuhui Xu, Juntao Tan, Doyen Sahoo, Silvio Savarese, Caiming Xiong, Huan Wang, and Shelby Heinecke. 2025c. Entropy-based block pruning for efficient large language models. *Preprint*, arXiv:2504.03794.

Lu Yin, You Wu, Zhenyu Zhang, Cheng-Yu Hsieh, Yaqing Wang, Yiling Jia, Mykola Pechenizkiy, Yi Liang, Zhangyang Wang, and Shiwei Liu. 2024. Outlier weighed layerwise sparsity (OWL): A missing secret sauce for pruning LLMs to high sparsity. In *International Conference on Machine Learning, ICML 2024, 21-27 July 2024, Vienna, Austria*, volume 225 of *Proceedings of the 41st International Conference on Machine Learning*, pages 57101–57115. PMLR.

Alexander Yom Din, Taelin Karidi, Leshem Choshen, and Mor Geva. 2024. Jump to conclusions: Short-cutting transformers with linear transformations. In *Proceedings of the Joint International Conference on Computational Linguistics and Language Resources (LREC-COLING 2024)*, pages 9615–9625, Torino, Italy. ELRA and ICCL.

Ofir Zafrir, Ariel Larey, Guy Boudoukh, Haihao Shen, and Moshe Wasserblat. 2021. Prune once for all: Sparse pre-trained language models. In *NeurIPS*

2021 Workshop on Efficient Natural Language and Speech Processing (ENLSP).

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4791–4800. Association for Computational Linguistics.

Chao Zeng, Songwei Liu, Shu Yang, Fangmin Chen, Xing Mei, and Lean Fu. 2024a. GQSA: group quantization and sparsity for accelerating large language model inference. *CoRR*, abs/2412.17560.

Hansi Zeng, Kai Hui, Honglei Zhuang, Zhen Qin, Zhenrui Yue, Hamed Zamani, and Dana Alon. 2025. Can pre-training indicators reliably predict fine-tuning outcomes of llms? *Preprint*, arXiv:2504.12491.

Ziqian Zeng, Yihuai Hong, Hongliang Dai, Huiping Zhuang, and Cen Chen. 2024b. Consistentee: a consistent and hardness-guided early exiting method for accelerating language models inference. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press.

Shuai Zhang, Meng Wang, Sijia Liu, Pin-Yu Chen, and Jinjun Xiong. 2021. Why lottery ticket wins? A theoretical perspective of sample complexity on sparse neural networks. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 2707–2720.

Yang Zhang, Yawei Li, Xinpeng Wang, Qianli Shen, Barbara Plank, Bernd Bischl, Mina Rezaei, and Kenji Kawaguchi. 2024. Finercut: Finer-grained interpretable layer pruning for large language models. In *Workshop on Machine Learning and Compression, NeurIPS 2024*.

Bowen Zheng, Ming Ma, Zhongqiao Lin, and Tianming Yang. 2025. A hybrid early-exit algorithm for large language models based on space alignment decoding (spade). *Preprint*, arXiv:2507.17618.

Longguang Zhong, Fanqi Wan, Ruijun Chen, Xiaojun Quan, and Liangzhi Li. 2025. Blockpruner: Fine-grained pruning for large language models. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria and virtual meeting, July 27–August 1, 2025*, page 5065–5080. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, page 19–27, USA. IEEE Computer Society.

A Additional Details on Depth Pruning and Probing Methods

A.1 Background on Transformer Blocks and Instability of Layer Pruning

To clarify the distinction among pruning approaches, we briefly review the structure of a transformer block and discuss the instability observed in layer pruning. A transformer block comprises a multi-headed self-attention (MHSA) module, a feed-forward network (FFN), layer normalization, and residual connections (Vaswani et al., 2017). These components are stacked to form a deep hierarchical model in which each block refines contextual representations from preceding layers.

While width pruning removes individual heads within each MHSA module or neuron groups within each FFN, depth pruning eliminates structural units along the model's depth, ranging from individual layers to entire blocks (Yin et al., 2024; Kim et al., 2024; Muñoz et al., 2025). Layer pruning targets specific subcomponents such as MHSA or FFN within each block, thereby skipping selected operations at a finer granularity (He et al., 2024; Zhang et al., 2024). In contrast, block pruning removes the entire transformer block, including its MHSA, FFN, normalization, and residual pathways (Zhong et al., 2025; Song et al., 2024; Yang et al., 2025c).

In practice, layer pruning often causes severe instability, as partial removal of a block disrupts normalization and residual dependencies. For instance, when pruning 25% of layers, the representative layer-level baseline JointLayerDrop (He et al., 2024) diverged and failed to produce valid perplexity values, yielding `NaN`. This instability illustrates that partial block removal breaks the transformer's internal functional coherence, leading to divergence during evaluation.

A.2 Complexity of Iterative Sensitivity-based Pruning

Consider the case of pruning eight blocks from a model with 32 blocks. To identify the optimal set at a pruning ratio of 25%, one would need to evaluate all possible combinations $\binom{32}{8}$, which corresponds to nearly ten million experiments. Instead of conducting such an exhaustive grid search, SLEB em-

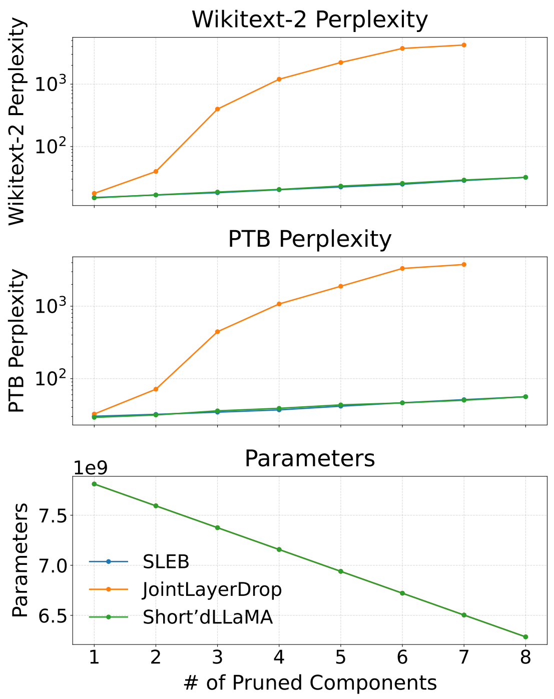

Figure 3: Perplexity comparison on LLaMA-3-8B between block-level and layer-level depth pruning. For layer pruning, two layers are grouped into a single block (*e.g.*, 16 layers → 8 blocks). At a pruning ratio of 25%, JointLayerDrop diverged, resulting in invalid (NaN) perplexity scores.

ploys a greedy search strategy by pruning the block with the lowest sensitivity score at each iteration, re-estimating the sensitivities of the remaining blocks, and repeating this process until the desired pruning ratio is reached (Song et al., 2024). In this greedy procedure, the total number of experiments is

$$32 + 31 + \cdots + 25 = \sum_{k=25}^{32} k = 216, \qquad (4)$$

which is substantially smaller and allows the method to reach a local optimum with far fewer evaluations, though it still requires considerable computational effort and time.

Empirical computational cost. Beyond the brute-force search complexity discussed above, we further examine the empirical computational cost of block importance estimation. The wall-clock time required for this step was measured using a single NVIDIA A100 GPU under identical experimental settings. Fig. 4 reports the measured time for TaBP and SLEB. TaBP requires 72 seconds to estimate block importance, whereas SLEB requires 227.5 seconds in total. The runtime of SLEB accumulates over eight greedy pruning phases, with

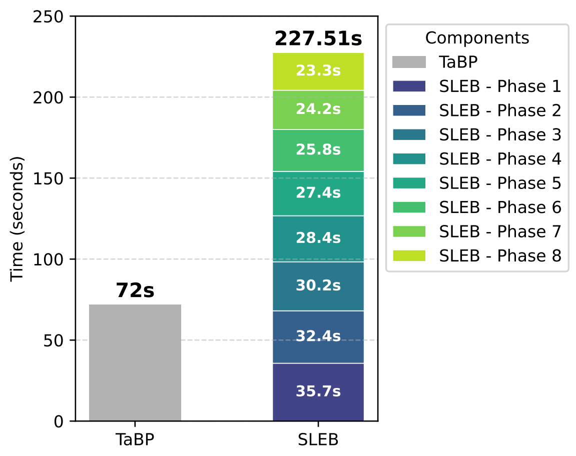

Figure 4: Wall-clock time for block importance estimation on a single NVIDIA A100 GPU. SLEB incurs significantly higher runtime due to iterative sensitivity re-estimation across pruning phases.

each phase involving repeated forward passes to re-estimate block sensitivities. In contrast, the runtime of TaBP remains constant regardless of the number of blocks to prune, since TaBP computes importance scores for all blocks in a single pass and then ranks them. As a result, the time complexity of TaBP depends on the total number of transformer blocks in the model and the size of the calibration set, rather than on the target pruning ratio.

This runtime gap becomes more pronounced in resource-constrained environments. For example, when the experiments are distributed across multiple NVIDIA Titan XP GPUs (4 GPUs with 12GB VRAM each), inter-GPU communication overhead further amplifies the cost of iterative sensitivity re-estimation in SLEB significantly. As a result, the relative runtime difference between TaBP and SLEB exceeds the ratio observed in the single-GPU setting. Overall, these results confirm the higher practical cost of iterative sensitivity-based pruning compared to the single-pass estimation used by TaBP.

A.3 Details of Probing Methods

LogitLens shows that hidden states can be linearly mapped into vocabulary distributions using the final projection matrix without additional training (Nostalgebraist, 2020; Shan et al., 2024). To improve the output quality for better analysis, TunedLens further enhances this by training block-specific affine LM heads as probes, demonstrating that while hidden representations encode predictive information on intermediate blocks, it is not directly decodable without auxiliary models (Belrose et al., 2025; Chuang et al., 2024). Yom Din et al. (2024) learn linear transformations to project hidden states into the final representation space, further supporting the claim that additional modeling is necessary to faithfully reveal latent predictions.

B Implementation Details

All experiments are implemented in PyTorch using Huggingface Transformers on four Nvidia Titan Xp GPUs, each with 12GB of memory. Evaluation was executed with `EleutherAI/lm-evaluation-harness` library.

For the general text generation in ablation study, models' behavior was observed using Wikitext-2. Since no answer key is specified in text generation, the next token was treated as the target and inference proceeded without logits processors. Statistical values were computed over the first 1,024 tokens of Wikitext-2.

Although ShortenedLLaMA also introduced a Taylor-based metric, only PPL sensitivity was adopted in comparison experiments due to the high computational overhead of gradient calculation.

For TaBP, the former half of the transformer blocks were excluded from pruning, following previous studies (Kim et al., 2024; Song et al., 2024), which reported that preserving the first few blocks is critical for performance preservation.

B.1 Dataset Configuration for LM Head Training.

To enhance generalization and robustness of block importance estimation, we constructed a mixed corpus from seven publicly available datasets: OpenWebText (Radford et al., 2019), BookCorpus (Zhu et al., 2015), ArXiv (Cohan et al., 2018), CNN/DailyMail (Hermann et al., 2015; See et al., 2017), XSUM (Narayan et al., 2018), WikiHow (Koupaee and Wang, 2018), and Alpaca (Taori et al., 2023). From each dataset, we used 200 samples (1,400 in total) and shuffled the concatenated set. This mixture spans open-domain, academic, instructional, and summarization styles, providing heterogeneous linguistic patterns for LM head training and improving generalization across text types.

B.2 Experimental Setup for Post-Pruning Fine-Tuning

Base model and pruning. All experiments are conducted on LLaMA-3-8B. We prune 8 transformer blocks, corresponding to a pruning ratio of 25%, using either TaBP or SLEB. Pruning is applied prior to fine-tuning, and the resulting pruned models are used as initialization for all subsequent fine-tuning experiments.

Fine-tuning configuration. To evaluate recovery through lightweight adaptation, we fine-tune the pruned models using the Alpaca dataset (Taori et al., 2023) with LoRA (Hu et al., 2022). We evaluate SLEB with two LoRA ranks (8 and 64) to examine whether increased adaptation capacity can improve its recovery, as SLEB shows weaker recoverabilty after pruning. TaBP is trained only with rank 8. Learning rates are selected through extensive empirical tuning for each configuration, and results are reported using the best-performing setting based on downstream task performance. Specifically, we use 2×10^{-5} for SLEB (rank 8), 1×10^{-5} for SLEB (rank 64), and 5×10^{-5} for TaBP (rank 8).

Training schedule and checkpoint selection. TaBP-pruned models are fine-tuned for 3 epochs, while SLEB-pruned models are fine-tuned for 3 and 10 epochs to ensure sufficient optimization. For each method, results are reported using the checkpoint that achieves the minimum validation loss over the entire training run, rather than the final epoch, to ensure a fair comparison of recovery potential.

C Granularity of Depth Pruning

As shown in Table 1 and 2, layer pruning can appear more stable than block pruning. This behavior arises from its fine-grained nature, which removes MHSA modules in the early stages and later includes FFNs in the pruning targets, where the former is known to have relatively minor influence (He et al., 2024). As the pruning ratio increases, however, the removal of interconnected components disrupts normalization and residual dependencies, leading to divergence, as also illustrated in Fig. 3. Hence, layer pruning may exhibit transient robustness at low sparsity but becomes increasingly unstable as pruning deepens. Moreover, since pruning only parts of a block compromises the functional integrity of transformer blocks, the pruned model cannot run properly unless the forward pass in the model implementation is manually modified.

Criteria (desirability)	Formula	#probs.	#dists.
Confidence score (maximum probability) ↑	$\mathrm{CONF}(x) = \max_{y \in V} p(y \mid x)$	1	1
Key probability (assigned to ground-truth y^*) ↑	$K(x) = p(y^* \mid x)$	1	1
Gap between top-1 and top-2 probabilities ↑	$G(x) = p(y_{(1)} \mid x) - p(y_{(2)} \mid x)$	2	1
Entropy over the token distribution ↓	$\mathrm{ENT}(x) = -\sum_{y \in V} p(y \mid x) \log p(y \mid x)$	All	1
Cross-entropy between output distributions ↓	$\mathrm{CE}(p, q) = -\sum_{y \in V} p(y \mid x) \log q(y \mid x)$	All	2
Kullback-Leibler divergence ↓	$\mathrm{KLD}(p \parallel q) = \sum_{y \in V} p(y \mid x) \log \frac{p(y \mid x)}{q(y \mid x)}$	All	2
Jensen-Shannon divergence ↓	$\mathrm{JSD}(p \parallel q) = \frac{1}{2}\,\mathrm{KLD}(p \parallel m) + \frac{1}{2}\,\mathrm{KLD}(q \parallel m)$	All	2

Table 4: Distribution-based criteria for the statistical value of a calibration sample, with their definitions and scopes. Here, p denotes the output distribution of the final block, whereas q denotes the output distribution of an intermediate block, and m is their average distribution.

D Statistical Measures

Table 4 summarizes the candidate statistical measures used to assess block-wise importance, including confidence score (Valade, 2024; Yang et al., 2025b), the gap between the top and secondary probabilities. (Schuster et al., 2022; Valade, 2024), and the entropy over all token probabilities in the distribution (Xin et al., 2020; Liu et al., 2020; Hu et al., 2023; Valade, 2024). Each measure captures models' internal behavior based on token probability distributions and is applied via aggregation methods. Ablation results in Table 5 report average accuracy after pruning and the area under the curve (AUC) across pruning ratios from 0% to 25% on LLaMA-3-8B. Among all candidates, entropy (ENT) consistently demonstrates strong performance across tasks, particularly in the MCQA setting, highlighting its reliability. Other criteria such as confidence score (CONF), gap (G), cross-entropy (CE), and KL-divergence (KLD) yield comparable yet slightly lower accuracy. Based on these findings, entropy is adopted as the default criterion in the main experiments, while other options remain viable depending on the application context.

E Additional Analysis on Calibration Dataset

To validate our method across the datasets and examine dataset-dependent tendencies, we further analyzed the effect of the calibration dataset and the choice of task-specific criteria in TaBP with LLaMA-3-8B.

Calibration dataset dependence. We observe that TaBP exhibits limited sensitivity to the choice of the calibration dataset. In particular, calibrating TaBP on a specific dataset does not consistently lead to improved zero-shot performance on the corresponding evaluation task. For example, using ARC-Easy or HellaSwag as the calibration dataset does not result in systematically higher accuracy on their respective test sets compared to calibrating on other datasets. These observations suggest that TaBP is not strongly dependent on the specific calibration data and instead captures pruning criteria that generalize across tasks.

We note an empirical caveat that reliable analysis requires calibration datasets with more than two answer choices. When the number of choices is limited to two, inherent model biases arising from binary choice settings (*e.g.*, 0/1, 1/2, or A/B labeling) can confound the analysis, as models may exhibit asymmetric preferences toward specific label positions or symbols (Wang et al., 2024). Moreover, with only two categories, the resulting distribution is too constrained for statistical measures with larger support, such as entropy, to be informative.

Impact of task-specific criteria. In Table 6, the reported results correspond to the best-performing configurations for each dataset with DDF. Specifically, entropy is used for ARC-Easy and ARC-Challenge, while gap is used for the remaining benchmarks, as these settings yield the strongest empirical performance among the seven criteria. While a more systematic investigation is required to draw definitive conclusions, we attribute this divergence to the distinct evaluation focus of each dataset. ARC-Easy and ARC-Challenge primarily assess scientific and logical reasoning (Clark et al., 2018), whereas the other benchmarks target different aspects of commonsense reasoning, such as situational plausibility and contextual inference (Zellers et al., 2019; Sakaguchi et al., 2021). These differences in evaluation objectives may favor dif-

Method			Avg.@2 ↑	Avg.@4 ↑	Avg.@8 ↑	AUC@8 ↑
SLEB			72.76	69.30	56.57	.6703
ShortenedLLaMA			73.52	65.32	56.60	.6633
EntroDrop			64.68	49.71	45.68	.4821
JointLayerDrop			62.37	59.54	56.05	.5286
LM Head	**Aggregation**	**Criteria**	**Avg.@2 ↑**	**Avg.@4 ↑**	**Avg.@8 ↑**	**AUC@8 ↑**
Frozen	DDF	CONF	70.12	68.40	**63.94**	.6832
		K	70.12	68.40	**63.94**	.6832
		G	70.12	67.72	60.98	.6742
		ENT	**75.04**	69.39	62.87	**.6922**
		CE	70.38	69.11	61.95	.6830
		KLD	70.12	68.48	**63.94**	.6833
		JSD	72.13	69.72	59.74	.6752
	SSN	CONF	66.13	64.90	55.36	.6245
		K	66.13	64.90	55.36	.6246
		G	67.09	65.97	56.24	.6316
		ENT	66.13	64.90	55.95	.6282
		CE	66.13	64.90	55.36	.6257
		KLD	66.13	64.90	57.88	.6323
		JSD	70.64	67.99	59.01	.6630
Trained	DDF	CONF	70.14	68.40	60.69	.6655
		K	70.14	68.40	60.69	.6655
		G	70.12	68.08	59.74	.6669
		ENT	74.59	**70.52**	60.93	.6913
		CE	70.12	68.33	62.57	.6708
		KLD	70.13	68.39	60.21	.6649
		JSD	70.11	67.72	59.74	.6610
	SSN	CONF	69.43	68.17	58.44	.6592
		K	69.43	68.17	58.44	.6592
		G	70.38	67.42	59.51	.6605
		ENT	69.43	68.17	59.54	.6602
		CE	69.43	68.17	58.44	.6592
		KLD	69.43	68.17	60.23	.6666
		JSD	69.43	67.42	59.51	.6622

Table 5: Average of zero-shot accuracy (%) of LLaMA-3-8B after pruning eight blocks and AUC over pruning ratio from 0% to 25%. For TaBP, calibration is performed using only the ARC-Easy dataset.

ferent pruning criteria.

To further understand this behavior, upon closer inspection, although both entropy and gap reflect model confidence, they operate at different scopes of the output distribution, which leads to different effectiveness across datasets. Entropy captures uncertainty over the entire set of answer choices, reflecting how well the model globally understands a given problem. In contrast, gap focuses on the relative margin between the top-ranked candidates, capturing how strongly the model prefers one option over the next best alternative.

This distinction is particularly relevant for ARC-Easy and ARC-Challenge, which primarily evaluate factual and scientific reasoning. For these tasks, when the model possesses the relevant knowledge, the output distribution typically concentrates sharply on the correct answer, yielding low entropy (Borchmann, 2025). Conversely, when the required knowledge or reasoning is missing, probability mass tends to spread across multiple choices, resulting in high entropy. As a result, entropy are well suited to detecting structural degradation in these datasets, as pruning-induced damage directly manifests as increased global uncertainty.

In contrast, datasets such as WinoGrande and HellaSwag emphasize comparative and contextual judgments. In these settings, multiple answer options can remain plausible even when the model identifies the correct one. Consequently, the output distribution may retain relatively high entropy, making entropy less effective. In such cases, the gap between the top two candidates more directly reflects the model's decision boundary, *i.e.*, the margin separating the top candidates. Thus, focusing on relative, local preferences is sufficient and can be more informative for capturing pruning effects (Yamakoshi et al., 2023).

Method	ARC-E ↑	ARC-C ↑	BoolQ ↑	COPA ↑	HeSw. ↑	PIQA ↑	Wino. ↑	Avg. ↑
Original Model	80.09	53.33	81.44	89.00	79.17	79.71	72.85	76.51
Prune 2 blocks (pruning ratio = 6.25%)								
SLEB	**76.60**	48.29	74.80	87.00	74.33	77.80	70.48	72.76
TaBP-Wino.	76.05	**50.17**	80.70	87.00	**76.25**	**78.07**	73.64	74.55
TaBP-HeSw.	73.65	48.38	67.52	**89.00**	74.51	76.55	73.48	71.87
TaBP-ARC-C	76.05	**50.17**	80.70	87.00	**76.25**	**78.07**	73.64	74.55
TaBP-ARC-E	76.01	**50.17**	**80.76**	87.00	76.24	**78.07**	**73.88**	**74.59**
Prune 4 blocks (pruning ratio = 12.5%)								
SLEB	**72.47**	43.26	66.12	**87.00**	70.69	**76.12**	69.46	69.30
TaBP-Wino.	67.05	46.42	62.08	76.00	69.86	74.10	67.48	66.14
TaBP-HeSw.	66.88	45.90	79.97	77.00	66.27	73.23	65.19	67.78
TaBP-ARC-C	70.50	46.25	**80.52**	82.00	**74.17**	76.06	**72.53**	**71.72**
TaBP-ARC-E	71.46	**47.44**	73.21	81.00	72.05	75.95	**72.53**	70.52
Prune 8 blocks (pruning ratio = 25%)								
SLEB	**61.78**	31.74	42.11	**77.00**	57.98	**71.82**	53.59	56.57
TaBP-Wino.	50.67	38.05	63.18	74.00	**61.19**	69.26	67.01	60.48
TaBP-HeSw.	57.95	39.76	**80.73**	73.00	54.32	69.53	61.64	**62.42**
TaBP-ARC-C	53.75	**40.61**	64.40	72.00	59.34	68.55	67.64	60.90
TaBP-ARC-E	53.83	**40.61**	64.34	72.00	59.35	68.61	**67.79**	60.93

Table 6: Zero-shot accuracy (%) comparison of LLaMA-3-8B on each task after pruning by our method (TaBP) and the state-of-the-art baseline SLEB . Best and the second best results are indicated as bold and underline, respectively.

Method	ARC-E	ARC-C	BoolQ	COPA	HeSw.	PIQA	Wino.	Avg.
Prune 8 blocks (pruning ratio = 25%)								
SLEB	61.78	31.74	42.11	77.00	57.98	71.82	53.59	56.57
TaBP	53.83	40.61	64.34	72.00	59.35	68.61	67.79	60.93
Fine-tune with Alpaca (LoRA) for 3 epochs								
SLEB (Rank 8)	63.26	35.15	40.89	79.00	58.36	71.98	53.43	57.44
SLEB (Rank 64)	63.13	33.70	41.77	82.00	58.47	71.82	52.80	57.67
TaBP (Rank 8)	59.47	41.47	68.96	76.00	61.72	68.82	70.64	**63.87**
Fine-tune with Alpaca (LoRA) for 10 epochs								
SLEB (Rank 8)	64.02	36.26	43.73	79.00	58.79	71.22	54.22	58.18
SLEB (Rank 64)	64.69	35.84	41.74	82.00	59.08	72.14	52.88	58.34

Table 7: Performance of LLaMA-3-8B pruned by our method (TaBP) and the state-of-the-art baseline SLEB, and subsequently fine-tuned. Results are reported at the checkpoint with the minimum validation loss over the training epochs (3 for TaBP, 10 for SLEB).

Taken together, these observations suggest that the differing effectiveness of entropy and gap arises from their distinct scopes: entropy reflects global distributional confidence that is critical for knowledge-intensive reasoning tasks, whereas gap captures local decision margins that are more informative for comparative and commonsense reasoning benchmarks. While this explanation is supported by consistent empirical trends in our experiments, we leave a more systematic analysis of this phenomenon for future work.

Generalization across datasets. Importantly, the effectiveness of TaBP is not limited to a specific benchmark. Although certain criteria align well with particular task families, TaBP consistently achieves competitive or superior performance across a diverse set of reasoning and commonsense tasks. This suggests that the proposed pruning framework and associated criteria capture general properties that extend beyond ARC-style benchmarks. Overall, these observations indicate that TaBP is robust to the choice of calibration dataset and that its task-aware criteria generalize across different benchmark categories, while leaving room for future work to more systematically characterize the relationship between task properties and optimal pruning criteria.

F Recovery Ability after Pruning

This section analyzes the recovery ability of pruned models through fine-tuning, comparing our method (TaBP) with the state-of-the-art baseline

SLEB. Recovery ability refers to how effectively a pruned model regains downstream task performance through additional training. Table 7 summarizes the recovery performance of pruned LLaMA-3-8B models before and after fine-tuning.

Without fine-tuning, TaBP already achieves a higher average performance than SLEB (60.93 vs. 56.57), indicating that TaBP better preserves task-relevant representations during pruning. After fine-tuning with Alpaca using LoRA, TaBP consistently demonstrates stronger recovery than SLEB, even in the more constrained settings. With LoRA rank 8, TaBP improves its average score to 63.87, substantially outperforming SLEB at both rank 8 (58.18) and rank 64 (58.34), despite the latter being trained for more epochs. Increasing the LoRA rank for SLEB yields only marginal improvements, whereas TaBP achieves higher absolute performance with a lower-rank adaptation.

These results suggest that TaBP produces pruned models that are more amenable to downstream recovery. Overall, the results demonstrate that TaBP not only yields stronger performance immediately after pruning but also provides superior recovery potential under lightweight fine-tuning regimes.