Sample-aware Adaptive Structured Pruning for Large Language Models

Jun Kong, Xinge Ma, Jin Wang*, Xuejie Zhang

School of Information Science and Engineering
Yunnan University
Kunming, China
{kongjun, maxinge}@mail.ynu.edu.cn, {wangjin, xjzhang}@ynu.edu.cn

Abstract

Large language models (LLMs) have achieved outstanding performance in natural language processing, but enormous model sizes and high computational costs limit their practical deployment. Structured pruning can effectively reduce the resource demands for deployment by removing redundant model parameters. However, the randomly selected calibration data and fixed single importance estimation metrics in existing structured pruning methods lead to degraded performance of pruned models. This study introduces AdaPruner, a sample-aware adaptive structured pruning framework for LLMs, aiming to optimize the calibration data and importance estimation metrics in the structured pruning process. Specifically, AdaPruner effectively removes redundant parameters from LLMs by constructing a structured pruning solution space and then employing Bayesian optimization to adaptively search for the optimal calibration data and importance estimation metrics. Experimental results show that the AdaPruner outperforms existing structured pruning methods on a family of LLMs with varying pruning ratios, demonstrating its applicability and robustness. Remarkably, at a 20% pruning ratio, the model pruned with AdaPruner maintains 97% of the performance of the unpruned model.

Code — https://github.com/JunKong5/AdaPruner

Introduction

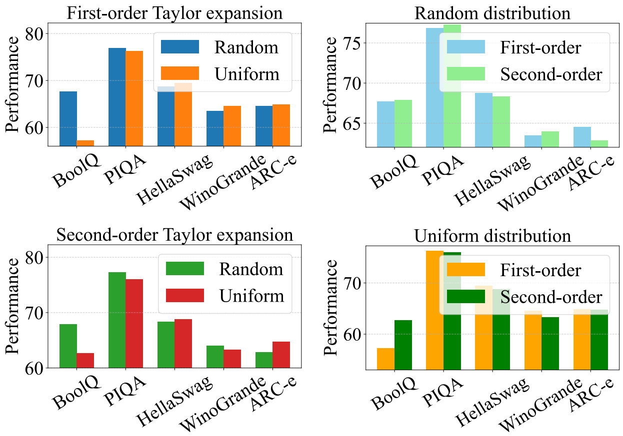
Large language models (LLMs), such as GPT-3 (Brown et al. 2020), OPT (Zhang et al. 2022), LLaMA (Touvron et al. 2023) and Vicuna (Chiang et al. 2023), have demonstrated significant accomplishments in the realm of natural language processing (Wei et al. 2022; Wu et al. 2020). Nevertheless, their exceptional capabilities are coupled with substantial model sizes and elevated computational expenses. Furthermore, owing to the scaling law (Hoffmann et al. 2022; Kaplan et al. 2020), LLMs tend to enhance model performance by progressively augmenting model parameters. Regrettably, larger model sizes entail heightened consumption of computational resources, presenting a notable obstacle to their practical deployment, particularly in settings with limited resources.

*Corresponding author.

Structured pruning (Xia et al. 2024) emerges as a pivotal technique for mitigating resource demands in the deployment of LLMs. In comparison to other strategies (Zhu et al. 2023) like unstructured pruning (Yin et al. 2023; Frantar and Alistarh 2023; Jaiswal et al. 2023), model quantization (Liu et al. 2023; Xiao et al. 2023), and knowledge distillation (Gu et al. 2024; Yuan et al. 2023; Hsieh et al. 2023), structured pruning not only provides a practical and hardware-independent solution but also offers an effective approach to streamline LLMs implementation on devices with limited computational resources. Its unique ability to selectively remove redundant model parameters while maintaining model integrity positions structured pruning as a cornerstone in optimizing the efficiency of LLMs deployment.

However, existing work on structured pruning commonly employs Taylor expansion as the metric for estimating the importance of structures. These methods hinge on a localized approximation of the loss function, necessitating additional calibration data for gradient information computation. As a result, the precision of the gradient is directly tied to the calibration data's quality, thereby influencing both the Taylor expansion approximation and the pruning decision. Subpar calibration data and unsuitable importance estimation metrics can lead to substantial performance degradation in pruned models. Thus, ensuring the quality of the calibration data and employing appropriate importance estimation metrics are crucial for effective structured pruning.

Hence, the Taylor expansion-based methodologies for structured pruning LLMs pose two notable challenges. Acquiring high-quality calibration data covering diverse parameters and sample spaces is crucial for accurate gradient computation. It aids in identifying parameters effectively during pruning, reducing computational overhead while preserving generalization. However, randomly selected calibration data lacks representativeness and diversity, leading to inaccurate gradient information and flawed pruning strategies. A preliminary experiment in Figure 1(a) examines the impact of different calibration data distributions on performance. It illustrates that uniformly distributed calibration data may degrade performance in some tasks and improve it in others compared to randomly distributed data. In addition, it highlights the significant influence of calibration data distributions on performance. Randomly selecting calibration data is considered suboptimal.



(a) Calibration data (b) Importance estimation

Figure 1: The impact of calibration data and importance estimation metrics.

Further, prevailing structured pruning approaches typically employ predetermined heuristic metrics to appraise sub-structure significance. However, these metrics often fall short of comprehensively assessing the model parameters. Furthermore, their disregard for alignment with calibration data results in inaccurate estimation of model weight importance, consequently undermining model performance. This deficiency can be ascribed to the reliance on model parameter importance estimation on gradient information, which is sensitive to variations in calibration data. Consequently, disparate calibration datasets lead to varying gradient magnitudes, causing parameters previously deemed crucial to lose importance with new calibration data and vice versa. As depicted in Figure 1(b), distinct importance estimation metrics exhibit varying sensitivity to calibration data. Thus, selecting compatible importance estimation metrics tailored to high-quality calibration data is imperative to mitigate performance degradation.

This study introduces AdaPruner, a sample-aware adaptive structured pruning framework for LLMs. AdaPruner aims to optimize both the calibration data and importance estimation simultaneously. Firstly, AdaPruner creates a structured pruning space containing a calibration data subspace and an importance estimation metrics subspace that are closely linked. Secondly, AdaPruner uses Bayesian optimization to search for high-quality calibration data and importance estimation metrics within this space. Lastly, AdaPruner conducts efficient pruning based on the obtained data and metrics, removing unnecessary structures from LLMs. AdaPruner streamlines the tedious manual design process and ensures stable performance by enabling the transfer of calibration data and importance estimation metrics to different pruning ratios and models.

The contributions can be summarized as follows:

- AdaPruner introduces a sample-aware approach to structured pruning, delineating interconnected calibration data and importance estimation metrics subspaces, enhancing their utilization in the pruning process.
- AdaPruner facilitates rapid and effective pruning by utilizing acquired calibration data and importance estimation metrics to eliminate redundant LLMs structures, reducing manual effort and ensuring consistent performance across different pruning rates and models.
- We conduct extensive experiments on a variety of language benchmarks. The AdaPruner outperforms the existing methods on the LLaMA series models, achieving superior average performance over the LLM-Pruner by 1.37%. The experimental results demonstrate the effectiveness of the proposed AdaPruner.

Sample-aware Adaptive Structured Pruning

This section introduces AdaPruner, a sample-ware adaptive structured pruning framework. As shown in Figure 2, AdaPruner aims to adaptively obtain the optimal calibration data and importance estimation metrics through Bayesian optimization to improve the performance of pruned LLMs.

Structured Pruning for LLMs

First, according to LLM-Pruner (Ma, Fang, and Wang 2023), structured pruning considers the dependencies between structures. Therefore, the dependency automatically identifies and groups coupling structures in LLMs.

Second, structured pruning is performed based on Taylor expansion. Specifically, given a calibration dataset $D = \{x_i\}_{i=1}^{N}$, where N is the number of samples contained in the calibration dataset, and the subsequent token prediction loss of the LLMs parameterized by θ on the calibration dataset is:

$$L(\theta, D) = \frac{1}{N}\sum_{i-1}^{N} F(\theta, x_i) \tag{1}$$

where $F(\theta, x_i)$ represents the subsequent token prediction loss of the sample x_i on the LLM.

The importance of θ_i can be quantified by measuring the impact of its removal on the loss. Structured pruning methods typically estimate the effect of removing parameters through Taylor expansion as follows:

$$\begin{aligned} I_{\theta_i} &= |L(\theta_i, D) - L(\theta_i = 0, D)| \\ &\approx |\frac{\partial L(\theta_i, D)}{\partial \theta_i}\theta_i - \frac{1}{2}{\theta_i}^{\top} H \theta_i + \mathrm{O}(\| \theta_i \|^3)| \end{aligned} \tag{2}$$

where $L(\theta_i = 0, D)$ denotes removing θ_i from the LLM, H denotes the Hessian matrix, and $\mathrm{O}(\| \theta_i\|^3)$ denotes the residual term that can be ignored in the calculation. Then, the importance of the structural group is obtained by aggregating the importance of the structural parameters within the group, i.e., $I(G) = \underset{i=1}{\overset{M}{Agg}}\, I(\theta_i)$. Here, Agg denotes aggregation functions such as summation, product, maximum, and last only. M denotes the number of structures in the group. Finally, the groups with lower importance scores are removed based on a predefined pruning ratio.

Solution Space for Structured Pruning

We jointly design a structured pruning solution space consisting of the calibration data subspace and the importance estimation metrics subspace.

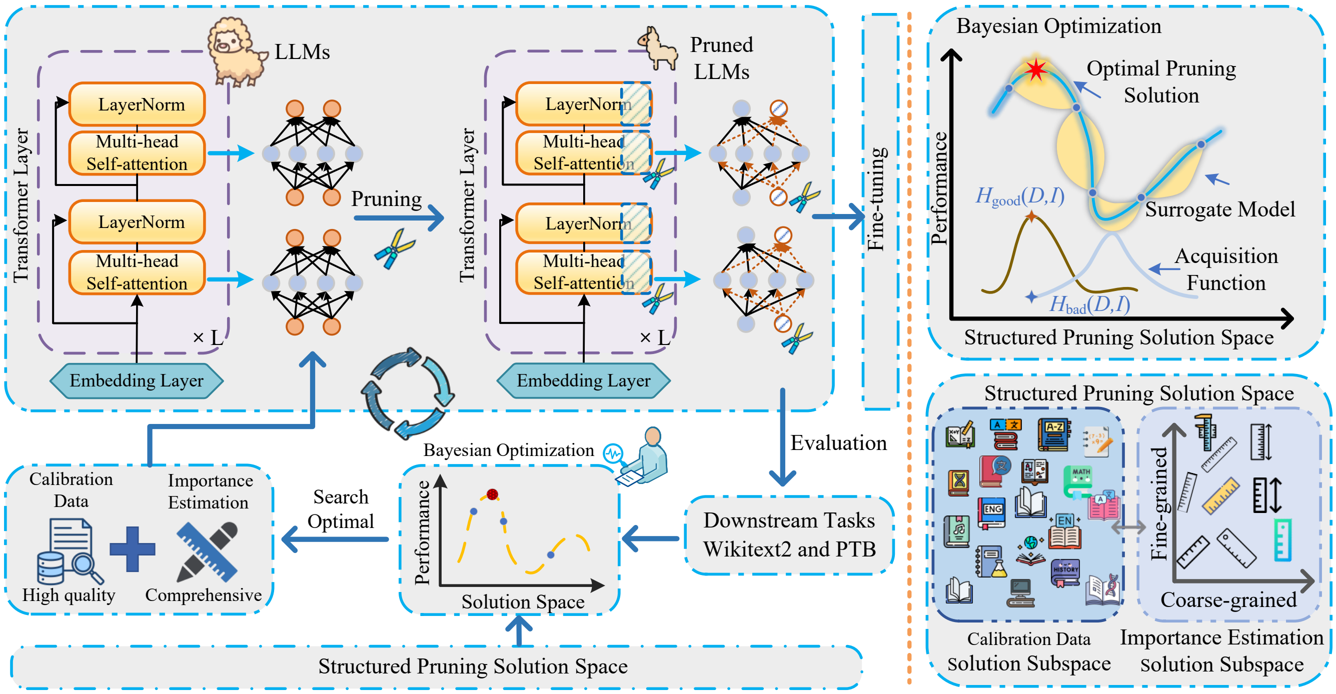


Figure 2: Overview of the AdaPruner Framework.

Subspace of calibration data. Due to the restricted access to the pre-training datasets, existing studies usually sample public datasets and randomly select a small portion of samples from them as the calibration data. Following LLM-Pruner, we construct the subspace of the calibration data using samples from the BookCorpus dataset, as shown in Figure 2. Meanwhile, the subspace solution of the calibration data is effectively reduced by filtering out the short texts with little information, accelerating the search process.

Subspace of importance estimation metrics. To comprehensively assess the importance of LLMs parameters, the importance estimation subspace comprises different granularity importance estimation metrics based on Taylor expansion, i.e., coarse-grained weight vectors and fine-grained weight element importance estimation metrics. The coarse-grained metric, as shown in Eq. (2), is used to estimate the importance of the overall structure weights. The fine-grained metric is introduced as follows:

$$\begin{aligned} I_{\theta_i^k} &= |L(\theta_i^k, D) - L(\theta_i^k = 0, D)| \\ &\approx |\frac{\partial L(\theta_i^k, D)}{\partial \theta_i^k}\theta_i^k - \frac{1}{2}{\theta_i^k}^{\top} H_{kk}\theta_i^k + \mathrm{O}(\| \theta_i^k\|^3)| \end{aligned} \tag{3}$$

where k denotes the index of the parameter. Due to the Hessian matrix H_{kk} can be approximated by the Fisher information matrix (Kwon et al. 2022), It can be redefined as:

$$\begin{aligned} I_{\theta_i^k} &= |L(\theta_i^k, D) - L(\theta_i^k = 0, D)| \\ &\approx |\frac{\partial L(\theta_i^k, D)}{\partial \theta_i^k}\theta_i^k - \frac{1}{2}\sum_{j=1}^{N}\left(\frac{\partial L(\theta_i^k, D)}{\partial \theta_i^k}\theta_i^k\right)^2 + \mathrm{O}(\| \theta_i^k\|^3)| \end{aligned} \tag{4}$$

The granularity of the weight vector is too coarse, while the granularity of the weight element is too fine. Therefore, balancing weight vector and weight element metrics is necessary. Therefore, the importance estimation metrics subspace is defined as follows:

$$I_{\theta_i, \theta_i^k} = \alpha_1 I_{\theta_i} + \alpha_2 I_{\theta_i^k} \tag{5}$$

where α_1 and α_2 are equilibrium coefficients for coarse- and fine-grained importance estimations, respectively. However, there is a significant order of magnitude difference between the manually designed first- and second-order gradients. We further extend the optimization of the alignment factor.

Adaptive Pruning via Bayesian Optimization

The effects of calibration data and importance estimation metrics on the performance of pruned models are interrelated. Independently optimizing either element may lead to a locally optimal solution. Therefore, AdaPruner uses Bayesian optimization to adaptively obtain optimal calibration data and importance estimation metrics in the structured pruning solution space. Specifically, the pruned model $f(\theta, D, I, \lambda)$ is evaluated with the dataset D_E, e.g., using the perplexity of the evaluation dataset as an evaluation metric. Thus, the optimization objective of LLMs pruning can be formulated as follows:

$$H = \min_{D, I \in Z} h(f(\theta, D, I, \lambda), D_E) \tag{6}$$

where $h(\cdot)$ denotes the evaluation metric, λ denotes the pruning ratio, and D and I denote the calibration data and importance estimation metrics to search, respectively. The pruned model without fine-tuning is used to evaluate the performance on a smaller evaluation set, such as Wik-iText2 and PTB, to minimize the evaluation cost. Their average perplexity is utilized as a metric. Lower perplexity scores indicate superior language modeling capabilities, reflecting a more effective pruning metric regarding mod-el performance.

Since $h(f(\theta, D, I, \lambda), D_E)$ is a black-box function, AdaPruner utilizes Bayesian Optimization with Tree-Structured Parzen Estimator (BO-TPE) to globally optimize Eq. (6) to obtain the optimal calibration data and importance estimation metrics, as shown in Figure 2. BO-TPE is an iterative process designed to optimize the objective function $h(f(\theta, D, I, \lambda), D_E)$ from the structured pruning solution space Z. It adaptively obtains the optimal calibration data and importance estimation metrics. It uses a probabilistic surrogate model M to estimate the objective function and updates the posterior estimate of $h(f(\theta, D, I, \lambda), D_E)$ based on each search step's results. Specifically, given the previous t-1 search results $\{(D_1, I_1), ..., (D_{t-1}, I_{t-1})\}$ and their evaluation results $H = [H(D_1, I_1), ..., H(D_{t-1}, I_{t-1})]$, we iteratively update the surrogate model M to estimate $h(f(\theta, D, I, \lambda), D_E)$. In each iteration t, to determine the next pruning solution (D_t, I_t), we use the expected improvement (EI) acquisition function to compute the probability of improvement below a threshold H^* given the observations and $\{(D_1, I_1), ..., (D_{t-1}, I_{t-1})\}$, as follows:

$$EI_{H^*(D,I)} = \int_{-\infty}^{H^*} (H^* - H) p(H|(D, I)) dH \quad (7)$$

Subsequently, AdaPruner stores (D, I) and $H(D, I)$ into the search history and fits a new agent model based on the updated record M. At the end of the loop, AdaPruner outputs the globally optimal (D, I). Specifically, instead of directly modeling, BO-TPE models $p(H|(D, I))$ by decomposing it into $p((D, I)|H)$ and $p(D, I)$ using Bayes theorem, as follows:

$$p(H|(D, I)) = \frac{p((D, I)|H)p(H)}{p(D, I)} \quad (8)$$

$$p((D, I)|H) = \begin{cases} H_{good}(D, I), \text{if } H < H^* \\ H_{bad}(D, I), \text{if } H \geq H^* \end{cases} \quad (9)$$

$$\begin{aligned} p(D, I) &= \textstyle\int_{\mathbb{R}} p((D, I)|H)p(H)dH \\ &= \gamma H_{good}(D, I) + (1 - \gamma) H_{\text{bad}}(D, I) \end{aligned} \quad (10)$$

The BO-TPE constructs different $p((D, I)|H)$ on different sides of the threshold, where $H_{good}(D, I)$ is the excellent density formed by searching into the pruning setup (D, I) to ensure that the model's perplexity is below the threshold H^*, and $H_{good}(D, I)$ is the lousy density formed by the residuals (D, I). The threshold H^* is determined by the quantile γ of the performance of the searched pruned model. Combining Eq. (8), (9), and (10), the final EI of BO-TPE is expressed as:

$$\begin{aligned} EI_{H^*(D,I)} &= \textstyle\int_{-\infty}^{H^*} (H^* - H) p(H|(D, I)) dH \\ &= \textstyle\int_{-\infty}^{H^*} (H^* - H) \frac{p((D,I)|H)p(H)}{p(D,I)} dH \\ &= \textstyle\int_{-\infty}^{H^*} (H^* - H) \frac{H_{good}(D,I)p(H)}{\gamma H_{good}(D,I) + (1-\gamma) H_{bad}(D,I)} dH \\ &= \frac{\int_{-\infty}^{H^*} (H^* - H) p(H) dH}{\gamma + (1-\gamma) \frac{H_{bad}(D,I)}{H_{good}(D,I)}} \propto \left(\gamma + \frac{H_{bad}(D,I)}{H_{good}(D,I)} (1 - \gamma)\right)^{-1} \end{aligned} \quad (11)$$

This shows that maximizing EI is proportional to the ratio of maximizing $\frac{H_{good}(D,I)}{H_{bad}(D,I)}$. The ultimate goal is to maximize the probability of $H_{good}(D, I)$ while decreasing the probability of $H_{bad}(D, I)$. After obtaining the optimal D and I, structured pruning is performed on the LLMs.

To improve the performance of the pruned model, we employ the Low-Rank Adaptation (LoRA) (Hu et al. 2022) to fine-tune the model on the Stanford Alpaca (Taori et al. 2023) dataset. Specifically, LoRA adds two low-rank matrices to the original weight matrix, avoiding the need for full-parameter fine-tuning, as follows:

$$\theta = \theta' x + \Delta\theta x = \theta' x + BAx \quad (12)$$

where A and B are two learnable low-rank matrices.

Experiments

Datasets

To evaluate the performance of LLMs before and after pruning, we use the perplexity (PPL) metric on the WikiText2 (Merity et al. 2017) and PTB (Marcus 1993) datasets to measure the language modeling capability. To evaluate the performance of the pruning method comprehensively and intuitively, we evaluate the zero-shot performance on seven commonsense reasoning datasets, including BoolQ (Clark et al. 2019), PIQA (Bisk et al. 2020), HellaSwag (Zellers et al. 2020), WinoGrande (Sakaguchi et al. 2021), ARC (including ARC-easy and ARC-challenge) (Clark et al. 2018), and OpenbookQA (Mihaylov et al. 2018). We report the accuracy of each dataset and the overall average accuracy across all datasets. A brief description of each dataset is shown in Appendix B.

Baselines

We comprehensively compare AdaPruner with existing structured pruning methods for LLMs. To ensure fairness in the comparison, all baselines use the same pruning and fine-tuning data set as LLM-Pruner, and 10 samples are selected as the calibration data. The baselines include Magnitude (Han, Mao, and Dally 2016), Wanda (Sun et al. 2023), LLM-Pruner (Ma, Fang, and Wang 2023), LoRAShear (Chen et al. 2023) and MoreauPruner (Wang et al. 2024). The details of the baseline models are described in Appendix B.

Implementation Details

To evaluate the effectiveness of AdaPrune, we conduct experiments on the LLaMA-7B (Touvron et al. 2023) and Vicuna-7B (Chiang et al. 2023) models. Meanwhile, samples with lengths less than 128 in BookCorpus (Zhu et al. 2015) are eliminated to narrow the calibration data subspace. The optimization range of the balance coefficients α_1 and α_2 for coarse and fine-grained importance estimation is set between 0 and 1. The estimation metrics alignment factors are optimized in the range (e5, e6, e7), (e-2, e-3), (1e-4). Both related works and detailed experimental settings are in Appendix A and B.

Pruning ratio	Method	WikiText2↓	PTB↓	BoolQ↑	PIQA↑	HellaSwag↑	WinoGrande↑	ARC-e↑	ARC-c↑	OBQA↑
Ratio=0%	LLaMA-7B	12.62	22.14	73.18	78.35	72.99	67.01	67.45	41.38	42.40
Ratio=20% w/o tune	Magnitude	582.41	1022.1	59.66	58.00	37.04	52.41	33.12	28.58	29.80
	Vector	22.28	41.78	61.44	71.71	57.27	54.22	55.77	33.96	38.40
	LLM-Pruner-E1	19.09	34.21	57.06	75.68	66.80	59.83	60.94	36.52	40.00
	LLM-Pruner-E2	19.77	36.66	59.39	75.57	65.34	61.33	59.18	37.12	39.80
	MoreauPruner	18.61	32.92	55.44	76.17	66.47	63.61	61.53	37.80	**40.60**
	AdaPruner	**17.72**	**31.10**	**62.39**	**76.33**	**68.03**	**63.64**	**63.64**	**38.65**	40.40
Ratio=20% w/ tune	Magnitude	21.78	38.64	61.89	70.81	58.34	56.87	54.87	34.02	38.40
	Vector	18.84	33.05	65.75	74.70	64.52	59.35	60.65	36.26	39.40
	LLM-Pruner-E1	17.58	30.11	64.62	77.20	68.80	63.14	64.31	36.77	39.80
	LLM-Pruner-E2	17.37	30.39	69.54	76.44	68.11	65.11	63.43	37.88	40.00
	LoRAShear	-	-	70.17	76.89	68.69	**65.83**	54.11	38.77	39.97
	MoreauPruner	17.01	30.27	66.61	77.04	68.32	65.59	65.57	38.40	**41.20**
	AdaPruner	**16.75**	**28.71**	**70.34**	**77.69**	**69.06**	65.40	**66.92**	**39.93**	40.80

Table 1: Comparative results of structured pruning on LLaMA-7B at 20% pruning ratio.The best results are shown in bold. And the proposed AdaPruner outperformed the baselines significantly (p <0.05).

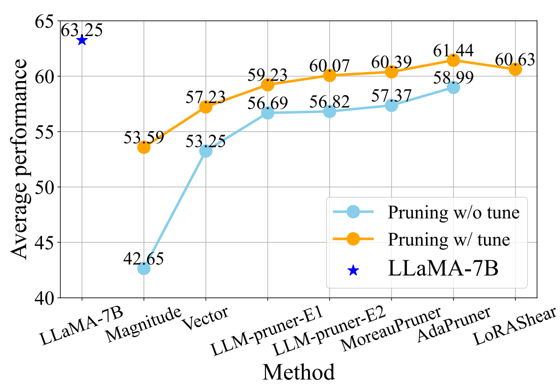


Figure 3: Average performance of structured pruning on LLa-MA-7B at 20% pruning ratio.

Comparative Results and Analysis

We evaluate the zero-shot capabilities of AdaPruner and baselines on 7 downstream commonsense reasoning tasks, as well as language modeling on the WikiText2 and PTB datasets. We performed 20% parameter pruning ratio experiments on LLaMA-7B, as shown in Table 1 and 3. The model pruned with AdaPruner exhibits the best average performance on commonsense reasoning benchmarks while retaining most language modeling capabilities. Specifically, the model the pruned model still retained 93.3% of the average zero-shot performance of the unpruned model without fine-tuning. LLM-Pruner retained 89.8% of the original model's performance, and AdaPruner improved by 3.5%. Notably, AdaPruner significantly outperforms existing structured pruning methods without fine-tuning, suggesting that AdaPruner can obtain a more efficient pruning structure.

The pruned LLaMA-7B retains 97% performance on the zero-shot evaluation dataset compared to the unpruned model with fine-tuning. Meanwhile, AdaPruner consistently outperforms existing structured pruning techniques, achieving 61.44% average accuracy. In addition, it outperforms LoRAShear and MoreauPruner on the most zero-shot commonsense reasoning tasks, resulting in the best average performance among all baseline methods. Specifically, AdaPruner outperforms LLM-Pruner-E2, LoRAShear, and MoreauPruner by 1.37%, 0.81%, and 1.05% on average performance, respectively. AdaPruner outperforms the other baseline methods in preserving text generation capabilities, further demonstrating the effectiveness of the AdaPruner method in preserving the language modeling ability and its superiority over existing structured pruning methods.

Impact of calibration data. To evaluate the importance of searching for calibration data, we removed the search for calibration data and replaced it with random selection (AdaPruner w/o D), As shown in Table 2. The calibration data obtained using random selection shows a performance degradation of all downstream tasks. Specifically, the calibration data obtained with random selection performs less than that obtained with AdaPruner adaptive optimization by 0.71% in average performance, demonstrating the validity of the calibration data adaptively obtained by AdaPruner and the importance of the calibration data in the pruning process. Notably, even when replacing randomized calibration data, the performance is still higher at 1.5% and 0.66% than that of LLM-Pruner-E1 and LLM-Pruner-E2, respectively, illustrating the robustness and validity of the searched importance estimation metrics under different calibration samples.

Impact of joint optimization. To demonstrate the effectiveness of end-to-end joint optimization, Table 2 compares

Method	BoolQ↑	PIQA↑	HellaSwag↑	WinoGrande↑	ARC-e↑	ARC-c↑	OBQA↑	Average↑
AdaPruner	**70.34**	**77.69**	**69.06**	**65.40**	**66.92**	**39.93**	**40.80**	**61.44**
Importance Estimation Metrics								
AdaPruner w/o I_1	70.00	76.61	68.95	64.01	66.20	37.88	40.60	60.61
AdaPruner w/o I_2	68.65	77.31	67.79	63.85	66.58	38.57	39.60	60.34
Calibration Data								
AdaPruner w/o D	69.63	77.86	68.95	64.56	65.03	38.48	40.60	60.73
Optimization Order								
AdaPruner w/o unified	69.81	77.09	69.03	65.23	66.37	39.16	40.00	60.98

Table 2: Ablation study for optimization of importance estimation metrics, calibration data, and different optimization orders.

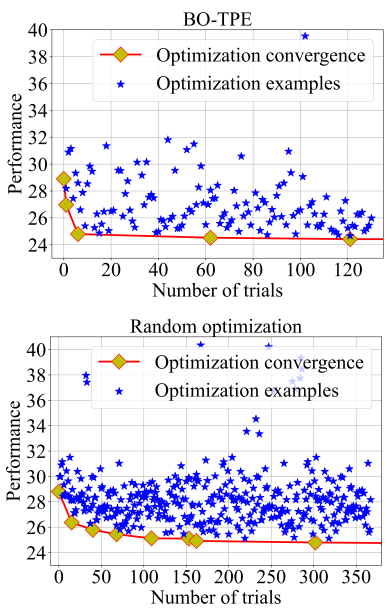


Figure 4: Comparison of Bayesian and stochastic optimization processes.

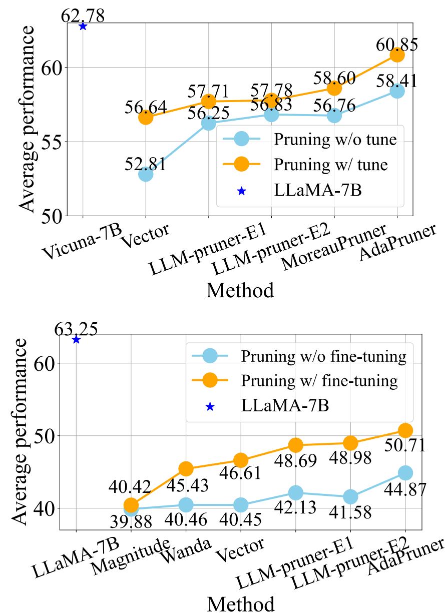


Figure 5: Average performance of structured pruning for Vicuna-7B at 20% pruning ratio and LLaMA-7B at 50% pruning ratio.

different optimization order strategies. The joint optimization of calibration data and importance estimation metrics is replaced with a combined optimization (AdaPruner w/o unified). It can be observed that the combined optimization shows performance degradation on all tasks. The combined optimization underperforms the joint optimization by 0.46% on the zero-shot average performance for commonsense reasoning. The reason is that the calibration data obtained by the combined optimization is not fully compatible with the importance estimation metrics, leading to inaccurate importance estimation and decreased model performance. This further proves that the joint optimization can find the globally optimal calibration data and importance estimation metrics, improving the pruned model's performance.

Ablation Study

Impact of importance estimation metrics. We evaluate the impact of importance estimation metrics obtained after optimization on pruning. Specifically, we remove the optimization of importance estimation metrics and fix using first-order (AdaPruner w/o I_1) and second-order (AdaPruner w/o I_2) Taylor-expanded as alternatives. Table 2 shows the pruned model using first-order and second-order Taylor-expanded. It can be observed that using different importance estimation metrics exhibits different degrees of performance degradation on all the tasks. AdaPruner outperforms the first-order approach regarding average accuracy on the zero-shot of commonsense inference by 0.83% and the second-order approach by 1.1%. Since AdaPruner finds more appropriate and comprehensive importance estimation metrics in

Pruning ratio	Method	WikiText2↓	PTB↓	BoolQ↑	PIQA↑	HellaSwag↑	WinoGrande↑	ARC-e↑	ARC-c↑	OBQA↑
Ratio=0%	Vicuna-7B	16.11	61.37	76.57	77.75	70.64	67.40	65.11	41.21	40.80
Ratio=20% w/ tune	Vector	19.94	74.66	63.15	74.59	61.95	60.30	60.48	36.60	39.40
	LLM-Pruner-E1	19.69	78.25	63.33	76.17	65.13	60.22	62.84	37.12	39.20
	LLM-Pruner-E2	18.97	76.78	60.40	75.63	65.45	63.22	63.05	37.71	39.00
	MoreauPruner	19.66	73.74	63.15	**76.77**	65.96	60.85	65.74	37.12	**40.60**
	AdaPruner	**18.05**	**73.56**	**71.93**	**76.77**	**67.06**	**64.80**	**66.12**	**40.27**	39.00

Table 3: Comparative results of structured pruning on Vicuna-7B at 20% pruning ratio with fine-tuning.

Pruning ratio	Method	WikiText2↓	PTB↓	BoolQ↑	PIQA↑	HellaSwag↑	WinoGrande↑	ARC-e↑	ARC-c↑	OBQA↑
Ratio=0%	LLaMA-7B	12.62	22.14	73.18	78.35	72.99	67.01	67.45	41.38	42.40
Ratio=50% w/ tune	Magnitude	78.80	164.32	47.40	54.36	33.49	53.10	37.88	26.60	30.12
	Wanda	43.89	85.87	50.90	57.38	38.12	55.68	42.68	**34.20**	36.25
	Vector	43.47	68.51	61.11	64.96	40.52	51.54	46.38	28.33	32.40
	LLM-Pruner-E1	38.12	66.35	60.28	69.31	47.06	53.43	45.96	29.18	35.60
	LLM-Pruner-E2	45.70	69.33	61.47	68.82	47.56	55.09	46.46	28.24	35.20
	AdaPruner	**34.29**	**53.40**	**61.89**	**70.29**	**47.98**	**55.72**	**52.61**	30.12	**36.40**

Table 4: Comparative results of structured pruning on LLaMA-7B at 50% pruning ratio with fine-tuning.

the solution space while balancing coarse and fine-grained importance estimation metrics. The results show that optimizing importance estimation metrics is crucial in improving performance.

Impact of search algorithms. To evaluate the effectiveness of Bayesian optimization, we use random search to replace Bayesian optimization, as shown in Figure 4. As can be seen from the figure, Bayesian optimization achieves faster convergence and better final search results compared to random search. Specifically, Bayesian optimization achieves convergence in less than 150 iterations, while random search requires about 300, showing the superiority of Bayesian optimization in terms of search efficiency and effectiveness.

Effects of Different Pruned Models

To validate the effectiveness and robustness of the AdaPruner method for pruning different models, we perform structured pruning of Vicuna-7B using the optimal calibration data and importance estimation obtained after optimization, as shown in Table 3 and Figure 5. The Vicuna-7B model is pruned at a pruning ratio 20% with fine-tuning. The table shows that the zero-shot average performance of AdaPruner consistently outperforms that of Vector, LLM-Pruner, and MoreauPruner. AdaPruner achieves 60.85% average performance, which is 3.07% and 2.25% higher than that of LLM-pruner-E2 and MoreauPruner, respectively, allowing the AdaPruner method can be applied to other LLMs while maintaining higher accuracy compared to existing structured pruning methods. These results also demonstrate the generalization ability and stability of the AdaPruner method for pruning different models. More results without fine-tuning are in Appendix C.

Effects of Different Pruning Ratios

To further analyze the performance and impact of AdaPruner under different pruning ratios, we conduct a 50% pruning ratio experiment on the LLaMA-7B model with fine-tuning, as shown in Tables 4 and Figure 5. It shows that AdaPruner's model performance consistently outperforms LLM-Pruner and other baseline methods even when the pruning ratio is increased to 50%. The AdaPruner outperforms LLM-Pruner in terms of perplexity on the WikiText2 and PTB datasets. Specifically, with a 50% pruning ratio and no fine-tuning, AdaPruner's perplexity on Wik-iText2 and PTB is 34.29 and 53.40, respectively, which are 11.41 and 15.93 lower than LLM-pruner-E2. In addition, AdaPruner achieves an average performance of 50.71% on seven zero-shot inference tasks, which is 1.73% higher than LLM-pruner-E2. It highlights the performance advantage of AdaPruner at high pruning ratios and confirms its performance stability when the pruning ratio increases.

Conclusions

This study proposes AdaPruner, a sample-aware adaptive structured pruning framework for LLMs, aiming to adaptively search for the optimal calibration data and importance estimation metrics to improve pruned LLMs' computational efficiency and performance. The experimental results show that AdaPruner maintains 97% of the performance over the unpruned model and demonstrate the effectiveness of the proposed method for pruning LLMs. In addition, AdaPruner's generalization ability and robustness for different models and pruning ratios. Future work will explore the potential of AdaPruner for a broader range of pruning solution space designs and model types.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant Nos.61966038 and 62266051, and the Yunnan Provincial Department of Education Science Foundation under Grant No.2024Y030. The authors would like to thank the anonymous reviewers for their constructive comments.

References

Bisk, Y.; Zellers, R.; Le Bras, R.; Gao, J.; and Choi, Y. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the 34th AAAI conference on artificial intelligence (AAAI 2020)*, 7432–7439.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language models are few-shot learners. In *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020)*.

Chen, T.; Ding, T.; Yadav, B.; Zharkov, I.; and Liang, L. 2023. Lorashear: Efficient large language model structured pruning and knowledge recovery.

Chiang, W.-L.; Li, Z.; Lin, Z.; Sheng, Y.; Wu, Z.; Zhang, H.; Zheng, L.; Zhuang, S.; Zhuang, Y.; Gonzalez, J. E.; and Others. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *https://vicuna. lmsys. org*, 2.

Clark, C.; Lee, K.; Chang, M. W.; Kwiatkowski, T.; Collins, M.; and Toutanova, K. 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 (NAACL 2019)*, 2924–2936.

Clark, P.; Cowhey, I.; Etzioni, O.; Khot, T.; Sabharwal, A.; Schoenick, C.; and Tafjord, O. 2018. Think you have solved question answering? Try arc, the ai2 reasoning challenge. arXiv:1803.05457v1.

Frantar, E.; and Alistarh, D. 2023. SparseGPT: Massive language models can be accurately pruned in one-shot. In *Proceedings of the 40th International Conference on Machine Learning (ICLM 2023)*, 10323–10337.

Gu, Y.; Dong, L.; Wei, F.; and Huang, M. 2024. MiniLLM: Knowledge distillation of large language models. In *Proceedings of The 12th International Conference on Learning Representations (ICLR 2024)*.

Han, S.; Mao, H.; and Dally, W. J. 2016. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. In *Proceedings of 4th International Conference on Learning Representations (ICLR 2016)*.

Hoffmann, J.; Cai, T.; Borgeaud, S.; Mensch, A.; Sifre, L.; Rae, J. W.; Simonyan, K.; and Elsen, E. 2022. Training compute-optimal large language models.

Hsieh, C. Y.; Li, C. L.; Yeh, C. K.; Nakhost, H.; Fujii, Y.; Ratner, A.; Krishna, R.; Lee, C. Y.; and Pfister, T. 2023. Distilling step-by-step! Outperforming larger language models with less training data and smaller model sizes. In *Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics (ACL 2023)*, 8003–8017.

Hu, E.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2022. Lora: Low-Rank Adaptation of Large Language Models. In *Proceedings of the10th International Conference on Learning Representations (ICLR 2022)*.

Jaiswal, A.; Liu, S.; Chen, T.; and Wang, Z. 2023. The emergence of essential sparsity in large pre-trained models: The weights that matter. In *Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS 2023)*, 38887–38901.

Kaplan, J.; Wu, J.; Amodei, D.; Henighan, T.; Gray, S.; Brown, T. B.; and Radford, A. 2020. Scaling laws for neural language models. arXiv:2001.08361v1.

Kwon, W.; Kim, S.; Mahoney, M. W.; Hassoun, J.; Keutzer, K.; and Gholami, A. 2022. A fast post-training pruning framework for transformers. In *Proceedings of the 36th Conference on Neural Information Processing Systems (NeurIPS 2022)*, 24101–24116.

Liu, Z.; Oguz, B.; Zhao, C.; Chang, E.; Stock, P.; Mehdad, Y.; Shi, Y.; Krishnamoorthi, R.; and Chandra, V. 2023. Llm-qat: Data-free quantization aware training for large language models. arXiv:2305.17888.

Ma, X.; Fang, G.; and Wang, X. 2023. Llm-pruner: On the structural pruning of large language models. In *Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS 2023)*, 21702–21720.

Marcus, M. P. 1993. Building a large annotated corpus of English:The Penn Treebank. *Computational linguistics*, 19: 313–330.

Merity, S.; Xiong, C.; Bradbury, J.; and Socher, R. 2017. Pointer sentinel mixture models. In *Proceedings of 5th International Conference on Learning Representations (ICLR 2017)*.

Mihaylov, T.; Clark, P.; Khot, T.; and Sabharwal, A. 2018. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, 2381–2391.

Sakaguchi, K.; Bras, R. L.; Bhagavatula, C.; and Choi, Y. 2021. WinoGrande: an adversarial Winograd schema challenge at scale. *Association for Computing Machinery*, 64(9): 99–106.

Sun, M.; Liu, Z.; Bair, A.; and Kolter, J. Z. 2023. A simple and effective pruning approach for large language models. arXiv:2306.11695.

Taori, R.; Gulrajani, I.; Zhang, T.; Dubois, Y.; Li, X.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023. Stanford alpaca: An instruction-following llama model. In *URL https://github. com/tatsu-lab/stanford_alpaca*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; Rodriguez, A.; Joulin, A.; Grave, E.; and Lample, G. 2023. Llama: Open and efficient foundation language models. arXiv:2302.13971.

Wang, Z.; Zhang, J.; Zhao, W.; Farnia, F.; and Yu, B. 2024. MoreauPruner: Robust pruning of large language models against weight perturbations. arXiv:2406.07017.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Chi, E.; Le, Q.; and Zhou, D. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th Conference on Neural Information Processing Systems (NeurIPS 2022)*, volume 35, 24824–24837.

Wu, Y.; Kuang, K.; Zhang, Y.; Liu, X.; Sun, C.; Xiao, J.; Zhuang, Y.; Si, L.; and Wu, F. 2020. De-biased court's view generation with causality. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, 763–780.

Xia, M.; Gao, T.; Zeng, Z.; Chen, D.; and Language, P. 2024. Sheared LLaMA: Accelerating Language Model Pre-training via Structured Pruning. In *Proceedings of th 12th International Conference on Learning Representations (ICLR 2024)*.

Xiao, G.; Lin, J.; Seznec, M.; Wu, H.; Demouth, J.; and Han, S. 2023. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proceedings of the 40th International Conference on Machine Learning (ICLM 2023)*, 38087–38099.

Yin, L.; Wu, Y.; Zhang, Z.; Hsieh, C.-Y.; Wang, Y.; Jia, Y.; Pechenizkiy, M.; Liang, Y.; Wang, Z.; and Liu, S. 2023. Outlier weighed layerwise sparsity (owl): A missing secret sauce for pruning LLMs to high sparsity. arXiv:2310.05175.

Yuan, S.; Chen, J.; Fu, Z.; Ge, X.; Shah, S.; Jankowski, C. R.; Xiao, Y.; and Yang, D. 2023. Distilling script knowledge from large language models for constrained language planning. In *Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics (ACL 2023)*, 4303–4325.

Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2020. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2022)*, 4791–4800.

Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, C.; Diab, M.; Li, X.; Lin, X. V.; Mihaylov, T.; Ott, M.; Shleifer, S.; Shuster, K.; Simig, D.; Koura, P. S.; Sridhar, A.; Wang, T.; and Zettlemoyer, L. 2022. Opt: Open pre-trained transformer language models. arXiv:2205.01068.

Zhu, X.; Li, J.; Liu, Y.; Ma, C.; and Wang, W. 2023. A survey on model compression for large language models. arXiv:2308.07633.

Zhu, Y.; Kiros, R.; Zemel, R.; Salakhutdinov, R.; Urtasun, R.; Torralba, A.; and Fidler, S. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV 2015)*, 19–27.