LaCo: Large Language Model Pruning via Layer Collapse

Anonymous ACL submission

Abstract

Large language models (LLMs) based on transformer are witnessing a notable trend of size expansion, which brings considerable costs to both model training and inference. However, existing methods such as model quantization, knowledge distillation, and model pruning are constrained by various issues, including hardware support limitations, the need for extensive training, and alterations to the internal structure of the model. In this paper, we propose a concise layer-wise pruning method called *Layer Collapse (LaCo)*, in which rear model layers collapse into a prior layer, enabling a rapid reduction in model size while preserving the model structure. Comprehensive experiments show that our method maintains an average task performance of over 80% at pruning ratios of 25-30%, significantly outperforming existing state-of-the-art structured pruning methods. We also conduct post-training experiments to confirm that the proposed pruning method effectively inherits the parameters of the original model. Finally, we discuss our motivation from the perspective of layer-wise similarity and evaluate the performance of the pruned LLMs across various pruning ratios.

1 Introduction

Recently, large language models (LLMs) based on transformer (Vaswani et al., 2017) have showcased impressive capabilities across diverse tasks. However, the prevailing trend in model development leans towards larger scales, placing substantial demands on computational power and resulting in challenges for many researchers and companies.

To mitigate the above challenge, various approaches have been explored to reduce the inference and training costs of models or to derive a smaller model from an LLM, including model quantization (Dettmers et al., 2022; Yao et al., 2022; Xiao et al., 2023), knowledge distillation (Liu et al., 2022; Hsieh et al., 2023; Shridhar et al., 2023), and model pruning (Zhang et al., 2022; Frantar and Alistarh, 2023; Ma et al., 2023). However, existing solutions exhibit certain notable drawbacks. Model quantization significantly impacts model performance and typically necessitates specific hardware support. Knowledge distillation often requires retraining a smaller model at a higher training cost and is usually task-specific. Existing Model pruning can be categorized into non-structured and structured pruning. Non-structured pruning often involves model sparsity, which generally leads to notable performance loss and also relies on hardware support. Structured pruning entails removing specific modules from the model, but it frequently modifies the model structure, diminishing the portability of the model.

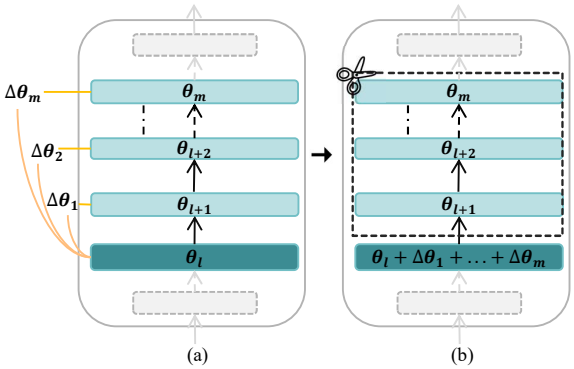


Figure 1: An example of *Reserving-Differences-while-Seeking-Common (RDSC) Layer Merge*. In (a), we perform parameter differencing, which we regard as Reserving-Differences. In (b), we conduct parameter merging, which we interpret as Seeking-Common.

Considering the above issues, we contemplate directly pruning the model with a new idea: to prune some layers directly from a well-trained LLM and substitute the parameters of one layer for multiple layers, enabling effective model pruning.

Specifically, we observe that for certain layers, merging the differential values of the parameters of this layer and the consecutive layers following

it often does not result in significant model performance loss, as illustrated in Figure 1. We term it the *Reserving-Differences-while-Seeking-Common (RDSC) Layer Merge*, as it incorporates parameter differencing and merging. Building upon this insight, we introduce a streamlined yet potent layer-wise pruning method dubbed *Layer Collapse (LaCo)*, in which rear layers collapse into a prior layer, with the objective of preserving the model's output representation as closely as feasible to the representation before pruning. In this paper:

• The *Layer Collapse* can directly remove 30%-50% of model layers without training while maintaining the model performance. Experiments on multiple benchmarks show that our approach outperforms state-of-the-art structured pruning methods under equivalent pruning ratios.

• The *Layer Collapse* also possesses excellent properties of preserving the internal structure of model layers, such as maintaining intermediate dimensions unchanged. Hence, it can be quickly adapted to existing applications.

• We further conduct post-training on the pruned model, confirming that the *Layer Collapse* can efficiently inherit parameters, requiring only a minimal amount of training to restore the pruned model to the loss convergence level of the original model. Furthermore, we discuss our motivation and evaluate the performance of pruned models using *LaCo* across different pruning ratios.

2 Method

2.1 Reserving-Differences-while-Seeking-Common Layer Merge

Given an LLM, for its l-th layer, we denote all its parameters, including those in self-attention (SAN) and MLP as $\boldsymbol{\theta}_l$. For the m consecutive layers following it, we merge the parameters of $\boldsymbol{\theta_{l+1}}, \boldsymbol{\theta_{l+2}}, \cdots, \boldsymbol{\theta_{l+m}}$ into $\boldsymbol{\theta_l}$ to form $\boldsymbol{\theta_l^*}$:

$$\begin{aligned}\boldsymbol{\theta}_l^* &= \boldsymbol{\theta}_l + (\boldsymbol{\theta}_{l+1} - \boldsymbol{\theta}_l) + \cdots + (\boldsymbol{\theta}_{i+m} - \boldsymbol{\theta}_l) \\ &= \boldsymbol{\theta}_l + \sum_{k=1}^{m} (\boldsymbol{\theta}_{l+k} - \boldsymbol{\theta}_l)\end{aligned} \tag{1}$$

where $(\boldsymbol{\theta_{l+k}} - \boldsymbol{\theta_l})$ is the parameter difference between each subsequent layer and the l-th layer. During parameter differencing and merging, given identical layer structures, we independently apply these processes to both SAN and MLP. After that, these m consecutive layers will be discarded. Subsequent model pruning will continuously involve RDSC Layer Merge, a process that can be regarded as the continual collapse of layers onto specific layers, hence the name *Layer Collapse*.

Algorithm 1 Layer Collapse

Input:
1: LLM $\mathcal{M}$
2: Number of layers combined in each merge $\mathcal{C}$
3: Layer range considered $[\mathcal{L}, \mathcal{H}]$
4: Minimum interval between two adjacent merged layers $\mathcal{I}$
5: Few-shot Calibration Samples $\mathcal{D}$
6: Threshold for representation similarity $\mathcal{T}$
Output: Pruned LLM $\mathcal{M}^*$
7: $\mathcal{M}^* \leftarrow \mathcal{M}$
8: $l \leftarrow \mathcal{H} - \mathcal{C}$
9: **while** $l >= \mathcal{L}$ **do**
10: $\quad\mathcal{K} \leftarrow$ Min($\mathcal{C}-1$, Layer_Count($\mathcal{M}^*$)$-l$)
11: $\quad\mathcal{M}_{tmp} \leftarrow$ RDSC_Lay_Merge($\mathcal{M}^*$, l, $\mathcal{K}$)
12: $\quad s \leftarrow$ Avg_Cos_Sim($\mathcal{M}_{tmp}$, $\mathcal{M}$, $\mathcal{D}$)
13: $\quad$**if** $s > \mathcal{T}$ **then**
14: $\quad\quad\mathcal{M}^* \leftarrow \mathcal{M}_{tmp}$
15: $\quad\quad l \leftarrow l - \mathcal{I}$
16: $\quad\quad$**if** $l >$ Layer_Count($\mathcal{M}^*$) **then**
17: $\quad\quad\quad l \leftarrow$Layer_Count($\mathcal{M}^*$)$-\mathcal{C}$
18: $\quad\quad$**end if**
19: $\quad$**else**
20: $\quad\quad l \leftarrow l - 1$
21: $\quad$**end if**
22: **end while**
23: **return** $\mathcal{M}^*$

2.2 Layer Collapse

We dynamically merge adjacent layers starting from the topmost layer of the model, while ensuring that the output representation of the pruned model on few-shot calibration samples remains as similar as possible to the original model, to ensure minimal performance loss. Algorithm 1 summarizes the procedure of *Layer Collapse*:

(1) Preparation (line 1-6)

For an LLM $\mathcal{M}$ to be pruned, we define the number of layers to be merged during each merging operation as $\mathcal{C}$. We configure the merging to operate within a certain range of layers, denoted as $[\mathcal{L}, \mathcal{H}]$. As the layer merging operation inevitably leads to a performance loss, to prevent consecutive layer merging from causing a sharp decline in the model performance, we set a minimum interval of layers between two merging operations as $\mathcal{I}$. Few-shot calibration samples $\mathcal{D}$, typically a few plain

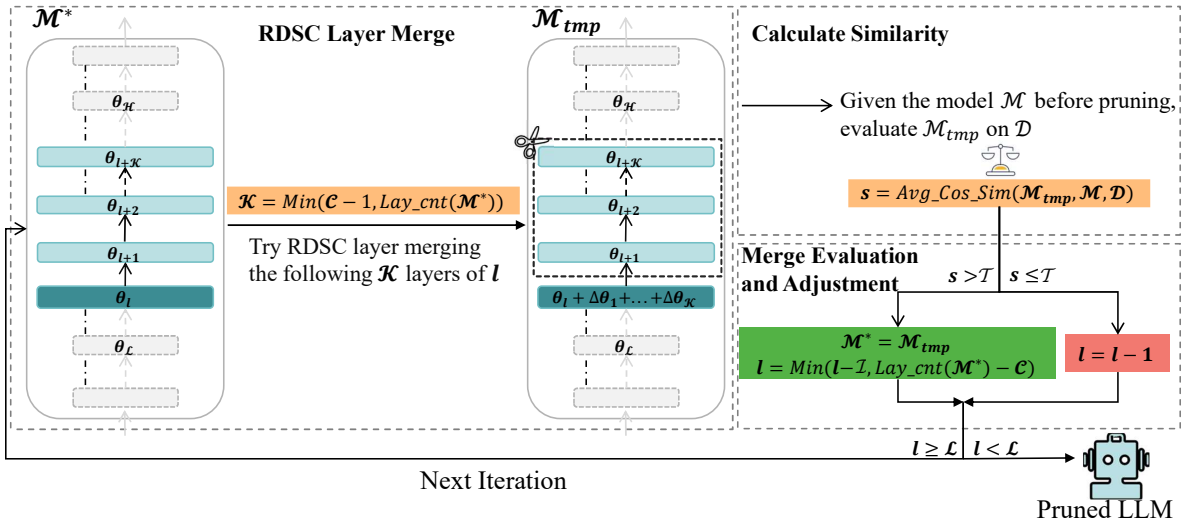


Figure 2: An illustration of Layer Collapse.

sentences, are used during the pruning process. We perform forward computations on $\mathcal{D}$ with both the pruned and original models to obtain the output representations and ensure that the similarity of representations is not less than the threshold $\mathcal{T}$.

(2) Pruning (line 7-23)

We present an illustration of *Layer Collapse*, as depicted in Figure 2. We begin by initializing the model $\mathcal{M}^*$ with the model $\mathcal{M}$ and set a layer pointer l to start from $\mathcal{H} - \mathcal{C}$. Then, the iterative process of pruning begins:

RDSC Layer Merge (line 10-11) During each iteration, our approach involves merging the $\mathcal{K}$ layers following layer l into layer l itself and then discarding the redundant $\mathcal{K}$ layers, where $\mathcal{K}$ is the minimum of $\mathcal{C}-1$ and the total layer count of $\mathcal{M}^* - l$, implying merging either the subsequent $\mathcal{C}-1$ layers or all layers following l, thus to prune the model $\mathcal{M}^*$, resulting in the interim model $\mathcal{M}_{tmp}$.

Calculate Similarity (line 12) We process each sentence in $\mathcal{D}$ using forward computations with $\mathcal{M}_{tmp}$ and $\mathcal{M}$ to derive their representations. For every sentence, we then calculate the cosine similarity between these representations from both models, averaging these values to obtain the overall similarity score s.

Merge Evaluation and Adjustment (line 13-21) Then, we evaluate s against the threshold $\mathcal{T}$. Should s exceed $\mathcal{T}$, the current merge is considered successful. Then, $\mathcal{M}_{tmp}$ is updated to $\mathcal{M}^*$ for the next iteration, and the pointer l is adjusted downwards by $\mathcal{I}$ layers. Conversely, l is simply reduced by a single layer. It is important to highlight that the instances may occur where l falls below the total layer count of $\mathcal{M}^*$ after a series of successive merges. Consequently, it is required to reset l to the layer count in $\mathcal{M}^* - \mathcal{C}$, as illustrated in line 15.

We iterate through the above process until l is less than $\mathcal{L}$ and output the pruned LLM.

2.3 Complexity Analysis

The complexity of *LaCo* primarily depends on model inference. In the worst-case scenario, where $\mathcal{L}$ is set to 0 and $\mathcal{H}$ is set to the total number of model layers. If in each iteration, the similarity s is less than $\mathcal{T}$, then all layers will be traversed. Thus, the worst-case time complexity is $O(\mathcal{H} \times ||\mathcal{D}||)$. For instance, considering Llama2-13B with 40 layers and $||\mathcal{D}||$ consisting of 10 sentences, the maximum number of inference steps would be only 400 sentences, which can be completed within minutes on a single GPU. Therefore, our approach demonstrates excellent time performance.

3 Experiments

3.1 Models

To assess the effectiveness of the proposed *LaCo*, we conduct experiments on popular English LLMs, Llama2-7B and 13B (Touvron et al., 2023). Additionally, we test the effectiveness on bilingual LLMs, specifically Baichuan2-7B and 13B (Yang et al., 2023), which support both Chinese and English. We leverage the base versions of these LLMs. All these models are decoder-only models based on the transformer architecture.

3.2 Benchmarks

To comprehensively evaluate the pruned model's capabilities, we utilized the OpenCompass evaluation framework (Contributors, 2023). Specifically, following OpenCompass categorization, we conduct evaluations in five aspects: Reasoning, Language, Knowledge, Examination and Understanding. We select several benchmarks from each category. **Reasoning:** CMNLI (Xu et al., 2020), HellaSwag (HeSw) (Zellers et al., 2019), PIQA (Bisk et al., 2019). **Language:** CHID (Zheng et al., 2019), WSC (Levesque et al., 2012). **Knowledge:** CommonSenseQA (CoQA) (Talmor et al., 2018), BoolQ (Clark et al., 2019). **Examination:** MMLU (Hendrycks et al., 2021), CMMLU (Li et al., 2023). **Understanding:** Race-High/Middle (H/M) (Lai et al., 2017), XSum (Narayan et al., 2018), C3 (Sun et al., 2020).

We conduct evaluations using official scripts from OpenCompass, all zero-shot or few-shot, without additional training. Two evaluation modes are utilized: perplexity (PPL) and generation (GEN) [1]. For CHID and XSum, we use the GEN mode. For the WSC dataset, we use both PPL (WSC_P) and GEN (WSC_G) modes. The remaining benchmarks are evaluated using the PPL mode. The evaluation results on each benchmark are converted to a score by OpenCompass, where a higher score indicates better performance. OpenCompass provides official evaluation results for the Baichuan2 and Llama2 series. However, to avoid discrepancies resulting from hardware and software environments, as well as potential errors in official results, we reproduce all results to ensure fairness.

3.3 Baselines

Since our *LaCo* involves structured pruning, which directly removes components from the LLM, we chose two current state-of-the-art (SOTA) structured pruning methods, LLM-Pruner (LLM-Pru.) (Ma et al., 2023) and SliceGPT (Ashkboos et al., 2024) which has surpassed the previous SOTA sparsity method, SparseGPT (Frantar and Alistarh, 2023) as our baselines. For experiments, we set the pruning ratio of these methods to be equivalent to or slightly smaller than our method to ensure fairness.

[1] opencompass.readthedocs.io/en/latest/get_started/faq.html

3.4 Settings

Since previous work mostly set pruning ratios below 30%, we heuristically adjust the hyperparameters to bring the model pruning ratio close to 30%, as shown in Appendix A Table 5. We randomly select 5 sentences from both the English and Chinese Wikipedia datasets for Baichuan2 and 10 sentences from English Wikipedia for Llama2 as few-shot calibration samples. All experiments are conducted on a server with 8 Nvidia A100 80GB GPUs.

3.5 Main Results

In Table 1, we present the results of four LLMs under different pruning methods across various benchmarks. "Dense" represents the official results of the unpruned LLMs in OpenCompass leaderboards, while "Dense*" represents our reproduction of the "Dense" results. "LLMPru." and "SliceGPT" correspond to the two baselines, respectively. "Ratio" refers to the overall pruning ratio, namely the proportion of the total number of pruned parameters to that of the unpruned model. "Lay." denotes the total number of layers in the model.

Comparing Dense and Dense*, the results show not much difference, with most discrepancies within 5%. This indicates our experimental setup is error-free. To ensure fairness, we compare the results against Dense* in the subsequent analyses.

Upon comparing LaCo with the baselines, from Table 1, it can be observed that LaCo achieves the best results on most benchmarks, despite our pruning ratio being slightly higher than the baselines.

To provide a more intuitive presentation of the results in Table 1, we compute the average scores of each pruner across all benchmarks (Avg.), the average scores per category (Reas., Lan., Know., Exam., Unde.), and the average performance percentages relative to Dense* across all benchmarks (Per.) in Table 2. Overall, our average scores are significantly higher than the baselines. It can also be inferred that Laco exhibits significantly superior performance overall in four out of five categories: Language, Knowledge, Examination, and Understanding. Even though there is a slight dip in performance in Reasoning, it remains comparable to the baselines. In addition, LaCo holds a significant advantage: relative to Dense*, our average performance percentage across all datasets is far superior to the baselines. The average percentage surpasses 80% in three out of four models, with the lowest being on Baichuan2-7B, yet still exceeding 73%.

LLM	Pruner	Ratio/Lay.	Reasoning			Language			Knowledge		Examination		Understanding			
			CMNLI	HeSw	PIQA	CHID	WSC_P	WSC_G	CoQA	BoolQ	MMLU	CMMLU	$Race_H$	$Race_M$	XSum	C3
Llama2 -7B	Dense	0%/32	34.90	74.00	78.30	46.50	-	66.30	66.50	74.90	46.80	31.80	37.50	40.20	19.70	42.80
	Dense*	0%/32	32.98	71.35	78.18	46.04	37.50	38.46	66.67	70.67	45.92	31.86	35.51	33.15	19.68	43.78
	LLMPru.	27.0%/32	34.33	**56.46**	**71.22**	25.25	36.54	0.96	42.51	55.20	23.33	25.25	22.56	22.35	11.51	25.64
	SliceGPT	26.4%/32	31.70	50.27	66.21	20.79	36.54	19.23	41.36	38.32	**28.92**	**25.37**	21.07	21.66	4.89	**39.78**
	LaCo	**27.1%/23**	**34.43**	55.69	69.80	**36.14**	**40.38**	**25.00**	**45.70**	**64.07**	26.45	25.24	**22.61**	**23.61**	**15.64**	39.67
Llama2 -13B	Dense	0%/40	41.40	77.50	79.80	53.00	-	66.30	66.70	82.40	55.00	38.40	58.90	63.00	23.40	46.10
	Dense*	0%/40	32.99	74.83	79.71	52.97	50.96	63.46	66.91	71.50	55.63	38.74	58.03	60.24	23.56	47.51
	LLMPru.	24.4%/40	**33.03**	**67.76**	**76.66**	35.64	40.38	0.00	50.86	56.42	25.21	24.71	22.47	22.08	**19.17**	32.33
	SliceGPT	23.6%/40	29.82	55.71	69.04	19.31	36.54	**36.54**	47.26	37.86	37.14	25.79	23.41	24.03	5.27	41.92
	LaCo	**24.6%/30**	32.86	64.39	74.27	**40.10**	**52.88**	35.58	**52.66**	**63.98**	**45.93**	**32.62**	**54.49**	**56.55**	14.45	**44.93**
Baic2. -7B	Dense	0%/32	32.90	67.00	76.20	82.70	-	66.30	63.00	63.20	54.70	57.00	52.50	50.90	20.90	64.50
	Dense*	0%/32	33.37	67.56	76.17	82.67	41.35	63.46	63.14	63.30	54.25	56.95	52.63	51.04	20.84	64.55
	LLMPru.	24.2%/32	32.28	**53.66**	**71.82**	69.80	**53.85**	0.00	**47.83**	**61.19**	24.93	25.69	21.96	22.28	**15.98**	41.64
	SliceGPT	22.2%/32	32.07	25.29	50.33	14.85	36.54	0.00	19.57	39.30	25.18	25.25	23.53	22.49	0.00	26.58
	LaCo	**24.2%/23**	**33.00**	52.28	68.50	**76.24**	42.31	**26.92**	47.26	56.15	**31.53**	**31.24**	**28.99**	**27.72**	12.03	**50.85**
Baic2. -13B	Dense	0%/40	32.70	70.80	78.10	83.20	-	63.20	65.60	67.00	59.50	61.30	67.20	68.90	25.20	65.60
	Dense*	0%/40	33.21	71.10	78.07	83.17	41.35	63.46	65.60	67.00	58.81	61.27	67.27	68.94	24.95	65.64
	LLMPru.	24.3%/40	**33.80**	53.57	**71.82**	72.77	37.50	0.00	38.82	56.54	23.19	25.18	21.17	21.61	**13.67**	39.89
	SliceGPT	22.8%/40	32.07	25.85	51.03	10.40	36.54	0.00	18.02	37.83	22.95	25.26	21.56	21.52	0.00	24.99
	LaCo	**24.7%/30**	33.03	**60.71**	68.88	**76.73**	**44.23**	**60.58**	**55.45**	**62.35**	**51.35**	**53.65**	**56.92**	**57.80**	12.32	**61.10**

Table 1: The main results of our experiments. “Dense" is the official LLM results in OpenCompass and “Dense*" is our reproduction. "LLMPru." and "SliceGPT" are two baseline comparisons.

However, none of the baselines have exceeded 70% on any model.

To demonstrate the stability of models pruned by our LaCo method, we compute the percentage of result on each benchmark relative to Dense*, as shown in Appendix C Table 7. We observe that models pruned by LaCo exhibit stability relative to the baselines, maintaining performance above 70% on most benchmarks. Furthermore, there were no instances of models crashing outright, with performance dropping below 30%.

It is worth noting that on three benchmarks evaluated through GEN mode, CHID, XSUM, and WSC_G, the LLMs pruned by LaCo maintain relatively stable performance, while models pruned by baselines exhibit poorly, with even multiple results becoming 0.00. GEN mode scores are based on the model's generated sentences, and the models pruned by baselines are prone to producing meaningless repetitive outputs. In Appendix C Table 8, We showcase an example from the Xsum benchmark, where Llama2-7B, pruned by baselines, produces nonsensical repeated outputs, whereas our

LLM	Pruner	Avg.	Per.	Reas.	Lan.	Know.	Exam.	Unde.
Llama2 -7B	Dense*	46.55	100%	60.83	40.67	68.67	38.89	33.03
	LLMPru.	32.36	67.79%	**54.00**	20.92	48.86	24.29	20.52
	SliceGPT	31.87	67.37%	49.39	25.52	39.84	**27.15**	21.85
	LaCo	**37.46**	**80.28%**	53.30	**33.84**	**54.89**	25.85	**25.38**
Llama2 -13B	Dense*	55.50	100%	62.51	55.80	69.20	47.18	47.34
	LLMPru.	36.19	65.87%	**59.15**	25.34	53.64	24.96	24.01
	SliceGPT	34.97	61.78%	51.52	30.80	42.56	31.46	23.66
	LaCo	**47.55**	**85.21%**	57.17	**42.85**	**58.32**	**39.28**	**42.60**
Baic2. -7B	Dense*	56.52	100%	59.03	62.49	63.22	55.60	47.26
	LLMPru.	38.78	69.65%	**52.59**	41.22	**54.51**	25.31	25.46
	SliceGPT	24.36	44.27%	35.90	17.13	29.44	25.22	18.15
	LaCo	**41.79**	**73.26%**	51.26	**48.49**	51.70	**31.38**	**29.90**
Baic2. -13B	Dense*	60.70	100%	60.79	62.66	66.30	60.04	56.70
	LLMPru.	36.40	60.70%	53.06	36.76	47.68	24.18	24.08
	SliceGPT	23.43	40.33%	36.32	15.65	27.92	24.10	17.02
	LaCo	**53.94**	**87.94%**	**54.21**	**60.51**	**58.90**	**52.50**	**47.04**

Table 2: The average scores and the percentages comparison with the Dense*.

LaCo yields outputs resembling normal sentences.

In summary, our LaCo stands out as a superior pruning method, effectively preserving model performance and demonstrating exceptional stability across various benchmarks. Additionally, our approach is straightforward, relying solely on parameter differences and additions without necessitating modifications to the model's internal structure, such as intermediate layer dimensions. This results in a more concise and efficient pruning solution.

4 Analysis

4.1 Post-training and Re-pruning

4.1.1 Post-training

Due to the inevitable performance loss caused by pruning, we investigate whether models pruned using our LaCo can effectively inherit parameters from the original model and quickly recover performance through post-training. Specifically, we select the pruned Llama2-7B and Baichuan2-7B models obtained through LaCo in the main experiments and post-train them. For training pruned Llama2-7B, we utilize approximately 1.0 billion tokens from the English dataset, while for pruned Baichuan2-7B, we employ approximately 1.25 billion tokens, with a 50% from English and Chinese. The detailed implementation can be found in the Appendix B.

In Figure 3, we present the loss curves. It can be observed that both models converge rapidly during training, with the loss showing a sharp decline after about 250 steps, then stabilizing. The pruned Llama2-7B and Baichuan2-7B models, both approximately 5 billion parameters, exhibit final convergence losses around 1.6 and 2.0, which are quite comparable to the reported values of 1.75 for Llama2-7B and 1.9 for Baichuan2-7B in their technical reports. The post-training of pruned Llama2-7B and Baichuan2-7B on 4 Nvidia A100 80GB GPUs takes approximately 28 hours and 35 hours, respectively. Training a 5B LLM from scratch requires at least 500 billion tokens on hundreds of A100 GPUs for several months. However, we achieve a loss-converged model of similar size with only one-thousandth of their cost. This indicates that the pruned models have effectively inherited the parameters of the original models, enabling them to rapidly recover performance with minimal post-training and achieve convergence.

We also evaluate the post-trained models on multiple benchmarks, and the detailed results can be found in Appendix C Table 9. We list the average score for each category and the overall average score in Table 3.

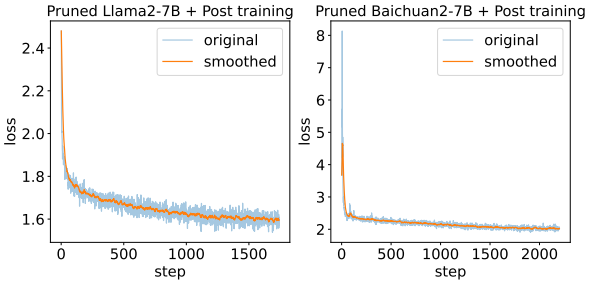


Figure 3: Loss curves for post-training.

LLM	Method	Avg.	Reas.	Lan.	Know.	Exam.	Unde.
Llama2 -7B	Dense*	46.55	60.83	40.67	68.67	38.89	33.03
	LaCo	37.46	53.30	33.84	54.89	25.85	25.38
	LaCo +post train	**40.33**	**56.66**	**36.43**	**61.85**	**27.40**	**26.70**
	LaCo +post train +re prune	32.40	48.07	20.26	49.46	25.72	24.56
Baic2. -7B	Dense*	56.52	59.03	62.49	63.22	55.60	47.26
	LaCo	**41.79**	51.26	**48.49**	51.70	**31.38**	29.90
	LaCo +post train	40.46	**51.67**	40.82	**53.97**	27.98	**31.28**

Table 3: The average scores across all categories and the overall average score of pruned models, post-trained models, as well as post-trained models followed by re-pruning.

From the tables, it is evident that the post-training of pruned Llama2-7B significantly improves its performance across various benchmarks. However, the performance of pruned Baichuan2-7B after post-training shows mixed results, with some benchmarks showing improvement while others exhibit a decrease and there is also a slight decrease in the overall score. We speculate that the pre-training data of Baichuan2-7B includes a variety of sources, resulting in a data distribution different from that of our post-training data, hindering the effectiveness of post-training. However, the consistent score improvement on pruned Llama2-7B indicates that models pruned using our LaCo indeed effectively inherit the parameters and can regain performance through low-cost post-training.

4.1.2 Re-pruning

Since it is possible to partially restore performance using post-training on an LLM with approximately 25%-30% of its parameters pruned, it raises the

question of whether we can further prune the post-trained model to obtain one with only around 50% of the original parameters while still maintaining relatively good model performance. To this end, we further prune the previously post-trained pruned Llama2-7B model using LaCo, resulting in a model with 17 layers, retaining 55% of the parameters of the original Llama2-7B model with 32 layers. We evaluate the re-pruned model. The detailed results are shown in Appendix C Table 9 and the average results are in Table 3.

From the tables, it can be seen that although the model has only 55% of the parameters of the original model, it still retains an average of approximately 70% of the original 7B model performance. However, our training data quality and scale are limited. With access to more and better training data, our LaCo method should be able to demonstrate even greater utility in this pruning+post-training+re-pruning pipeline on larger models.

4.2 Layer-wise Similarity

This section discusses our motivation, i.e., why we consider merging adjacent layers. Our motivation primarily stems from the observation that the changes in parameters and output representations between adjacent layers within the LLM are not particularly significant.

In Figure 4, we illustrate the L2 similarities between the SAN q,k,v matrices of each layer and their corresponding matrices from the subsequent one layer as well as the up and down matrices of the MLP for both Llama2-7B and Baichuan2-7B. The results indicate that the maximum L2 values between corresponding matrices in adjacent layers are generally no more than 200. However, the sizes of the up and down matrices of the MLP reach 11008*4096, and the SAN q,k,v matrices also contain 4096*4096 parameters. Therefore, the change in each element of the matrices between adjacent layers is very small.

In Figure 5 (a), we randomly select 20 sentences from Wikipedia and calculate the cosine similarity between the hidden-states of adjacent layers outputs. The results show that for both Baichuan2-7B and Llama2-7B, the representation similarity between adjacent layers from layers 3 to 28 is typically very close to 1, which suggests that the representations between adjacent layers of LLMs also exhibit high similarity. Therefore, the high similarity in parameters and representations between adjacent layers leads us to consider whether a single layer can replace multiple layers following it.

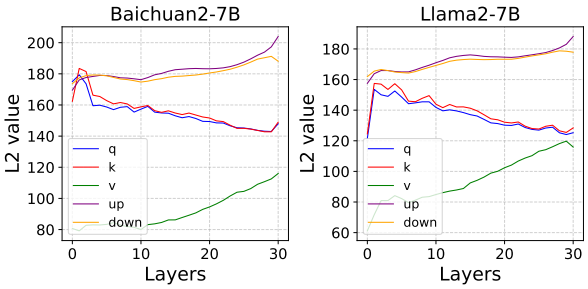
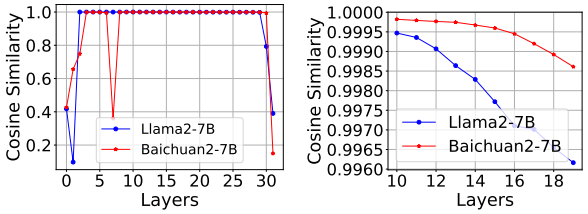


Figure 4: The L2 similarity of corresponding matrices between adjacent layers.



(a) The cosine similarity of output representations between adjacent layers.

(b) The similarity of output representations before and after RDSC Layer Merge.

Figure 5: The cosine similarity of layer representations.

Moreover, the similarity in parameters suggests that we should pay more attention to the differences between layers. Inspired by previous model merging work (Yu et al., 2023; Matena and Raffel, 2022), we come up with collecting the parameter differences between layers and then merging them into a single layer.

To verify that RDSC Layer Merge can indeed replace multiple layers with a merged single layer, we conduct the following experiments. Within layers 10 to 19, we attempt to merge consecutive 4 layers into one layer sequentially, then evaluate the cosine similarity between the output of the merged layer and the output of the original last layer among those 4 layers, as shown in Figure 5 (b). The results showcase that the lowest cosine similarity on the 4096-dimensional vectors is above 0.996, confirming the effectiveness of the proposed RDSC Layer Merge in preserving representations.

4.3 Varying Pruning Ratio

In this section, we explore the performance of *LaCo* at different pruning ratios. We conduct experiments on Llama2-7B and Llama2-13B, controlling the pruning ratios at approximately 10%, 25% (our main experiments), and around 50% by setting different hyperparameters (as shown in Appendix A Table 6). We evaluate pruned models accordingly.

LLM	Ratio/Lay.	Avg.	Reas.	Lan.	Know.	Exam.	Unde.
Llama2 -7B	0%/32	46.55	60.83	40.67	68.67	38.89	33.03
	12.0%/28	36.13	44.46	**36.31**	**56.35**	**26.34**	24.54
	27.1%/23	**37.46**	**53.30**	33.84	54.89	25.85	**25.38**
	45.0%/17	30.00	43.66	19.27	48.06	24.78	21.44
Llama2 -13B	0%/40	55.50	62.51	55.80	69.20	47.18	47.34
	14.6%/34	**53.89**	**60.56**	**54.51**	**63.58**	**46.10**	**47.46**
	24.7%/30	47.55	57.17	42.85	58.32	39.28	42.60
	49.7%/20	38.27	48.20	26.89	49.26	32.82	36.58

Table 4: The model performance at different pruning ratios.

The average results are shown in Table 4 and the detailed results are shown in Appendix C Table 10.

From the results, it can be observed that as the pruning ratio increases, there is an overall decrease in model performance. However, from around 10-15% pruning ratio to around 25% pruning ratio, the model performance does not decrease significantly, which indicates that our method remains relatively stable within this range. Furthermore, at close to 50% pruning ratio, the model still maintains approximately 70% of its performance, which demonstrates that our method does not cause the model to crash even when about half of its parameters are removed.

5 Related Work

How to reduce the excessively high training and inference costs caused by the increase in the size of LLMs has been a critical issue of concern. In recent years, the methods to alleviate this issue have mainly consisted of model quantization, knowledge distillation, and model pruning.

Model Quantization reduces the size of models by converting their weights from high-precision floating-point representations to lower-precision floating-point or integer representations. SmoothQuant (Xiao et al., 2023) quantizes both weights and activation values and smooths the activation outliers. Gptq (Frantar et al., 2022) leverages approximate second-order information to weight quantization. Qlora (Dettmers et al., 2023a) backpropagates gradients through a frozen, 4-bit quantized model into Low Rank Adapters. OmniQuant (Shao et al., 2023) quantizes LLMs by optimizing various quantization parameters.

Knowledge Distillation transfers knowledge from a large model to a smaller one. Distilling step-by-step (Hsieh et al., 2023) trains smaller models that outperform LLMs. DISCO (Chen et al., 2023) distills counterfactual knowledge from LLMs. SOCRATIC COT (Shridhar et al., 2023) distills the ability of Chain-of-Thought from LLMs. ZEPHYR (Tunstall et al., 2023) applies distilled direct preference optimization to learn a chat model.

Model Pruning refers to the technique for improving the efficiency of models by sparsification or removing parameters. Non-structured pruning often involves model sparsity. SparseGPT (Frantar and Alistarh, 2023) works by reducing the pruning problem to a set of extremely large-scale instances of sparse regression. SpQR (Dettmers et al., 2023b) involves the identification and isolation of outlier weights during LLM sparsification. Structured pruning primarily discards portions of model modules. LLM-Pruner (Ma et al., 2023) selectively removes non-critical structures based on gradient information. ShearedLLaMA (Xia et al., 2023) employs targeted structured pruning and dynamic batch loading to prune the Llama2.

However, model quantization and sparsification methods have a significant impact on model performance and typically require special hardware support. Knowledge distillation entails considerable training costs and is usually task-specific. Existing structured model pruning methods often disrupt the inherent structure of the model. In contrast, our *LaCo* does not disrupt the model structure during pruning and is also more concise, while maintaining excellent performance.

6 Conclusion

In this paper, we propose a concise layer-wise structured pruning method called *Layer Collapse*, which involves the collapse of rear model layers into the preceding layer, facilitating a swift reduction in model size. Our *LaCo* does not require special hardware support and preserves the intrinsic structure of the model. Experimental results demonstrate that the proposed *LaCo* significantly outperforms existing SOTA structured pruning methods. Subsequently, we conduct post-training on the pruned models, verifying that *LaCo* effectively inherits the parameters of the original model and enables rapid convergence of the pruned model during post-training. Later, we also discuss our motivation from the perspective of layer-wise similarity. Finally, we explore the performance of models pruned by *LaCo* at different pruning ratios.

Limitations

Due to *LaCo*'s pruning process primarily relying on layer-wise iterations, it cannot directly control the pruning ratio like previous methods. Instead, it requires tuning hyperparameters such as the representation similarity threshold $\mathcal{T}$ for control. In future work, we will summarize additional experimental patterns regarding how to set hyperparameters to achieve a specific pruning ratio.

References

Saleh Ashkboos, Maximilian L Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. 2024. Slicegpt: Compress large language models by deleting rows and columns. *arXiv preprint arXiv:2401.15024*.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2019. Piqa: Reasoning about physical commonsense in natural language.

Zeming Chen, Qiyue Gao, Antoine Bosselut, Ashish Sabharwal, and Kyle Richardson. 2023. Disco: distilling counterfactuals with large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5514–5528.

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. *arXiv preprint arXiv:1905.10044*.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023a. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. 2023b. Spqr: A sparse-quantized representation for near-lossless llm weight compression. *arXiv preprint arXiv:2306.03078*.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

hiyouga. 2023. Llama factory. https://github.com/hiyouga/LLaMA-Factory.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. 2023. Cmmlu: Measuring massive multitask language understanding in chinese.

Chang Liu, Chongyang Tao, Jiazhan Feng, and Dongyan Zhao. 2022. Multi-granularity structural knowledge distillation for language model compression. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1001–1011.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *arXiv preprint arXiv:2305.11627*.

Michael S Matena and Colin A Raffel. 2022. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716.

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.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. 2023. Omniquant: Omnidirectionally calibrated quantization for large language models. *arXiv preprint arXiv:2308.13137*.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7059–7073.

Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. 2020. Investigating prior knowledge for challenging chinese machine reading comprehension. *Transactions of the Association for Computational Linguistics*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. 2023. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2023. Sheared llama: Accelerating language model pre-training via structured pruning. *arXiv preprint arXiv:2310.06694*.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, et al. 2020. Clue: A chinese language understanding evaluation benchmark. *arXiv preprint arXiv:2004.05986*.

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, et al. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. 2022. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35:27168–27183.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2023. Language models are super mario: Absorbing abilities from homologous models as a free lunch. *arXiv preprint arXiv:2311.03099*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence?

Qingru Zhang, Simiao Zuo, Chen Liang, Alexander Bukharin, Pengcheng He, Weizhu Chen, and Tuo Zhao. 2022. Platon: Pruning large transformer models with upper confidence bound of weight importance. In *International Conference on Machine Learning*, pages 26809–26823. PMLR.

Chujie Zheng, Minlie Huang, and Aixin Sun. 2019. ChID: A large-scale Chinese IDiom dataset for cloze test. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 778–787, Florence, Italy. Association for Computational Linguistics.

A Hyperparameter Settings

LLM	$\mathcal{C}$	$\mathcal{L}$	$\mathcal{H}$	$\mathcal{I}$	$\mathcal{T}$
Llama2-7B	4	1	32	2	0.65
Llama2-13B	6	1	40	2	0.75
Baichuan2-7B	4	1	32	2	0.70
Llama2-13B	6	1	40	2	0.70

Table 5: Hyperparameter settings for main results.

LLM (Ratio/Lay.)	$\mathcal{C}$	$\mathcal{L}$	$\mathcal{H}$	$\mathcal{I}$	$\mathcal{T}$
Llama2-7B (12.0%/28)	5	1	32	2	0.85
Llama2-7B (27.1%/22)	4	1	32	2	0.65
Llama2-7B (45.0%/17)	6	1	32	2	0.45
Llama2-13B (14.6%/34)	7	1	40	2	0.85
Llama2-13B (24.7%/30)	6	1	40	2	0.75
Llama2-13B (49.7%/20)	7	1	40	2	0.45

Table 6: Hyperparameter settings for varying pruning ratios.

B Post-Training Implementation Details

For post-training, we utilize the code framework provided by the LLaMA-Factory repository (hiyouga, 2023) along with DeepSpeed ZeRO-2. The sequence length is set to 4096 following the default settings of the Llama2-7B and Baichuan2-7B. We use Adam optimizer with the learning rate of 2e-4, where we set the Adam parameters $\beta_1 = 0.9$ and $\beta_2 = 0.95$. The batch size is set to 8 per GPU, resulting in a total batch size of 32. We configure the gradient accumulation steps to be 4 and utilize a cosine learning rate scheduler. Additionally, we apply weight decay of 0.1 and set the maximum gradient normalization to 1.0.

C Supplementary Results

LLM	Pruner	Ratio/Lay.	Reasoning(%)			Language(%)			Knowledge(%)		Examination(%)		Understanding(%)			
			CMNLI	HeSw	PIQA	CHID	WSC_P	WSC_G	CoQA	BoolQ	MMLU	CMMLU	$Race_H$	$Race_M$	XSum	C3
Llama2 -7B	Dense*	0%/32	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	LLMPru.	27.0%/32	104.09	**79.13**	**91.10**	54.84	97.44	2.50	63.76	78.11	50.81	79.25	63.53	67.42	58.49	58.57
	SliceGPT	26.4%/32	96.12	70.46	84.69	45.16	97.44	50.00	62.04	54.22	**62.98**	**79.63**	59.34	65.34	24.85	**90.86**
	LaCo	**27.1%/22**	**104.40**	78.05	89.28	**78.50**	**107.68**	**65.00**	**68.55**	**90.66**	57.60	79.22	**63.67**	**71.22**	**79.47**	90.61
Llama2 -13B	Dense*	0%/40	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	LLMPru.	24.4%/40	**100.12**	**90.55**	**96.17**	67.28	79.24	0.00	76.01	78.91	45.32	63.78	38.72	36.65	**81.37**	68.05
	SliceGPT	23.6%/40	90.39	74.45	86.61	36.45	71.70	**57.58**	70.63	52.95	66.76	66.57	40.34	39.89	22.37	88.23
	LaCo	**24.6%/30**	99.61	86.05	93.18	**75.70**	**103.77**	56.07	**78.70**	**89.48**	**82.56**	**84.20**	**93.90**	**93.87**	61.33	**94.57**
Baic2. -7B	Dense*	0%/32	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	LLMPru.	24.2%/32	96.73	**79.43**	**94.29**	84.43	**130.23**	0.00	**75.75**	**96.67**	45.95	45.11	41.73	43.65	**76.68**	64.51
	SliceGPT	22.2%/32	96.10	37.43	66.08	17.96	88.37	0.00	30.99	62.09	46.41	44.34	44.71	44.06	0.00	41.18
	LaCo	**24.2%/23**	**98.89**	77.38	89.93	**92.22**	102.32	**42.42**	74.85	88.70	**58.12**	**54.86**	**55.08**	**54.31**	57.73	**78.78**
Baic2. -13B	Dense*	0%/40	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	LLMPru.	24.3%/40	**101.78**	75.34	**91.99**	87.50	90.69	0.00	59.18	84.39	39.43	41.10	31.47	31.35	**54.79**	60.77
	SliceGPT	22.8%/40	96.57	36.36	65.36	12.50	88.37	0.00	27.47	56.46	39.02	41.23	32.05	31.22	0.00	38.07
	LaCo	**24.7%/30**	99.46	**85.39**	88.23	**92.26**	**106.96**	**95.46**	**84.53**	**93.06**	**87.32**	**87.56**	**84.61**	**83.84**	49.38	**93.08**

Table 7: The percentage of each model's score on each benchmark relative to the score of Dense* in the main results.

Prompt	Document: The 18-year-old scored 88.40 to make history in what was the fifth and the final stop of the World Cup season.\nShe came ahead of Sweden's Emma Dahlstrom and Swiss Mathilde Gremaud.\nBoston-born Atkin, who initially competed for the US before switching to Great Britain aged 15, was making her 15th appearance at a World Cup event.\nAtkin will be competing at the Freestyle World Championships in Sierra Nevada, Spain (9-19 March). The event will be live on the BBC Sport website, app, connected TV and red button.\nBased on the previous text, provide a brief single summary:
Pruner	**Generated Responses**
LLMPru.	\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n
SliceGPT	of the 19900s of the 1900s of the 1900s of the 1900s.
LaCo	Boston-born Atkin, who initially competed for the US before switching to Britain aged 15, was making her 15th appearance at a World Cup event.\nThe 18-year-old scored 88.40 to make history in what was the fifth and the final stop of the World Cup season.

Table 8: A response on the Xsum benchmark from Llama2-7B after pruning with different pruners.

LLM	Method	Reasoning			Language			Knowledge		Examination		Understanding			
		CMNLI	HeSw	PIQA	CHID	WSC_P	WSC_G	CoQA	BoolQ	MMLU	CMMLU	$Race_H$	$Race_M$	XSum	C3
Llama2 -7B	Dense*	32.98	71.35	78.18	46.04	37.50	38.46	66.67	70.67	45.92	31.86	35.51	33.15	19.68	43.78
	LaCo	34.43	55.69	69.80	36.14	**40.38**	25.00	45.70	64.07	26.45	25.24	22.61	23.61	**15.64**	**39.67**
	LaCo +post train	**34.92**	**61.88**	**73.18**	**38.12**	36.54	**34.62**	**57.49**	**66.21**	**29.47**	**25.33**	**28.33**	**29.87**	10.02	38.58
	LaCo +post train +re prune	33.80	45.35	65.07	23.27	36.54	0.96	38.49	60.43	26.07	25.37	23.07	22.98	15.48	36.71
Baichuan2 -7B	Dense*	33.37	67.56	76.17	82.67	41.35	63.46	63.14	63.30	54.25	56.95	52.63	51.04	20.84	64.55
	LaCo	**33.00**	52.28	68.50	76.24	**42.31**	**26.92**	47.26	56.15	**31.53**	**31.24**	**28.99**	**27.72**	12.03	50.85
	LaCo +post train	32.92	**52.67**	**69.42**	**78.22**	40.38	3.85	**52.01**	**55.93**	28.72	27.25	25.01	26.25	**15.82**	**58.03**

Table 9: The detailed scores across all benchmarks of pruned models, post-trained models, as well as post-trained models followed by re-pruning.

LLM	Ratio/Lay.	Reasoning			Language			Knowledge		Examination		Understanding			
		CMNLI	HeSw	PIQA	CHID	WSC_P	WSC_G	CoQA	BoolQ	MMLU	CMMLU	$Race_H$	$Race_M$	XSum	C3
Llama2 -7B	0%/32	32.98	71.35	78.18	46.04	37.50	38.46	66.67	70.67	45.92	31.86	35.51	33.15	19.68	43.78
	12.0%/28	32.99	**55.91**	**74.48**	**42.57**	36.54	**29.81**	**52.58**	60.12	25.59	**27.10**	22.01	21.73	**17.97**	36.44
	27.1%/22	**34.43**	55.69	69.80	36.14	**40.38**	25.00	45.70	**64.07**	**26.45**	25.24	**22.61**	**23.61**	15.64	**39.67**
	45.0%/17	32.58	38.33	60.07	20.30	36.54	0.96	34.73	61.38	23.98	25.59	22.38	23.26	1.28	38.85
Llama2 -13B	0%/40	32.99	74.83	79.71	52.97	50.96	63.46	66.91	71.50	55.63	38.74	58.03	60.24	23.56	47.51
	14.6%/34	32.99	**71.88**	**76.82**	**51.98**	**63.46**	**48.08**	**63.72**	63.43	**53.97**	**38.23**	**59.35**	**61.49**	**21.32**	**47.67**
	24.7%/30	32.86	64.39	74.27	40.10	52.88	35.58	52.66	**63.98**	45.93	32.62	54.49	56.55	14.45	44.93
	49.7%/20	**34.22**	46.55	63.82	13.37	56.73	10.58	36.28	62.23	38.41	27.24	51.97	56.41	1.56	36.38

Table 10: The detailed results of models pruned at different pruning ratios using LaCo across all benchmarks.