PASER: Post-Training Data Selection for Efficient Pruned Large Language Model Recovery

Anonymous Authors[1]

Abstract

Model pruning is an effective approach for compressing large language models. However, this process often leads to significant degradation of model capabilities. While post-training techniques such as instruction tuning are commonly employed to recover model performance, existing methods often overlook the uneven deterioration of model capabilities and incur high computational costs. Moreover, some instruction data irrelevant to model capability recovery may introduce negative effects. To address these challenges, we propose the **P**ost-training d**A**ta **S**election method for **E**fficient pruned large language model **R**ecovery (**PASER**). PASER aims to identify instructions where model capabilities are most severely compromised within a certain recovery data budget. Our approach first applies manifold learning and spectral clustering to group recovery data in the semantic space, revealing capability-specific instruction sets. We then adaptively allocate the data budget to different clusters based on the degrees of model capability degradation. In each cluster, we prioritize data samples where model performance has declined dramatically. To mitigate potential negative transfer, we also detect and filter out conflicting or irrelevant recovery data. Extensive experiments demonstrate that PASER significantly outperforms conventional baselines, effectively recovering the general capabilities of pruned LLMs while utilizing merely 4%-20% of the original post-training data.

1. Introduction

Model pruning, which aims at reducing model parameter amounts while maintaining model capabilities, has been a promising approach for large language model (LLM) compression. Mainstream LLM pruning schemes include unstructured (Frantar & Alistarh, 2023), semi-structured (Sun et al., 2024), and structured pruning (Ma et al., 2023). In practice, the capability degradation of the pruned model compared with the original LLM is almost unavoidable, especially under high pruning ratios. This degradation phenomenon is often more severe for the structured pruning scheme (Dong et al., 2024), which has been widely adopted in industrial LLM compression thanks to its hardware-friendly property. Therefore, first *pruning*, then recovery *post-training* has been a standard pipeline (Ma et al., 2023; Zhao et al.). Among various types of data including pre-training corpora and extensive fine-tuning datasets (Xia et al., 2024; Sun et al., 2024), instruction tuning data has demonstrated unique advantages for efficient capability recovery (Ma et al., 2023; Zhao et al.; Zhang et al., 2024; Chen et al., 2023). Compared to pre-training which requires massive computational resources, instruction tuning enables effective recovery with a much smaller data scale by explicit supervision. Furthermore, through the diverse task coverage, like language modeling, common sense reasoning, mathematical problem solving, and code generation, instruction tuning preserves the model's general-purpose capabilities while preventing over-specialized recovery.

Conventional schemes (Ma et al., 2023) usually employ the full version of publicly available instruction tuning datasets like Alpaca (Taori et al., 2023) to conduct the recovery post-training. However, this can bring significant computation overhead and even unsatisfactory recovery performance (See Appendix A). An intuitive solution is to take part of the original data for training, thus consuming less data and correspondingly reducing the computation resource demand. Nevertheless, directly utilizing the uniformly split data subset (e.g., first 20% of the data), can lead to sub-optimal performance, or even performance degradation. Moreover, the recovered performance considerably varies for models trained with different subsets. Therefore, selecting the most valuable instruction-tuning data that can contribute to recovery performance and reduce training costs becomes crucial. Though previous works have noticed the significance of selecting high-quality data to conduct the general instruction tuning (Wang et al., 2024), few of them are specifically

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

designed for the recovery post-training. Note that general high quality does not necessarily mean useful for recovery.

Considering the above limitations, the ideal recovery training data selection approach should exhibit the following properties: (1) **Targeted and Balanced Capability Recovery**: Given the uneven deterioration of different capabilities in the pruning (see Appendix A), the ideal selection method should identify and prioritize the severely-degraded ones, while ensuring balanced recovery of the model's overall functionality. Therefore, it needs to select a comprehensive instruction set that effectively targets the most affected capabilities while still allocating appropriate recovery data to less impacted ones. (2) **Recovery Training Efficiency**: Limited computing resources pose serious efficiency challenges to the LLM recovery post-training. An ideal method should be able to select instructions that are both most beneficial for recovery and low in computational cost, thereby accelerating the training process and optimizing resource utilization. (3) **Mitigation of Negative Transfer**: Recognizing that not all instruction data is beneficial for model recovery, an optimal approach should not only identify the most advantageous instructions, but also filter out potentially harmful or irrelevant ones. This significantly reduces the risk of negative transfer during the recovery training, ensuring that the selected data contributes positively for model recovery.

To achieve such goals, we propose the **P**ost-training d**A**ta **S**election method for **E**fficient pruned large language model **R**ecovery (**PASER**). First, we perform semantic-structural recovery instruction clustering to obtain different data groups corresponding to different LLM capabilities. Second, we select recovery instructions in a capability degradation-aware manner, where the overall data budget is allocated to different clusters based on their corresponding capability degradation degrees. In particular, when selecting samples within each capability cluster, the computation cost of each sample is also taken into consideration to optimize the efficiency of the recovery process. Finally, we construct the concept consistency graph to maintain the semantic consistency across selected instructions, thus preventing introducing conflicting or irrelevant samples. We take the LLaMA 2/3 and Baichuan2 as the target LLMs and perform the experiments under different LLM pruning schemes and different-sized instruction tuning datasets. The comparison with random and conventional instruction tuning data selection baselines demonstrates that PASER can more effectively enhance the recovered LLM performance on language modeling and various reasoning tasks. Meanwhile, the recovery training overhead can also be reduced significantly.

2. Related Works

Large Language Model Pruning can be generally divided into three categories: unstructured pruning, semi-structured pruning, and structured pruning. Unstructured pruning removes individual weights without structural constraints, with representative works including SparseGPT (Frantar & Alistarh, 2023), Wanda (Sun et al., 2024), BESA (Xu et al., 2024b), and OWL (Yin et al., 2024). This technique allows for maximum flexibility in weight selection and can achieve high compression rates while maintaining model performance. However, the resulting irregular sparsity patterns limits the practical acceleration. Semi-structured pruning (Guo et al., 2024; Malla et al., 2024; Frantar & Alistarh, 2023; Sun et al., 2024) targets specific patterns like N:M sparsity, balancing flexibility and hardware efficiency. Structured pruning approaches like LLM-Pruner (Ma et al., 2023) and SliceGPT (Ashkboos et al., 2024) remove entire structural components, offering better hardware compatibility and attracting industry attention (Ko et al., 2023; An et al., 2024; Song et al., 2024; Xia et al., 2024). However, structured pruning faces more severe performance degradation, highlighting the importance of recovery post-training.

Instruction Tuning has emerged as a crucial technique for enhancing LLMs (Wei et al.; Wang et al., 2023), improving their adaptability to novel tasks (Sanh et al.; Liang et al.; Zhou et al., 2024). Recent works have explored instruction tuning as a post-compression recovery mechanism (Zhao et al.; Ma et al., 2023). While promising, this combination faces challenges from reduced model capacity and computational costs. Most current approaches use general instruction datasets without considering compressed model's characteristics or disproportionately affected capabilities. Our work addresses these gaps by proposing a novel framework for post-training data selection in pruned LLM recovery.

3. Methodology

In this section, we first formulate the problem and then introduce three main components of the PASER framework (shown in Figure 1): semantic-structural recovery instruction clustering, capability degradation-aware instruction selection, and negative transfer mitigation. Furthermore, we provide the time complexity analysis for PASER process.

3.1. Problem Formulation

Let M_o denote the original large language model and M_p the pruned version of this model. We define the instruction tuning dataset as $D = \{(x_i, y_i)\}_{i=1}^N$, where x_i represents an instruction and y_i its corresponding output. Our goal is to select a subset $S \subset D$ to efficiently recover the performance of M_p. We formulate the problem as an optimization task:

$$S^* = \underset{S \subset D, |S| \leq B}{\arg\min} \ \mathbb{E}_{(x,y)\sim\mathcal{T}}[\mathcal{L}(M_r(S), x, y)], \quad \text{s.t.} \quad M_r(S) = \text{RecoveryTrain}(M_p, S) \tag{1}$$

where $M_r(S)$ is the recovered model after training on sub-

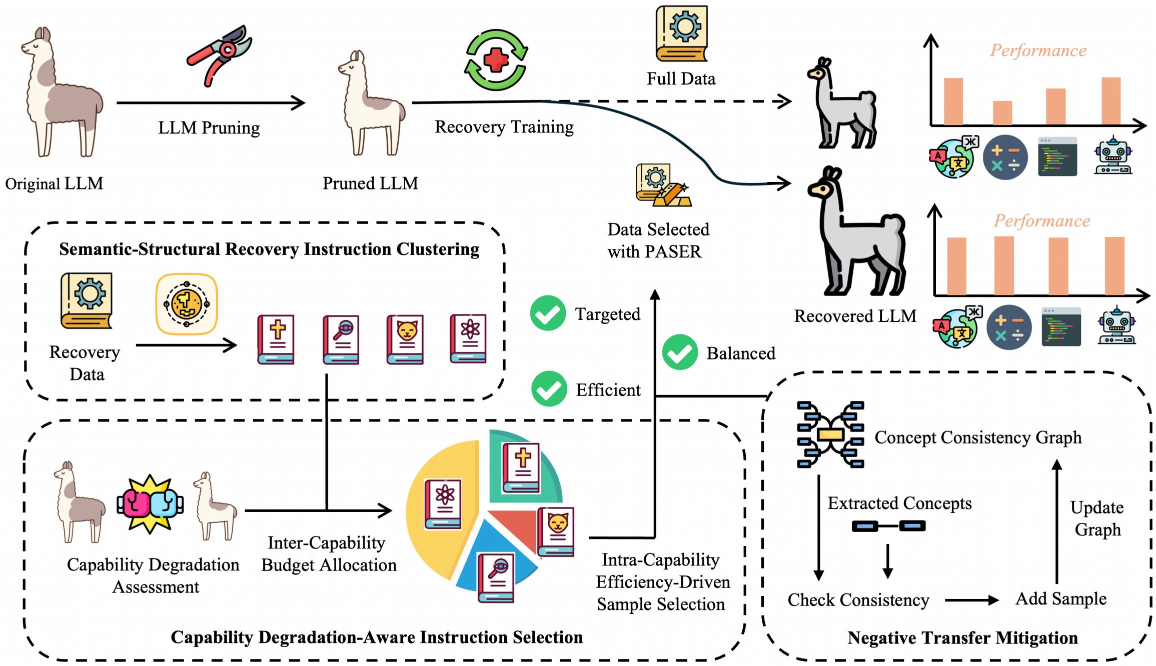

Figure 1: Overall Framework for our proposed PASER recovery data selection framework.

set S, $\mathcal{T}$ is the distribution of downstream evaluation tasks, $\mathcal{L}$ is a loss function. $B(B < N)$ is the recovery data budget, i.e., the maximum number of samples allowed in the selected subset.

3.2. Semantic-Structural Recovery Instruction Clustering

During the LLM pruning process, different model capabilities can be affected unevenly by pruning. To ensure targeted and balanced recovery, we need to identify and group data points that focus on similar capabilities. To achieve this goal, we hypothesize that distinct geometric topological structures of recovery instruction data in the high-dimensional semantic space may correspond to different aspects of LLM capabilities. This hypothesis is based on the intuition that instructions requiring similar capabilities are likely to cluster together in the semantic space, forming identifiable topological structures. In detail, we employ a two-step approach on the embedding space of instructions. First, we apply a diffusion kernel to SentenceBERT (Reimers & Gurevych, 2019) embeddings for manifold learning:

$$e(x_i) = \text{DiffusionKernel}(\text{SentenceBERT}(x_i)). \quad (2)$$

Here, $e(x_i)$ is the obtained low-dimensional manifold representation of instruction x_i. This process helps uncover the intrinsic geometric structure in the semantic space while reducing dimensionality and preserving non-linear relationships. Then, non-negative matrix factorization (NMF)-based spectral clustering (Ding et al., 2005) is conducted based on such $e(x_i)$ to identify natural groupings of instructions that potentially correspond to different LLM capabilities, leading to K clusters as follows:

$$\begin{aligned} C &= \{c_1, \ldots, c_K\} \\ &= \text{NMFSpectralClustering}(\{e(x_i)|(x_i, y_i) \in D\}). \end{aligned} \quad (3)$$

The detailed process of these two steps are provided as below. In the first step of manifold learning, we first obtain the SentenceBERT embedding of each instruction x_i. Then, an adjacency matrix A is constructed based on the pairwise Euclidean distances of these embeddings: $A_{ij} = \exp(-\frac{\|\text{SentenceBERT}(x_i) - \text{SentenceBERT}(x_j)\|_2^2}{2\sigma^2})$, where σ is a scaling parameter, typically set to the median of all pairwise distances. The degree matrix D is then computed as a diagonal matrix where each diagonal element is the sum of the corresponding row in A:$D_{ii} = \sum_{j=1}^{n} A_{ij}$. Using these matrices, we define the normalized graph Laplacian $L = I - D^{-1/2}AD^{-1/2}$, where I is the identity matrix. We then apply the diffusion kernel to this Laplacian $K_t = \exp(-tL)$, where K_t is the diffusion kernel at time t. The diffusion time t is selected using the spectral gap method: $t_{opt} = \arg\max_t \left(\frac{d\log(\lambda_2(t))}{d\log(t)}\right)$, where $\lambda_2(t)$ is the second eigenvalue of K_t. The low-dimensional manifold representation $e(x_i)$ is then obtained by selecting the top d eigenvectors of $K_{t_{opt}}$: $e(x_i) = [\phi_1(x_i), \phi_2(x_i), ..., \phi_d(x_i)]$, where ϕ_j are the eigenvectors of $K_{t_{opt}}$ corresponding to the d largest eigenvalues.

In the second step, we perform NMF-based spectral clustering on these low-dimensional representations. Specifically, we construct a similarity matrix S from the manifold representations $S_{ij} = \exp(-\frac{\|e(x_i)-e(x_j)\|^2}{2\sigma^2})$. We then determine the optimal number of clusters K by performing NMF with different values of k and selecting the one that minimizes the Frobenius norm of the approximation error $K = \arg\min_k \|S - W_k H_k^T\|_F$, where W_k and H_k are non-negative matrices resulting from NMF with k components. With this optimal K, we decompose the similarity matrix S using NMF such that $S \approx WH^T$, where W and H are non-negative matrices with K columns. Finally, we assign each data point to a cluster based on the maximum value in each row of W, where $c_i = \arg\max_j W_{ij}, i = 1, ..., N$. This results in K clusters $C = \{c_1, \ldots, c_K\}$, where the number of clusters K is adaptively determined through the above process.

3.3. Capability Degradation-aware Instruction Selection

Capability Degradation Assessment To prioritize the severely affected capabilities and finally achieve the balanced recovery of pruned LLMs, we need a measure of capability degradation to guide the data selection. For each cluster c_k obtained in Section 3.2, we define the capability degradation score (CDS) with the Jensen-Shannon divergence (JSD) (Fuglede & Topsoe, 2004) as follows:

$$\frac{1}{|c_k|}\sum_{(x,y)\in c_k}\frac{1}{|y|}\sum_{m=1}^{|y|}\text{JSD}(P(t_m|M_p,x)||P(t_m|M_o,x)). \tag{4}$$

Here, $P(t_m|M_p,x)$ represents the output probability distribution on the m-th token of the pruned model M_p given input x. Taking a token $t_m^i (1 \le i \le |\text{Voc}|)$ in this distribution as an example, its corresponding probability is:

$$P(t_m^i|M_p,x) = \frac{\exp(\frac{\text{logit}(t_m^i)}{\tau})}{\sum_{j=1}^{|\text{Voc}|}\exp(\frac{\text{logit}(t_m^j)}{\tau})}, \tag{5}$$

where τ is the softmax temperature and the $|\text{Voc}|$ indicates the vocabulary size. $\text{logit}(\cdot)$ is the logit value for tokens produced by LLM. Similarly, the $P(t_m|M_o,x)$ represents the output probability distribution for the original model M_o. The JSD is actually the symmetrized and smoothed version of the Kullback–Leibler divergence (KLD) (Kullback & Leibler, 1951): $\text{JSD}(X||Y) = \frac{1}{2}\text{KLD}(X||M) + \frac{1}{2}\text{KLD}(Y||M)$. The distribution $M = \frac{1}{2}(X+Y)$ is the mixed distribution of X and Y.

Thus, the obtained CDS quantifies the average performance degradation for data points in each capability cluster. The choice of JSD over simple loss variations as the performance degradation signal is motivated by its unique properties. First, its symmetry ensures consistent comparison between the pruned model M_p and the original model M_o, while its bounded range (0 to 1) provides a normalized measure of divergence. This facilitates easier interpretation and comparison across different capability clusters. Moreover, JSD's robustness to outliers and its information-theoretic foundation allow for a more nuanced assessment of capability degradation, capturing subtle changes in model outputs that might not be apparent from loss or accuracy values alone (Dutta et al., 2024) due to the sampling uncertainty. The smoothing effect introduced by the use of the mixed distribution in JSD calculation also contributes to a more stable assessment across diverse instruction types. By employing JSD in our CDS calculation, we obtain a comprehensive and reliable assessment of capability degradation, enabling more accurate identification and prioritization of the capabilities most severely affected by model pruning.

Inter-Capability Budget Allocation Sampling a subset of high-quality data from the original set to achieve better training performance is the objective of the data selection process. To ensure the efficiency on data utilization and training process, the instruction data budget $B (B < N)$ should be maintained. Under this budget, we design an adaptive selection approach based on the above CDS for balanced recovery while focusing on severely affected capabilities. In detail, we allocate sampling budget to each cluster proportionally to its corresponding CDS:

$$n_k = \left\lfloor B \cdot \frac{\text{CDS}(c_k)}{\sum_{j=1}^{K}\text{CDS}(c_j)} \right\rfloor. \tag{6}$$

n_k is the sample number budget allocated to cluster c_k.

Intra-Capability Efficiency-Driven Sample Selection To maximize computational efficiency during the recovery post-training phase, we need to select samples that offer the highest recovery benefit relative to their computational cost. Within each cluster c_k, we select top n_k samples based on their Individual Efficiency Scores (IES):

$$\text{IES}(x,y) = \frac{\frac{1}{|y|}\sum_{m=1}^{|y|}\text{JSD}(P(t_m|M_p,x)||P(t_m|M_o,x))}{\log \text{ComputationalCost}(x,y)}. \tag{7}$$

Here, ComputationalCost is instantiated with the quadratic term of sequence length $(|x|+|y|)^2$ as the approximated complexity for LLM training. The use of JSD captures the degree of divergence between the pruned and original models' outputs, indicating areas where capabilities have been most affected and thus offering the highest potential for targeted recovery. The logarithmic term balances the consideration of computational cost, allowing for a more careful selection that favors efficient samples without overly penalizing high-potential, moderately costly instances.

3.4. Negative Transfer Mitigation

To prevent performance degradation due to conflicting or irrelevant information, we need to detect and mitigate potential negative transfer. We introduce a Concept Consistency

Graph (CCG) to model relationships between concepts in the selected data. Here, a concept refers to a key knowledge unit or semantic element extracted from an instruction tuning sample. Concepts play a crucial role in capturing the essential information within instructions and help to identify potential conflicts that could lead to negative transfer. By managing relationships between concepts, we aim to maintain semantic consistency across the selected instruction tuning dataset, thereby reducing the risk of learning conflicting or irrelevant information during the recovery process. The formal definition for CCG is provided as follows:

Definition 1 (Concept Consistency Graph)**.** *A CCG is a graph $G = (V, E)$ where vertices V represent concepts, and an edge $(v_i, v_j) \in E$ exists if concepts v_i and v_j co-occur in at least one instruction tuning sample without conflict.*

For each new sample (x, y), we extract its concept $C(x, y)$ and check for consistency: IsConsistent$(x, y) = \forall v_i, v_j \in C(x, y) : (v_i, v_j) \in E$ or $\{v_i, v_j\} \not\subset V$. We only add samples that are consistent with the existing CCG, updating the graph with each addition. This approach ensures that we maintain a coherent set of instructions, minimizing the risk of negative transfer by avoiding the introduction of conflicting concepts during the recovery training process. The full version of our algorithm is provided in Appendix B.

3.5. Time Complexity Analysis

We provide a comprehensive analysis of the time complexity for the PASER algorithm.

Theorem 1. *The overall time complexity of PASER is $O(N \log N + NC^2)$, where N is the number of instructions in D, and C is the maximum number of concepts in any instruction tuning sample.*

Proof. We analyze each step of the algorithm in detail: The Semantic-Structural Recovery Instruction Clustering step, including SentenceBERT embedding, Diffusion Kernel computation, and NMF Spectral Clustering, has a dominant complexity of $O(N \log N)$. For the Capability Degradation Assessment step, computing JSD for each sample and calculating CDS for each cluster take $O(N)$ time in total. The Inter-capability Budget Allocation, which involves allocating the budget to clusters, has a time complexity of $O(K)$, where K is the number of clusters. However, since $K \leq N$, this step does not dominate the overall complexity. During Intra-capability Efficiency-Driven Sample Selection, for each cluster c_k, we perform sorting by JSD ($O(|c_k| \log |c_k|)$), iterate through sorted samples ($O(|c_k|)$), perform consistency checks (IsConsistent, $O(C^2)$ per sample), and update the CCG ($O(C^2)$ per sample). Considering all clusters, this step's total complexity is $O(N \log N + NC^2)$. Thus, the overall time complexity is dominated by the clustering step and the intra-capability sample selection step. Therefore, the total time complexity is $O(N \log N + NC^2)$.

In practice, C is often much smaller than N ($C \ll N$) and can be considered as a constant factor for large N. Thus, we can simplify the complexity to $O(N \log N)$. This analysis demonstrates that PASER is computationally efficient and scalable for large instruction tuning datasets.

4. Experiments

4.1. Experiment Setup

Target LLMs The experiments are performed on several open-source popular English LLMs: LLaMA2-7B/13B/70B (Touvron et al., 2023) (hf version), LLaMA3-8B (Dubey et al., 2024)(instruct version), and bilingual LLMs: Baichuan2-7B/13B (Yang et al., 2023)(base version), which support both English and Chinese.

Instruction Tuning Datasets As for the original recovery post-training data, we choose two different-size instruction tuning datasets: Alpaca (Taori et al., 2023) and LaMini (Wu et al., 2024). Alpaca contains 52K instruction-following samples generated using OpenAI's *text-davinci-003* model based on 175 human-written seed tasks. LaMini contains a total of 2.58M pairs of instructions and responses synthesized with *gpt-3.5-turbo* based on several existing resources of prompts, including self-instruct (Wang et al., 2023), P3 (Sanh et al., 2022), FLAN (Longpre et al., 2023) and Alpaca (Taori et al., 2023).

Base Pruning Schemes Different pruning schemes are incorporated into our experiments to explore the applicability of PASER, ranging from structured pruning methods: LLM-Pruner (Ma et al., 2023), SliceGPT (Ashkboos et al., 2024), semi-structured pruning method: Wanda (Sun et al., 2024), and unstructured pruning method: SparseGPT (Frantar & Alistarh, 2023).

Instruction Tuning Data Selection Baselines In addition to the random data selection, we also compare PASER with several recent general instruction tuning data selection baselines: Instruction Mining (Cao et al.), IFD (Li et al., 2024a), Nuggets (Li et al., 2024b). Note none of these baselines are customized for post-pruning recovery training scenario. Besides, the evaluation performance of recovery training with the full original dataset is also compared.

Evaluation Datasets and Tasks To thoroughly evaluate the performance of recovered LLMs, we employ seven common sense reasoning datasets:BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), ARC-Easy (Clark et al., 2018), ARC-Challenge (Clark et al., 2018), and OpenbookQA (Mihaylov et al., 2018). In the practice, we relies on the open-source library[1] to implement the evaluation, where the model needs to rank the choices in the multi-

[1] https://github.com/EleutherAI/lm-evaluation-harness

Table 1: Recovery performance of different instruction tuning data selection methods under various pruning schemes on LLaMA2-7B model. The 'bold' represents the best performance under the same pruning scheme. Here, the Alpaca is taken as the original dataset.

Pruning	Recovery Post-training	WikiText2↓	PTB↓	BoolQ	PIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Average
w/o pruning	w/o Training	12.62	22.14	71.13	78.40	72.79	67.17	69.36	40.70	40.80	62.91
LLM-Pruner ratio=25%	w/o Training	20.34	38.81	61.87	76.61	65.86	60.22	63.13	37.37	39.40	57.78
	Full Data	736.42	1273.10	37.83	53.21	26.42	49.57	25.29	28.16	29.00	35.64
	Random	93.77	180.62	57.61	64.37	45.39	55.87	43.78	31.94	34.90	47.69
	Instruction Mining	23.31	40.63	61.59	75.68	66.08	60.71	62.34	37.96	39.20	57.65
	IFD	19.76	33.30	63.55	77.23	67.21	60.90	63.46	37.81	40.00	58.59
	Nuggets	20.02	35.19	63.62	**77.43**	67.36	61.08	63.77	37.64	**39.90**	58.69
	PASER	**16.40**	**26.35**	**67.25**	77.29	**68.98**	**66.97**	**67.84**	**39.54**	39.80	**61.10**
SliceGPT ratio=25%	w/o Training	44.53	80.07	65.54	66.87	54.16	63.38	58.46	34.56	36.90	54.27
	Full Data	38.24	68.53	68.75	69.84	57.92	66.18	62.37	36.82	38.30	57.17
	Random	41.86	74.92	66.89	68.21	55.79	64.56	60.23	35.47	37.60	55.54
	Instruction Mining	39.75	71.28	67.87	68.93	56.42	65.76	61.89	36.23	37.60	56.39
	IFD	37.75	67.48	69.23	70.54	58.38	67.12	63.75	37.18	38.40	57.80
	Nuggets	23.86	35.42	69.89	71.21	58.79	67.56	**72.23**	37.47	38.60	59.39
	PASER	**12.24**	**21.53**	**72.75**	**79.84**	**73.92**	**69.18**	71.37	**41.82**	**41.30**	**64.31**
Wanda sparsity=2:4	w/o Training	42.10	76.85	69.30	71.99	53.06	62.75	60.94	28.07	34.60	54.39
	Full Data	27.63	50.22	70.77	74.87	63.78	65.26	65.30	34.04	37.10	58.73
	Random	35.98	65.24	69.68	73.14	58.65	63.69	63.16	31.91	36.20	56.63
	Instruction Mining	31.47	57.17	70.61	73.85	61.27	64.13	64.72	33.79	36.80	57.88
	IFD	25.82	46.78	71.06	75.57	64.15	65.38	66.55	35.63	37.60	59.42
	Nuggets	23.98	43.24	**71.68**	76.14	64.65	65.69	66.16	36.91	38.20	59.92
	PASER	**14.13**	**27.22**	70.77	**77.87**	**71.78**	**66.26**	**68.30**	**39.04**	**40.10**	**62.02**
SparseGPT sparsity=50%	w/o Training	19.26	36.41	71.22	75.60	62.85	66.06	69.11	36.86	37.80	59.93
	Full Data	25.83	47.26	69.10	74.15	59.68	67.76	63.74	**39.59**	37.80	58.83
	Random	28.74	50.85	67.84	75.39	57.14	68.92	59.76	37.34	36.60	57.57
	Instruction Mining	24.08	45.51	70.50	74.47	61.91	65.40	67.73	36.49	37.40	59.13
	IFD	21.19	40.05	71.06	75.13	62.79	65.72	68.80	36.23	37.20	59.56
	Nuggets	16.21	28.95	71.64	75.67	63.33	66.05	69.49	36.60	37.40	60.03
	PASER	**13.33**	**23.77**	**74.79**	**78.38**	**66.62**	**69.03**	**72.57**	38.70	**39.40**	**62.78**

ple choice tasks or generate the answer in the open-ended generation tasks. The whole process is conducted in the *zero-shot* manner. Besides, we follow (Ma et al., 2023) to evaluate the language modeling capability with the zero-shot perplexity (PPL) analysis on WikiText2 (Merity et al., 2022) and PTB (Marcus et al., 1993). For enhancing evaluation comprehensiveness, we also conduct experiments on mathematical problem solving and code generation tasks, as shown in Appendix C and D, respectively.

4.2. Experiment Results and Analysis

Recovery Performance for Different Pruning Schemes We evaluate the recovery performance of LLaMA2-7B using different instruction tuning data selection methods under structured pruning, semi-structured pruning, and unstructured pruning, respectively. According to the results in Table 1, directly employing full data can indeed bring the suboptimal recovery performance, especially under the LLM-Pruner. This is because the full version of data contains some irrelevant or conflicting information for capability recovery, resulting in the negative transfer during the training phase. Meanwhile, even the general instruction tuning data selection methods like IFD and Nuggets can bring better reasoning and language model performance than full data and random in most cases, validating the necessity of conducting recovery data selection. Furthermore, we can find that previous selection methods can hardly help model recover to the level of unpruned status, under the limited data budget. However, our PASER can not only outperform baselines, but also reduce the averaged reasoning performance degradation to less than 3% under LLM-Pruner, Wanda, and SparseGPT. Especially, when pruning LLaMA2-7B with SliceGPT, our PASER can improve the average reasoning performance to 64.31, higher than the unpruned model. Besides, its zero-shot perplexity on WikiText2 and PTB is also lower than unpruned model slightly. This suggests that allocating recovery budget according to capability degradation levels and prioritizing most-affected samples exhibit the potential of help pruned LLM recover to the capability level of unpruned status. Besides, PASER can also be extended to other LLM post-compression scenarios, like the post-quantization recovery. The corresponding results and analysis are provided in Appendix E.

Robustness over Various Target Large Language Models To validate whether PASER can maintain the robust effectiveness among various target LLMs, we conduct the experi-

Table 2: Recovery performance of different instruction tuning data selection methods on different target LLMs under LLM-Pruner scheme. The 'bold' represents the best performance on the same target LLM. Here, the Alpaca is taken as the original dataset. The "Reason" indicates the averaged performance on 7 common sense reasoning datasets.

Model	Benchmark	w/o pruning	w/o Training	Full Data	Random	Instruction Mining	IFD	Nuggets	PASER
LLaMA2-13B ratio=50%	WikiText2↓	11.58	73.52	27.74	39.85	44.37	38.61	33.50	**21.67**
	PTB↓	20.24	151.19	45.08	76.20	80.82	73.25	61.26	**35.09**
	Reason↑	64.78	48.86	56.40	54.62	54.09	54.77	55.25	**57.62**
LLaMA2-70B ratio=50%	WikiText2↓	8.92	46.81	31.76	25.34	23.16	22.87	19.63	**12.35**
	PTB↓	15.59	92.36	56.83	48.72	43.45	43.68	36.24	**21.82**
	Reason↑	71.72	61.14	65.56	64.03	66.74	67.27	67.73	**69.62**
LLaMA3-8B ratio=25%	WikiText2↓	7.36	15.47	9.58	12.52	13.25	11.04	10.31	**8.09**
	PTB↓	12.87	28.31	16.73	22.17	23.47	19.31	18.02	**14.16**
	Reason↑	70.14	63.45	67.84	65.60	65.47	66.64	67.30	**69.83**
Baichuan2-7B ratio=25%	WikiText2↓	14.42	28.30	25.29	27.04	34.24	24.83	21.48	**16.92**
	PTB↓	26.78	53.34	35.81	46.83	60.93	37.81	37.65	**30.76**
	Reason↑	64.19	56.33	57.39	57.09	54.78	57.36	57.84	**59.70**
Baichuan2-13B ratio=50%	WikiText2↓	11.23	58.41	24.35	40.44	36.82	33.45	28.96	**14.62**
	PTB↓	18.04	116.26	42.68	76.57	70.45	63.23	53.31	**29.82**
	Reason↑	67.25	57.59	61.64	59.12	59.38	60.34	61.09	**63.75**

ments on LLaMA2-7B/13B/70B, LLaMA3-8B, Baichuan2-7B/13B, under LLM-Pruner. According to results in Table 2, we can first observe the model capability under high pruning ratio (50%) is hard to recover to unpruned level, especially for relatively smaller model like LLaMA2-13B and Baichuan2-13B. Though, PASER can still outperform random and best-performing data selection baseline, Nuggets, by 4.41 and 2.31 points, respectively on average. Especially, for LLaMA2-70B, our PASER can control the averaged reasoning performance degradation to less than 3%. This can be explained that the structure redundancy in 70B model is relatively higher, paving the way for effective recovery through data selection under high pruning ratios. As for the second smallest model, LLaMA3-8B, PASER can recover the reasoning performance to the 99.56% of the unpruned status, which further demonstrates the robustness of PASER over different target LLMs. Finally, the performance of various recovery methods including PASER on Baichuan2-7B is not satisfying enough given only 25% pruning ratio, which can be attributed to that the pruning process has severely damaged the model internal structure.

Recovery Performance with Different Instruction Tuning Datasets In addition to the recovery performance on Alpaca shown in Table 1, we also explore the corresponding results on another larger dataset, LaMini. Especially, considering the space limitation and more severe performance degradation of structured pruning schemes, we provide the experiments results on LLM-Pruner and SliceGPT on Table 3. From this table, we can observe that PASER can still consistently outperform other data selection and random methods. Besides, comparing the results in Table 1 and 3, it can be found that improving the data scale (from 10K to 10K samples) indeed facilitates the recovery performance though the significantly increased computational overhead,

Table 3: Recovery performance of different instruction tuning data selection methods under two structured pruning schemes on LLaMA2-7B model. The 'bold' represents the best performance under the same pruning scheme. Here, the LaMini is taken as the original dataset. The "Reason" indicates the averaged performance on 7 reasoning datasets

Pruning	Recovery Post-training	WikiText2↓	PTB↓	Reason↑
w/o pruning	w/o Training	12.62	22.14	62.91
LLM-Pruner ratio=25%	w/o Training	20.34	38.81	57.78
	Full Data	16.28	27.12	62.68
	Random	18.40	32.15	60.93
	Instruction Mining	17.83	28.87	60.76
	IFD	18.54	31.23	60.65
	Nuggets	18.27	30.90	60.99
	PASER	**13.45**	**22.63**	**63.79**
SliceGPT ratio=25%	w/o Training	44.53	80.07	54.27
	Full Data	24.36	35.64	58.31
	Random	39.86	70.92	56.68
	Instruction Mining	37.75	67.28	57.53
	IFD	25.75	53.48	58.94
	Nuggets	21.86	31.42	60.96
	PASER	**14.27**	**23.53**	**65.74**

which is consistent with the Scaling Law (Kaplan et al., 2020). We can also notice that the performance of full data on LaMini is relatively competitive, which is because the proportion of conflicting or negative data for recovery is much lower than that in Alpaca.

Ablation Study To validate the contribution of each component in PASER, we conduct comprehensive ablation studies. Specifically, we evaluate three variants: (1) PASER w/o S^2RIC: replacing semantic-structural clustering with random clustering while keeping other modules unchanged; (2) PASER w/o CDAIS: randomly sampling equal number of instructions from each cluster instead of using capa-

Table 4: Ablation study results on LLaMA2-7B for each component under different pruning schemes. The "Reason" indicates the averaged performance on 7 common sense reasoning datasets.

Ablation Variant	LLM-Pruner (25%)			SliceGPT (25%)			Wanda (2:4)			SparseGPT (50%)		
	WikiText2↓	PTB↓	Reason↑	WikiText2↓	PTB↓	Reason↑	WikiText2↓	PTB ↓	Reason↑	WikiText2↓	PTB↓	Reason↑
w/o S^{2}RIC	18.73	32.84	59.67	14.83	25.42	63.03	15.84	30.25	61.19	14.89	26.31	62.60
w/o CDAIS	17.56	30.15	60.26	14.16	24.92	62.68	15.46	29.48	61.23	14.62	25.84	62.49
w/o NTM	19.82	35.60	59.25	15.37	27.81	61.92	16.79	31.52	61.34	15.91	28.19	61.76
PASER	16.40	26.35	61.10	12.24	21.53	64.31	14.13	27.22	62.02	13.33	23.77	62.78

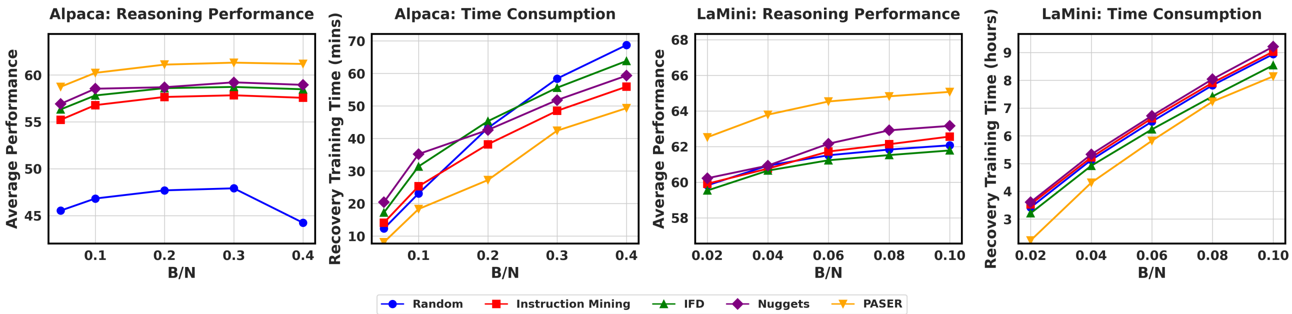

Figure 2: Average reasoning performance and recovery post-training time consumption curves corresponding to different instruction tuning data selection methods. The left two subfigures are for Alpaca while right two are subfigures for LaMini.

bility degradation-aware selection; (3) PASER w/o NTM: removing the negative transfer mitigation module. The results in Table 4 demonstrate that all three components contribute positively to model recovery across different pruning schemes. The semantic-structural clustering effectively identifies capability-specific instruction groups, leading to 0.18-1.43 points improvement in reasoning performance. Its removal causes degradation in both language modeling (increased perplexity) and reasoning tasks, particularly evident under structured pruning schemes like LLM-Pruner and SliceGPT. The capability degradation-aware selection mechanism enhances recovery efficiency through adaptive budget allocation, contributing 0.29-1.63 points improvement in reasoning tasks while maintaining stable language modeling performance. Notably, negative transfer mitigation shows significant impact (0.68-2.39 points improvement), especially under high pruning ratios, highlighting its importance in preventing conflicting information during recovery training. These improvements are consistently observed across different pruning schemes, with particularly pronounced effects in structured pruning where capability degradation tends to be more severe and uneven.

Recovery Post-training Efficiency Analysis To highlight PASER's advantages on recovery post-training efficiency, we conduct the experiments under different data budgets B and different datasets and record the corresponding averaged reasoning performance and training time in Figure 2. From the first and third subfigures, we can observe that PASER can obtain best recovery performance under different B/N on Alpaca and LaMini. Interestingly, in the first subfigure, when rising B/N from 0.3 to 0.4, the reasoning performance of Random even decreases. It is because expanding data scale also introduces the conflicting or negative data existing in the original dataset. From the second and fourth subfigures, PASER consistently consumes the least training time, which can be attributed to the efficiency-driven sample selection process in PASER. This advantage can be more obvious under low B/N like 0.02 on LaMini. This is because increasing data budget will force PASER to select some relatively more time-consuming samples given the fixed original dataset, weakening its efficiency superiority.

In addition, more experiment results and analysis can be found in Appendix C, D, E, F, G, H, I, and J.

5. Conclusion and Future Works

Recovery post-training has been an important procedure after large language model pruning to restore the critical capabilities. Previous works directly utilize the full instruction tuning dataset, facing high computation cost and risks of untargeted recovery and negative transfer. In this work, we propose the post-training data selection method for efficient pruned model recovery. According to capability degradation degrees, we allocate selection budget among different capability data obtained through semantic-structural clustering. We then select samples where model behavior has been severely affected while considering computation cost, and introduce a concept consistency graph to mitigate negative transfer. Extensive experiments on different LLMs demonstrate the effectiveness of our framework. Future work will explore other optimization approaches like data augmentation and revision to further improve recovery efficiency.

Impact Statement

This work aims to improve the efficiency of recovering pruned large language model capabilities while reducing computational costs. Our approach has several broader implications for society and the field of machine learning: On the positive side, PASER could help reduce the environmental impact of training large language models by enabling more efficient recovery processes that require only 4-20% of the original post-training data. This aligns with growing efforts to make AI development more environmentally sustainable. Additionally, by reducing computational requirements, our method could make working with large language models more accessible to researchers and organizations with limited computing resources, potentially democratizing access to AI technology. However, we acknowledge potential risks and challenges. As our method helps recover model capabilities more efficiently, it could accelerate the deployment of compressed language models in real-world applications. This warrants careful consideration of model safety, reliability, and potential misuse. While PASER focuses on maintaining model performance, future work should examine how capability recovery impacts model biases and fairness. We encourage implementing appropriate safeguards when deploying recovered models in practice. We also note that as this technology develops, it will be important to monitor its effects on energy consumption patterns in AI development and establish best practices for responsible use. The AI community should continue discussing how efficiency improvements like PASER can best serve society's interests while minimizing potential harms. Our hope is that by making language model compression more practical and efficient, this work will contribute to more sustainable and accessible AI development, while acknowledging the need for careful consideration of safety and ethical implications in its deployment.

References

An, Y., Zhao, X., Yu, T., Tang, M., and Wang, J. Fluctuation-based adaptive structured pruning for large language models. In Wooldridge, M. J., Dy, J. G., and Natarajan, S. (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pp. 10865–10873. AAAI Press, 2024. doi: 10.1609/AAAI.V38I10.28960. URL `https://doi.org/10.1609/aaai.v38i10.28960`.

Ashkboos, S., Croci, M. L., Nascimento, M. G. D., Hoefler, T., and Hensman, J. Slicegpt: Compress large language models by deleting rows and columns. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL `https://openreview.net/forum?id=vXxardq6db`.

Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., Jiang, E., Cai, C., Terry, M., Le, Q., et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Bisk, Y., Zellers, R., Gao, J., Choi, Y., et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Cao, Y., Kang, Y., Wang, C., and Sun, L. Instruction mining: Instruction data selection for tuning large language models.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. D. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Chen, T., Ding, T., Yadav, B., Zharkov, I., and Liang, L. Lorashear: Efficient large language model structured pruning and knowledge recovery. *arXiv preprint arXiv:2310.18356*, 2023.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of NAACL-HLT*, pp. 2924–2936, 2019.

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

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Ding, C., He, X., and Simon, H. D. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proceedings of the 2005 SIAM international conference on data mining*, pp. 606–610. SIAM, 2005.

Dong, H., Chen, B., and Chi, Y. Prompt-prompted adaptive structured pruning for efficient llm generation. In *First Conference on Language Modeling*, 2024.

Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Dutta, A., Krishnan, S., Kwatra, N., and Ramjee, R. Accuracy is not all you need. *arXiv preprint arXiv:2407.09141*, 2024.

Frantar, E. and Alistarh, D. Optimal brain compression: A framework for accurate post-training quantization and pruning. *Advances in Neural Information Processing Systems*, 35:4475–4488, 2022.

Frantar, E. and Alistarh, D. Sparsegpt: Massive language models can be accurately pruned in one-shot. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 10323–10337. PMLR, 2023. URL https://proceedings.mlr.press/v202/frantar23a.html.

Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. Optq: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

Fuglede, B. and Topsoe, F. Jensen-shannon divergence and hilbert space embedding. In *International symposium onInformation theory, 2004. ISIT 2004. Proceedings.*, pp. 31. IEEE, 2004.

Guo, Z., Kamigaito, H., and Wanatnabe, T. Dependency-aware semi-structured sparsity of GLU variants in large language models. *CoRR*, abs/2405.01943, 2024. doi: 10.48550/ARXIV.2405.01943. URL https://doi.org/10.48550/arXiv.2405.01943.

Hu, E. J., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., and Liu, Q. Tinybert: Distilling bert for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4163–4174, 2020.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Ko, J., Park, S., Kim, Y., Ahn, S., Chang, D., Ahn, E., and Yun, S. NASH: A simple unified framework of structured pruning for accelerating encoder-decoder language models. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pp. 6076–6093. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-EMNLP.404. URL https://doi.org/10.18653/v1/2023.findings-emnlp.404.

Kullback, S. and Leibler, R. A. On information and sufficiency. *The annals of mathematical statistics*, 22(1): 79–86, 1951.

Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., Slone, A., Anil, C., Schlag, I., Gutman-Solo, T., et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857, 2022.

Li, M., Zhang, Y., Li, Z., Chen, J., Chen, L., Cheng, N., Wang, J., Zhou, T., and Xiao, J. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7595–7628, 2024a.

Li, Y., Hui, B., Xia, X., Yang, J., Yang, M., Zhang, L., Si, S., Chen, L., Liu, J., Liu, T., Huang, F., and Li, Y. One-shot learning as instruction data prospector for large language models. In Ku, L., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 4586–4601. Association for Computational Linguistics, 2024b. URL https://aclanthology.org/2024.acl-long.252.

Liang, S., Tian, R., Zhu, K., Qin, Y., Wang, H., Cong, X., Liu, Z., Liu, X., and Sun, M. Exploring format consistency for instruction tuning. *Transactions on Machine Learning Research*.

Longpre, S., Hou, L., Vu, T., Webson, A., Chung, H. W., Tay, Y., Zhou, D., Le, Q. V., Zoph, B., Wei, J., et al. The flan collection: Designing data and methods for effective instruction tuning. In *International Conference on Machine Learning*, pp. 22631–22648. PMLR, 2023.

Ma, X., Fang, G., and Wang, X. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720, 2023.

Malla, S., Choi, J. H., and Choi, C. COPAL: continual pruning in large language generative models. In *Forty-first International Conference on Machine Learning, ICML*

2024, Vienna, Austria, July 21-27, 2024. OpenReview.net, 2024. URL https://openreview.net/forum?id=Lt8Lk7IQ5b.

Marcus, M., Santorini, B., and Marcinkiewicz, M. A. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2022.

Mihaylov, T., Clark, P., Khot, T., and Sabharwal, A. 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*, pp. 2381–2391, 2018.

Muralidharan, S., Sreenivas, S. T., Joshi, R., Chochowski, M., Patwary, M., Shoeybi, M., Catanzaro, B., Kautz, J., and Molchanov, P. Compact language models via pruning and knowledge distillation. *arXiv preprint arXiv:2407.14679*, 2024.

Park, G., Kim, M., Lee, S., Kim, J., Kwon, B., Kwon, S. J., Kim, B., Lee, Y., Lee, D., et al. Lut-gemm: Quantized matrix multiplication based on luts for efficient inference in large-scale generative language models. In *The Twelfth International Conference on Learning Representations*.

Reimers, N. and Gurevych, I. Sentence-bert: Sentence embeddings using siamese bert-networks. In Inui, K., Jiang, J., Ng, V., and Wan, X. (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pp. 3980–3990. Association for Computational Linguistics, 2019. doi: 10.18653/V1/D19-1410. URL https://doi.org/10.18653/v1/D19-1410.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Sanh, V., Webson, A., Raffel, C., Bach, S., Sutawika, L., Alyafeai, Z., Chaffin, A., Stiegler, A., Raja, A., Dey, M., et al. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Sanh, V., Webson, A., Raffel, C., Bach, S. H., Sutawika, L., Alyafeai, Z., Chaffin, A., Stiegler, A., Le Scao, T., Raja, A., et al. Multitask prompted training enables zero-shot task generalization. In *ICLR 2022-Tenth International Conference on Learning Representations*, 2022.

Song, J., Oh, K., Kim, T., Kim, H., Kim, Y., and Kim, J. SLEB: streamlining llms through redundancy verification and elimination of transformer blocks. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=fuX4hyLPmO.

Sun, M., Liu, Z., Bair, A., and Kolter, J. Z. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=PxoFut3dWW.

Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., and Hashimoto, T. Stanford alpaca: An instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca, 2023. Accessed on: Enter the date you accessed or used the dataset.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Wang, J., Zhang, B., Du, Q., Zhang, J., and Chu, D. A survey on data selection for llm instruction tuning. *arXiv preprint arXiv:2402.05123*, 2024.

Wang, Y., Kordi, Y., Mishra, S., Liu, A., Smith, N. A., Khashabi, D., and Hajishirzi, H. Self-instruct: Aligning language models with self-generated instructions. In *The 61st Annual Meeting Of The Association For Computational Linguistics*, 2023.

Wei, J., Bosma, M., Zhao, V., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Wu, M., Waheed, A., Zhang, C., Abdul-Mageed, M., and Aji, A. Lamini-lm: A diverse herd of distilled models from large-scale instructions. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 944–964, 2024.

Xia, M., Gao, T., Zeng, Z., and Chen, D. Sheared llama: Accelerating language model pre-training via structured pruning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=09iOdaeOzp.

Xu, C., Saranathan, G., Alam, M. P., Shah, A., Lim, J., Wong, S. Y., Martin, F., and Bhattacharya, S. Data efficient evaluation of large language models and text-to-image models via adaptive sampling. *arXiv preprint arXiv:2406.15527*, 2024a.

Xu, P., Shao, W., Chen, M., Tang, S., Zhang, K., Gao, P., An, F., Qiao, Y., and Luo, P. BESA: pruning large language models with blockwise parameter-efficient sparsity allocation. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024b. URL https://openreview.net/forum?id=gC6JTEU3jl.

Yang, A., Xiao, B., Wang, B., Zhang, B., Bian, C., Yin, C., Lv, C., Pan, D., Wang, D., Yan, D., et al. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*, 2023.

Yao, Z., Yazdani Aminabadi, R., Zhang, M., Wu, X., Li, C., and He, Y. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35: 27168–27183, 2022.

Yin, L., Wu, Y., Zhang, Z., Hsieh, C., Wang, Y., Jia, Y., Li, G., Jaiswal, A. K., Pechenizkiy, M., Liang, Y., Bendersky, M., Wang, Z., and Liu, S. Outlier weighed layerwise sparsity (OWL): A missing secret sauce for pruning llms to high sparsity. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=ahEm3l2P6w.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019.

Zhang, M., Chen, H., Shen, C., Yang, Z., Ou, L., Yu, X., and Zhuang, B. Loraprune: Structured pruning meets low-rank parameter-efficient fine-tuning. In *Findings of the Association for Computational Linguistics ACL 2024*, pp. 3013–3026, 2024.

Zhao, W., Huang, Y., Han, X., Liu, Z., Zhang, Z., Li, K., Chen, C., Yang, T., and Sun, M. Ca-lora: Adapting existing lora for compressed llms to enable efficient multi-tasking on personal devices. In *First Conference on Language Modeling*.

Zhou, C., Liu, P., Xu, P., Iyer, S., Sun, J., Mao, Y., Ma, X., Efrat, A., Yu, P., Yu, L., et al. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36, 2024.

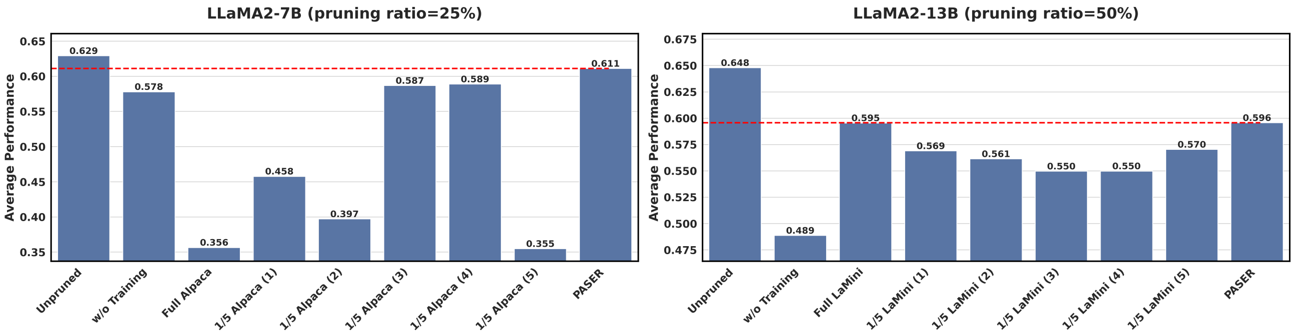

Figure 3: Average performance on seven common LLM reasoning evaluation tasks after recovery post-training with different data. The numbers in brackets represent the group index of the data subset in the full dataset. *Unpruned* indicates the original model and *w/o Training* indicates the pruned model (using LLM-Pruner (Ma et al., 2023)) without the recovery post-training.

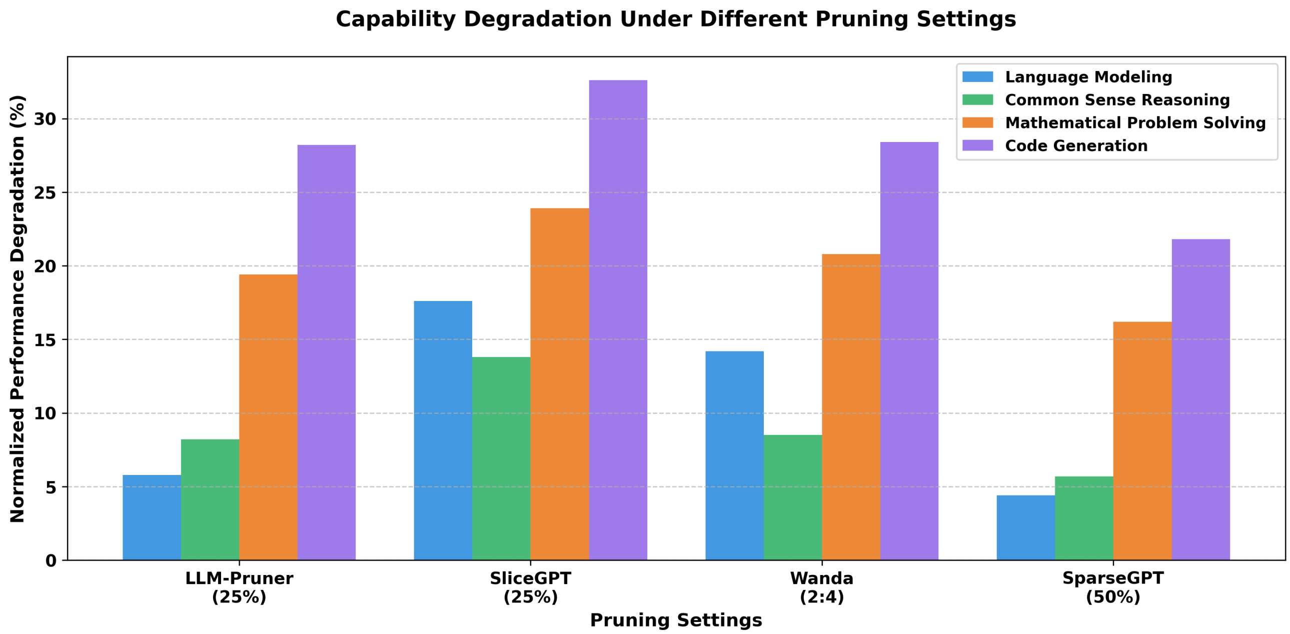

Figure 4: Normalized Performance Degradation (%) on four various capabilities under four LLM pruning settings.

A. Supporting Materials for Introduction Part

We provide the performance comparison in Figure 3 which supports the claim in the second paragraph of Section 1 about the recovery performance deterioration. From this figure, we can find that employing the full version of recovery data or uniformly split subset to conduct recovery training can hardly achieve satisfying performance.

Besides, we also provide the evidence for the uneven deterioration of different LLM capabilities during the pruning process (corresponding to the third paragraph in Section 1). From the Figure 4, we can observe that the four critical capabilities: language modeling, common sense reasoning, mathematical problem solving, and code generation exhibit significant difference on the performance degradation degrees. This phenomenon exits widely in the provided four pruning settings, which implies the necessity of performing targeted and balanced capability recovery. In fact, even among the various common sense reasoning tasks, this kind of uneven capability deterioration effect is still evident.

B. Pseudocode

We provide the full version of our pseudocode in Algorithm 1.

Algorithm 1 Post-training data Selection for efficient pruned large language model recovery (PASER)

Require: M_o: original model, M_p: pruned model, D: instruction tuning dataset, B: data budget, U: computational cost budget

```
function PASER (M_o, M_p, D, B, U)
  C ← NMFSpectralClustering({e(x_i)|(x_i, y_i) ∈ D})
  for c_k ∈ C do
    CDS(c_k) ← CapabilityDegradationScore(c_k, M_o, M_p)
  end for
  {n_k} ← AllocateSamples({CDS(c_k)}, B)
  S ← ∅, G ← InitializeCCG()
  for c_k ∈ C do
    L_k ← SortByIESDescending(c_k)
    i ← 0, count ← 0
    while count < n_k and i < |L_k| do
      (x, y) ← L_k[i]
      if IsConsistent(x, y, G) and Σ_{(x',y') ∈ S ∪ {(x,y)}} ComputationalCost(x', y') ≤ U then
        S ← S ∪ {(x, y)}
        G ← UpdateCCG(G, x, y)
        count ← count + 1
      end if
      i ← i + 1
    end while
  end for
  Return S
end function
```

C. Evaluation on Mathematical Problem Solving Tasks

To validate the effectiveness of PASER beyond common sense reasoning tasks, we conduct additional experiments on mathematical problem solving capabilities. Specifically, we employ two widely-adopted mathematical problem solving benchmarks:

- **GSM8K** (Cobbe et al., 2021): A dataset containing 8.5K high-quality grade school math word problems that test various mathematical problem solving capabilities, including arithmetic, algebra, and word problem solving.
- **Minerva Math** (Lewkowycz et al., 2022): A comprehensive mathematical evaluation dataset covering diverse topics in mathematics ranging from arithmetic to calculus, with problems requiring multi-step reasoning.

Table 5: Recovery performance of different instruction tuning data selection methods on mathematical problem solving tasks under various pruning schemes. The 'bold' represents the best performance under the same pruning scheme.

Recovery Method	LLM-Pruner (25%)		SliceGPT (25%)		Wanda (2:4)		SparseGPT (50%)	
	GSM8K	Minerva	GSM8K	Minerva	GSM8K	Minerva	GSM8K	Minerva
w/o Training	44.3	17.8	42.5	16.9	43.8	17.4	43.1	17.2
Full Data	46.5	19.1	44.8	18.3	45.9	18.7	45.2	18.5
Random	45.8	18.4	43.9	17.8	44.7	18.1	44.3	17.9
Instruction Mining	46.2	18.9	44.5	18.1	45.4	18.5	44.9	18.3
IFD	46.8	19.3	45.1	18.5	45.8	18.8	45.4	18.6
Nuggets	47.1	19.5	45.4	18.7	46.2	19.0	45.7	18.8
PASER	**49.4**	**21.2**	**47.8**	**20.5**	**48.5**	**20.8**	**47.2**	**20.1**

The recovery performance under different pruning schemes is presented in Table 5. From these results, we can observe that PASER consistently outperforms baseline methods across all pruning schemes on both mathematical problem solving benchmarks. The improvements are particularly significant under the LLM-Pruner scheme, where PASER achieves 5.1% and 3.4% absolute improvements over w/o Training on GSM8K and Minerva Math, respectively. While different pruning schemes affect the base performance levels, PASER maintains its effectiveness in recovery. For example, under the more aggressive SparseGPT (50%) setting, PASER still achieves 4.1% and 2.9% improvements on GSM8K and Minerva Math over w/o Training. Compared to Full Data training, PASER achieves better performance while using only 20% of the instruction data, demonstrating its efficiency in recovering mathematical problem solving capabilities.

These results, combined with the common sense reasoning results presented in the main paper, demonstrate that PASER is effective across diverse tasks. The strong performance on mathematical tasks is particularly noteworthy given that these problems often require precise, step-by-step reasoning and have less tolerance for errors compared to common sense reasoning tasks. This validates the effectiveness of our capability degradation score in identifying and prioritizing recovery for severely affected capabilities, even in domains requiring high precision.

Table 6: Recovery performance of different instruction tuning data selection methods on code generation tasks under various pruning schemes. The 'bold' represents the best performance under the same pruning scheme. 'P@k' indicates 'Pass@k'.

Recovery Method	LLM-Pruner (25%)						SliceGPT (25%)					
	HumanEval			MBPP			HumanEval			MBPP		
	P@1	P@10	P@100	P@1	P@10	P@100	P@1	P@10	P@100	P@1	P@10	P@100
w/o Training	3.4	6.2	10.4	7.1	15.5	24.6	3.0	5.9	8.7	6.2	11.8	21.5
Full Data	7.8	15.1	19.0	15.2	26.3	39.5	2.9	5.1	11.8	9.3	19.7	36.4
Random	6.2	13.7	20.5	12.8	23.8	35.0	3.1	5.8	14.2	8.5	17.2	38.6
Instruction Mining	8.9	17.8	28.4	15.7	29.2	43.1	6.3	11.4	23.8	12.8	24.5	45.2
IFD	10.5	21.2	35.6	18.2	34.5	50.7	8.7	16.8	31.2	16.4	30.8	52.4
Nuggets	11.8	22.9	38.3	18.9	35.8	52.4	9.5	18.5	33.9	17.6	32.6	51.9
PASER	**14.4**	**27.6**	**48.2**	**23.1**	**42.6**	**62.0**	**12.9**	**25.2**	**44.5**	**22.3**	**41.0**	**63.7**

D. Evaluation on Code Generation Tasks

To further explore the PASER's effectiveness on recovering code generation capability, we take two structured pruning schemes (LLM-Pruner, SliceGPT) and perform exhaustive evaluations on two major code generation benchmarks:

- **HumanEval** (Chen et al., 2021): A widely-used code generation benchmark consisting of 164 hand-crafted Python programming problems that test various programming concepts. Each problem contains a function signature with docstring and test cases, requiring models to complete the implementation. The benchmark evaluates functional correctness using the Pass@k metric, which measures the percentage of problems where a correct solution appears in k samples.
- **MBPP** (Austin et al., 2021): A programming benchmark containing 974 Python programming problems focused on basic programming tasks. Each problem includes a natural language description, test cases, and a reference solution, making it particularly suitable for evaluating language-to-code generation capabilities. MBPP uses the same Pass@k evaluation metric as HumanEval but generally features simpler problems with a broader coverage of basic programming concepts.

In our experiments, models are evaluated in zero-shot on HumanEval and 3-shot on MBPP. The results under Pass@k ($k = 1, 10, 100$) metrics are present in Table 6. As shown in the table, code generation capability experiences severe degradation after pruning. The Pass@1 performance on HumanEval drops to merely 3.4% under LLM-Pruner without recovery training. This dramatic decline indicates that code generation, as a complex reasoning task, is particularly vulnerable during model compression. Through capability degradation-aware budget allocation and targeted sample selection, PASER demonstrates remarkable effectiveness in recovering this severely impacted capability. Under LLM-Pruner, it achieves 14.4% Pass@1 on HumanEval, not only substantially outperforming other recovery methods but also surpassing the full data training baseline. The improvement becomes even more pronounced at higher sampling rates, i.e., PASER reaches 48.2% Pass@100 compared to Random's 20.5% and Instruction Mining's 28.4%. This significant performance gap validates our approach of prioritizing recovery resources for severely degraded capabilities and selecting the most relevant instruction samples for recovery training. The superiority of PASER remains consistent across different evaluation settings. On MBPP, which features simpler programming tasks, PASER still maintains a clear advantage over baseline methods, achieving 23.1%

Pass@1 and 62.0% Pass@100 under LLM-Pruner. When tested with a different pruning scheme (SliceGPT), which causes even more severe initial degradation (3.0% Pass@1 on HumanEval), PASER successfully recovers the performance to 12.9% Pass@1 and 44.5% Pass@100.

These results comprehensively demonstrate that our capability-aware recovery strategy effectively addresses the disproportionate impact of model compression on complex reasoning abilities, enabling targeted and efficient recovery of critical model capabilities.

Table 7: Recovery performance of different instruction tuning data selection methods under various LLM quantization schemes on LLaMA2-7B model. The 'bold' represents the best performance under the same quantization scheme. Here, the Alpaca is taken as the original dataset.

Quantization	Recovery Post-training	WikiText2↓	PTB↓	BoolQ	PIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Average
w/o Quant	w/o training	12.62	22.14	71.13	78.40	72.79	67.17	69.36	40.70	40.80	62.91
RTN 4 bits	w/o training	18.14	33.28	66.52	74.95	69.24	63.91	65.58	38.07	35.10	59.05
	Full Data	15.83	27.41	67.35	75.70	69.94	64.57	66.22	38.48	35.90	59.74
	Random	16.72	29.56	64.53	73.08	67.48	62.28	63.93	37.13	33.90	57.48
	Instruction Mining	16.05	27.83	66.73	75.15	69.43	64.08	65.74	38.18	35.30	59.23
	IFD	15.21	25.74	68.16	76.40	70.60	65.18	66.83	38.83	37.40	60.49
	Nuggets	14.68	24.53	68.99	77.13	71.28	65.82	67.46	39.21	38.70	61.23
	PASER	**14.21**	**23.37**	**70.43**	**78.41**	**72.47**	**66.92**	**68.54**	**39.81**	**41.50**	**62.58**
GPTQ 4 bits	w/o training	15.96	27.86	67.82	76.15	70.35	64.95	66.59	38.69	36.90	60.21
	Full Data	15.62	26.95	68.00	76.31	70.50	65.09	66.73	38.78	37.40	60.40
	Random	16.31	28.74	66.81	75.24	69.49	64.14	65.79	38.22	35.70	59.34
	Instruction Mining	15.37	26.42	68.31	76.58	70.75	65.33	66.96	38.93	37.90	60.68
	IFD	14.83	25.16	68.96	77.15	71.29	65.83	67.47	**40.21**	39.00	61.42
	Nuggets	13.52	22.93	69.74	77.83	71.93	66.43	68.06	39.56	40.20	61.96
	PASER	**12.95**	**21.84**	**71.20**	**79.12**	**73.12**	**67.53**	**69.14**	40.18	**42.90**	**63.31**

Table 8: Recovery performance of different instruction tuning data selection methods under various LLM quantization schemes on LLaMA2-13B model. The 'bold' represents the best performance under the same quantization scheme. Here, the Alpaca is taken as the original dataset.

Quantization	Recovery Post-training	WikiText2↓	PTB↓	BoolQ	PIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Average
w/o Quant	w/o training	11.58	20.24	69.02	78.73	76.60	69.69	73.23	44.20	42.00	64.78
RTN 4 bits	w/o training	17.53	32.34	63.15	74.59	72.62	65.94	69.17	41.49	37.00	60.57
	Full Data	16.95	31.02	63.59	75.02	73.04	66.33	69.58	41.75	37.50	60.97
	Random	17.86	33.15	62.00	73.48	71.55	64.94	68.13	40.84	35.20	59.45
	Instruction Mining	17.24	31.68	62.83	74.27	72.32	65.65	68.87	41.29	36.10	60.19
	IFD	15.63	28.39	65.03	76.37	74.34	67.49	70.80	42.46	39.60	62.30
	Nuggets	15.08	27.15	65.45	76.76	**76.72**	67.84	71.17	42.70	40.20	62.98
	PASER	**12.34**	**23.08**	**67.33**	**78.50**	76.41	**69.37**	**72.78**	**43.67**	**41.70**	**64.25**
GPTQ 4 bits	w/o training	14.74	26.86	64.68	76.04	74.02	67.20	70.49	42.28	39.10	61.97
	Full Data	16.02	29.34	63.62	75.05	73.07	66.35	69.61	41.76	37.40	60.98
	Random	14.58	26.52	64.82	76.17	74.15	67.32	70.61	42.36	39.30	62.10
	Instruction Mining	13.67	24.59	66.37	77.58	75.51	68.56	71.91	**44.15**	41.30	63.63
	IFD	13.51	24.26	**68.46**	77.66	75.59	68.63	71.99	43.20	41.40	63.85
	Nuggets	12.76	22.92	67.21	78.34	76.25	69.24	72.63	43.59	42.70	64.28
	PASER	**11.25**	**20.93**	68.11	**79.16**	**77.05**	**69.97**	**73.39**	44.04	**44.20**	**65.13**

E. Extended Experiments on Post-quantization Recovery Training

Though the method descriptions and the experiments in the main body are mainly around the LLM pruning scenario, our PASER framework can actually be extended seamlessly to other LLM compression scenario. To further demonstrate its applicability, we conduct the experiments on post-quantization recovery training and compare our PASER with corresponding instruction tuning data selection baselines. In detail, we choose two most representative methods: Round-To-Nearest (RTN) (Frantar & Alistarh, 2022; Yao et al., 2022), GPTQ (Frantar et al., 2023), to perform the LLM quantization. It should be clarified that RTN method, which rounds all weights to the nearest quantized value on exactly the same asymmetric

Table 9: Knowledge distillation recovery performance of different instruction tuning data selection methods under various pruning schemes on LLaMA2-7B model. The 'bold' represents the best performance under the same pruning scheme. Here, the Alpaca is taken as the original dataset.

Pruning	Recovery Post-training	WikiText2↓	PTB↓	BoolQ	PIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Average
w/o pruning	w/o Training	12.62	22.14	71.13	78.40	72.79	67.17	69.36	40.70	40.80	62.91
LLM-Pruner ratio=25%	w/o Training	20.34	38.81	61.87	76.61	65.86	60.22	63.13	37.37	39.40	57.78
	Full Data	24.72	43.91	63.30	76.01	67.18	62.27	64.23	36.86	39.20	58.44
	Random	23.82	41.20	**68.03**	74.89	66.27	64.51	64.65	32.58	38.30	58.46
	Instruction Mining	22.65	39.40	62.17	75.98	66.74	61.29	63.01	38.32	39.60	58.16
	IFD	19.17	32.30	64.13	77.55	67.89	61.56	64.09	38.19	**40.40**	59.12
	Nuggets	18.64	32.19	64.46	76.66	67.26	64.88	66.50	36.52	39.20	59.35
	PASER	**15.91**	**25.39**	67.89	**77.81**	**69.62**	**67.63**	**68.46**	**39.87**	40.20	**61.64**
SliceGPT ratio=25%	w/o training	44.53	80.07	65.54	66.87	54.16	63.38	58.46	34.56	36.90	54.27
	Full Data	35.48	66.25	69.35	70.34	58.50	66.76	62.95	37.14	38.70	57.68
	Random	38.63	65.67	67.19	68.59	56.21	64.94	60.63	35.61	37.80	55.85
	Instruction Mining	35.56	62.14	68.41	69.51	57.08	66.33	62.51	36.59	38.00	56.92
	IFD	33.50	61.33	69.51	70.82	58.70	67.49	64.09	37.22	38.50	58.05
	Nuggets	21.39	32.83	70.17	71.49	59.11	67.94	**72.51**	37.54	38.70	59.64
	PASER	**11.87**	**20.91**	**73.43**	**80.32**	**74.46**	**69.76**	71.95	**42.26**	**41.70**	**64.84**
Wanda sparsity=2:4	w/o training	42.10	76.85	69.30	71.99	53.06	62.75	60.94	28.07	34.60	54.39
	Full Data	25.92	47.85	71.09	75.14	64.10	65.62	65.64	34.38	37.50	59.07
	Random	34.98	63.47	70.18	73.62	59.15	63.83	63.70	32.13	36.50	57.02
	Instruction Mining	30.56	55.56	71.03	73.97	61.69	64.56	64.86	33.93	37.00	58.15
	IFD	24.08	41.44	71.78	75.89	64.83	65.72	**68.89**	35.97	38.00	60.15
	Nuggets	23.14	40.10	**72.26**	76.50	65.33	66.03	66.52	37.27	38.60	60.36
	PASER	**13.84**	**23.54**	71.25	**78.15**	**72.06**	**66.64**	68.66	**39.38**	**40.50**	**62.38**
SparseGPT sparsity=50%	w/o training	19.26	36.41	71.22	75.60	62.85	66.06	69.11	36.86	37.80	59.93
	Full Data	28.17	52.82	68.52	75.77	57.84	69.26	60.43	37.72	37.00	58.08
	Random	25.31	43.22	69.74	74.91	60.28	68.10	64.06	**39.95**	**39.80**	59.55
	Instruction Mining	21.56	39.61	71.12	74.85	62.53	66.06	68.07	36.85	37.80	59.61
	IFD	17.76	31.25	71.70	75.76	63.43	66.06	69.14	36.59	37.60	60.04
	Nuggets	14.83	25.38	72.18	75.95	63.91	66.29	69.75	36.86	37.70	60.38
	PASER	**13.00**	**22.24**	**75.07**	**78.66**	**66.90**	**69.31**	**72.85**	38.89	39.60	**63.04**

per-row grid, is actually the fundamental technique in most works about LLM quantization (Frantar & Alistarh, 2022; Yao et al., 2022; Park et al.). Its runtimes scales well to the models with many billion parameters due to the direct rounding. According to the results provided in Table 7 and 8, we can observe that the PASER can still effectively enhance the recovery performance and outperform the data selection baselines on averaged reasoning performance and zero-shot perplexity for both LLaMA2-7B and LLaMA2-13B models. Meanwhile, recovery data selection baselines can indeed achieve the stronger performance than full data and random baselines, which validates the necessity of conducting recovery data selection even in the LLM quantization scenario. Furthermore, comparing these results with Table 1 and 2, it can be noticed that the improvement space of PASER in Table 7 and 8 has been reduced to some extent. This is because the post-compression performance of such quantization schemes has been competitive enough, which can reflected from the w/o training row.

F. Extended Experiments on Recovery Training with Knowledge Distillation

Inspired by (Muralidharan et al., 2024), we explore the knowledge distillation as the recovery post-training paradigm instead of the standard supervised learning with the groundtruth label. Here, we set the original model M_o as the teacher and the pruned model M_p as the student. The mean KL divergence (Kullback & Leibler, 1951) between the output probability distribution of M_o and that of M_p is taken as the objective function. Comparing the corresponding results under different pruning schemes in Table 9 with that in Table 1, we can first observe that knowledge distillation can effectively improve the recovery performance on both reasoning and language modeling tasks in most cases. In particular, the reasoning performance of PASER is improved by 0.348 points on average among such four pruning schemes. Interestingly, the knowledge distillation recovery performance of Full Data under LLM-Pruner is much better than that with standard label-supervised learning. This demonstrates that knowledge distillation is also a promising approach to avoid the misleading information from the irrelevant or conflicting samples existing in the original dataset. Because its learning process directly imitates the unpruned model behavior instead of the provided labels, thus better preserving the thinking and decision-making consistency with the original model. As a summary, distilling the knowledge of unpruned model into the pruned model can be regarded as an effective way to enhance the recovery performance, though bring more memory overhead. Furthermore, stronger layer-wise distillation can also be taken into consideration (Jiao et al., 2020).

Exploration on Combined Training Strategies Given the complementary potential of knowledge distillation (KD) and supervised fine-tuning (SF), we further explore whether combining these two approaches could lead to enhanced recovery performance. Specifically, we investigate two cascading strategies: (1) first applying KD followed by SF, and (2) first conducting SF followed by KD. Table 10 presents the results under different pruning schemes.

Table 10: Recovery performance comparison between different combinations of knowledge distillation (KD) and supervised fine-tuning (SF) under various pruning schemes. The 'bold' represents the best performance under the same pruning scheme.

Recovery Training	WikiText2↓	PTB↓	BoolQ	PIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Average
LLM-Pruner (ratio=25%)										
KD	15.91	25.39	67.89	77.81	69.62	67.63	68.46	39.87	40.20	**61.64**
SF	16.40	26.35	67.25	77.29	68.98	66.97	67.84	39.54	39.80	61.10
First KD, then SF	16.15	25.87	67.57	77.55	69.31	67.30	68.15	39.71	40.00	61.37
First SF, then KD	16.28	26.02	67.41	77.43	69.15	67.11	67.96	39.63	39.90	61.23
SliceGPT (ratio=25%)										
KD	11.87	20.91	73.43	80.32	74.46	69.76	71.95	42.26	41.70	**64.84**
SF	12.24	21.53	72.75	79.84	73.92	69.18	71.37	41.82	41.30	64.31
First KD, then SF	12.06	21.24	73.12	80.05	74.18	69.45	71.62	42.03	41.50	64.56
First SF, then KD	12.15	21.38	72.94	79.95	74.05	69.32	71.51	41.95	41.40	64.45
Wanda (sparsity=2:4)										
KD	13.84	23.54	71.25	78.15	72.06	66.64	68.66	39.38	40.50	**62.38**
SF	14.13	27.22	70.77	77.87	71.78	66.26	68.30	39.04	40.10	62.02
First KD, then SF	13.97	25.31	71.02	78.03	71.94	66.47	68.49	39.23	40.30	62.21
First SF, then KD	14.05	26.28	70.89	77.95	71.85	66.35	68.41	39.15	40.20	62.11
SparseGPT (sparsity=50%)										
KD	13.00	22.24	75.07	78.66	66.90	69.31	72.85	38.89	39.60	**63.04**
SF	13.33	23.77	74.79	78.38	66.62	69.03	72.57	38.70	39.40	62.78
First KD, then SF	13.15	22.96	74.94	78.53	66.78	69.18	72.72	38.81	39.50	62.92
First SF, then KD	13.24	23.35	74.85	78.45	66.70	69.11	72.64	38.75	39.45	62.85

Interestingly, the results show that neither cascading strategy consistently outperforms individual KD or SF approaches. This suggests that these two training paradigms might actually serve similar functions in recovering model capabilities, making their combination redundant rather than complementary. Knowledge distillation shows slightly better performance across all pruning schemes, which could be attributed to its ability to capture the nuanced knowledge encoded in the teacher model's full output distribution. However, the marginal gains from combining approaches do not justify the additional computational overhead required for cascaded training.

G. Detailed Ablation Study Results

In this section, we present comprehensive ablation results of the three key components in PASER: semantic-structural recovery instruction clustering (S^2RIC), capability degradation-aware instruction selection (CDAIS), and negative transfer mitigation (NTM). Table 11 shows the detailed performance across different evaluation metrics.

The detailed results reveal the distinct contributions of each component under different pruning schemes. For structured pruning like LLM-Pruner, removing S^2RIC leads to significant degradation in both language modeling (perplexity increases from 16.40 to 18.73 on WikiText2) and reasoning tasks (average score drops by 1.43 points), highlighting its importance in addressing uneven capability degradation. The impact of CDAIS is particularly evident under SliceGPT, where its removal causes a 1.63-point drop in average reasoning performance while maintaining relatively stable language modeling metrics, demonstrating its effectiveness in balancing recovery priorities. Under semi-structured pruning (Wanda), all three components show more balanced contributions, with performance drops ranging from 0.68 to 0.83 points when each is removed. This suggests that semi-structured pruning requires a more holistic recovery approach. For unstructured pruning (SparseGPT) where capability degradation tends to be more uniform, NTM plays a particularly crucial role - its removal leads to the largest drop in language modeling performance (perplexity increases from 13.33 to 15.91 on WikiText2) and affects complex reasoning tasks like WinoGrande and ARC-e significantly. Notably, the full PASER framework consistently achieves the best performance across almost all metrics under various pruning schemes, with only occasional exceptions in individual tasks (e.g., OBQA in LLM-Pruner and PIQA in SparseGPT). This comprehensive superiority validates our design choice of combining these three components for effective pruned model recovery.

Table 11: The detailed ablation study for our proposed three components under various pruning schemes on LLaMA2-7B model. The ‘bold’ represents the best performance under the same pruning scheme. Here, the Alpaca is taken as the original dataset.

Pruning	Recovery Post-training	WikiText2↓	PTB↓	BoolQ	PIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Average
LLM-Pruner ratio=25%	PASER w/o S^2RIC	18.73	32.84	65.31	76.84	67.59	64.85	65.92	37.96	39.20	59.67
	PASER w/o CDAIS	17.56	30.15	66.27	77.03	68.15	65.73	66.58	38.54	39.50	60.26
	PASER w/o NTM	19.82	35.60	64.83	**77.52**	67.34	64.48	63.59	36.78	**40.20**	59.25
	PASER	**16.40**	**26.35**	**67.25**	77.29	**68.98**	**66.97**	**67.84**	**39.54**	39.80	**61.10**
SliceGPT ratio=25%	PASER w/o S^2RIC	14.83	25.42	71.15	78.91	72.25	67.84	69.95	40.82	40.30	63.03
	PASER w/o CDAIS	14.16	24.92	70.89	78.56	71.84	67.45	69.58	40.47	40.00	62.68
	PASER w/o NTM	15.37	27.81	69.97	77.33	70.68	65.92	68.03	39.39	**42.10**	61.92
	PASER	**12.24**	**21.53**	**72.75**	**79.84**	**73.92**	**69.18**	**71.37**	**41.82**	41.30	**64.31**
Wanda sparsity=2:4	PASER w/o S^2RIC	15.84	30.25	69.26	77.42	70.31	65.82	67.84	38.67	39.00	61.19
	PASER w/o CDAIS	15.46	29.48	69.14	77.35	70.27	65.74	67.79	38.75	39.60	61.23
	PASER w/o NTM	16.79	31.52	69.51	76.92	70.76	65.23	67.28	38.47	**41.20**	61.34
	PASER	**14.13**	**27.22**	**70.77**	**77.87**	**71.78**	**66.26**	**68.30**	**39.04**	40.10	**62.02**
SparseGPT sparsity=50%	PASER w/o S^2RIC	14.89	26.31	73.25	77.45	**70.15**	68.47	69.28	39.82	39.80	62.60
	PASER w/o CDAIS	14.62	25.84	72.91	77.50	69.93	68.12	69.05	**39.94**	40.00	62.49
	PASER w/o NTM	15.91	28.19	71.53	**78.62**	65.48	67.21	69.79	39.18	**40.50**	61.76
	PASER	**13.33**	**23.77**	**74.79**	78.38	66.62	**69.03**	**72.57**	38.70	39.40	**62.78**

H. Exploration on Other Possible Clustering Methods

To discuss the impact of different instruction tuning data clustering approaches, we replace our Semantic-structural Recovery Instruction Clustering (S^2RIC) module with some other common text clustering method: NMF_TFIDF, LDA_TFIDF, KMeans_TFIDF, Spectral_MTEB, Spectral_BERT (Xu et al., 2024a). The reasoning performance comparison among different PASER versions with such clustering methods is provided in Table 12. From the table, we can find that integrating other instruction clustering methods with PASER can bring the performance decline to some extent among all four pruning schemes. Especially, the clustering method with traditional statistics-based text representation technique, TFIDF, generally behaves worse than semantic embedding-based text representation techniques like BERT. Therefore, we can conclude that our semantic-structural recovery instruction clustering is at least a competitive approach as the clustering component of PASER. Though, comparing these results with those in Table 1, we can observe the advantages of PASER over other general instruction tuning data selection methods can still be stably maintained. This further demonstrates that the potential of the clustering-based data selection for effective and balanced LLM capability recovery.

I. Case Study for Recovery Instruction Clustering

To illustrate the effectiveness of our Semantic-Structural Recovery Instruction Clustering (S^2RIC) approach for grouping samples focusing on similar capabilities together, we conduct a case study of clustered instruction samples from the Alpaca dataset. Specifically, we provide representative samples from several obtained clusters as follows.

I.1. Cluster 1: Basic Factual Knowledge and Information Retrieval

- **Instruction**: “Find the five largest cities in France.”
- **Instruction**: “What is the capital of France?”
- **Instruction**: “Find the population density of United States.”

These instructions primarily test the model’s ability to recall basic facts and information, corresponding to general knowledge capabilities.

I.2. Cluster 2: Language Understanding and Translation

- **Instruction**: “Translate the word ’giraffe’ to French.”

Table 12: Recovery performance of multiple PASER versions integrated with different data clustering approaches under various pruning schemes on LLaMA2-7B model. The PASER(S^2RIC) is the version we employ in the main body. The ‘bold’ represents the best performance under the same pruning scheme. Here, the Alpaca is taken as the original dataset.

Pruning	Recovery Post-training	WikiText2↓	PTB↓	BoolQ	PIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Average
w/o pruning	w/o Training	12.62	22.14	71.13	78.40	72.79	67.17	69.36	40.70	40.80	62.91
LLM-Pruner ratio=25%	w/o Training	20.34	38.81	61.87	76.61	65.86	60.22	63.13	37.37	39.40	57.78
	PASER(NMF_TFIDF)	17.82	29.45	65.93	76.88	67.42	65.19	66.37	38.81	39.60	60.03
	PASER(LDA_TFIDF)	17.56	28.91	66.18	77.02	67.76	65.58	66.92	38.95	39.70	60.30
	PASER(KMeans_TFIDF)	17.21	28.13	66.47	77.15	68.04	65.92	67.23	39.12	**39.80**	60.53
	PASER(Spectral_MTEB)	16.82	27.24	66.89	77.23	68.46	66.38	67.56	39.31	**39.80**	60.80
	PASER(Spectral_BERT)	16.61	26.79	67.06	77.26	68.72	66.68	67.71	39.43	**39.80**	60.95
	PASER(S^2RIC)	**16.40**	**26.35**	**67.25**	**77.29**	**68.98**	**66.97**	**67.84**	**39.54**	**39.80**	**61.10**
SliceGPT ratio=25%	w/o Training	44.53	80.07	65.54	66.87	54.16	63.38	58.46	34.56	36.90	54.27
	PASER(NMF_TFIDF)	14.27	24.36	70.89	78.76	72.13	67.69	70.12	**41.95**	40.80	63.21
	PASER(LDA_TFIDF)	14.86	25.19	70.31	78.42	71.64	67.25	69.58	40.37	40.60	62.60
	PASER(KMeans_TFIDF)	13.58	23.42	71.46	79.07	72.61	68.14	70.48	41.08	41.00	63.41
	PASER(Spectral_MTEB)	12.91	22.47	72.08	79.41	73.18	68.62	70.87	41.43	41.10	63.81
	PASER(Spectral_BERT)	12.58	22.01	72.41	79.63	73.55	68.91	71.12	41.63	41.20	64.06
	PASER	**12.24**	**21.53**	**72.75**	**79.84**	**73.92**	**69.18**	**71.37**	41.82	**41.30**	**64.31**
Wanda sparsity=2:4	w/o Training	42.10	76.85	69.30	71.99	53.06	62.75	60.94	28.07	34.60	54.39
	PASER(NMF_TFIDF)	16.18	30.94	70.09	76.68	69.98	64.82	66.92	38.14	39.60	60.89
	PASER(LDA_TFIDF)	18.74	34.98	69.85	76.31	69.42	64.37	66.48	37.82	39.40	60.52
	PASER(KMeans_TFIDF)	15.49	29.76	**70.92**	77.03	70.51	65.28	67.38	38.47	**40.30**	61.41
	PASER(Spectral_MTEB)	14.81	28.49	70.54	77.42	71.12	65.75	67.82	38.74	39.90	61.61
	PASER(Spectral_BERT)	14.47	27.86	70.66	77.65	71.45	66.01	68.06	38.89	40.00	61.82
	PASER	**14.13**	**27.22**	70.77	**77.87**	**71.78**	**66.26**	**68.30**	**39.04**	40.10	**62.02**
SparseGPT sparsity=50%	w/o Training	19.26	36.41	71.22	75.60	62.85	66.06	69.11	36.86	37.80	59.93
	PASER(NMF_TFIDF)	15.97	28.13	72.63	76.94	64.37	67.18	70.39	37.54	38.60	61.09
	PASER(LDA_TFIDF)	15.41	27.09	73.12	77.31	64.93	67.63	70.92	37.86	38.80	61.51
	PASER(KMeans_TFIDF)	14.72	25.91	73.61	77.66	65.46	68.09	71.48	38.19	39.00	61.93
	PASER(Spectral_MTEB)	14.03	24.84	74.16	78.01	66.02	68.54	71.98	38.44	39.20	62.34
	PASER(Spectral_BERT)	13.68	24.31	74.48	78.21	66.32	68.79	72.28	**38.75**	39.30	62.59
	PASER	**13.33**	**23.77**	**74.79**	**78.38**	**66.62**	**69.03**	**72.57**	38.70	**39.40**	**62.78**

- **Instruction**: “Pick the correct Spanish translation of “Hello”.”
- **Instruction**: “Difference in meaning between "done freely" and "freely done¨? For instance, is there any difference in meaning between these two sentences?”’

This cluster focuses on language-related tasks, including translation, idiomatic expressions, and grammatical analysis.

I.3. Cluster 3: Logical Reasoning and Problem Solving

- **Instruction**: “A friend shares the following text with you and asks for your opinion: ’Purple-eyed individuals have a stronger psychic connection to the cosmos and have more chances to predict the future.’ Analyze the statements and point out logical fallacies or unsupported claims.”
- **Instruction**: “Explain how to solve a Sudoku puzzle in three steps.”
- **Instruction**: “Answer this math question: What is the value of 3 to the power of 5?”

These instructions test the model’s ability to perform mathematical calculations, logical deductions, and pattern recognition.

I.4. Cluster 4: Creative Writing and Text Generation

- **Instruction**: “Write a microblog post about a recent experience you had.”
- **Instruction**: “Compose a haiku about the wonders of technology.”
- **Instruction**: “Create an illustration of the inside of a castle.”

This cluster groups tasks that require creative text generation, showcasing the model's ability to produce original content across various formats and topics.

I.5. Cluster 5: Summarization and Information Extraction

- **Instruction**: "Summarize the techniques used for summarizing text."
- **Instruction**: "Extract the main argument from the passage."
- **Instruction**: "Provide a brief summary of the article "A Brief History of the Automobile Industry."

These instructions focus on the model's capability to condense information and identify key points from longer texts.

This study demonstrates that S^2RIC effectively groups instructions targeting similar LLM capabilities, enabling PASER to allocate recovery efforts, i.e., data budget, more strategically. By focusing intensively on clusters where capabilities have degraded most after the pruning, while maintaining awareness of all capability clusters, PASER achieves both targeted and balanced recovery. This dual approach ensures efficient and comprehensive restoration of the model's diverse functionalities, optimizing the recovery process for pruned LLMs.

J. Case Study for Negative Transfer Mitigation

To illustrate the negative transfer mitigation process in PASER, we present the following case study, including both initial concept consistency graph construction and new sample evaluation.

J.1. Constructing Initial Concept Consistency Graph

We begin with three accepted instruction-output pairs to construct our initial Concept Consistency Graph (CCG):

1. **Instruction:** "Explain the basic principles of quantum computing."

   Output: "Quantum computing is based on the principles of quantum mechanics. It uses quantum bits or qubits, which can exist in multiple states simultaneously due to superposition. Quantum entanglement allows qubits to be correlated with each other. These properties enable quantum computers to perform certain calculations much faster than classical computers for specific problems."

   Concepts: {quantum computing, qubit, superposition, entanglement}

2. **Instruction:** "Describe the main components of a classical computer."

   Output: "A classical computer typically consists of several key components: the Central Processing Unit (CPU) which performs calculations and controls other components; Random Access Memory (RAM) for temporary data storage; a hard drive for long-term data storage; and various input/output devices. These components work together to process and store information using binary logic."

   Concepts: {CPU, RAM, hard drive, binary logic}

3. **Instruction:** "What are the fundamental concepts in deep learning?"

   Output: "Deep learning is a subset of machine learning based on artificial neural networks. Key concepts include: neural networks with multiple layers (deep neural networks), which can learn hierarchical representations of data; backpropagation, an algorithm for training these networks by adjusting weights; and various optimization techniques to improve learning. Deep learning has shown remarkable success in tasks such as image and speech recognition."

   Concepts: {deep learning, neural network, backpropagation, optimization}

Based on these samples, we construct an initial Concept Consistency Graph (CCG):

$$
\begin{aligned}
V = \{&\text{quantum computing, qubit, superposition, entanglement,}\\
&\text{CPU, RAM, hard drive, binary logic,}\\
&\text{deep learning, neural network, backpropagation, optimization}\}
\end{aligned}
$$

The edges E in the graph represent the co-occurrence of concepts within each instruction-output pair.

$$\begin{aligned} E = \{&(\text{quantum computing, qubit}), (\text{quantum computing, superposition}),\\ &(\text{quantum computing, entanglement}), (\text{qubit, superposition}),\\ &(\text{qubit, entanglement}), (\text{superposition, entanglement}),\\ &(\text{CPU, RAM}), (\text{CPU, hard drive}), (\text{CPU, binary logic}),\\ &(\text{RAM, hard drive}), (\text{RAM, binary logic}), (\text{hard drive, binary logic}),\\ &(\text{deep learning, neural network}), (\text{deep learning, backpropagation}),\\ &(\text{deep learning, optimization}), (\text{neural network, backpropagation}),\\ &(\text{neural network, optimization}), (\text{backpropagation, optimization})\} \end{aligned}$$

This explicit definition of E clearly shows the relationships between concepts within each domain, while highlighting the absence of edges between concepts from different domains.

J.2. Evaluating New Samples

We now evaluate two new samples using PASER's negative transfer mitigation process.

J.2.1. Sample 1: Accepted

Instruction: "How does increasing the number of qubits affect a quantum computer's performance?"

Output: "Increasing the number of qubits in a quantum computer can significantly enhance its performance and capabilities:

1. Exponential growth in computational power: Each additional qubit doubles the number of quantum states that can be represented, leading to an exponential increase in computational power.

2. Handling more complex problems: More qubits allow quantum computers to tackle larger and more complex problems that are intractable for classical computers.

3. Improved error correction: A larger number of qubits can be used to implement more robust quantum error correction codes, improving the overall stability and reliability of quantum computations.

4. Enhanced quantum parallelism: More qubits enable greater parallelism in quantum algorithms, potentially leading to faster solution times for certain problems.

5. Challenges in scalability: However, increasing the number of qubits also introduces challenges in maintaining coherence and reducing error rates, which are active areas of research in quantum computing.

In summary, while increasing the number of qubits can dramatically improve a quantum computer's capabilities, it also comes with technical challenges that researchers are working to overcome."

Concepts: {qubit, quantum computer, computational power, quantum states, error correction, quantum parallelism, coherence}

Applying PASER's negative transfer mitigation process:

1. **Concept Extraction:** $C(x, y) =$ {qubit, quantum computer, computational power, quantum states, error correction, quantum parallelism, coherence}
2. **Consistency Check:** For each pair $(v_i, v_j) \in C(x, y)$:
   - (qubit, quantum computer): both $\in V$ and $(v_i, v_j) \in E$
   - (qubit, computational power): qubit $\in V$, computational power $\notin V$
   - (qubit, quantum states): qubit $\in V$, quantum states $\notin V$
   - Other pairs: at least one concept $\notin V$
3. **Decision:** IsConsistent(x, y) = True, because for all pairs, either $(v_i, v_j) \in E$ or $\{v_i, v_j\} \not\subset V$

PASER accepts this sample as it is consistent with the existing CCG and introduces relevant new concepts without conflicts.

J.2.2. Sample 2: Rejected

Instruction: "Explain how quantum computing can be used to significantly speed up deep learning algorithms."

Output: "Quantum computing has the potential to speed up certain aspects of deep learning algorithms. Quantum algorithms could potentially optimize the training process of neural networks, especially in handling large-dimensional data. However, it's important to note that the exact advantages and applications are still being researched, and classical computers remain more practical for most current deep learning tasks."

Concepts: {quantum computing, deep learning, neural network, speedup}

Applying PASER's negative transfer mitigation process:

1. **Concept Extraction:** $C(x, y) = \{$quantum computing, deep learning, neural network, speedup$\}$

2. **Consistency Check:** For each pair $(v_i, v_j) \in C(x, y)$:

   - (quantum computing, deep learning): both $\in V$, but $(v_i, v_j) \notin E$
   - (quantum computing, neural network): both $\in V$, but $(v_i, v_j) \notin E$
   - (deep learning, neural network): both $\in V$ and $(v_i, v_j) \in E$
   - (speedup, any other concept): speedup $\notin V$

3. **Decision:** IsConsistent(x, y) = False, because the pairs (quantum computing, deep learning) and (quantum computing, neural network) have both concepts in V, but these edges do not exist in E. This introduces new relationships between existing concepts that are not present in the current CCG.

PASER rejects this sample because it introduces direct relationships between quantum computing and deep learning/neural networks, which were not present in the initial CCG. While these concepts existed separately in the CCG, their combination in this context could lead to potential misunderstandings or oversimplifications about the current state and capabilities of quantum computing in machine learning.

J.3. Conclusion

This case study demonstrates PASER's negative transfer mitigation process in action. By accepting Sample 1, PASER allows for the introduction of new, relevant concepts that expand the concept consistency graph without introducing conflicts. By rejecting Sample 2, PASER prevents the introduction of potentially misleading relationships between existing concepts from different domains, thus mitigating the risk of negative transfer during the recovery process.

K. Implementation Details

Most of experiments are conducted on the server with 8 $\times$ RTX 6000 Ada GPUs. During the recovery post-training phase, we take the the low-rank approximation, LoRA (Hu et al.), to improve the efficiency. The corresponding hyperparameters are set as following: rank=8, batch size=64, epochs=2, learning rate = 1e-4 (Alpaca series experiments), 5e-5 (LaMini series experiments). As for the structured pruning, we set the pruning ratio as 25% for LLaMA2-7B/LLaMA3-8B/Baichuan2-7B and 50% for LLaMA2-13B/LLaMA2-70B/Baichuan-13B models. For the other two kinds of pruning schemes, we follow the previous work (Frantar & Alistarh, 2023); Specifically, we adopt the 2:4 semi-structured sparsity patterns and implement 50% unstructured weight sparsity. Except the experiments for recovery post-training efficiency analysis, we set the ratio of recovery data budget B to original dataset size N as 20% for Alpaca and 4% for LaMini. As for the implementation of concept extraction in Section 3.4, we use the open-source library `rake-nltk` [2]. To ensure statistical robustness, all the results reported in this paper are the averages of five runs with different seeds. Statistical significance is also assessed using two-tailed independent t-tests, with results considered significant when $p < 0.01$. For facilitating the reproduction of our work, we provide the code in the supplementary materials, also seen in anonymous github `https://anonymous.4open.science/r/PASER-E606`.

[2] https://pypi.org/project/rake-nltk/

L. Limitation Analysis

While PASER demonstrates significant improvements in recovery performance and efficiency for pruned large language models, there are several limitations to consider:

- **Computational overhead:** Although PASER reduces the recovery training time, the initial clustering and data selection process introduces some computational overhead. For very large instruction tuning datasets, this overhead may become non-trivial.
- **Dependence on initial pruning quality:** The effectiveness of PASER may vary depending on the quality and method of the initial model pruning. Poorly pruned models might not benefit as much from the targeted recovery approach.
- **Potential bias in capability recovery:** While PASER aims for balanced capability recovery, there might still be some bias towards certain capabilities based on the initial clustering results and the composition of the instruction tuning dataset.
- **Scalability to extremely large models:** The paper primarily demonstrates results on models up to 70B parameters. The scalability and effectiveness of PASER on even larger models (e.g., 100B+ parameters) need further investigation.
- **Long-term Stability:** The long-term stability of models recovered using PASER, especially under continued fine-tuning or adaptation, has not been thoroughly examined in this work.

Limitations of Concept Consistency Graph. In addition to the above analysis, we further discuss the concept consistency graph's potential limitations. Indeed, while CCG helps mitigate negative transfer, we acknowledge there are scenarios where semantic conflicts might not be fully captured:

- **Cross-domain Knowledge Integration:** When instructions involve integrating knowledge from multiple distinct domains, CCG might miss subtle conflicts in their interactions. For example, when concepts from physics and biology are combined in interdisciplinary problems, their complex relationships and potential incompatibilities may not be fully reflected in simple co-occurrence patterns.
- **Context-dependent Semantics:** The same concept pairs might have different relationships depending on context. For instance, terms like "positive" and "negative" could be contradictory in sentiment analysis but complementary in mathematics, making it challenging for CCG to maintain consistent concept relationships across different contexts.
- **Temporal or Version-specific Conflicts:** In rapidly evolving domains like technology or scientific research, concept relationships might change over time. An instruction about "state-of-the-art performance" or "current best practices" could contain outdated or conflicting information that is not immediately apparent from concept co-occurrence analysis.
- **Nuanced Conceptual Dependencies:** When instructions involve subtle logical dependencies or conditional relationships between concepts, the binary edge representation in CCG might not fully capture these complex interactions. This is particularly evident in reasoning tasks where conclusions depend on specific combinations of conditions.

Our results acknowledge these inherent limitations while demonstrating CCG's overall effectiveness in practical applications.

M. Ethics Statement

The development and deployment of technologies like PASER for efficient recovery of pruned large language models necessitates careful consideration of ethical implications. While PASER contributes to reducing environmental impact and potentially democratizing AI access by lowering computational requirements, it also raises concerns about potential misuse, bias amplification, and privacy. It's crucial to remain vigilant about these risks, implement robust safeguards, and maintain transparency in the recovery process. Continuous monitoring for fairness and bias in model outputs is essential, as is responsible deployment with appropriate human oversight, especially in high-stakes applications. As the field evolves, ongoing ethical assessment and dialogue with stakeholders are vital to ensure that advancements in large language model efficiency contribute positively to society while minimizing potential harm. Ultimately, the goal should be to harness the benefits of improved model recovery techniques like PASER while proactively addressing the complex ethical challenges they present.