International Journal of Computer Vision (2026) 134:313
https://doi.org/10.1007/s11263-026-02865-5

Structural Pruning of Large Vision Language Models: A Comprehensive Study on Pruning Dynamics, Recovery, and Data Efficiency

Yiran Huang[1,2,3] · Lukas Thede[3,4] · Massimiliano Mancini[5] · Wenjia Xu[6] · Zeynep Akata[1,2,3]

Received: 15 January 2026 / Accepted: 24 April 2026

Abstract

While Large Vision Language Models (LVLMs) demonstrate impressive capabilities, their substantial computational and memory requirements pose deployment challenges on resource-constrained edge devices. Current parameter reduction techniques primarily involve training LVLMs from small language models, but these methods offer limited flexibility and remain computationally intensive. We study a complementary route: compressing existing LVLMs by applying structured pruning to the language model backbone, followed by lightweight recovery training. Specifically, we investigate two structural pruning paradigms: layerwise and widthwise pruning, and pair them with supervised finetuning and knowledge distillation on logits and hidden states. Additionally, we assess the feasibility of conducting recovery training with only a small fraction of the available data. Our results show that widthwise pruning generally maintains better performance in low-resource scenarios, where computational resources are limited or there is insufficient finetuning data. As for the recovery training, finetuning only the multimodal projector is sufficient at small compression levels. Furthermore, a combination of supervised finetuning and hidden-state distillation yields optimal recovery across various pruning levels. Notably, effective recovery can be achieved using just 5% of the original data, while retaining over 95% of the original performance. Through empirical study on three representative LVLM families ranging from 3B to 7B parameters, this study offers actionable insights for practitioners to compress LVLMs without extensive computation resources or sufficient data.

Keywords Multimodal LLM · Model compression · Structured pruning · Knowledge distillation

Communicated by Francesco Locatello.

✉ Yiran Huang
yiran.huang@tum.de

Lukas Thede
lukas.thede@uni-tuebingen.de

Massimiliano Mancini
massimiliano.mancini@unitn.it

Wenjia Xu
xuwenjia@bupt.edu.cn

Zeynep Akata
zeynep.akata@tum.de

1 Technical University of Munich, Munich, Germany

2 Munich Center for Machine Learning, Munich, Germany

3 Helmholtz Munich, Munich, Germany

4 University of Tübingen, Tübingen AI Center, Tübingen, Germany

5 University of Trento, Trento, Italy

6 Beijing University of Posts and Telecommunications, Beijing, China

1 Introduction

State-of-the-art LVLMs (Chen et al., 2024; Chu et al., 2023; Liu et al., 2024) based on Large Language Models (LLMs) (Jiang, 2024; Touvron et al., 2023) require substantial resources. For instance, models in the LLaVA (Liu et al., 2024) family typically range from 7 to 34 billion parameters, and even compact models like Bunny-v1.0 (3 billion parameters) (He et al., 2024) pose significant deployment challenges in resource-constrained environments. Reducing the size of these models without compromising performance is crucial for adapting them to diverse deployment scenarios with varying resource constraints.

Since the language backbone typically constitutes the vast majority of an LVLM's parameters, it represents the primary target for compression (Chen et al., 2024; He et al., 2024; Liu et al., 2024). Existing approaches address this mainly by building LVLMs from Small Language Models (SLMs) (Chu et al., 2023; He et al., 2024; Zhu et al., 2024). However, these methods suffer from fundamental limitations: they are constrained by the fixed size of the underlying SLM and

Published online: 09 June 2026

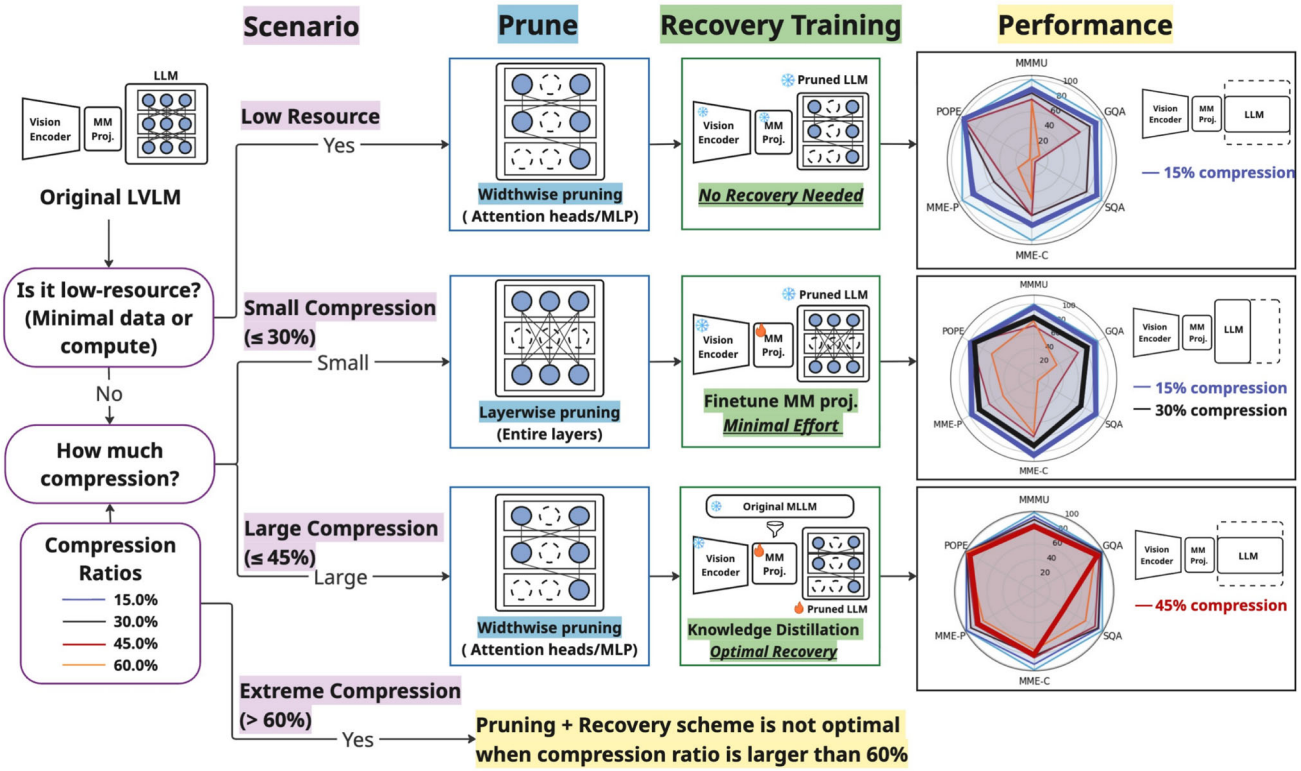

Fig. 1 Compression decision flow for LVLMs. The left panel presents a decision flowchart that guides the choice of pruning and recovery based on the given scenarios, i.e., resource availability and compression ratio requirements. (i) widthwise pruning only (no recovery) in *extremely low?resource settings*; (ii) layerwise pruning with MM?projector fine?tuning for *moderate compression* ($\leq$ 30%); and (iii) widthwise pruning + knowledge distillation for *high compression* ($\leq$ 45%). The right panel shows spider plots of the retained performance across multimodal benchmarks, demonstrating each strategy's effectiveness at various compression levels

require expensive, from-scratch training to meet target specifications. In contrast, we investigate an orthogonal and more flexible approach: directly compressing existing LVLMs via structured pruning of their language backbone. This strategy allows customizable model sizing and performance tuning without retraining from scratch.

However, simply applying established pruning techniques to this new domain is non-trivial. While pruning has been extensively studied for unimodal LLMs, its effectiveness in LVLMs remains largely unexplored. LVLMs present unique challenges: pruning the language backbone risks not only degrading linguistic capabilities but also disrupting the learned vision-language alignment. Consequently, understanding how different pruning paradigms and recovery strategies interact under varying resource constraints is essential for practitioners, yet no systematic study currently exists.

To address this gap, we conduct a comprehensive investigation into structured pruning for LVLMs. We focus on structured pruning because it provides immediate efficiency gains on commodity hardware by physically reducing matrix dimensions, unlike unstructured pruning, which requires specialized sparse kernels to realize speedups (Frantar and Alistarh, 2023; Sun et al., 2024). Concretely, we apply two pruning paradigms originally developed for LLMs to the LVLM setting. The first, layerwise pruning (Men et al., 2024), removes entire transformer blocks, leveraging evidence that many layers are redundant. The second, widthwise pruning (Ma et al., 2023), drops unimportant attention heads and MLP neurons, reflecting observations that only a subset of these sub-components is essential. Crucially, we pair these pruning strategies with various recovery training methods, including supervised finetuning and knowledge distillation on both logits and hidden states. Finally, we vary the pruning ratio and the amount of available data to map out the accuracy/efficiency frontier.

This work extends our preliminary conference version (Huang et al., 2025) significantly in scope and depth. First, we expand the empirical evaluation from two to three representative LVLM families by adding Mini-InternVL-Chat-4B (Chen et al., 2024), in addition to LLaVA-v1.5-7B (Liu et al., 2024) and Bunny-v1-3B (He et al., 2024), ensuring our findings generalize across different architectures. Second, we broaden the evaluation suite to include domain-specific

and reasoning-intensive benchmarks such as AI2D (Kembhavi et al., 2016), MathVista (Lu et al., 2023), and DocVQA (Mathew et al., 2021), providing a more granular view of how compression affects complex reasoning versus simple perception. Third, we introduce a novel analysis disentangling the degradation of modality alignment versus language capabilities, offering an explanation for performance drops. Finally, we provide a new sensitivity analysis regarding calibration dataset size and a qualitative study of failure modes, offering a comprehensive guide for practical deployment. Our systematic empirical analysis provides insights into how pruning and recovery techniques affect LVLM performance under varying compression levels and data availability conditions. Specifically, we found:

- We confirm that widthwise pruning outperforms layerwise pruning in low-resource settings. A new analysis of reasoning benchmarks reveals that layerwise pruning disproportionately degrades performance on complex reasoning tasks compared to widthwise pruning.
- Degradation Dynamics: We demonstrate that low compression ratios primarily damage multimodal alignment rather than the language backbone, whereas high ratios degrade both. This finding explains why finetuning only the projector is sufficient for mild compression.
- Optimal Recovery: We validate that Supervised Finetuning (SFT) combined with hidden-state distillation (L2) consistently yields the best recovery.
- Data Efficiency and Robustness: We show that calibration is highly data-efficient. Furthermore, for recovery training, effective results can be achieved with as little as 5% of the original training data.

We highlight our key findings in Fig. 1. Our findings enable practitioners to efficiently compress LVLMs, allowing researchers to build upon empirically supported strategies without undertaking extensive experimentation themselves.

2 Related Work

2.1 Pruning.

Model compression via pruning has been extensively studied to alleviate computational burdens. Early approaches focused on *unstructured pruning* (Dong et al., 2017; Farina et al., 2024; Frankle and Carbin, 2019; Lee et al., 2020; Park et al., 2020; Sanh et al., 2020), which zeroes out individual weights. While these methods achieve high sparsity with minimal accuracy loss, they typically require specialized hardware or sparse kernels for realizing actual speedups. *Semi-structured pruning* offers a middle ground (Zhou et al., 2021), though it often imposes specific hardware constraints. Consequently, *structured pruning* (Hoefler et al., 2021; Li et al., 2017; Liu et al., 2021; You et al., 2019; Zhang et al., 2022) has gained traction for its ability to remove coherent parameter groups, directly reducing model size and latency on general-purpose hardware.

In the context of Large Language Models (LLMs), recent work demonstrates that structured pruning can be effectively applied to different architectural dimensions. This includes *layerwise* pruning, which removes full layers (Chen et al., 2024; Dery et al., 2024; Men et al., 2024; Song et al., 2024), and *widthwise* pruning, which targets components such as attention heads and MLP neurons (Fang et al., 2023; Ma et al., 2023; Xia et al., 2024), often with only sparse performance degradation. Our study builds on these advances by applying both layerwise and widthwise structured pruning to the language backbone of LVLMs, systematically evaluating their efficacy when paired with subsequent recovery training.

2.2 Knowledge Distillation (KD)

KD facilitates the transfer of learned patterns from a high-capacity "teacher" model to a more compact "student" model (Gou et al., 2021; Hinton et al., 2015; Sanh et al., 2020).

In the context of language modeling, this transfer is typically achieved by aligning the student's output distributions with those of the teacher (Hsieh et al., 2023; Liang et al., 2021). Beyond final outputs, the student can also be trained to mimic the teacher's internal representations, such as hidden states (Jiao et al., 2020; Sun et al., 2019) or self-attention patterns (Wang et al., 2021, 2020). Recent methodological refinements have further optimized this process. For instance, multi-stage distillation allows for the transfer of intermediate features to capture more granular model behaviors (Hsieh et al., 2023), while (Gu et al., 2023) utilizes reverse Kullback-Leibler (KL) divergence to prevent the student from over-focusing on the teacher's low-probability output regions. Notably, KD has emerged as a recovery strategy following aggressive pruning (Hoffmann et al., 2021; Muralidharan et al., 2024), in which the uncompressed model serves as the teacher to restore lost performance.

While KD is increasingly used in the vision-language domain to distill large LVLMs into smaller architectures (Cai et al., 2025; Kim et al., 2025), its use as a *post-pruning recovery mechanism* for LVLMs remains under-explored. We address this gap by empirically evaluating various KD strategies against conventional supervised finetuning to identify the most effective recovery schemes for pruned LVLMs.

2.3 Efficient LVLMs

Research on accelerating LVLMs generally follows two paradigms: input-level optimization and architectural efficiency.

A significant body of work focuses on visual token reduction (Shang et al., 2025; Tang et al., 2025; Wang et al., 2025). For instance, LLaVA-PruMerge (Shang et al., 2025) utilizes a dynamic approach of token pruning and merging for adaptive compression, while CoViPAL (Tang et al., 2025) introduces a model-agnostic plug-and-play module for token pruning. While these token-level methods reduce computational overhead by lowering input dimensionality, they differ fundamentally from our approach as they leave the underlying model architecture uncompressed.

Alternatively, researchers have developed lightweight LVLMs by integrating smaller language backbones, as seen in LLaVA-Phi (Zhu et al., 2024), MobileVLM (Chu et al., 2023), and Bunny (He et al., 2024). While these models are inherently efficient, they are constrained by the fixed dimensions of their underlying small language models (SLMs). In contrast, our study focuses on the structured pruning and recovery of existing, large-scale LVLMs. This enables customizable model sizing and performance tuning, offering greater flexibility than training small-scale models from scratch.

3 Methodology

This section outlines our approach to compressing LVLMs. We first introduce two pruning strategies: layerwise and widthwise pruning. We then describe methods to recover model performance through supervised finetuning and knowledge distillation.

Notation. Given a triplet $\mathbf{X} = \{\mathbf{x}_v, \mathbf{x}_p, \mathbf{x}_r\}$, the objective of an LVLM m_θ, parameterized by $\theta = \{\psi, \phi, \mathbf{W}\}$, is to generate a response $\mathbf{x}_r$ based on an input image $\mathbf{x}_v$ and a text prompt $\mathbf{x}_p$, such that $m_\theta(\mathbf{x}_v, \mathbf{x}_p) = \mathbf{x}_r$. The LVLM typically consists of a vision encoder $g_\psi(\cdot)$, an LLM $f_\phi(\cdot)$, and a multimodal projector $\mathbf{W}$ aligning the two modalities. The prompt $\mathbf{x}_p$ is tokenized into $\mathbf{T}_p$, while the vision encoder processes the image $\mathbf{x}_v$ to extract visual features, which are then converted into language embedding tokens $\mathbf{T}_v$ via the multimodal projector:

$$\mathbf{T}_v = \mathbf{W} \cdot g_\psi(\mathbf{x}_v) \quad \text{and} \quad f_\phi(\mathbf{T}_v \odot \mathbf{T}_p) = \mathbf{x}_r. \tag{1}$$

The concatenated visual tokens $\mathbf{T}_v$ and prompt tokens $\mathbf{T}_p$ are fed into the LLM's M layers, producing hidden states $\{\mathbf{H}_i \in \mathbb{R}^{T\times d}\}_{i=1}^{M}$, where T is the number of tokens and d is the hidden dimension. Finally, the probabilities $p_{m_\theta}(\mathbf{x}_r|\mathbf{x}_v, \mathbf{x}_p, \tau)$ are computed by passing the final hidden state through the classification head with softmax temperature τ.

3.1 Pruning

Large Transformers are largely over-parameterized, as whole layers can be dropped with little accuracy loss (Fan et al., 2019; Sajjad et al., 2023), and only a few attention heads or MLP units per layer truly matter (Hudson and Manning, 2019; Mccarley et al., 2019; Michel et al., 2019; Voita et al., 2019). Motivated by these findings, we explore two pruning paradigms specifically targeting the language model backbone within LVLMs: layerwise pruning, which removes entire transformer layers, and widthwise pruning, which eliminates the least important components within each layer. To determine which layers or components to prune, we draw a small subset of n samples from the original visual instruct-tuning dataset as the calibration dataset $\mathcal{D} = \{\mathbf{x}_v^j, \mathbf{x}_p^j, \mathbf{x}_r^j\}_{j=1}^{n}$. The importance of each layer or component is assessed, and the lowest-importance components are pruned.

Layerwise Pruning. To identify the redundant layers, we use the Block Influence (BI) score (Men et al., 2024), which quantifies the importance of layer i through the cosine distance between input $\mathbf{H}_i$ and output hidden states $\mathbf{H}_{i+1}$. The key assumption is that layers that cause larger changes in hidden states have a greater influence on model performance. The BI score of layer i is then calculated by

$$\mathrm{BI}_i(\mathcal{D}) = 1 - \mathbb{E}_{\mathbf{X}\sim\mathcal{D},t}\left[\frac{\mathbf{H}_{i,t}^{\mathsf{T}}\mathbf{H}_{i+1,t}}{\|\mathbf{H}_{i,t}\|_2\|\mathbf{H}_{i+1,t}\|_2}\right], \tag{2}$$

where $\mathbf{H}_{i,t}$ represents the t^{th} row of $\mathbf{H}_i$. After calculating the BI scores, the layers are ranked by importance, and the lowest-scoring layers are pruned.

Widthwise Pruning To address the widthwise redundancy, we apply dependency-based structural pruning. Following (Fang et al., 2023) and Ma et al. (2023), we build a dependency graph inside each LLM layer. Let N_i and N_j represent two neurons in the layer, where $\mathrm{In}(N_i)$ and $\mathrm{Out}(N_i)$ represent the neurons connected to N_i as inputs and outputs, respectively. Neuron N_j is dependent on on N_i if

$$\begin{aligned} &N_j \in \mathrm{Out}(N_i) \wedge \mathrm{Num}_{\mathrm{In}(N_j)} = 1, \\ &\text{or } N_j \in \mathrm{In}(N_i) \wedge \mathrm{Num}_{\mathrm{Out}(N_j)} = 1 \end{aligned} \tag{3}$$

where $\mathrm{Num}_{\mathrm{In}(N_j)}$ refers to the number of input neurons of N_j and $\mathrm{Num}_{\mathrm{Out}(N_j)}$ is the number of the output neurons of N_j. In words, N_i is the only downstream or upstream node of N_j. If neuron N_i is pruned, all its dependent neurons N_j must also be pruned. This process results in a set of dependency graphs $G = \{w_i^k\}_{i=1}^{M}$, where M is the number of structures in the graph and w_i^k represents the k^{th} weight parameter within a structure.

We assess their importance at the group level since all weights within a graph must be pruned together. Group

importance is evaluated by comparing the loss of vision language modeling $\mathcal{L}_{CE}(m_\theta(\mathbf{x}_v, \mathbf{x}_q), \mathbf{x}_r)$ in the calibration data set, with and without weight. To efficiently approximate the importance, we apply a Taylor expansion using gradient information:

$$I_{w_i^k}(\mathbf{X}) = |\mathcal{L}_{CE}(\mathbf{X}, m_\theta) - \mathcal{L}_{CE}(\mathbf{X}, m_\theta^{w_i^k=0})| \approx \left| \frac{\partial \mathcal{L}_{CE}(\mathbf{X}, m_\theta)}{\partial w_i^k} w_i^k \right| . \tag{4}$$

We then prune the graphs with the lowest group importance I_G:

$$I_G(\mathcal{D}) = \mathbb{E}_{\mathbf{X}\sim\mathcal{D}} \left[\sum_i^M \sum_k I_{w_i^k}(\mathbf{X}) \right]. \tag{5}$$

3.2 Recovery Training

Pruning LVLMs degrades performance, affecting both language modeling and cross-modality alignment. To mitigate this, we investigate two recovery training methods: supervised finetuning (Sec. 3.2.1) and knowledge distillation (Sec. 3.2.2). We consider the original uncompressed model as the teacher m_θ^{T}, the pruned model as the student $m_{\theta'}^{\mathrm{S}}$, and a recovery dataset $\mathcal{D}$.

3.2.1 Recovery Training with Supervised Finetuning (SFT)

We first focus on training only the multimodal projector to realign the vision and language spaces. Second, we jointly finetune the projector and the pruned language model while keeping the vision encoder fixed, as finetuning the vision encoder has been shown to yield negligible gains (Karamcheti et al., 2024) and increases training cost. We use the cross-entropy loss for supervised finetuning:

$$\mathcal{L}_{sft}(m_{\theta'}^{\mathrm{S}}, \mathcal{D}) = \mathbb{E}_{\mathbf{X}\sim\mathcal{D}}[\mathcal{L}_{CE}(m_{\theta'}^{\mathrm{S}}(\mathbf{x}_v, \mathbf{x}_p), \mathbf{x}_r)]. \tag{6}$$

3.2.2 Recovery Training with Knowledge Distillation

KD allows the pruned model to regain lost performance by mimicking the decision-making process of the more capable teacher (original model). We explore two main strategies, logits-based KD and hidden state based KD.

Logits-based KD aligns the output probability distributions of the pruned model with those of the teacher model. The logits-based KD loss is defined as

$$\mathcal{L}_{logits}(m_{\theta'}^{\mathrm{S}}, m_\theta^{\mathrm{T}}, \mathcal{D}) = \mathbb{E}_{\mathbf{X}\sim\mathcal{D}}\big[\mathcal{L}_{KD}(p_{m_\theta^{\mathrm{T}}}(\mathbf{x}_r|\mathbf{x}_v, \mathbf{x}_p, \tau), p_{m_{\theta'}^{\mathrm{S}}}(\mathbf{x}_r|\mathbf{x}_v, \mathbf{x}_p, \tau))\big]. \tag{7}$$

We explore two losses to evaluate the differences between the logit distributions of the student p_θ and the teacher $q_{\theta'}$: Kullback–Leibler divergence (KL), denoted as $\mathbf{KL}(p_\theta \| q_{\theta'})$ and its reversed form (RKL), denoted as $\mathbf{KL}(q_{\theta'} \| p_\theta)$. The standard KD objective, minimizing the approximated forward KL, encourages the student distribution to match all modes of the teacher distribution. In contrast, using RKL encourages $q_{\theta'}$ to focus on the major modes of p_θ while assigning low probabilities to its less significant regions. This helps the student model avoid learning unnecessary long-tail variations of the teacher distribution and instead focus on generating more accurate responses (Gu et al., 2023; Holtzman et al., 2019).

Hidden State Matching involves aligning the pruned model's intermediate representations (hidden states) $\mathbf{H}_i^{m_{\theta'}^{\mathrm{S}}}$ with the teacher model's $\mathbf{H}_i^{m_\theta^{\mathrm{T}}}$. The corresponding loss for a layer i can be defined as

$$\mathcal{L}_{match}(m_{\theta'}^{\mathrm{S}}, m_\theta^{\mathrm{T}}, \mathcal{D}) = \mathbb{E}_{\mathbf{X}\sim\mathcal{D}} \left[\mathcal{L}_{feat}(\mathbf{H}_i^{m_{\theta'}^{\mathrm{S}}}, \mathbf{H}_i^{m_\theta^{\mathrm{T}}}) \right], \tag{8}$$

where $\mathcal{L}_{feat}$ refers to a feature matching loss. Both (Yang et al., 2024) and Popp et al. (2024) suggest that applying a feature-based L2 distillation loss improves the student model's performance, particularly for pre-trained vision-language models. Consequently, we employ L2 loss as the feature matching loss $\mathcal{L}_{feat}(x, y) = \|x - y\|_2^2$. The total loss for recovery training is computed as:

$$\mathcal{L}(m_\theta^{\mathrm{S}}, m_\theta^{\mathrm{T}}, \mathcal{D}) = \alpha \mathcal{L}_{sft}(m_{\theta'}^{\mathrm{S}}, \mathcal{D}) + \beta \mathcal{L}_{logits}(m_{\theta'}^{\mathrm{S}}, m_\theta^{\mathrm{T}}, \mathcal{D}) + \gamma \mathcal{L}_{match}(m_{\theta'}^{\mathrm{S}}, m_\theta^{\mathrm{T}}, \mathcal{D}) \tag{9}$$

where α, β, and γ are the coefficients that balance three loss components.

4 Experiments

In this section, we first introduce our experimental setup and then demonstrate the main findings on model pruning (Sec. 4.2) and performance recovery (Sec. 4.3, Sec. 4.4). We highlight the findings on recovery training using only a small fraction of data (Sec. 4.5) and present the model compression results following our best practices (Sec. 4.6), including the qualitative performance and failure case.

4.1 Experimental Setup

Model. To ensure architectural diversity, we evaluate pruning and recovery methods on three LVLMs from distinct model families: LLaVA-v1.5-7B (LLaVA) (Liu et al., 2024),

Table 1 Architecture details of the uncompressed models. We present the number of parameters, along with the vision encoder, multimodal projector and the language decoder of the models included in our study

Model	Parameters	Vision Encoder	Multimodal Projector	Language Decoder
LLaVA-v1.5-7B	7.0B	CLIP-ViT-L (0.3B)	mlp2x-gelu (0.01B)	Vicuna-v1.5 (6.7B)
Mini-InternVL-Chat-4B-V1.5	4.1B	InternViT-300M-448px L (0.3B)	mlp2x-gelu (0.02B)	Phi-3-mini-128k-instruct (3.8B)
Bunny-v1.0-3B	3.2B	SigLIP-SO (0.4B)	mlp2x-gelu (0.02B)	Phi-2 (2.8B)

Mini-InternVL-Chat-4B (InternVL) (Chen et al., 2024), and Bunny-v1.0-3B (Bunny) (He et al., 2024). As detailed in Table 1, the language backbone accounts for the majority of the total parameter count across all three architectures. This structural imbalance underscores the need to target the language component to achieve meaningful compression.

Data. We exclusively utilize each model's original visual instruction tuning dataset for both pruning calibration and recovery training: LLaVA-v1.5-mix665k (Liu et al., 2024) for LLaVA, InternVL-Chat-V1-2-SFT-Data (Chen et al., 2024) for InternVL, and Bunny-695K (He et al., 2024) for Bunny. During the pruning stage, we draw a small subset of n samples from the respective training data to serve as the calibration dataset for computing importance scores. For the subsequent recovery phase, we investigate the impact of data scale by finetuning on varying fractions (5%, 10%, 20%, and 100%) of the full dataset.

Evaluation We evaluate the pruned and recovered models on a diverse suite of benchmarks designed to challenge various facets of multimodal capability. To assess domain-specific knowledge, we employ MMMU (Yue et al., 2024), SQA (Lu et al., 2022). For advanced reasoning, we employ MathVista (Lu et al., 2023) and AI2D (Kembhavi et al., 2016). Complementing these are assessments of fine-grained perception and hallucination, including POPE (Li et al., 2023) and the separate Cognition and Perception subsets of MME (Yin et al., 2023). Finally, we evaluate general visual understanding and document understanding via GQA (Hudson and Manning, 2019) and DocVQA (Mathew et al., 2021). All assessments are conducted using the `lmms-eval` suite (Bo et al., 2024) to ensure reproducibility. Crucially, these benchmarks span a wide range of prompt formats, requiring the models to adapt to multiple-choice questions (e.g., MMMU, SQA, AI2D), binary Yes/No classification (e.g., POPE, MME), and open-ended free generation (e.g., MathVista, DocVQA, GQA). For clarity, we report performance as a percentage relative to the uncompressed model's score on each benchmark.

4.2 Structural Pruning Paradigms: Widthwise vs. Layerwise Analysis

In this section, we compare two structural pruning strategies: widthwise pruning and layerwise pruning. We evaluate the relative strengths of these methods by analyzing their zero-shot performance immediately after pruning in Sec. 4.2.1, their post-training recovery capabilities after finetuning in Sec. 4.2.2, and their efficiency regarding actual GPU memory and computational cost reduction in Sec. 4.2.3.

4.2.1 Zero-Shot Performance

Our evaluation shows that widthwise pruning consistently outperforms layerwise pruning when the model is not fine-tuned. As shown in Table 2, widthwise pruning keeps much more of the original capability at moderate pruning ratios (15%-30%). For example, at a 15% ratio, LLaVA retains 90.87% of its performance with widthwise pruning, compared to only 72.22% with layerwise pruning. However, there is a limit: beyond a 45% ratio, both methods fail, and benchmark scores on SQA and DocVQA drop to near-zero.

Notably, the contrast between widthwise and layerwise pruning becomes much clearer once we stratify benchmarks by either (i) output format or (ii) reasoning demand. First, on tasks that require free-form responses (GQA, DocVQA, and MathVista), the gap is substantially larger than the aggregate trend (Fig. 2). For instance, on Bunny at a 15% pruning ratio, widthwise pruning improves overall performance by only 2.2% relative to layerwise pruning, whereas the margin on generative tasks increases to 31.5%. Second, a similar pattern appears on reasoning-heavy benchmarks (MathVista and AI2D). Table 2 shows that widthwise pruning reaches 65.64 on AI2D versus 52.73 for layerwise pruning, corresponding to an absolute gap of 12.91 points (nearly 20% relative). Taken together, these results indicate that *layer removal* is disproportionately harmful on benchmarks that place higher demands on multi-step reasoning and/or free-form generation.

Crucially, our findings in the multimodal domain strongly align with recent discoveries in the unimodal LLM pruning literature, pointing to a fundamental property of transformer architectures. For instance, Sreenivas et al. (2024) recently demonstrated a similar phenomenon when pruning LLMs: widthwise pruning proved vastly superior to layerwise pruning, especially on reasoning-intensive benchmarks such as GSM8K (Cobbe et al., 2021). The fact that layer removal disproportionately impairs complex reasoning across both pure language models and vision-language models underscores that this degradation is not merely an artifact of multimodal integration. Instead, it exposes a structural vulnerability inherent to the underlying language backbone itself.

We attribute these trends to two primary structural factors: feature alignment and architectural depth. First, the general performance drop in layerwise pruning stems from feature misalignment. In a pre-trained network, Layer $N+1$ is optimized to process the specific feature distribution output by Layer N. Removing a layer forces Layer N to feed directly into Layer $N+2$. This creates a distribution shift that subsequent layers are not calibrated to handle, disrupting the forward pass for all tasks regardless of complexity. Widthwise pruning avoids this shock by maintaining the original layer interfaces, preserving the expected feature flow. Second, the disproportionate decline in free-form generation and

Table 2 Pruning results for LLaVA-v1.5-7B, Mini-InternVL-Chat-4, and Bunny-v1-3B across benchmarks. Widthwise pruning generally results in better performance without finetuning compared to layerwise pruning

Method	Size	Ratio	MMMU	SQA	MathVista	AI2D	MME-C	MME-P	POPE	DocVQA	GQA	AVG	AVG-%
LLaVA-v1.5-7B	7.0B	0%	35.10	68.67	26.70	54.80	363.21	1511.33	86.99	28.10	61.98	53.70	100.00%
Widthwise	6.0B	15%	32.40	63.21	23.10	45.40	268.93	1432.47	86.57	23.90	59.34	48.80	90.87%
	5.0B	30%	31.00	54.29	15.80	37.80	253.21	1174.93	86.29	17.90	52.59	42.90	79.88%
	3.8B	45%	27.60	12.10	5.40	13.40	70.00	347.45	45.96	5.30	20.86	17.42	32.43%
	2.8B	60%	23.30	0.40	1.30	1.20	2.14	19.24	3.94	0.80	0.43	3.62	6.75%
Layerwise	6.0B	15%	32.70	59.64	14.80	32.10	210.71	921.88	78.69	16.50	42.18	38.78	72.22%
	5.0B	30%	31.80	55.23	13.50	29.70	202.14	701.83	86.38	14.60	42.77	37.15	69.18%
	4.0B	45%	26.90	3.82	5.40	10.50	132.86	616.63	51.69	6.10	14.39	18.47	34.40%
	2.8B	60%	25.80	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	2.87	5.34%
Mini-InternVL-Chat-4B	4.1B	0%	43.20	93.30	53.70	76.90	547.50	1596.71	88.00	87.70	62.57	72.63	100.00%
Widthwise	3.5B	15%	42.70	92.96	38.80	72.40	527.86	1594.37	88.09	80.20	58.43	68.81	94.74%
	3.0B	30%	33.30	68.82	28.80	44.50	397.14	1300.60	85.20	64.20	42.39	53.54	73.72%
	2.5B	45%	24.00	12.30	6.20	10.80	260.20	980.00	83.00	25.00	22.00	29.43	40.52%
	1.6B	60%	22.30	0.00	0.40	0.50	0.51	14.34	10.00	0.00	3.72	5.12	7.05%
Layerwise	3.5B	15%	43.60	93.12	22.50	70.20	510.10	1588.30	88.15	78.30	52.35	65.71	90.48%
	3.0B	30%	28.30	62.82	18.40	24.50	220.90	1200.60	73.20	53.10	27.39	41.71	57.42%
	2.5B	45%	24.68	8.80	1.20	8.80	220.10	887.35	80.00	12.00	16.00	24.82	34.17%
	1.6B	60%	21.30	0.00	0.40	0.50	0.31	6.34	0.20	0.00	2.53	3.48	4.79%
Bunny-v1-3B	3.2B	0%	34.10	70.70	25.80	60.49	289.30	1487.71	87.82	22.80	54.72	51.89	100.00%
Widthwise	2.8B	15%	30.90	65.64	23.40	55.73	242.50	1207.85	87.94	18.20	51.83	47.15	90.87%
	2.4B	30%	28.40	55.73	14.20	43.97	199.64	807.95	87.13	12.50	45.65	39.21	75.58%
	2.0B	45%	25.70	3.42	5.10	5.56	200.00	618.25	83.12	4.80	37.92	24.61	47.44%
	1.5B	60%	24.80	0.00	1.20	0.20	141.07	293.23	2.34	0.80	6.12	7.53	14.51%
Layerwise	2.8B	15%	33.80	69.66	18.80	52.73	271.43	1456.41	87.91	15.80	29.42	46.10	88.84%
	2.4B	30%	29.00	28.76	10.10	34.97	272.86	1273.34	86.50	10.50	24.77	35.82	69.03%
	2.0B	45%	23.90	3.47	8.40	3.56	191.43	867.37	80.09	11.20	16.85	23.86	45.99%
	1.5B	60%	26.60	17.15	4.20	0.20	0.71	55.92	0.02	3.90	0.02	6.11	11.77%

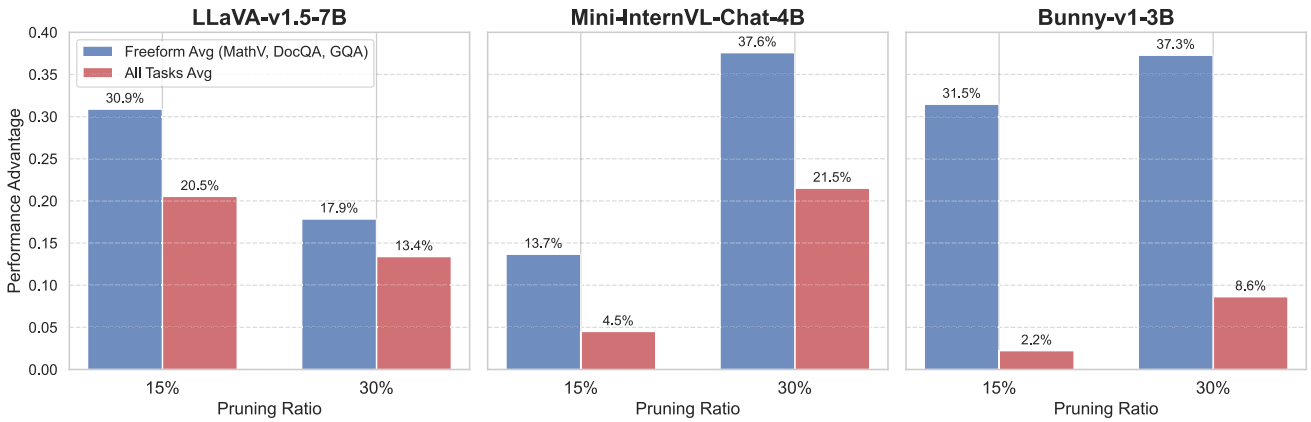

Fig. 2 Comparative Analysis of widthwise vs. layerwise Pruning. We evaluate the relative performance advantage of widthwise over layerwise pruning ($\Delta = (S_{\text{width}} - S_{\text{depth}})/S_{\text{width}}$) at 15% and 30% sparsity ratios. Widthwise pruning consistently outperforms layerwise strategies, with a significantly larger gap observed in complex freeform generation tasks (MathVista, DocVQA, GQA) compared to the global average, suggesting depth is critical for open-ended reasoning

reasoning performance (e.g., MathVista) is directly linked to the loss of processing depth. Although non-linearity exists in both dimensions, depth is specifically required to build sequential reasoning chains (Liu et al., 2024; Telgarsky, 2016). Theoretical studies show that complex logic relies on a deep sequence of transformations, which cannot be efficiently approximated by a wider, shallower network. Layerwise pruning explicitly shortens this "chain of thought", limiting the model's ability to perform multi-step reasoning. In contrast, widthwise pruning reduces redundancy within layers but preserves the full sequence of steps required for complex tasks.

4.2.2 The Impact of Recovery Training after Pruning

While zero-shot performance highlights the immediate structural damage of pruning, recovery training reveals the model's capacity to heal. In this section, we compare the intrinsic recoverability of widthwise versus layerwise structures under standard conditions, explicitly deferring the optimization of specific training recipes, such as parameter selection and distillation objectives, to Sec. 4.3) and Sec. 4.4).

When recovery training is applied, it reshapes the performance landscape (see Fig. 3, red lines). Layerwise pruning demonstrates superior recoverability at low compression ratios ($\leq$ 30%). However, at high compression ratios ($\geq$ 45%), the trend shifts: widthwise pruning generally yields superior recovery performance across most models evaluated (e.g., Mini-InternVL and Bunny) or, at a minimum, achieves parity with layerwise pruning (as observed with LLaVA).

This reversal is driven by the shifting balance between the two factors identified in Sec. 4.2.1: feature misalignment, which is learnable, and processing depth, which is structural. At low compression ratios, the primary limitation of layerwise pruning was the immediate feature distribution shift between unconnected layers. Recovery training effectively mitigates this issue by realigning the feature spaces of adjacent layers. Once this misalignment is corrected, layerwise pruning proves advantageous because it preserves the full dimensionality (width) of the remaining layers, maintaining a richer representation subspace than the thinned-out layers of widthwise pruning. However, as the compression ratio increases, the model hits the structural depth constraint discussed in Sec. 4.2.1. While finetuning can address feature alignment, it cannot overcome the fundamental need for sequential non-linear transformations required for complex reasoning (Telgarsky, 2016). Thus, widthwise pruning prevails in the high-compression regime simply because it sustains the necessary architectural framework for reasoning, a structural advantage that recovery training alone cannot replicate.

Takeaway. Widthwise pruning proves more effective than layerwise pruning in obtaining the best pruned model. With recovery training, layerwise pruning shows a slight advantage at compression ratios no greater 30%, while widthwise pruning performs better at higher compression ratios.

4.2.3 Hardware Efficiency

Table 3 compares the resource efficiency of widthwise and layerwise pruning across three metrics: GPU memory usage, computational cost (FLOPs), and real-world inference latency. As expected, increasing the pruning ratio linearly reduces memory consumption and computational overhead across all models. For instance, at a 60% compression ratio, resource consumption decreases by approximately half for both methods.

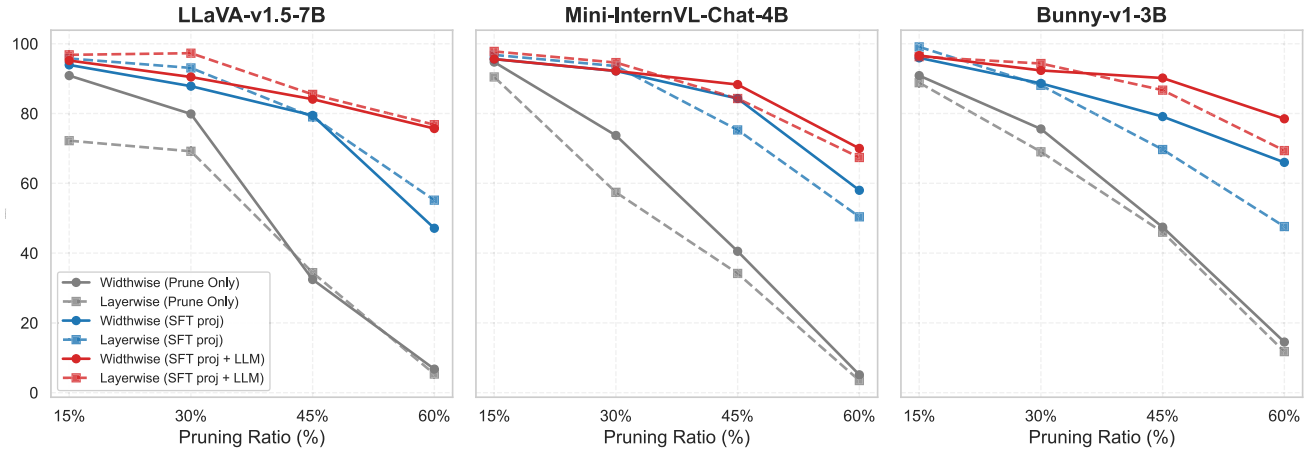

Fig. 3 Evolution of pruning efficacy across supervised finetuning(SFT) regimes. The plots illustrate the average performance across LLaVA-v1.5-7B, Mini-InternVL-Chat-4, and Bunny-v1-3B. ((1) Robustness of widthwise pruning: in the absence of SFT (gray lines), widthwise pruning (solid) consistently maintains higher performance than layerwise pruning (dashed). (2) Regime shifts with SFT: Post-SFT (blue/red lines), the landscape inverts at lower ratios ($\leq$ 30%), where layerwise pruning becomes superior. However, at higher compression rates ($\geq$ 45%), widthwise pruning regains its advantage. (3) Efficiency of multimodal projector SFT: SFT only the projector (blue) yields performance comparable to full projector+LLM SFT (red) at low sparsity, highlighting a cost-effective strategy for moderate model compression

Table 3 Memory (GiB), FLOPS (T), and Latency(ms) for Bunny, InternVL, and LLaVA at various compression ratios for widthwise and layerwise pruning

Ratio	Bunny			InternVL			LLaVA		
	Mem.	FLOPS	Latency	Mem.	FLOPS	Latency	Mem.	FLOPS	Latency
0%	6.02	4.77	50 ± 1.2	8.22	5.12	58 ± 1.5	13.23	9.57	105 ± 1.5
Widthwise Pruning									
15%	5.25	4.14	47.2 ± 1.1	7.05	4.41	54.8 ± 1.4	11.26	8.21	98.2 ± 3.1
30%	4.49	3.50	43 ± 1.0	5.82	3.73	49.2 ± 1.2	9.32	6.89	84.4 ± 2.5
45%	3.68	2.84	38 ± 0.8	4.61	3.06	46 ± 1.1	7.29	5.49	64.5 ± 1.9
60%	2.92	2.20	32.3 ± 0.6	3.43	2.38	36.7 ± 0.9	5.31	4.17	55.4 ± 1.4
Layerwise Pruning									
15%	5.28	4.16	46.5 ± 0.8	7.16	4.45	54.3 ± 1.1	11.33	8.03	95 ± 8.1
30%	4.55	3.56	41.2 ± 0.9	6.11	3.79	48.7 ± 1.3	9.44	6.92	80.7 ± 0.6
45%	3.82	2.95	36.1 ± 0.7	5.05	3.12	42.3 ± 0.9	7.54	5.55	58.5 ± 1.6
60%	2.94	2.22	30.2 ± 0.5	3.55	2.41	32.4 ± 0.8	5.37	3.90	49.2 ± 1.2

Our results highlight a strong consistency between widthwise and layerwise pruning in terms of memory and FLOPs. Although the two methods remove parameters from different parts of the transformer structure, they yield nearly identical theoretical hardware costs when the overall compression ratio is held constant. For example, at 30% compression on LLaVA, the difference in FLOPs between widthwise (6.89 T) and layerwise (6.92 T) pruning is less than 0.5%.

However, measuring actual inference latency reveals a notable divergence. As shown in Table 3, layerwise pruning consistently yields lower inference latency than widthwise pruning across all tested models and compression ratios. For example, on LLaVA at 30% compression, layerwise pruning achieves a latency of 80.7 ms, compared to 84.4 ms for widthwise pruning. This empirical advantage stems from fundamental differences in how each pruning strategy interacts with GPU execution. Layerwise pruning removes entire transformer blocks, directly reducing the number of sequential operations and eliminating multiple CUDA kernel launches per removed layer. Widthwise pruning also delivers meaningful speedups by reducing the size of matrix operations within each layer, but it preserves the original network depth, meaning the fixed overhead of launching kernels for every layer remains unchanged. Additionally, while smaller matrix dimensions reduce total computation, the relationship between matrix size and execution time is not strictly linear on modern GPUs due to memory bandwidth constraints and parallelization thresholds.

Consequently, while both strategies provide immense memory and speed gains, layerwise pruning is more efficient for actual hardware execution. Nevertheless, because widthwise pruning better preserves reasoning capabilities (as

Table 4 Disentangling multimodal versus language degradation. We compare average performance (AVG) and relative degradation (Deg) for LLaVA-v1.5-7B under widthwise pruning. Comparisons between multimodal (mm) and text-only (txt) inputs reveal that low compression ratios disproportionately affect modality alignment (Deg_{mm}) compared to language capabilities (Deg_{txt})

Ratio	AVG_{mm}	AVG_{txt}	Deg_{mm}	Deg_{txt}
0.00%	53.70	40.30	0.00%	0.00%
15.00%	48.80	38.87	9.13%	1.07%
30.00%	42.90	36.12	20.11%	10.38%
45.00%	17.42	17.36	67.56%	56.92%
60.00%	3.62	2.80	93.26%	93.05%

discussed in Section 4.2.1), selecting the superior overall strategy requires a careful trade-off, balancing deployment needs for absolute latency reduction against the retention of task accuracy.

4.3 Supervised Finetuning for Performance Recovery

Pruning the language backbones in LVLMs inevitably degrades their capabilities. To understand the mechanism of this degradation, we first disentangle the effects of pruning on modality alignment versus core language reasoning. We evaluate pruned models on the same benchmarks using both multimodal inputs and text-only inputs. Table 4 reveals a distinct disparity in performance degradation. At a low compression ratio of 15%, multimodal performance drops significantly (9.13%) while text-only performance remains largely intact (1.07% degradation). This implies that mild pruning primarily disrupts the *alignment* between the vision and language components rather than the language model itself. However, as the compression ratio increases, both multimodal and text-only performance degrade sharply. This indicates that at higher pruning ratios, the degradation stems from damage to both the modality alignment and the fundamental language backbone.

To address these issues, we investigate two recovery strategies: (1) supervised finetuning (SFT) only the multimodal projector, and (2) jointly SFT both the projector and the LLM. These experiments allow us to isolate the source of degradation and determine the most efficient recovery method. Following prior work suggesting that training the vision encoder can be detrimental (Karamcheti et al., 2024), we keep the vision encoder frozen in all setups. For the LLM, we employ Low-Rank Adaptation (LoRA) (Hu et al., 2021) to facilitate efficient finetuning.

SFT the multimodal projector As shown in Fig. 3 (blue lines), SFT the projector significantly restores performance, particularly at lower compression ratios. When the compression ratio is under 30%, updating the projector alone yields results comparable to jointly SFT the LLM. Notably, at a 15% compression ratio, this approach retains over 95% of the original performance for all the models. Even at a severe compression ratio of 60%, projector SFT recovers 60–80% of the original performance by re-aligning the visual features with the pruned language model. These results confirm that pruning induces a misalignment between visual and textual representations, which can be largely corrected by updating the projection layer.

SFT both the projector and the LLM While projector SFT is effective for realignment, it cannot fully compensate for the loss of linguistic knowledge at higher compression rates. We observe consistent performance gains from additionally SFT the pruned LLM (red lines in Fig. 3), specifically when the compression ratio exceeds 40%. This aligns with our observation in Table 4 that the language backbone itself degrades significantly at these ratios. By SFT the LLM, we significantly restore these lost reasoning capabilities. For instance, at 45% compression, joint SFT restores over 80% of the original model's performance across all three models, underscoring its necessity when the language backbone is heavily compressed.

Takeaway. When a small compression ratio of around 15% is required, SFT the multimodal projector alone is typically sufficient to recover most of the model's performance. For higher compression ratios ($\geq$ 45%), incorporating SFT of the LLM yields additional performance improvements.

4.4 Knowledge Distillation for Performance Recovery.

Standard SFT recovers performance by retraining the model on ground-truth labels. However, the unpruned teacher model contains richer information–both in its output probability distributions and internal representations–that SFT ignores. We investigate whether distilling this knowledge back into the pruned student can further enhance recovery. We first compare standalone KD strategies against SFT, and then identify the optimal combination of SFT and KD.

Comparison of KD strategies vs. SFT. In Fig. 4, we evaluate three KD objectives without SFT: standard KL Divergence (KL), Reverse KL (RKL), and L2 distance on hidden states (L2). At a low pruning ratio (15%), all methods perform comparably to SFT. However, as the ratio increases, all KD methods fail to surpass SFT. Notably, L2 performance drops the fastest, becoming the least effective standalone strategy at high compression rates.

The degradation of KD-only methods underscores that without ground-truth labels (SFT), the pruned model lacks the strong supervision needed to realign its output space, especially at high pruning ratios. Regarding the specific KD trends ($RKL > KL > L2$), we attribute RKL's superiority

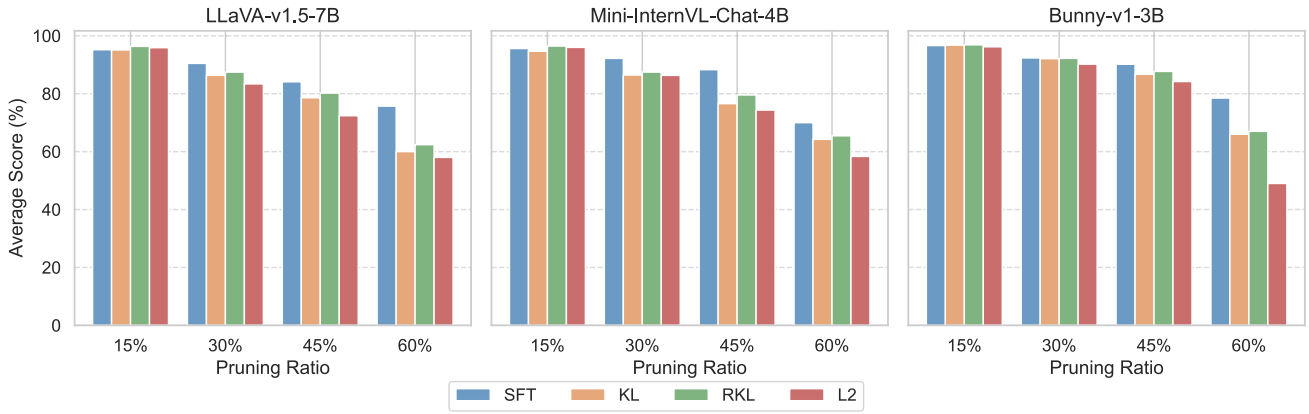

Fig. 4 Performance comparison of KD strategies versus SFT across varying pruning ratios for LLaVA, InternVL, and Bunny models. KD achieves comparable results to SFT at low pruning ratios but performs significantly worse at higher ratios

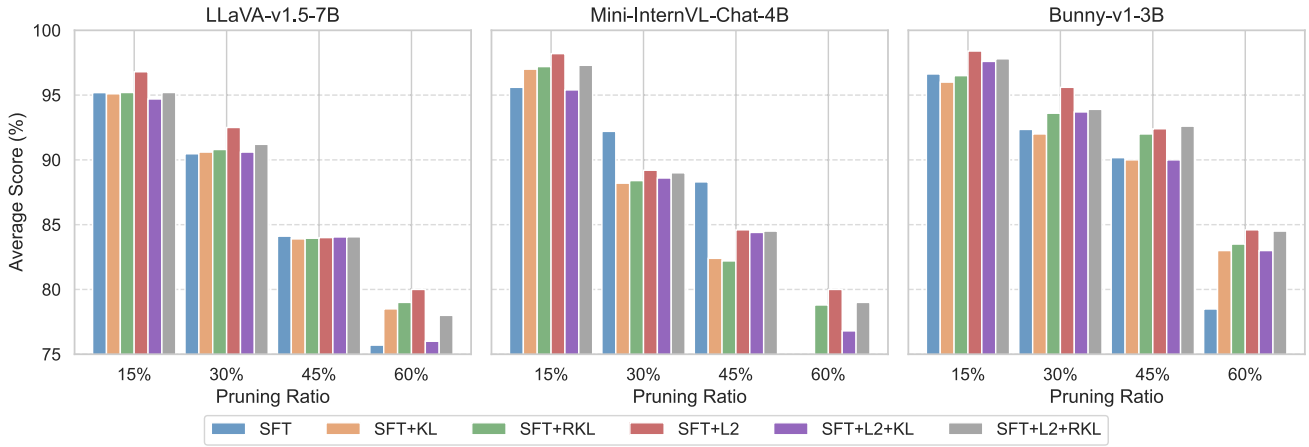

Fig. 5 Performance comparison of combined recovery training techniques for LLaVA, InternVL, and Bunny models. Among the evaluated strategies, **SFT+L2** consistently demonstrates superior robustness and outperforms other loss combinations across varying pruning ratios

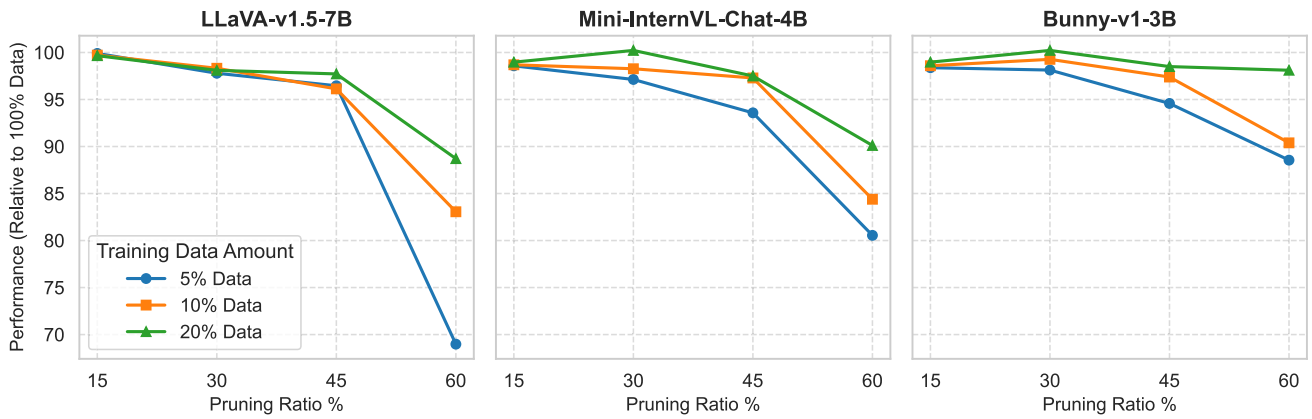

Fig. 6 Comparison of performance using different percentages of data for recovery training. For smaller compression ratios, even a small percentage of the training data (as low as 5%) is sufficient to recover most of the original performance. However, as the compression ratio increases, more training data is required to achieve higher recovery performance

over KL to its optimization behaviors. Standard KL is "mean-seeking": it forces the student to match the teacher's entire probability distribution, including low-probability tails. A heavily pruned model lacks the capacity to capture these nuances, resulting in uncertain predictions. In contrast, RKL is "mode-seeking": it penalizes the student only when it predicts high probability for an incorrect token, effectively encouraging the model to focus solely on the teacher's most likely prediction, as also noted in Gu et al. (2023). For a capacity-constrained model, focusing on the single-best answer is more efficient than trying to mimic the teacher's full complexity. Finally, L2 performs worst on its own because it aligns internal features but does not directly supervise the final classification head. Without output-level guidance, the model cannot effectively map these aligned features to correct text tokens.

Synergy between SFT and KD To combine the benefits of hard labels and soft knowledge, we experiment with joint objectives in Fig. 5, including dual combinations (e.g., SFT+KL, SFT+L2) and triple combinations (e.g., SFT+L2+KL). Interestingly, the performance trend reverses: while L2 was the weakest standalone method, **SFT+L2** consistently outperforms all other strategies, including the more complex triple combinations. This success stems from the orthogonality of the objectives: SFT optimizes the final output (the "what") while L2 acts as a regularizer for internal representations (the "how"). Adding further output-level constraints (KL or RKL) on top of SFT+L2 yields diminishing returns or slight degradation, likely due to optimization conflicts between the hard labels and soft targets. Thus, the simple combination of SFT and feature alignment (L2) provides the most robust recovery.

Takeaway Knowledge distillation, particularly when combined with SFT and using L2 loss to map the intermediate states, delivers the most effective performance recovery after pruning across all compression ratios.

4.5 Data Efficient Recovery

We investigate the trade-off between training cost (data usage) and model performance. Fig. 6 illustrates the recovery performance of widthwise-pruned models finetuned with varying fractions of the original dataset, using the optimal SFT+L2 strategy established in Section 4.4.

For mild-to-moderate compression ratios ($\leq$ 45%), we observe a rapid performance saturation. Remarkably, using just 5% of the training data allows the model to recover over 90% of the performance achieved with the full dataset. This trend holds consistently across different architectures. However, as the compression ratio increases beyond this point (e.g., 60%), the performance gap between using partial data and full data widens, indicating that heavily pruned models require more extensive training examples to recover.

These findings suggest that the nature of the recovery task shifts with pruning severity. At lower compression ratios, the model retains most of its knowledge and structural integrity. In this regime, the recovery phase primarily serves as a calibration step that realigns the remaining weights and can be performed with a small, representative subset of data. In contrast, at high compression ratios, the model suffers significant capacity loss. The recovery process becomes a re-learning task, in which the model must discover new internal pathways to compensate for the removal of neurons. This requires a larger and more diverse set of examples to generalize effectively.

Takeaway With a compression ratio below 50%, using only 5% of the dataset is sufficient to achieve performance comparable to training on the full dataset. However, for compression ratios greater than 50%, full-data training is necessary to recover performance effectively.

4.6 Key Insights for Model Compression

Based on the empirical results from the previous section, we outline the following practices for compressing LVLMs:

- **Widthwise pruning is more effective**, yielding an efficient model even without recovery training.
- **With recovery training**, layerwise pruning excels for smaller compression ratios ($\leq$ 30%), while widthwise pruning performs better at higher ratios ($\geq$ 45%).
- **For small compression ratios** ($\leq$ 15%), finetuning just the multimodal projector is often sufficient to restore performance.
- **For recovery training**, combining SFT with KD of the intermediate representations using L2 loss consistently achieves the highest performance.
- **Recovery is highly data-efficient,** requiring only 5% of the original data to match full-data training results, though full datasets are still needed for high compression ratios.

Guided by these insights, we present the comprehensive performance benchmarks of our pruned and recovered models in Table 5.

Qualitative Analysis and Failure Cases. Beyond numerical metrics, we validate the practical utility of our compressed and recovered models following our established insights through qualitative examples in Table 6. For LLaVA-v1.5-7B models compressed by up to 45% (retaining approximately 3.8B parameters), the results demonstrate that despite significant pruning, these models retain a strong capacity for understanding complex visual inputs. For instance, they

Table 5 Performance of the compressed models across various benchmarks. The Ratio indicates the proportion of LLM parameters removed. Layerwise pruning is applied for ratios below 45%, while widthwise pruning is used for ratios 45% and above

Method	Ratio	MMMU	SQA	MathVista	AI2D	MME-C	MME-P	POPE	DocVQA	GQA	AVG	AVG-%
LLaVA-v1.5-7B												
-	0.00%	35.10	68.67	26.70	54.80	363.21	1511.33	86.99	28.10	61.98	53.70	100.00%
Layerwise+FT+L2	15.00%	36.40	68.42	25.50	53.70	337.86	1442.35	86.94	27.60	61.20	52.68	98.10%
Layerwise+FT+L2	30.00%	36.00	68.82	23.40	50.50	318.57	1496.60	85.98	26.10	60.34	51.75	96.38%
Widthwise+FT+L2	45.00%	30.80	52.92	20.20	48.40	215.00	1191.17	85.74	23.70	57.74	45.10	83.99%
Widthwise+FT+L2	60.00%	27.70	46.26	18.00	46.30	211.79	1085.97	84.06	22.20	52.32	41.96	78.13%
InternVL-Chat-4B												
-	0.00%	43.20	93.30	53.70	76.90	547.50	1596.71	88.00	87.70	62.50	72.62	100.00%
Layerwise+FT+L2	15.00%	42.80	92.47	53.22	76.21	542.61	1582.42	87.21	86.92	62.01	71.98	99.12%
Layerwise+FT+L2	30.00%	41.45	89.53	51.53	73.79	525.36	1532.14	84.44	84.15	60.04	69.69	95.96%
Widthwise+FT+L2	45.00%	37.24	80.43	46.29	66.30	472.00	1376.53	75.87	75.61	53.94	62.61	86.22%
Widthwise+FT+L2	60.00%	34.29	74.06	42.63	61.05	434.62	1267.52	69.86	69.62	49.67	57.65	79.39%
Bunny-v1-3B												
-	0.00%	34.10	70.72	25.80	60.49	289.31	1487.71	87.82	22.80	54.72	51.89	100.00%
Layerwise+FT+L2	15.00%	33.00	70.00	24.20	59.80	304.29	1457.06	87.97	21.80	54.56	51.36	98.98%
Layerwise+FT+L2	30.00%	32.35	68.12	22.60	57.20	252.51	1349.91	87.53	20.80	53.08	48.97	94.37%
Widthwise+FT+L2	45.00%	29.15	63.06	21.00	55.50	244.64	1281.66	87.09	19.80	52.31	46.95	90.48%
Widthwise+FT+L2	60.00%	28.10	53.20	19.40	53.90	216.07	1115.33	86.73	18.80	48.72	43.51	83.86%

Table 6 Qualitative analysis of compressed LLaVA-v1.5-7B models following our established insights. **Bold text** indicates correctly identified visual details or counts, while *italics* denote incorrect descriptions, hallucinations, or miscounts

		 **Q1**: What is funny in the image?	 **Q2**: How many uncut fruits are in the image?
Ratio	**Strategy**	**Q1: Response 1**	**Q2: Response 2**
0%	-	The image is funny because it is **a parody of the famous Mona Lisa painting**, but with a **dog's head and paws** instead of a woman's. The dog is wearing the same **dark veil and dress** as seen in the original artwork. This is humorous because it replaces a serious and **iconic piece of art** with a cute dog, creating a silly and unexpected contrast.	There are **three uncut mangosteens** in the image. They are positioned in the background and to the left, showing their **dark purple skin and green stems**. Two other fruits have been cut open in the foreground to reveal the white flesh inside.
15%	Layerwise +FT+L2	The image is funny because it features **a dog wearing a Renaissance-style dress and a bandana**, posing as if it's **posing for a painting**. This is humorous because it is a playful representation of a dog, which is not something you would typically see in a **Renaissance-style dress**.	There are **three uncut fruits** in the picture. These are **mangosteens** with their **green caps** still attached. There are also *three* other fruits that have been sliced open to show the inside.
30%	Layerwise +FT+L2	The image is funny because it features **a dog dressed up in a human-like attire**, wearing a **long black dress** and *a hat*. This unusual scene is not something you would typically see in real life, as **dogs don't wear clothes** like humans.	There are **three uncut** *purple onions* in the background. The image also shows some **fruit that has been cut in half** on a wooden surface.
45%	widthwise +FT+L2	The image is funny because it features **a dog dressed in a human-like dress**, posing in a **pose as if it were a person**. The dog appears to be **dressed in a dress**, which is a humorous and unexpected scene, as dogs are not typically dressed like humans. **The dog's pose and the overall scene create a sense of humor and playfulness.**	There are *two* uncut fruits. They look like *pomegranates* or some kind of tropical fruit.
60%	widthwise +FT+L2	The image is funny because *there is a woman sitting in a chair.* She is **wearing a black dress** and *has long hair.* It is funny because *she* has a **serious expression on *her* face.**	There is *one large purple cabbage* in the center. *It is sitting on a wooden table next to some garlic.* The image shows a *healthy vegetable harvest.*

effectively identify key elements in the first image, such as the dog's attire, and generate contextually accurate descriptions of the humor. However, as the pruning ratio increases, fine-grained object recognition and counting gradually degrade. In the second image, while the model correctly counts three uncut fruits at 15% and 30% pruning ratios, it begins to exhibit semantic shifts, misidentifying the mangosteens as "purple onions" at 30%, and eventually miscounting them as "two pomegranates" at 45%. Furthermore, the analysis reveals a clear performance boundary at extreme pruning ratios (e.g., 60%). At this level, visual-semantic alignment degrades significantly: the model hallucinates a woman based on the global compositional structure (resembling the Mona Lisa) rather than local visual features, and completely fails the counting task by hallucinating a "large purple cabbage" and "garlic." This confirms that while our method effectively preserves broad capabilities across a wide range of sparsity levels, there is a critical threshold beyond which semantic granularity and precise visual grounding are compromised.

5 Discussion

In this section, we discuss the extent to which pruning the model makes sense (Sec. 5.2), and compare our best practices with quantization and explore their integration (Sec. 5.3). Finally, we discuss the limitations of structural pruning and recovery training as compression techniques, as well as potential directions for future work (Sec. 5.4).

5.1 Impact of Calibration Dataset Size

We investigate the sensitivity of the pruning process to the size of the calibration dataset. Fig. 7 reports the performance of LLaVA-v1.5-7B under widthwise and layerwise pruning using calibration sets ranging from 1 to 50 samples.

We observe a consistent trend where model performance improves as the calibration set size increases from 1 to 15 samples. However, beyond 10 samples, the performance effectively plateaus. For instance, with widthwise pruning at a 15% ratio, the score rises from 43.60 (1 sample) to 48.80 (15 samples) and remains constant thereafter. This saturation point is consistent across different pruning ratios and strategies, suggesting that a very small subset of data is sufficient to calculate reliable importance scores.

The rapid stabilization of performance indicates that the "importance" of specific weights or layers is a structural property of the model rather than being highly dependent on specific data examples. In large pre-trained models, redundant neurons tend to remain inactive or contribute minimally across a wide variety of inputs. Once the calibration set spans a broad range of features, adding more samples does not significantly alter the ranking of parameters deemed "safe" to prune, allowing for highly data-efficient calibration.

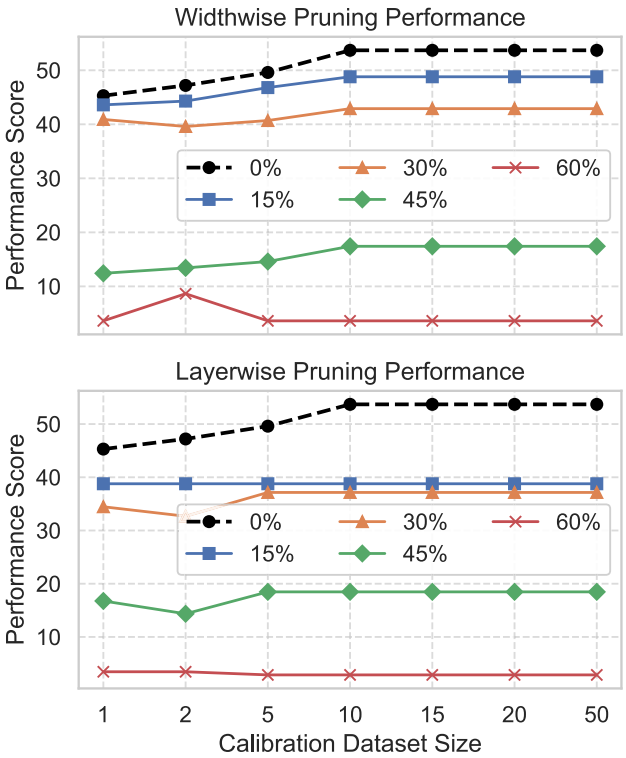

Fig. 7 Impact of calibration dataset size on zero-shot performance (LLaVA-v1.5-7B). We evaluate model performance across varying pruning ratios (15%–60%) and paradigms (Widthwise vs. Layerwise) using calibration subsets ranging from 1 to 50 samples. The results demonstrate that pruning sensitivity is largely independent of calibration size, with importance scores converging rapidly

5.2 How much can we prune the model?

A critical question in model compression is determining the maximum sparsity level that maintains acceptable performance. Table 5 summarizes the capabilities of LLaVA, InternVL, and Bunny across varying pruning ratios using our optimized recovery strategy.

The Safe Zone ($\leq$ 30%). Our results indicate that LVLMs exhibit significant redundancy. At a pruning ratio of 15%, all models retain approximately 98–99% of their original capabilities. Even at a 30% compression ratio, the performance loss is minimal. This suggests that up to 30% of the parameters in current LVLMs contribute little to the model's core reasoning and multimodal abilities.

The Tipping Point (45%). Performance degradation accelerates significantly once the pruning ratio exceeds 30%. At 45% compression, the retained performance for LLaVA and InternVL drops to 83.99% and 86.22%, respectively, falling below the 90% fidelity threshold. While Bunny remains slightly more robust (retaining 90.48%), the trend

Table 7 Memory usage and average performance of LLaVA-v1.5-7B when combining structural pruning (layerwise) and int8 quantization. The results demonstrate that these complementary techniques can be seamlessly combined to achieve compounded memory reductions

Model	Memory (GiB)	Ratio	Avg
LLaVA-7B	13.2	0%	53.7
LLaVA-7B.int8()	7.5	0%	52.5
LLaVA-6B	11.3	15%	52.68
LLaVA-6B.int8()	6.5	15%	51.9
LLaVA-5B	9.4	30%	51.75
LLaVA-5B.int8()	5.4	30%	50.8

is clear: aggressive pruning beyond this point compromises the model's structural integrity.

Consequently, if the goal is to preserve high fidelity ($>$ 90% performance), the compression ratio should generally be kept below 30%. Within this range, pruning offers significant efficiency gains with negligible impact on multimodal reasoning; beyond it, users must accept a trade-off between higher efficiency and noticeable performance degradation.

5.3 Combination with Quantization

While our study focuses primarily on structural pruning, integrating quantization into our framework can further optimize memory efficiency. Here, we demonstrate that pruning and quantization are highly complementary techniques.

Using LLM.int8() (Dettmers et al., 2022) as a representative method, Table 7 shows that applying quantization to our already-pruned models yields compound memory savings. For instance, quantizing the 30% pruned LLaVA-5B model reduces the total memory footprint to just 5.4 GiB (down from the 13.5 GiB baseline) while maintaining a strong average performance score of 50.83. This confirms that our structured pruning framework can be seamlessly paired with quantization to maximize memory efficiency without severe performance degradation.

5.4 Limitation and Future Work

Our experiments demonstrate the effectiveness of structural pruning with recovery training at moderate compression ratios (up to 30%). However, our study has boundaries. First, we focused exclusively on structured pruning for hardware universality; we did not investigate unstructured or semi-structured methods (Frantar and Alistarh, 2023; Sun et al., 2024), which represent an orthogonal direction for compression requiring different hardware considerations. Second, we kept the vision encoder frozen. While motivated by the parameter imbalance shown in Table 1, future holistic compression frameworks could explore pruning the vision backbone, particularly for architectures where the vision encoder is larger. Finally, beyond this threshold, performance loss becomes increasingly difficult to recover, suggesting that for applications requiring more aggressive compression, the extreme pruning of a large model is not a viable approach. Due to computational constraints, this work focuses on two pruning techniques applied to three different models. Future work could extend these findings to include a broader range of pruning techniques and models, further refining these strategies.

6 Conclusion

We systematically evaluated two structural pruning schemes, widthwise and layerwise, on LLaVA-v1.5-7B, InternVL-Chat-4B, and Bunny-v1-3B, and paired them with lightweight recovery through supervised finetuning and knowledge distillation. From these experiments, we distilled a decision chart that guides practitioners in choosing the pruning route and recovery budget for different target compression ratios. Our findings provide a concrete path to fit LVLMs within strict memory, compute, or energy budgets without surrendering performance.

Funding Open Access funding enabled and organized by Projekt DEAL.

Data Availability The data used in this study are derived from publicly available datasets. The original visual instruction tuning datasets–LLaVA-v1.5-mix665k (Liu et al., 2024), InternVL-Chat-V1-2-SFT-Data (Chen et al., 2024), and Bunny-695K (He et al., 2024), were utilized for pruning calibration and recovery training. Evaluation was conducted using the lmms-eval (Bo et al., 2024) suite across several public benchmarks, including MMMU (Yue et al., 2024), SQA (Lu et al., 2022), MathVista (Lu et al., 2023), AI2D, POPE (Li et al., 2023), MME (Yin et al., 2023), GQA (Hudson and Manning, 2019), and DocVQA (Mathew et al., 2021). The code and pruned model checkpoints will be made available on GitHub upon publication.

References

Bo, L., Peiyuan, Z., Kaichen, Z., Fanyi, P., Xinrun, D., Yuhao, D., Haotian, L., Yuanhan, Z., Ge, Z., Chunyuan, L., & Ziwei, L. (2024). LMMs-Eval: Accelerating the Development of Large Multimoal Models. Zenodo. https://github.com/EvolvingLMMs-Lab/lmms-eval.

Cai, Y., Zhang, J., He, H., He, X., Tong, A., Gan, Z., Wang, C., Xue, Z., Liu, Y., & Bai, X. (2025). Llava-kd: A framework of distilling multimodal large language models. *Proceedings of the IEEE/CVF international conference on computer vision*, (pp. 239–249).

Chen, X., Hu, Y., Zhang, J., Wang, Y., Li, C., & Chen, H. (2024). Streamlining redundant layers to compress large language models. *The thirteenth international conference on learning representations*.

Chen, Z., Wu, J., Wang, W., Su, W., Chen, G., Xing, S., Zhong, M., Zhang, Q., Zhu, X., Lu, L., and others. (2024). Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, (pp. 24185–24198)

Chu, X., Qiao, L., Lin, X., Xu, S., Yang, Y., Hu, Y., Wei, F., Zhang, X., Zhang, B., Wei, X., and others. (2023). Mobilevlm: A fast, reproducible and strong vision language assistant for mobile devices. arXiv:2312.16886

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., and others. (2021). Training verifiers to solve math word problems. arXiv:2110.14168.

Dery, L., Kolawole, S., Kagy, J.-F., Smith, V., Neubig, G., & Talwalkar, A. (2024). Everybody prune now: Structured pruning of LLMs with only forward passes. arXiv:2402.05406.

Dettmers, T., Lewis, M., Belkada, Y., & Zettlemoyer, L. (2022). Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems, 35*, 30318–30332.

Dong, X., Chen, S., & Pan, S.J. (2017). Learning to prune deep neural networks via layer-wise optimal brain surgeon. arXiv:1705.07565.

Fan, A., Grave, E., & Joulin, A. (2019). Reducing transformer depth on demand with structured dropout. arXiv:1909.11556.

Fang, G., Ma, X., Song, M., Mi, M.B., & Wang, X. (2023). Depgraph: Towards any structural pruning. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, (pp. 16091–16101) .

Farina, M., Mancini, M., Cunegatti, E., Liu, G., Iacca, G., & Ricci, E. (2024). Multiflow: Shifting towards task-agnostic vision-language pruning. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, (pp. 16185–16195).

Frankle, J., & Carbin, M. (2019). The lottery ticket hypothesis: Finding sparse, trainable neural networks. arXiv:1803.03635.

Frantar, E., & Alistarh, D. (2023). Sparsegpt: Massive language models can be accurately pruned in one-shot. *International conference on machine learning*, PMLR. (pp. 10323–10337).

Gou, J., Yu, B., Maybank, S. J., & Tao, D. (2021). Knowledge distillation: A survey. *International Journal of Computer Vision, 129*(6), 1789–1819. https://doi.org/10.1007/s11263-021-01453-z

Gu, Y., Dong, L., Wei, F., & Huang, M. (2023). Knowledge distillation of large language models. arXiv:2306.08543.

He, M., Liu, Y., Wu, B., Yuan, J., Wang, Y., Huang, T., & Zhao, B. (2024). Efficient multimodal learning from data-centric perspective. arXiv:2402.11530.

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. arXiv:1503.02531.

Hoefler, T., Alistarh, D., Ben-Nun, T., Dryden, N., & Peste, A. (2021). Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *Journal of Machine Learning Research, 23*, 1–124.

Hoffmann, J., Agnihotri, S., Saikia, T., & Brox, T. (2021). Towards improving robustness of compressed cnns. *ICML workshop on uncertainty and robustness in deep learning (UDL)*, vol. 4.

Holtzman, A., Buys, J., Du, L., Forbes, M., & Choi, Y. (2019). The curious case of neural text degeneration. arXiv:1904.09751.

Hsieh, C.-Y., Li, C.-L., Yeh, C.-K., Nakhost, H., Fujii, Y., Ratner, A., Krishna, R., Lee, C.-Y., & Pfister, T. (2023). Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. arXiv:2305.02301

Hu, E.J., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., and others. (2021). Lora: Low-rank adaptation of large language models. *International conference on learning representations*.

Huang, Y., Thede, L., Mancini, M., Xu, W., & Akata, Z. (2025). Investigating structural pruning and recovery techniques for compressing multimodal large language models: An empirical study. *DAGM german conference on pattern recognition*, Springer. (pp. 320–336).

Hudson, D.A., & Manning, C.D. (2019). Gqa: A new dataset for real-world visual reasoning and compositional question answering. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, (pp. 6700–6709).

Jiang, F. (2024). Identifying and mitigating vulnerabilities in llm-integrated applications. Master's thesis, University of Washington.

Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., & Liu, Q. (2020). TinyBERT: Distilling BERT for natural language understanding. arXiv:1909.10351.

Karamcheti, S., Nair, S., Balakrishna, A., Liang, P., Kollar, T., & Sadigh, D. (2024). Prismatic vlms: Investigating the design space of visually-conditioned language models. arXiv:2402.07865.

Kembhavi, A., Salvato, M., Kolve, E., Seo, M., Hajishirzi, H., & Farhadi, A. (2016). A diagram is worth a dozen images. *European conference on computer vision*, Springer. (pp. 235–251).

Kim, J., Kim, K., Seo, S., & Park, C. (2025). Compodistill: Attention distillation for compositional reasoning in multimodal llms. arXiv:2510.12184.

Lee, N., Ajanthan, T., Gould, S., & Torr, P.H.S. (2020). A signal propagation perspective for pruning neural networks at initialization. arXiv:1906.06307.

Li, Y., Du, Y., Zhou, K., Wang, J., Zhao, W.X., & Wen, J.-R. (2023). Evaluating object hallucination in large vision-language models. arXiv:2305.10355.

Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H.P. (2017). Pruning filters for efficient convnets. *International conference on learning representations*.

Liang, K.J., Hao, W., Shen, D., Zhou, Y., Chen, W., Chen, C., & Carin, L. (2021). MixKD: Towards efficient distillation of large-scale language models. arXiv:2011.00593.

Liu, H., Li, C., Li, Y. & Lee, Y.J. (2024). Improved baselines with visual instruction tuning. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, (pp. 26296–26306).

Liu, L., Zhang, S., Kuang, Z., Zhou, A., Xue, J.-H., Wang, X., Chen, Y., Yang, W., Liao, Q., & Zhang, W. (2021). Group fisher pruning for practical network compression. arXiv:2108.00708

Liu, Z., Zhao, C., Iandola, F., Lai, C., Tian, Y., Fedorov, I., Xiong, Y., Chang, E., Shi, Y., Krishnamoorthi, R., and others. (2024). Mobilellm: Optimizing sub-billion parameter language models for on-device use cases. *Forty-first international conference on machine learning*.

Lu, P., Bansal, H., Xia, T., Liu, J., Li, C., Hajishirzi, H., Cheng, H., Chang, K.-W., Galley, M., & Gao, J. (2023). Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. arXiv:2310.02255

Lu, P., Mishra, S., Xia, T., Qiu, L., Chang, K.-W., Zhu, S.-C., Tafjord, O., Clark, P., & Kalyan, A. (2022). Learn to explain: Multimodal reasoning via thought chains for science question answering.

Advances in Neural Information Processing Systems, 35, 2507–2521.

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

Mathew, M., Karatzas, D., & Jawahar, C.(2021). Docvqa: A dataset for vqa on document images. *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, (pp. 2200–2209).

McCarley, J., Chakravarti, R., & Sil, A. (2019). Structured pruning of a bert-based question answering model. arXiv:1910.06360.

Men, X., Xu, M., Zhang, Q., Wang, B., Lin, H., Lu, Y., Han, X., & Chen, W. (2024). Shortgpt: Layers in large language models are more redundant than you expect. arXiv:2403.03853

Michel, P., Levy, O., & Neubig, G. (2019). Are sixteen heads really better than one? *Advances in neural information processing systems,32*.

Muralidharan, S., Turuvekere Sreenivas, S., Joshi, R., Chochowski, M., Patwary, M., Shoeybi, M., Catanzaro, B., Kautz, J., & Molchanov, P. (2024). Compact language models via pruning and knowledge distillation. *Advances in Neural Information Processing Systems, 37*, 41076–41102.

Park, S., Lee, J., Mo, S., & Shin, J. (2020). Lookahead: A far-sighted alternative of magnitude-based pruning. arXiv:2002.04809.

Popp, N., Metzen, J.H., & Hein, M. (2024). Zero-shot distillation for image encoders: How to make effective use of synthetic data. arXiv:2404.16637.

Sajjad, H., Dalvi, F., Durrani, N., & Nakov, P. (2023). On the effect of dropping layers of pre-trained transformer models. *Computer Speech & Language, 77*, 101429.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2020). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter . arXiv:1910.01108.

Sanh, V., Wolf, T., & Rush, A.M. (2020). Movement pruning: Adaptive sparsity by fine-tuning. arXiv:2005.07683.

Shang, Y., Cai, M., Xu, B., Lee, Y.J., & Yan, Y. (2025). Llava-prumerge: Adaptive token reduction for efficient large multimodal models. *Proceedings of the IEEE/CVF international conference on computer vision*, (pp. 22857–22867).

Song, J., Oh, K., Kim, T., Kim, H., Kim, Y., and others. (2024). Sleb: Streamlining llms through redundancy verification and elimination of transformer blocks. *Forty-first international conference on machine learning*.

Sreenivas, S.T., Muralidharan, S., Joshi, R., Chochowski, M., Mahabaleshwarkar, A.S., Shen, G., Zeng, J., Chen, Z., Suhara, Y., Diao, S., and others. (2024). Llm pruning and distillation in practice: The minitron approach. arXiv:2408.11796.

Sun, S., Cheng, Y., Gan, Z., & Liu, J. (2019). Patient knowledge distillation for BERT model compression. arXiv:1908.09355.

Sun, M., Liu, Z., Bair, A., & Kolter, J.Z. (2024). A simple and effective pruning approach for large language models. *The twelfth international conference on learning representations*.

Tang, Z., Ma, Z., Wang, S., Li, Z., Zhang, L., Zhao, H., Li, Y., & Wang, Q. (2025). Covipal: Layer-wise contextualized visual token pruning for large vision-language models. *Findings of the association for computational linguistics: EMNLP 2025*, (pp. 20701–20714).

Telgarsky, M. (2016). Benefits of depth in neural networks. *Conference on learning theory (COLT)*, (pp. 1517–1539).

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

Voita, E., Talbot, D., Moiseev, F., Sennrich, R., & Titov, I. (2019). Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *Proceedings of the 57th annual meeting of the association for computational linguistics*, (pp. 5797–5808).

Wang, W., Bao, H., Huang, S., Dong, L., & Wei, F. (2021). MiniLMv2: Multi-head self-attention relation distillation for compressing pretrained transformers. arXiv:2012.15828.

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. arXiv:2002.10957.

Wang, H., Xu, Y., Xu, Z., Gao, J., Liu, Y., Hu, W., Wang, K., & Zhang, Z. (2025). Autoprune: Each complexity deserves a pruning policy. arXiv:2509.23931.

Xia, M., Gao, T., Zeng, Z., & Chen, D. (2024). Sheared LLaMA: Accelerating language model pre-training via structured pruning. arXiv:2310.06694.

Yang, C., An, Z., Huang, L., Bi, J., Yu, X., Yang, H., Diao, B., & Xu, Y. (2024). Clip-kd: An empirical study of clip model distillation. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, (pp. 15952–15962).

Yin, S., Fu, C., Zhao, S., Li, K., Sun, X., Xu, T., & Chen, E. (2023). A survey on multimodal large language models. arXiv:2306.13549.

You, Z., Yan, K., Ye, J., Ma, M., & Wang, P. (2019). Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks. arXiv:1909.08174.

Yue, X., Ni, Y., Zhang, K., Zheng, T., Liu, R., Zhang, G., Stevens, S., Jiang, D., Ren, W., Sun, Y., Wei, C., Yu, B., Yuan, R., Sun, R., Yin, M., Zheng, B., Yang, Z., Liu, Y., Huang, W., Sun, H., Su, Y., & Chen, W. (2024). Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. *Proceedings of CVPR*.

Zhang, C., Bengio, S., & Singer, Y. (2022). Are all layers created equal? *Journal of Machine Learning Research, 23*(67), 1–28.

Zhou, A., Ma, Y., Zhu, J., Liu, J., Zhang, Z., Yuan, K., Sun, W., & Li, H. (2021). Learning n: M fine-grained structured sparse neural networks from scratch. *International conference on learning representations*.

Zhu, M., Zhu, Y., Liu, X., Liu, N., Xu, Z., Shen, C., Peng, Y., Ou, Z., Feng, F., & Tang, J. (2024). A comprehensive overhaul of multimodal assistant with small language models. arXiv:2403.06199

Zhu, Y., Zhu, M., Liu, N., Xu, Z., & Peng, Y. (2024). Llava-phi: Efficient multi-modal assistant with small language model. *Proceedings of the 1st international workshop on efficient multimedia computing under limited*, (pp. 18–22).