LoGIC: Multi-LoRA Guided Importance Consensus for Multi-Task Pruning in Vision Transformers

Yu-Hong Chou*, **Rui Fang***, **Hsi-Wen Chen, Ming-Syan Chen**

Department of Electrical Engineering, National Taiwan University
{yhchou, rfang, hwchen}@arbor.ee.ntu.edu.tw, mschen@ntu.edu.tw

Abstract

Deploying Vision Transformers (ViTs) in real-world multi-task learning remains challenging due to their massive computational costs and the difficulty of pruning shared backbones without harming task performance. Single-task pruning often causes destructive interference by discarding weights critical to other tasks, while existing multi-task pruning strategies remain costly and unscalable for billion-parameter models. We propose **Multi-LoRA Guided Importance Consensus (LoGIC)**, a unified framework for efficient and robust multi-task ViT pruning. LoGIC follows a two-phase procedure: *(i) task-consistent pruning of LoRA modules*, guided by a task-adaptive gating mechanism that balances shared and task-specific contributions while enforcing structured sparsity for deployment; and *(ii) cross-task consensus pruning* of the frozen ViT backbone, which retains both universally shared and task-specialized capabilities, enabling aggressive sparsity without sacrificing accuracy. Across five diverse vision benchmarks, LoGIC achieves up to 50% structured sparsity while maintaining competitive accuracy and surpassing all baselines.

Code — https://github.com/AnderStudio/LoGIC

Introduction

Vision Transformers (ViTs) (Dosovitskiy et al. 2020a; Liu et al. 2021; Arnab et al. 2021) have shown strong performance across diverse vision tasks, including image captioning (Li et al. 2023), object detection (Zhang et al. 2021), and semantic segmentation (Zhang et al. 2022). This progress stems largely from pretraining large-scale ViT backbones on extensive datasets, then adapting them via fine-tuning (Tay et al. 2021; Goyal et al. 2023). However, fully fine-tuning for each task is computationally costly and storage-inefficient, especially in multi-task settings that require deploying many task-specific variants. To address this, parameter-efficient tuning methods such as Low-Rank Adaptation (LoRA) (Hu et al. 2022; Zhu et al. 2024; Yang et al. 2023b) have gained traction. LoRA employs low-rank matrices to encode task-specific knowledge while freezing backbone parameters, allowing a single backbone to support multiple tasks. This design balances efficiency and adaptability, making it especially effective for multi-task learning.

To balance accuracy and efficiency in LoRA, a key challenge is choosing its rank (Albert et al. 2025), which is usually fixed beforehand and often leads to empirically suboptimal choices (Xie and Liao 2023; Ahmed et al. 2025). Pruning (Parikh et al. 2024; Yang et al. 2023a; Xu et al. 2024) offers a natural remedy by iteratively removing redundancy to improve inference efficiency. Recent work prunes LoRA using parameter or output based signals (He et al. 2022; Zhang et al. 2023; Zhou et al. 2025), showing that only a fraction of weight in LoRA contributes meaningfully to predictions. However, these methods focus mainly on LoRA and overlook its interaction with the backbone. In practice, the backbone typically contains 10 to even 100 times more parameters than LoRA, making backbone pruning far more important, yet LoRA's task specific signals and structural coupling make pruning the backbone much more challenging.

More recently, LoRAPrune (Zhang et al. 2024) integrates LoRA into pruning to retain task-relevant backbone parameters, but its reliance on a single LoRA limits applicability in multi-task settings. Extending it to multiple tasks introduces cross-task dependencies: different LoRAs impose conflicting sparsity patterns on the shared backbone (Sun et al. 2022; Xiang et al. 2024; Jeong et al. 2021), and pruning based on one task's signal may remove parameters vital for others, leading to unpredictable cross-task degradation. We theoretically show that pruning in multi-LoRA ViTs induces inherent *structure dependency* between the backbone and all LoRA modules, and that this further escalates into *task dependency* in multi-task settings, where adapters can interfere with one another through the shared backbone. In our preliminary experiment, pruning the backbone using the image-classification LoRA and reusing it for semantic segmentation led to a 12.3% performance drop.

To address these challenges, we propose the **Multi-LoRA Guided Importance Consensus (LoGIC)**, a unified framework for pruning ViTs in multi-task settings. LoGIC operates in two phases. In the first phase, we prune LoRA modules to remove redundant parameters. Allowing each task to define its own structured pruning mask causes (i) inconsistent sparsity patterns across tasks, hindering deployment, and (ii) redundant parameters due to missed cross-task sharing. We therefore enforce *task-consistent pruning masks*

*These authors contributed equally.

within each layer to enable deployment-friendly structured sparsity. Because uniform sparsity may limit task-specific flexibility, we introduce a *task-adaptive gating mechanism* that learns soft combinations of shared and task-specific LoRA paths, balancing knowledge sharing with specialization and providing pruning signals to reduce interference and redundancy. In the second phase, pruning the shared ViT backbone poses two further challenges: (iii) parameters redundant for one task may be essential for others, and (iv) identifying universally useful parameters is difficult. LoGIC addresses this by unifying *LoRA-guided gradient sensitivity*, *routing-based task usage*, and *LoRA adaptation magnitude* into a consensus-based score, ensuring pruning removes only parameters consistently unimportant across tasks while preserving shared and task-specific representations.

- We propose **LoGIC**, the first unified pruning framework that jointly optimizes a sparsified shared ViT backbone and interchangeable LoRA modules, enabling scalable multi-task deployment under resource constraints.
- LoGIC employs a *shared pruning mask with task-wise gating* to enforce consistent sparsity while preserving task-specific adaptability, and a *consensus-based backbone pruning strategy* that unifies multi-task signals to retain both universal and task-critical parameters.
- We conduct extensive experiments on five vision tasks using two large-scale ViT backbones. Results show that LoGIC achieves full-precision performance with only 50% parameter utilization, without any loss in accuracy.

Related Work

While pruning can improve efficiency by focusing on impactful parameters, multi-task settings pose unique challenges, as tasks often rely on different parameter subsets, rendering single-task strategies suboptimal (Ye et al. 2023; He et al. 2021). To address this, DiSparse (Sun et al. 2022) introduces a disentangled pruning framework that computes task-specific saliency scores and prunes shared parameters only when unanimously deemed unimportant. Although effective in mitigating performance degradation, its unanimity rule limits scalability as the number of tasks grows. AdapMTL (Xiang et al. 2024) extends this approach by jointly learning weights and structured masks via differentiable thresholding, employing separate shared and task-specific thresholds alongside adaptive task reweighting (Chen et al. 2018) to handle imbalance. Unlike DiSparse, AdapMTL trains from scratch, making it more suitable for resource-constrained deployments. Nevertheless, most multi-task pruning methods still rely on full-model fine-tuning, which incurs high training overhead and undermines parameter efficiency in large Transformers. Moreover, many importance estimates remain heuristic or gradient-based (Xiang et al. 2024; Sun et al. 2022), failing to fully capture task-disentangled signals.

To address the inefficiency of full fine-tuning, recent work has explored parameter-efficient strategies, with Low-Rank Adaptation (LoRA) (Hu et al. 2022) becoming widely adopted in multi-task learning (Chen et al. 2023; Fu et al. 2023). To further cut adaptation costs, several methods prune LoRA modules directly, using heuristics (He et al. 2022), singular value pruning (Zhang et al. 2023), gating (Ding et al. 2023), or output-based criteria (Zhou et al. 2025) to remove redundant parameters. However, these approaches focus exclusively on LoRA, overlooking substantial redundancy within the backbone. LoRAPrune (Zhang et al. 2024) addresses this by estimating task-specific signals from LoRA to identify and prune the backbone parameters. Yet, it remains limited in multi-task contexts, where different LoRA branches activate distinct backbone regions, creating conflicts and inefficiencies.

Problem Formulation

Here, we present the pruning problem for Vision Transformers with multiple LoRA modules in a multi-task setting.

Low-rank Adaptation. To efficiently adapt pretrained ViTs for multiple tasks, we employ LoRAs (Hu et al. 2022), which constrain parameter updates to a low-rank subspace while keeping the pretrained weights frozen. During fine-tuning, the backbone parameters are excluded from gradient updates, while the inserted LoRA modules remain trainable.

Consider a pretrained ViT with layers $\ell \in \{1, \dots, L\}$, where each layer is parameterized by weights $W_\ell \in \mathbb{R}^{d\times k}$. For each task $t \in \{1, \dots, T\}$, we attach a task-specific LoRA module to layer ℓ, defined as follows.

$$\Delta W_\ell^{(t)} = B_\ell^{(t)} {A_\ell^{(t)}}^\top, \quad A_\ell^{(t)} \in \mathbb{R}^{k\times r},\ B_\ell^{(t)} \in \mathbb{R}^{d\times r},$$

where $r \ll \min(d, k)$. The resulting effective weight for task t at layer ℓ is $W_{\ell,\text{eff}}^{(t)} = W_\ell + \Delta W_\ell^{(t)}$. Given an input $x^{(t)}$, the prediction for task t is obtained through the full forward pass across all layers,

$$\hat{y}^{(t)} = f_t\Big(x^{(t)};\, \{W_{\ell,\text{eff}}^{(t)}\}_{\ell=1}^{L}\Big),$$

where f_t denotes the task-specific classifier. The training objective for task t is then defined as follows.

$$\min_{\{A_\ell^{(t)}, B_\ell^{(t)}\}_{\ell=1}^{L}} \mathcal{L}^{(t)}\Big(\hat{y}^{(t)}, y^{(t)}\Big).$$

Each ViT layer is trained with its corresponding LoRA weights $(A_\ell^{(t)}, B_\ell^{(t)})$, enabling efficient adaptation while keeping the pretrained backbone frozen. However, the layer-wise placement of adapters introduces additional pruning dependencies (Zhang et al. 2024): sparsity decisions in one layer can reduce the effectiveness of adapters in subsequent layers, leading to inter-layer coupling. Moreover, within each layer, pruning backbone and adapter components independently risks disrupting their mutual reliance, creating intra-layer coupling. Therefore, an effective pruning method must jointly consider both backbone and adapter components to sustain efficiency while preserving task performance in multi-task settings.

Layer-wise Pruning with Multi-LoRA. While LoRA enables efficient multi-task adaptation, it also poses significant challenges for pruning. Each layer integrates both the pretrained backbone and multiple task-specific LoRA modules,

so pruning solely the backbone may render certain adapters ineffective, whereas pruning adapters in isolation risks undermining cross-task consistency (Zhang et al. 2024).

Specifically, we introduce two pruning mask types per layer ℓ: a backbone mask M_{W_ℓ} and task-specific adapter masks $M_{A^{(t)}\ell}, MB_\ell^{(t)}$. For the backbone, we define

$$\tilde{W}_\ell = W_\ell \odot M_{W_\ell}\text{ , where } M_{W_\ell} \in \{0,1\}^{d\times k}.$$

For each task t, one may apply a LoRA mask $M_{\Delta W_\ell^{(t)}}$, but this is less efficient as it ignores LoRA's low-rank structure. Hence, we prune the LoRA update in a factorized manner,

$$\Delta\tilde{W}_\ell^{(t)} = \left(B_\ell^{(t)} \odot M_{B_\ell^{(t)}}\right)\left(A_\ell^{(t)} \odot M_{A_\ell^{(t)}}\right)^\top,$$

where $M_{A_\ell^{(t)}} \in \{0,1\}^{k\times r}$ and $M_{B_\ell^{(t)}} \in \{0,1\}^{d\times r}$. Thus, the pruned layer becomes $\tilde{W}_{\ell,\text{eff}}^{(t)} = \tilde{W}_\ell + \Delta\tilde{W}_\ell^{(t)}$. Here, M_{W_ℓ} controls sparsity in the shared backbone, while $M_{A_\ell^{(t)}}$ and $M_{B_\ell^{(t)}}$ enable fine-grained pruning within each LoRA adapter. This design avoids applying a coarse binary mask directly to the full low-rank update, instead pruning its factorized components for more structured compression. Then, the pruning objective is defined as follows.

$$\min_{\{M_{W_\ell}, M_{A_\ell^{(t)}}, M_{B_\ell^{(t)}}\}_{\ell=1}^L} \mathcal{L}^{(t)}\left(\hat{y}^{(t)}, y^{(t)}\right) + \lambda_W\,\theta_W + \lambda_\Delta\,\theta_\Delta^{(t)}.$$

Each ViT layer maintains a shared backbone mask and task-specific adapter masks. The backbone sparsity penalty is $\theta_W = \sum_{\ell=1}^L \|M_{W_\ell}\|_0$, while the adapter penalty for task t is $\theta_\Delta^{(t)} = \sum_{\ell=1}^L (\|M_{A_\ell^{(t)}}\|_0 + \|M_{B_\ell^{(t)}}\|_0)$. Here, M_{W_ℓ} enforces structural sparsity shared across tasks, and $(M_{A_\ell^{(t)}}, M_{B_\ell^{(t)}})$ allow task-specific fine-grained pruning. These masks are interdependent: pruning in the backbone constrains the capacity of LoRA modules, creating layer-wise dependencies between shared and task-specific components.

Naive weight pruning with arbitrary patterns often leads to infeasible GPU deployment. Thus, we adopt structural pruning (Zhang et al. 2024; Liu et al. 2018; Molchanov et al. 2019), which removes entire rows or columns of weight matrices to ensure hardware-friendly sparsity. Beyond computational efficiency, this approach preserves the low-rank structure intrinsic to LoRA, better aligning parameter distributions. Intuitively, it reduces the effective rank of the backbone to encode shared information, while allowing LoRA to retain task-specific adaptations.

Theoretical Motivation

Here, we first analyze the structural dependency arising when pruning the backbone within LoRA, and then extend the analysis to task dependency in the multi-LoRA setting.

Consider a linear layer with a backbone $W \in \mathbb{R}^{d\times k}$ and a LoRA update $\Delta W = BA^\top$, where $A \in \mathbb{R}^{k\times r}$ and $B \in \mathbb{R}^{d\times r}$. Given inputs $X \in \mathbb{R}^{k\times n}$ and targets $Y \in \mathbb{R}^{d\times n}$, we seek to minimize the empirical squared loss as follows.

$$\| Y - (W + \Delta W)X \|_F^2.$$

Define $Z := YX^\top(XX^\top)^{-1}$, which reduces to $Z = YX^\top$ under the whitening assumption $XX^\top = I$. Then the optimization problem is equivalent to

$$\min_{W\in\mathcal{S}(S),\ \Delta W\in\mathcal{R}(r)} \| Z - W - \Delta W \|_F^2, \tag{1}$$

where the feasible sets correspond to structural pruning and low-rank constraints $\mathcal{S}(S) = \{\, W : \text{supp}(W) \subseteq S\}$ and $\mathcal{R}(r) = \{\, \Delta W : \text{rank}(\Delta W) \le r \,\}$.[1]

Theorem 1 (Structure Dependency)**.** *For the problem in* (1)*, the following hold: If* $(W^*, \Delta W^*)$ *is a global minimizer of* (1)*, then*

$$W^* = \mathsf{P}_{\mathcal{S}(S)}(Z - \Delta W^*), \quad \Delta W^* = \mathsf{T}_r(Z - W^*),$$

where $\mathsf{P}_{\mathcal{S}(S)}$ *denotes orthogonal projection onto* $\mathcal{S}(S)$ *and* $\mathsf{T}_r(\cdot)$ *is the best rank-*r *approximation (truncated SVD). Conversely, if* $(W', \Delta W')$ *satisfies these two equalities, then it is a block-wise (coordinatewise) minimizer of* (1) *and hence a stationary point. Moreover, if the truncated SVD at rank* r *is unique (i.e.,* $\sigma_r(Z - W') > \sigma_{r+1}(Z - W')$*) and the projection onto* $\mathcal{S}(S)$ *is unique, then* $(W', \Delta W')$ *is a strict local minimizer.*

Theorem 1 demonstrates that pruning and LoRA are inherently coupled, since the regression target Z is decomposed into a sparse backbone W and a low-rank component ΔW, each required to explain the residual the other cannot. Pruning ΔW (e.g., reducing rank r) increases the residual $Z - \Delta W$, forcing W to use more of its support S, while pruning W (shrinking S) enlarges $Z - W$, requiring ΔW to assume greater expressive load. Consequently, the optimal solution depends jointly on both components. Structural pruning can thus be aligned with LoRA's low-rankness: W captures entries LoRA cannot efficiently represent, while ΔW offsets dimensions pruned from the backbone. This interdependence shows why pruning strategies must be jointly designed rather than applied to W and ΔW in isolation.

Then, we generalize Theorems 1 into a multi-task setting.

Theorem 2 (Task Dependency)**.** *With a shared backbone and task weights* ρ_t*, pruning a single task* t *(e.g., lowering its LoRA rank) causes a backbone shift and cross-task changes that scale linearly with* ρ_t*:*

$$\|W' - W^*\|_F \;\lesssim\; \rho_t\, \|\Delta W'^{(t)} - \Delta W^{*(t)}\|_F,$$

and

$$\|\Delta W'^{(s)} - \Delta W^{*(s)}\|_F \;\lesssim\; \kappa_s\, \rho_t\, \|\Delta W'^{(t)} - \Delta W^{*(t)}\|_F.$$

Here $\kappa_s \ge 1$ *is a task-specific sensitivity scalar (larger when task* s *is less spectrally separated). Thus, even a modest change for one task propagates to* all *others, with severity proportional to* ρ_t *and amplified by* κ_s*.*

[1]The backbone support set $\mathcal{S}(S)$ can be realized using a binary mask M_W applied to the backbone weights, i.e., $\tilde{W} = M_W \odot W$. Likewise, the low-rank constraint $\mathcal{R}(r)$ is enforced through structured pruning of the LoRA factors, i.e., $\Delta\tilde{W} = (M_B \odot B)(M_A \odot A)^\top$, where the effective rank is determined by the number of active columns retained by M_A and M_B.

Theorem 2 shows that pruning the LoRA component of a single task t (e.g., by reducing its rank r_t) does not remain confined to that task but propagates to the shared backbone and, indirectly, to all other tasks. The magnitude of the backbone shift scales with the task's weight ρ_t, while the resulting changes in other tasks' LoRA modules are further amplified by their sensitivity factors κ_s. Intuitively, when one task's representational capacity is reduced, the backbone compensates, thereby altering the residuals available to the other tasks. Tasks with larger κ_s (i.e., less spectral separation) are especially vulnerable to these effects, often suffering greater accuracy degradation.

In summary, Theorem 1 shows that pruning one component affects the other's optimal solution, while Theorem 2 reveals that pruning one task influences others via the shared backbone. These insights motivate LoGIC's design: it prunes adapters first to reduce redundancy, uses task routing for usage signals, and applies a consensus score for backbone pruning.

Method

In the following, we introduce **Multi-LoRA Guided Importance Consensus (LoGIC)**, a unified pruning framework for multi-task ViTs equipped with multiple LoRA adapters. As illustrated in Figure 1, we adopt a **Multi-LoRA architecture** (Agiza, Neseem, and Reda 2024; He, Duan, and Zhu 2025), where each linear layer is augmented with one shared LoRA module for cross-task knowledge transfer and T task-specific adapters for specialized corrections. While LoRA inherently offers strong parameter efficiency ($r \ll \min(d, k)$), pruning in multi-task settings remains challenging due to (i) structural dependencies between LoRA modules and the backbone, caused by conflicting sparsity patterns in task-specific adapters that complicate integration and deployment, and (ii) cross-task interference, where pruning shared backbone parameters using single-task signals risks discarding weights critical for other tasks. To address these challenges, LoGIC integrates two complementary components: structured LoRA pruning with task-consistent masks to reduce redundancy, and backbone refinement guided by multi-task signals from LoRA to balance pruning effectiveness with robust task performance.

Task-Consistent Pruning with Multi-LoRA

First, we prune the LoRA modules using *progressive pruning with task-consistent masks*, progressively reducing redundancy while ensuring hardware-friendly structured sparsity. Then, we introduce a *task-adaptive gating mechanism* that is jointly used to train and prune LoRA modules, dynamically balancing shared and task-specific contributions while providing reliable pruning signals.

Progressive Pruning for Task-Specific LoRA. A key challenge in multi-task pruning is achieving high sparsity without degrading accuracy. Simply pruning once often leads to irreversible accuracy loss, as the model cannot adapt to the removed parameters. Moreover, pruning decisions must contend with two forms of coupling: *inter-layer coupling*, where sparsity choices in earlier layers reduce the effectiveness of adapters in subsequent layers, and *intra-layer coupling*, where the utility of a LoRA branch depends on which backbone or other adapter parameters are preserved within the same layer. To address these challenges, we adopt *progressive pruning*, an iterative procedure that alternates between pruning and fine-tuning. This approach progressively removes redundant parameters while allowing the remaining modules to adapt, thereby alleviating both inter- and intra-layer coupling and maintaining task performance even under aggressive sparsity.

Formally, all tasks are trained with a composite loss defined as the average of task-specific losses, i.e., $L_{\text{total}} = \frac{1}{T}\sum_{t=1}^{T} L^{(t)}$, where $L^{(t)}$ denotes the loss for task t. The pretrained backbone weights W_ℓ remain frozen, and the backbone pruning mask M_{W_ℓ} is fixed, so pruning focuses exclusively on the LoRA modules. We optimize adapter masks $M_{A_\ell}^{(t)}$ and $M_{B_\ell}^{(t)}$, while LoRA parameters $A_\ell^{(t)}$ and $B_\ell^{(t)}$ are updated continuously via gradient descent. The masks are updated *periodically* based on adaptation signals (e.g., gradient magnitude or gating usage), allowing pruning and fine-tuning to co-adapt and preserve accuracy.

Building on this procedure, we enforce *task-consistency* of masks within each LoRA, requiring all tasks to share M_{A_ℓ} and M_{B_ℓ} to mitigate cross-task pruning dependencies and enable structured deployment. Without this constraint, tasks may induce inconsistent sparsity patterns that (i) hinder the use of structured sparsity accelerators (e.g., TensorRT, XLA) (Jeong et al. 2021) and (ii) preserve redundant parameters as overlapping subspaces are independently retained by different tasks (Xiang et al. 2024). To further ensure hardware-friendliness, we adopt *structural pruning* (Xu et al. 2024; Zhang et al. 2024), applying M_{A_ℓ} and M_{B_ℓ} in regular patterns such as row-wise or column-wise groups. Unlike unstructured pruning, this strategy removes predefined parameter groups, enabling efficient GPU execution while preserving the low-rank structure for a coherent, rank-reduced update. We also apply structural pruning to the backbone weights M_{W_ℓ}, preserving critical structures in the shared representation while encouraging LoRA to capture task-specific adaptations, thereby achieving better alignment between LoRA and backbone masks.

Task-Adaptive Gating for Shared LoRA. While progressive pruning reduces redundancy in task-specific adapters, it does not fully account for the shared LoRA, which interacts with all task-specific branches. Moreover, enforcing a single shared pruning mask across tasks poses challenges, as it compels all adapters to prune at the same rate even though tasks differ in pruning tolerance: some can be pruned more aggressively, whereas others require additional capacity to maintain accuracy. To address this, we introduce a *task-adaptive gating mechanism* integrated with progressive pruning. Each Transformer block is equipped with a differentiable, task-dependent gate $\mathcal{G}_\ell$ that dynamically modulates the contributions of shared and task-specific LoRA branches, thereby enabling fine-grained balancing between shared representation and task-specific specialization (Ma et al. 2018; Stickland and Murray 2019). For a task

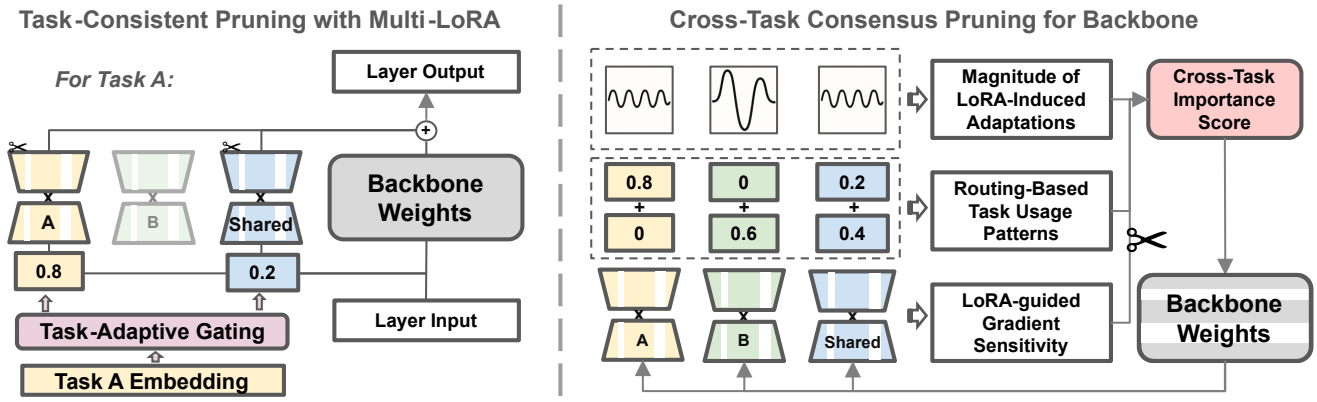

Figure 1: Overall Architecture of LoGIC. a) Task-Consistent Pruning with Multi-LoRA preserves task-specific specialization in LoRA, and b) Cross-Task Consensus Pruning removes redundant backbone parameters across tasks.

t, gating coefficients are computed as follows.

$$[\mathcal{G}^{(t)}_{\ell,\text{shared}}, \mathcal{G}^{(t)}_{\ell,\text{specific}}] = \sigma(W_\ell h^{(t)} + b_\ell),$$

where $h^{(t)}$ is the task embedding, W_ℓ and b_ℓ are learnable parameters, and $\sigma(\cdot)$ is the sigmoid function. These coefficients weight the pruned LoRA perturbations before being added to the block output.

Importantly, the gating outputs play a dual role: they not only provide pruning signals but also guide the updates of LoRA parameters. To reflect this, we extend the pruning-aware objective such that (i) the task loss updates $A^{(t)}_\ell$ and $B^{(t)}_\ell$ in proportion to the gating coefficients, and (ii) sparsity penalties are scaled by each task's reliance on shared versus task-specific branches. For each layer ℓ, we decompose the pruning-aware objective into two components. The first term captures the gated task loss:

$$L_{\text{task},\ell} = \frac{1}{T}\sum_{t=1}^{T}\left(\mathcal{G}^{(t)}_{\ell,\text{specific}} L^{(t)}(A^{(t)}_\ell, B^{(t)}_\ell) + \mathcal{G}^{(t)}_{\ell,\text{shared}} L^{(t)}(A^{(0)}_\ell, B^{(0)}_\ell)\right),$$

where the gating coefficients $\mathcal{G}^{(t)}_\ell$ determine how much each task relies on the task-specific adapters $(A^{(t)}_\ell, B^{(t)}_\ell)$ versus the shared adapter $(A^{(0)}_\ell, B^{(0)}_\ell)$. This ensures that LoRA updates are scaled in proportion to task reliance.

The second term regularizes the LoRA mask through task-specific and shared routing gates:

$$\theta_{\Delta,\ell} = \frac{1}{T}\sum_{t=1}^{T}(\mathcal{G}^{(t)}_{\ell,\text{specific}} + \mathcal{G}^{(t)}_{\ell,\text{shared}})\,(\|M_{A_\ell}\|_0 + \|M_{B_\ell}\|_0),$$

where task-specific gates weight the 0-norm penalties to prune more aggressively for tasks heavily reliant on their own LoRA branches, reducing redundancy while safeguarding critical capacity. Shared gates penalize sparsity in proportion to cross-task usage, preventing excessive pruning of parameters essential for knowledge transfer. Overall, the gating coefficients adaptively scale LoRA updates based on each task's reliance on task-specific versus shared adapters.

Cross-Task Consensus Pruning for Backbone

After pruning LoRA modules, we extend pruning to the frozen ViT backbone. Since direct gradient updates are unavailable in parameter-efficient tuning (Hu et al. 2022; Zhu et al. 2024), backbone importance estimates via fine-tuning signals are often coarse, so pruning must necessarily rely on indirect signals. This introduces *intra-layer coupling*, as prediction quality depends on which backbone or adapter parameters are preserved. Thus, pruning must jointly consider both to retain task-critical capacity while removing redundancy. To this end, we define an importance score $I_{i,j}$ that integrates multiple task-aware signals to guide backbone pruning effectively.

$$I_{i,j} = \sum_t \hat{I}^{(t)}_{i,j} \cdot w^{(t)} \cdot \tilde{n}^{(t)},$$

where $I_{i,j}$ denotes the importance of weight (i,j) in layer ℓ of W_ℓ. Directly deriving this score through full backpropagation across all tasks would be prohibitively costly. Therefore, we approximate it using *three complementary components*: (1) gradient sensitivity $\hat{I}^{(t)}_{i,j}$, (2) routing-based task usage patterns $w^{(t)}$, and (3) the magnitude of LoRA-induced adaptations $\tilde{n}^{(t)}$. This consensus-driven criterion ensures that only parameters consistently deemed unimportant across all tasks are pruned, while preserving shared representations crucial for generalization and specialized ones essential for individual tasks.

LoRA-guided Gradient Sensitivity. A central signal in our consensus scoring is the sensitivity of each pretrained weight to task performance, as captured through LoRA updates. To estimate this, we adapt the first-order Taylor expansion–based importance estimation technique (Zhang et al. 2024), which allows us to approximate each weight's contribution without requiring full backpropagation through the frozen backbone. For task t, the gradient-based importance score $\hat{I}^{(t)}_{i,j}$ estimates the squared first-order influence of weight (i,j) on task t's loss:

$$\hat{I}^{(t)}_{i,j} = \left(\frac{\partial L^{(t)}}{\partial B^{(t)}_{i,:}} A^{(t)}_{j,:} + B^{(t)}_{i,:}\frac{\partial L^{(t)}}{\partial A^{(t)}_{j,:}} - \frac{\partial L^{(t)}}{\partial B^{(t)}_{i,:}}\frac{\partial L^{(t)}}{\partial A^{(t)}_{j,:}}\right) \cdot \left[W_{i,j} + \left(B^{(t)}A^{(t)\top}\right)_{i,j}\right]^2,$$

which approximates how perturbations to weight (i,j) affect the task loss via LoRA updates, providing a first-order

estimate of its contribution to performance. Computing such sensitivity exactly across all tasks would be prohibitively costly due to repeated backpropagation through the frozen backbone. The Taylor approximation offers a tractable alternative that captures gradient-driven importance while avoiding the expense of full second-order computations. Thus, gradient sensitivity provides a principled, task-level baseline for weight importance. When aggregated across tasks in the consensus framework, it ensures pruning decisions reflect both shared and task-specific gradient signals, forming the foundation for balanced multi-task pruning.

Routing-Based Task Usage Patterns. While gradient sensitivity captures the local impact of each weight on task performance, it overlooks how frequently tasks rely on specific LoRA branches. To address this, we incorporate gating outputs from the Task-Adaptive Routing module into our importance estimation. These gating coefficients quantify the relative contributions of shared and task-specific adapters at each layer (Ma et al. 2018; Stickland and Murray 2019). While computing exact routing dynamics is costly, we use gating outputs from the task-adaptive gating module as a proxy. For each task t, we define the usage weight as $w^{(t)} = \mathcal{G}^{(t)}_{\ell,\text{specific}} + \mathcal{G}^{(t)}_{\ell,\text{shared}}$, where $\mathcal{G}^{(t)}_{\ell,\text{specific}}$ measures task t's reliance on its dedicated adapter, and $\mathcal{G}^{(t)}_{\ell,\text{shared}}$ quantifies reliance on the shared adapter at layer ℓ. This adaptively scales weight importance by each LoRA's utilization, ensuring that critical parameters, whether task-specific or shared, are retained, aligning pruning with task demands.

Magnitude of LoRA-Induced Adaptations. While gradient sensitivity and routing-based usage capture how weights contribute to and are utilized by tasks, they overlook the extent to which tasks adapt the pretrained backbone. Tasks that apply larger LoRA updates are generally more sensitive to pruning, since removing such parameters risks eliminating essential adaptations (Kim, Kim, and Kang 2024). We define the adaptation strength of each task-specific LoRA branch t as $n^{(t)} = \|A^{(t)}\|_F \cdot \|B^{(t)}\|_F$, where $\|\cdot\|_F$ denotes the Frobenius norm, capturing the effective scale of its low-rank adaptation. We then normalize $n^{(t)}$ via a softmax to obtain $\tilde{n}^{(t)}$, so that importance reflects collective knowledge across tasks rather than isolated task-specific updates. This additional signal complements routing-based usage by highlighting parameters that, though rarely activated, may enable substantial, high-impact modifications. In doing so, it preserves such "low-frequency, high-impact" weights, which are essential for sustaining robust multi-task performance under sparsity.

By integrating three complementary signals, (i) LoRA-guided gradient sensitivity, which measures each weight's impact on task loss; (ii) routing-based usage, which indicates task reliance on shared versus task-specific branches; and (iii) adaptation magnitude, which highlights substantial task-driven updates, LoGIC effectively balances shared and task-specific capacity, preserving critical weights and sustaining strong multi-task performance under high sparsity.

Experiments

Dataset. We evaluate our method on benchmarks spanning pixel-level, image-level, and instance-level tasks. NYU Depth v2 (NYUv2) (Silberman et al. 2012) contains 1,449 indoor images annotated for semantic segmentation (13 classes), monocular depth estimation, and surface normal prediction, making it a standard benchmark for multi-task dense prediction. For fine-grained image classification, we use Oxford Flowers-102 (Nilsback and Zisserman 2008), which covers 102 flower species with high intra-class variance. For instance-level evaluation, we adopt CPPE-5 (Dagli and Shaikh 2023), a dataset of about 1,000 real-world images annotated with bounding boxes for five personal protective equipment categories.

Evaluation Metrics. Performance is measured with standard task-specific metrics: Mean Intersection over Union (mIoU) (Long, Shelhamer, and Darrell 2015) for semantic segmentation, Threshold Accuracy (Silberman et al. 2012) for depth estimation, Angular Accuracy (Silberman et al. 2012) for surface normals, Top-1 Accuracy for classification, and mean Average Precision (mAP) under the COCO protocol (Lin et al. 2014) for detection. We also report the average score (AVG) across all five metrics to assess overall multi-task performance. Beyond accuracy, we evaluate model compactness and efficiency using two measures: sparsity, defined as the proportion of pruned weights, and model size, with three components for the multi-task setting, the ViT backbone, the LoRA modules, and the resulting total parameter count for deployment. These jointly capture the trade-off between accuracy and efficiency.

Architecture. We use two ViT backbones: ViT-L (Dosovitskiy et al. 2020b) (304M parameters, pretrained on ImageNet-21K) and Swin-L (Liu et al. 2021) (195M parameters, pretrained on ImageNet-22K with shifted-window attention), both initialized from HuggingFace checkpoints (Wolf et al. 2019). For dense prediction, we add an Atrous Spatial Pyramid Pooling (ASPP) module (Chen et al. 2017) to enhance multi-scale features. Task-specific heads are lightweight: 1×1 convolutions for segmentation, depth, and normals; linear layers for classification; and a DETR-style head with 100 queries (Carion et al. 2020).

Baselines. We consider two deployment strategies. Full Fine-Tuning (FFT) trains a separate model per task, providing full capacity without sharing. Multi-Task Tuning (MTT) jointly updates all weights across tasks. For efficiency, we also evaluate LoRA (Hu et al. 2022), which adds task-specific low-rank adapters using only about 1% of the parameters, enabling lightweight multi-task adaptation. For pruning baselines, Magnitude Pruning (Lee et al. 2021) removes low-magnitude weights during fine-tuning, while Hessian-based Pruning (Kuznedelev et al. 2022) uses second-order information for more informed decisions. LoRAPrune (Zhang et al. 2024) combines LoRA with backbone pruning to reduce inference cost, but its single shared adapter often leads to conflicts in multi-task settings. AdaLoRA (Zhang et al. 2023) reallocates LoRA capacity using task-specific importance scores and performs

	Sparsity	Approach	Model Size			SEG	DEPTH	NORMAL	CLS	OD
			Backbone	LoRA	Total	mIoU ↑	$\sigma_{1.25}$ ↑	Angle 30° ↑	Acc ↑	mAP ↑
ViT-L	0%	Full Fine Tuning (FFT)	304M x 5	0M	1520M	0.7444	0.7044	0.6309	0.9297	0.4457
		Multi-Task Tuning (MTT)	304M x 1	0M	304M	0.7061	0.7005	0.5202	0.8287	0.4503
		LoRA	304M x 1	3.8M x 5	323M	0.7531	0.6874	0.6006	0.9771	0.4172
		FFT + AdaLoRA	304M x 5	3.1M x 5	1535.5M	0.6857	0.6113	0.4603	0.9618	0.4415
	30%	FFT + Magnitude Pruning	225M x 5	0M	1125M	0.7012	0.6537	0.4929	0.8345	0.4325
		FFT + Hessian-based Pruning	225M x 5	0M	1125M	0.6789	0.6488	0.4799	0.8095	0.4256
		FFT + LoRAPrune	225M x 5	3.1M x 5	1140.5M	0.7242	0.6824	0.4930	0.9602	0.4313
		MTT + Magnitude Pruning	225M x 1	0M	225M	0.6722	0.6435	0.4834	0.7783	0.4410
		MTT + Hessian-based Pruning	225M x 1	0M	225M	0.6567	0.5395	0.4760	0.7599	0.4103
		LoGIC (Ours)	225M x 1	3.1M x 6	243.6M	**0.7506**	**0.7118**	**0.6020**	**0.9648**	**0.4424**
	50%	FFT + Magnitude Pruning	146M x 5	0M	730M	0.6059	0.5785	0.4299	0.7068	0.4152
		FFT + Hessian-based Pruning	146M x 5	0M	730M	0.5876	0.5783	0.3759	0.6292	0.4256
		FFT + LoRAPrune	146M x 5	2.8M x 5	744M	0.6540	0.5965	0.4563	0.9006	0.3945
		MTT + Magnitude Pruning	146M x 1	0M	146M	0.5712	0.5401	0.4209	0.5581	0.4324
		MTT + Hessian-based Pruning	146M x 1	0M	146M	0.4696	0.4806	0.3514	0.4098	0.4010
		LoGIC (Ours)	146M x 1	2.8M x 6	162.8M	**0.7130**	**0.6625**	**0.5739**	**0.9006**	**0.4400**
Swin-L	0%	Full Fine Tuning (FFT)	195M x 5	0M	975M	0.7552	0.7486	0.5948	0.9786	0.4386
		Multi-Task Tuning (MTT)	195M x 1	0M	195M	0.7303	0.7224	0.4675	0.9572	0.4193
		LoRA	195M x 1	2.6M x 5	208M	0.7398	0.7198	0.5394	0.9908	0.4422
		FFT + AdaLoRA	195M x 5	2.2M x 5	986M	0.6725	0.5603	0.3721	0.9786	0.3991
	30%	FFT + Magnitude Pruning	139M x 5	0M	695M	0.6803	0.6724	0.4398	0.9410	0.4252
		FFT + Hessian-based Pruning	139M x 5	0M	695M	0.6655	0.5095	0.4192	0.9284	0.4356
		FFT + LoRAPrune	139M x 5	2.2M x 5	706M	0.6888	0.6249	0.4108	0.9801	**0.4432**
		MTT + Magnitude Pruning	139M x 1	0M	139M	0.6902	0.6611	0.4478	0.9312	0.4196
		MTT + Hessian-based Pruning	139M x 1	0M	139M	0.6652	0.3269	0.4188	0.9067	0.3911
		LoGIC (Ours)	139M x 1	2.2M x 6	152.2M	**0.7249**	**0.7108**	**0.5421**	**0.9817**	0.4303
	50%	FFT + Magnitude Pruning	103M x 5	0M	515M	0.6042	0.5495	0.4035	0.8766	0.4097
		FFT + Hessian-based Pruning	103M x 5	0M	515M	0.5985	0.5321	0.4087	0.8324	0.3987
		FFT + LoRAPrune	103M x 5	2.0M x 5	525M	0.4782	0.4594	0.3123	0.8746	0.3959
		MTT + Magnitude Pruning	103M x 1	0M	103M	0.5704	0.5507	0.3845	0.8364	0.4005
		MTT + Hessian-based Pruning	103M x 1	0M	103M	0.4358	0.3844	0.2692	0.2018	0.3622
		LoGIC (Ours)	103M x 1	2.0M x 6	115M	**0.6201**	**0.5800**	**0.4536**	**0.9174**	**0.4120**

Table 1: Comparison of different pruning approaches on ViT-L (304 M) and Swin-L (195 M) across five tasks. Our method **LoGIC** achieves strong performance while training only a small fraction of parameters.

well in single-task adaptation, but in multi-task settings its fixed backbone causes cross-task interference and reduces robustness under high sparsity. We use 30% LoRA sparsity for the AdaLoRA baseline.

Quantitative Results. As shown in Table 1, LoGIC outperforms all MTT pruning methods by explicitly modeling the dependency between LoRA modules and the shared backbone. While prior approaches neglect the coupling across tasks and introduce interference, LoGIC selectively prunes only those parameters that are consistently unimportant across all tasks. By effectively extracting shared information across different tasks, LoGIC achieves substantial performance gains. At 30% sparsity on ViT-L, it reaches an average score of 0.6943, compared to 0.6037 for the best competing MTT method. Even under 50% sparsity, LoGIC maintains strong performance (0.6580), surpassing both FFT-based and MTT-based baselines. On Swin-L, LoGIC achieves 0.6780 at 30% sparsity and 0.5966 at 50%, outperforming all other methods at the same sparsity levels. Notably, compared to the FFT model that employs five independently fine-tuned backbones, LoGIC achieves comparable multi-task performance using only 16% of the total deployment parameters. Although LoRAPrune supports backbone pruning alongside LoRA modules, it is limited to single-task settings and thus falls short of LoGIC in both parameter efficiency and overall performance.

Conclusion

We propose **LoGIC**, a unified multi-task ViT pruning framework that integrates shared and task-specific LoRA modules with adaptive routing and consensus-based importance estimation to preserve accuracy under heavy compression. We show that pruning in multi-LoRA ViTs is inherently coupled across tasks, making independent pruning suboptimal. Leveraging this insight, LoGIC attains high sparsity while maintaining accuracy, matching full-precision performance with only 50% of the parameters and surpassing all baselines. We also plan to extend LoGIC to zero-shot and online pruning settings.

Acknowledgments

This work was supported by the NTU–Delta Electronics Innovation Research Funding Project; by the Ministry of Education, Taiwan, through the Higher Education Sprout Project—The Featured Area Research Center Program; and by the National Science and Technology Council, Taiwan, under Grant NSTC 114-2223-E-002-009.

References

Agiza, A.; Neseem, M.; and Reda, S. 2024. Mtlora: Low-rank adaptation approach for efficient multi-task learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 16196–16205.

Ahmed, S.; Al Arafat, A.; Najafi, D.; Mahmood, A.; Rizve, M. N.; Al Nahian, M.; Zhou, R.; Angizi, S.; and Rakin, A. S. 2025. DeepCompress-ViT: Rethinking Model Compression to Enhance Efficiency of Vision Transformers at the Edge. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 30147–30156.

Albert, P.; Zhang, F. Z.; Saratchandran, H.; Rodriguez-Opazo, C.; van den Hengel, A.; and Abbasnejad, E. 2025. RandLoRA: Full rank parameter-efficient fine-tuning of large models. In *The Thirteenth International Conference on Learning Representations*.

Arnab, A.; Dehghani, M.; Heigold, G.; Sun, C.; Lučić, M.; and Schmid, C. 2021. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, 6836–6846.

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, 213–229. Springer.

Chen, J.; Zhang, A.; Shi, X.; Li, M.; Smola, A.; and Yang, D. 2023. Parameter-Efficient Fine-Tuning Design Spaces. In *The Eleventh International Conference on Learning Representations*.

Chen, L.-C.; Papandreou, G.; Schroff, F.; and Adam, H. 2017. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*.

Chen, Z.; Badrinarayanan, V.; Lee, C.-Y.; and Rabinovich, A. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*, 794–803. PMLR.

Dagli, R.; and Shaikh, A. M. 2023. Cppe-5: Medical personal protective equipment dataset. *SN Computer Science*, 4(3): 263.

Ding, N.; Lv, X.; Wang, Q.; Chen, Y.; Zhou, B.; Liu, Z.; and Sun, M. 2023. Sparse Low-rank Adaptation of Pre-trained Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 4133–4145.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020a. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020b. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Fu, Z.; Yang, H.; So, A. M.-C.; Lam, W.; Bing, L.; and Collier, N. 2023. On the effectiveness of parameter-efficient fine-tuning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 12799–12807.

Goyal, S.; Kumar, A.; Garg, S.; Kolter, Z.; and Raghunathan, A. 2023. Finetune like you pretrain: Improved finetuning of zero-shot vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19338–19347.

He, J.; Duan, Z.; and Zhu, F. 2025. CL-LoRA: Continual Low-Rank Adaptation for Rehearsal-Free Class-Incremental Learning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 30534–30544.

He, S.; Ding, L.; Dong, D.; Zhang, M.; and Tao, D. 2022. SparseAdapter: An Easy Approach for Improving the Parameter-Efficiency of Adapters. In *2022 Findings of the Association for Computational Linguistics: EMNLP 2022*.

He, X.; Gao, D.; Zhou, Z.; Tong, Y.; and Thiele, L. 2021. Pruning-aware merging for efficient multitask inference. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 585–595.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2): 3.

Jeong, E.; Kim, J.; Tan, S.; Lee, J.; and Ha, S. 2021. Deep learning inference parallelization on heterogeneous processors with tensorrt. *IEEE Embedded Systems Letters*, 14(1): 15–18.

Kim, J.; Kim, G.; and Kang, S. 2024. Lottery Rank-Pruning Adaptation Parameter Efficient Fine-Tuning. *Mathematics*, 12(23): 3744.

Kuznedelev, D.; Kurtic, E.; Frantar, E.; and Alistarh, D. 2022. ovit: An accurate second-order pruning framework for vision transformers. *openreview*.

Lee, J.; Park, S.; Mo, S.; Ahn, S.; and Shin, J. 2021. Layer-adaptive Sparsity for the Magnitude-based Pruning. In *International Conference on Learning Representations*.

Li, T.; Chang, H.; Mishra, S.; Zhang, H.; Katabi, D.; and Krishnan, D. 2023. Mage: Masked generative encoder to unify representation learning and image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2142–2152.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, 740–755. Springer.

Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; and Guo, B. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, 10012–10022.

Liu, Z.; Sun, M.; Zhou, T.; Huang, G.; and Darrell, T. 2018. Rethinking the Value of Network Pruning. In *International Conference on Learning Representations*.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3431–3440.

Ma, J.; Zhao, Z.; Yi, X.; Chen, J.; Hong, L.; and Chi, E. H. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 1930–1939.

Molchanov, P.; Mallya, A.; Tyree, S.; Frosio, I.; and Kautz, J. 2019. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11264–11272.

Nilsback, M.-E.; and Zisserman, A. 2008. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, 722–729. IEEE.

Parikh, D.; Li, S.; Zhang, B.; Kannan, R.; Busart, C.; and Prasanna, V. 2024. Accelerating vit inference on fpga through static and dynamic pruning. In *2024 IEEE 32nd Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, 78–89. IEEE.

Silberman, N.; Hoiem, D.; Kohli, P.; and Fergus, R. 2012. Indoor segmentation and support inference from rgbd images. In *European conference on computer vision*, 746–760. Springer.

Stickland, A. C.; and Murray, I. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, 5986–5995. PMLR.

Sun, X.; Hassani, A.; Wang, Z.; Huang, G.; and Shi, H. 2022. Disparse: Disentangled sparsification for multitask model compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12382–12392.

Tay, Y.; Dehghani, M.; Rao, J.; Fedus, W.; Abnar, S.; Chung, H. W.; Narang, S.; Yogatama, D.; Vaswani, A.; and Metzler, D. 2021. Scale Efficiently: Insights from Pretraining and Finetuning Transformers. In *International Conference on Learning Representations*.

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Xiang, M.; Tang, J.; Yang, Q.; Guan, H.; and Liu, T. 2024. AdapMTL: Adaptive Pruning Framework for Multitask Learning Model. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 5121–5130.

Xie, Y.; and Liao, Y. 2023. Efficient-ViT: A light-weight classification model based on CNN and ViT. In *Proceedings of the 2023 6th International Conference on Image and Graphics Processing*, 64–70.

Xu, K.; Wang, Z.; Chen, C.; Geng, X.; Lin, J.; Yang, X.; Wu, M.; Li, X.; and Lin, W. 2024. Lpvit: Low-power semi-structured pruning for vision transformers. In *European Conference on Computer Vision*, 269–287. Springer.

Yang, H.; Yin, H.; Shen, M.; Molchanov, P.; Li, H.; and Kautz, J. 2023a. Global vision transformer pruning with hessian-aware saliency. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 18547–18557.

Yang, Y.; Chiang, H.-Y.; Li, G.; Marculescu, D.; and Marculescu, R. 2023b. Efficient low-rank backpropagation for vision transformer adaptation. *Advances in Neural Information Processing Systems*, 36: 14725–14736.

Ye, H.; Zhang, B.; Chen, T.; Fan, J.; and Wang, B. 2023. Performance-aware approximation of global channel pruning for multitask cnns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(8): 10267–10284.

Zhang, B.; Tian, Z.; Tang, Q.; Chu, X.; Wei, X.; Shen, C.; et al. 2022. Segvit: Semantic segmentation with plain vision transformers. *Advances in Neural Information Processing Systems*, 35: 4971–4982.

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

Zhang, Q.; Chen, M.; Bukharin, A.; He, P.; Cheng, Y.; Chen, W.; and Zhao, T. 2023. Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning. In *International Conference on Learning Representations*. Openreview.

Zhang, Z.; Lu, X.; Cao, G.; Yang, Y.; Jiao, L.; and Liu, F. 2021. ViT-YOLO: Transformer-based YOLO for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2799–2808.

Zhou, H.; Lu, X.; Xu, W.; Zhu, C.; Zhao, T.; and Yang, M. 2025. LoRA-drop: Efficient LoRA Parameter Pruning based on Output Evaluation. In *COLING*.

Zhu, Y.; Shen, Z.; Zhao, Z.; Wang, S.; Wang, X.; Zhao, X.; Shen, D.; and Wang, Q. 2024. Melo: Low-rank adaptation is better than fine-tuning for medical image diagnosis. In *2024 IEEE International Symposium on Biomedical Imaging (ISBI)*, 1–5. IEEE.