TDSS: Task Dynamic-Synergistic Skill Adaptation for Boosting Efficient and Scalable Multi-Task Learning in Dense Visual Prediction

Haiming Yao [1,3], Qiyu Chen[2,3], Wei Luo[1,3], Zheng Zhang[3], Jianxing Liao[3], Wei You[3]*

[1]Department of Precision Instruments, Tsinghua University, Beijing
[2]Institute of Automation, Chinese Academy of Sciences, Beijing
[3]Advanced Computing and Storage Lab, Huawei Technologies Co. Ltd
{yhm22, luow23}@mails.tsinghua.edu.cn; chenqiyu2021@ia.ac.cn; {zhangzheng147, liaojianxing, youwei22}@huawei.com

Abstract

The transfer of knowledge from large-scale pre-trained models to diverse downstream tasks has achieved remarkable success. Beyond the traditional full fine-tuning paradigm, Parameter-Efficient Fine-Tuning (PEFT) has emerged as a more efficient model adaptation approach. However, applying existing PEFT methods to adapt dense vision models, particularly in multi-task settings, remains inadequately explored due to their low efficiency, limited task scalability, and neglect of cross-task fine-tuning interactions. To address these challenges, we propose the **T**ask **D**ynamic-**S**ynergistic **S**kill Adaptation, termed *TDSS*, an efficient and scalable multi-task model adaptation framework for dense visual predictions. *TDSS* comprises two key components: Task-Dynamic Skill Adapters (TDSA) and Task-Synergistic Adaptation Interaction (TSAI). Specifically, TDSA are inserted in parallel into pre-trained vision models to extract task-specific adapted features through the construction of skill representation experts and task dynamic gating. TSAI is developed to enhance cross-task adaptation interaction by bridging global generic and task-specific adapted features. Extensive experiments on multi-task dense visual predictions demonstrate that *TDSS* surpasses existing state-of-the-art parameter-efficient fine-tuning methods, while exhibiting remarkable efficiency and scalability in parameters and computational complexity.

1 Introduction

The fine-tuning paradigm enables knowledge transfer from models pre-trained on large datasets to downstream tasks and is widely used in natural language processing (Devlin et al. 2018; Brown et al. 2020) and computer vision (He et al. 2020; Chotard et al. 2023). However, full model fine-tuning faces challenges, such as catastrophic forgetting of pre-trained semantics, as well as high computational and storage demands, especially as the number of downstream tasks increases. To address these challenges, researchers have increasingly focused on Parameter-Efficient Fine-Tuning (PEFT) (Yu et al. 2022), a promising paradigm that freezes the majority of pre-trained parameters while fine-tuning only a small portion. This strategy achieves a better balance between parameter efficiency and task performance, making it well-suited for scalable adaptation in resource-constrained environments.

While PEFT has shown significant progress in large-scale language models (Ding et al. 2023) and natural language processing (Houlsby et al. 2019), its potential in the visual domain—especially for challenging dense vision tasks like semantic segmentation and depth estimation—has yet to be thoroughly investigated. Furthermore, most existing PEFT approaches are tailored to specific single-task settings, with limited consideration for the more challenging context of multi-task learning (MTL) (Zhang and Yang 2021; Vandenhende et al. 2021b), which requires a unified model to effectively adapt across multiple tasks simultaneously.

To apply PEFT for adapting pre-trained vision models (PVMs) to multiple downstream dense prediction tasks in MTL settings, in contrast to full-model fine-tuning illustrated in Fig. 1(a), the model must keep the PVM parameters frozen while fine-tuning only a small subset within the PEFT modules. However, when directly utilizing existing PEFT modules (*e.g.*, Adapter (Houlsby et al. 2019), LoRA (Hu et al. 2022)) for multi-task model adaptation, the current practice, as shown in Fig. 1(b), is to customize a PEFT module for each task and integrate it into the PVMs for fine-tuning. Although this approach is intuitive and easy to implement, we argue that it fails to achieve efficient and scalable multi-task adaptation from two critical perspectives: *integration topology* and *task-adaptation relationship*. First, these methods primarily integrate PEFT and PVM modules using a serial main-path adaptation topology, as shown in Fig. 1(d). This implies that the PVMs must be executed separately for each task to meet multi-task adaptation requirements, causing the backbone execution time to increase proportionally with the number of tasks. Additionally, these methods decouple the fine-tuning of different tasks, as shown in Fig. 1(e), requiring a task-specific PEFT module for each task. This results in the adaptation modules scaling proportionally with the number of tasks, while also failing to capture the interactions among the fine-tuning processes of different tasks, which have been shown to be beneficial for MTL (Zhang et al. 2025). Consequently, based on the above two perspectives, we conclude that existing PEFT methods for multi-task dense vision adaptation still face three key limitations:**1) backbone execution**, **2) task scalability**, and **3) cross-task interaction**.

*Corresponding author.

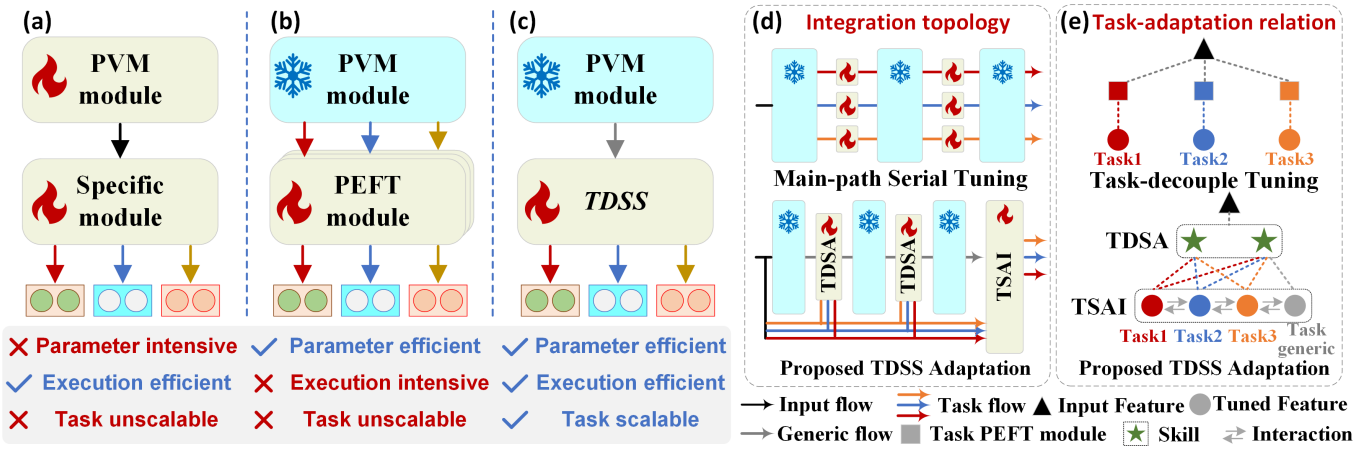

Figure 1: Comparison of adaptation paradigms. (a) Full fine-tuning methods use a specific module design and train all model parameters. (b) Existing PEFT-based methods require a specific PEFT module for each task and separate PVM execution per task. (c) The proposed *TDSS* needs only one module and a single PVM execution to get representations of different tasks simultaneously. (d) Comparison between the topology of existing main-path serial adaptation and the proposed *TDSS* adaptation. (e) Comparison between existing task-decoupled adaptation and the proposed *TDSS* adaptation.

Although recent studies such as Polyhistor (Liu et al. 2022), MTLoRA (Agiza, Neseem, and Reda 2024), and TADFormer (Baek et al. 2025) have extended PEFT to MTL by employing parameter-sharing strategies or designing task-aware fine-tuning mechanisms, they still exhibit inherent limitations with respect to the challenges outlined above. Going beyond these prior approaches, we propose Task Dynamic-Synergistic Skill (*TDSS*), a novel PEFT-based adaptation framework designed for dense vision prediction in MTL settings. As shown in Fig. 1(c), *TDSS* requires only one adaptation module and a single backbone execution to obtain adapted representations for various dense tasks simultaneously. Specifically, *TDSS* incorporates two key components: the Task-Dynamic Skill Adapters (TDSA) and Task-Synergistic Adaptation Interaction (TSAI), designed to achieve efficient and scalable multi-task adaptation by revisiting the two key perspectives outlined above. First, regarding the *integration topology*, unlike existing PEFT methods, the proposed TDSA adopts a parallel lateral adaptation topology when integrated with the PVMs, as illustrated in Fig. 1(d). Specifically, each task has its own dedicated lateral path to encode task-specific adapted information. This design enables the PVMs to obtain representations for multiple tasks in a single execution. More importantly, regarding the *task-adaptation relation* illustrated in Fig. 1(e), inspired by mixture-of-experts (MoE) models (Cai et al. 2024), TDSA constructs skill representation experts shared across tasks and employs dynamic task gating to combine these skills for task adaptation. Meanwhile, TSAI is introduced to facilitate cross-task adaptation interactions. These two components jointly enhance the model's performance, efficiency, and scalability.

We benchmark our method against state-of-the-art approaches, demonstrating superior performance with fewer trainable parameters, as well as improved efficiency and scalability. Our key contributions include:

- From two key perspectives, we elucidate three major limitations of existing PEFT-based adaptation methods for dense visual prediction in the context of MTL.
- To address the limitations, we propose *TDSS*, an efficient and scalable PEFT-based MTL dense vision adaptation approach for that integrates two key designs: TDSA and TSAI, which are built on the observed insight.
- Experiments show that the proposed method obtains a +8.37 % Δm (Agiza, Neseem, and Reda 2024) over single-task full fine-tuning on the Pascal Context dataset (Everingham et al. 2010), using only 3.52M trainable parameters, outperforming other existing PEFT methods in both performance and efficiency.

2 Related Work

2.1 Multi-Task Learning for Dense Vision

MTL (Zhang and Yang 2018, 2021) solves multiple related tasks within a unified framework by sharing generalizable features (Vandenhende et al. 2021a), reducing redundancy and parameter usage. Moreover, the cross-task complementarity can also enhance performance (Fifty et al. 2021). In dense vision tasks (Vandenhende et al. 2021a), this typically includes semantic segmentation, depth estimation, and surface normal prediction. Existing approaches mainly follow encoder-based (Xu et al. 2023; Xu, Yang, and Zhang 2023) or decoder-based (Ye and Xu 2024; Xin et al. 2024b) designs, aiming to enable task interaction through tailored architectures. However, they often require full fine-tuning, leading to a large number of trainable parameters.

2.2 Parameter-Efficient Fine-tuning

Existing PEFT methods can be categorized into several major paradigms based on their mechanisms. The conventional

approach (He et al. 2020; Chotard et al. 2023) of freezing the pre-trained backbone and fine-tuning only the output/decoder layers often fails to sufficiently adapt the backbone for downstream tasks, leading to suboptimal performance. Modern approaches (Ding et al. 2023; Fu et al. 2023; He et al. 2022) instead update a small subset of parameters in the backbone while retaining its inherent structure to achieve task-specific adaptation. Representative methods include Adapter (Houlsby et al. 2019) fine-tuning, which inserts lightweight Adapter modules into pre-trained models (*e.g.*, after Attention/feed-forward network (FFN) blocks in Transformers) to adapt their features; LoRA (Hu et al. 2022) fine-tuning, which employs low-rank approximation to update the weight matrices in Attention; and Feature Transform (Lian et al. 2022) fine-tuning, which introduces scaling and shifting operations to calibrate the feature distribution.

2.3 Multi-Tasking PEFT

To mitigate the drawbacks of full fine-tuning in multi-MTL, recent works (Liu et al. 2022; Agiza, Neseem, and Reda 2024; Xin et al. 2024a) have increasingly explored PEFT techniques to enable more efficient adaptation. For example, HyperPerformer (Karimi Mahabadi et al. 2021) employs a hyper-network to generate adapter weights tailored to different tasks. Polyhistor (Liu et al. 2022) proposes task-specific adapters with cross-layer and cross-task parameter sharing. MTLoRA (Agiza, Neseem, and Reda 2024) extends LoRA to the MTL setting, and TADFormer (Baek et al. 2025) introduces a Transformer-based architecture that dynamically adjusts to task-specific requirements. While inter-task interaction has been shown to be crucial for effective MTL (Zhang et al. 2025), most existing multi-tasking PEFT methods typically decouple fine-tuning across tasks, neglecting cross-task fine-tuning interactions as previously noted.

3 Preliminaries

We briefly revisit the foundations of this work, including hierarchical vision Transformers(ViT) and representative parameter-efficient tuning (PEFT) methods.

3.1 Pre-trained Vision Transformers

We adopt a hierarchical vision Transformer, where the input image is split into patches and embedded into tokens via a patch embedding layer. These tokens form a sequence processed by multiple Transformer blocks, each comprising multi-head self-attention (e.g., window or shifted-window attention (Liu et al. 2021)) and a feed-forward network (FFN), both with pre-normalization and residual connections. Token resolution is progressively reduced via patch merging across stages, enabling hierarchical representation. These backbones are typically pretrained on large-scale datasets (e.g., ImageNet) and adapted to downstream tasks.

3.2 PEFT Mechanisms

Adapter Tuning (Houlsby et al. 2019) introduces a lightweight bottleneck module comprising a down-projection $W_d \in \mathbb{R}^{C\times R}$, activation Φ, and an up-projection $W_u \in \mathbb{R}^{R\times C}$, where C and R denote input and reduced dimensions. Given input $\boldsymbol{x} \in \mathbb{R}^{L\times C}$, a sequence of L tokens with C-dimensional features, the forward process is:

$$\text{Adapter}\left(\boldsymbol{x}\right) = \Phi\left(\boldsymbol{x} \cdot W_d\right) \cdot W_u. \tag{1}$$

where $W = \left[W_d; W_u^T\right] \in \mathbb{R}^{C\times 2R}$ is the learnable parameters within the adapter.

Feature Transform Tuning (Lian et al. 2022) (FT) applies an affine transformation using learnable scaling and shifting parameters $\gamma \in \mathbb{R}^C$ and $\beta \in \mathbb{R}^C$ for improved task adaptation as follows:

$$\text{FT}(\boldsymbol{x}) = \gamma \odot \boldsymbol{x} + \beta, \tag{2}$$

where $\odot$ denotes element-wise multiplication. Both γ and β are independent of the input $\boldsymbol{x}$ and are learned to capture the whole dataset distribution (Lian et al. 2022).

4 Methodology

4.1 Overall Framework

As illustrated in Fig. 2(a), the proposed *TDSS* framework operates as follows: Given an input image with a size of $H \times W \times 3$, the pre-trained ViT is first employed to extract pre-trained visual knowledge. Subsequently, Task-Dynamic Skill Adapters (TDSA) are integrated in parallel into the ViT blocks, where multiple lateral paths capture task-specific adaptation signals, enabling the multi-task adaptation process. To facilitate cross-task adaptation interactions, Task-Synergistic Adaptation Interaction (TSAI) is introduced to fuse the representations from each task path. The details of these components are elaborated in the subsequent sections.

4.2 Task-Dynamic Skill Adapters

The proposed TDSA is designed to be seamlessly integrated into the pre-trained ViT modules. Its lateral insertion topology, as illustrated in Fig. 2(b), enables simultaneous adaptation to multiple tasks. As shown in Fig. 2(c), the TDSA architecture comprises three key components: 1) a set of shared down-projection and up-projection layers; 2) skill representation experts that encode shared skill patterns across tasks; and 3) a dynamic task gating mechanism that selectively activates task-relevant skills.

Lateral Insertion Topology. As shown in Fig. 2(b), the proposed TDSA module is inserted into each Attention and FFN layer of the pre-trained ViT blocks. It extracts the input feature $\boldsymbol{x}$ and generates task-specific adapted features $\boldsymbol{x}^t$, which are injected into corresponding lateral branches for each task. Compared to existing adapter-based adaptation architectures (He et al. 2022; Lu et al. 2024), our lateral insertion topology allows the shared ViT backbone to be executed only once for all tasks, significantly improving computational efficiency. Moreover, as noted in (Sung, Cho, and Bansal 2022), this topology avoids gradient propagation through the backbone, which reduces GPU memory usage.

Skill Representation Experts. As shown in Fig. 2(c), given the input feature $\boldsymbol{x} \in \mathbb{R}^{L\times C}$ from a pre-trained ViT block, TDSA first applies a shared down-projection layer $W_d^s \in \mathbb{R}^{C\times R}$ to obtain low-rank representations $\boldsymbol{h} = \boldsymbol{x} \cdot W_d^s \in \mathbb{R}^{L\times R}$. In this low-rank space, we introduce a set

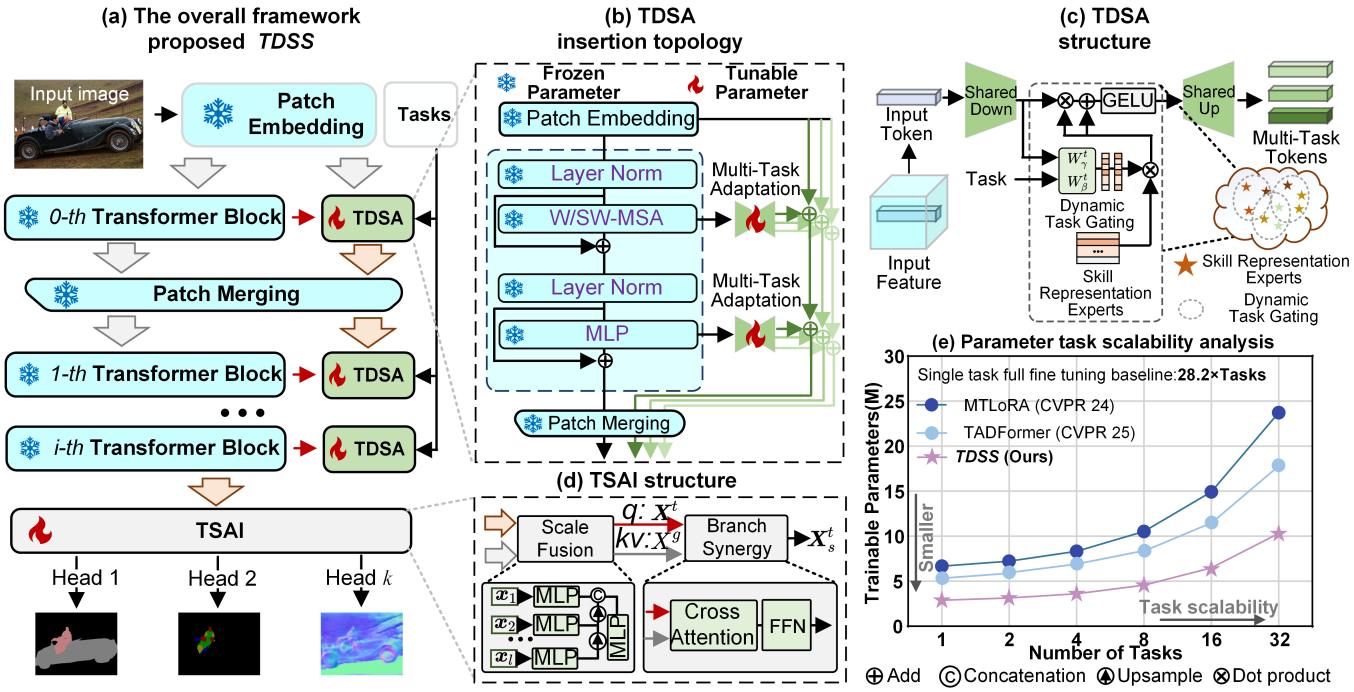

Figure 2: (a) Pipeline of our *TDSS*, where only the parameters in TDSA and TSAI are tunable. (b) Insertion topology of the TDSA in the backbone. (c) Details of the TDSA. (d) Details of the TSAI. (e) Comparison of the trainable parameters among single-task full fine-tuning, MTLoRA (Agiza, Neseem, and Reda 2024), TADFormer (Baek et al. 2025), and our method.

of learnable expert vectors $\boldsymbol{S} \in \mathbb{R}^{M\times R}$, consisting of M vectors, each of dimension R, where each vector represents an adaptation-related concept—we name it a "skill." This terminology is inspired by the way humans learn and combine different skills to perform various tasks (Yoo, Cho, and Woo 2022). These skills form a shared pool designed to capture general knowledge used for adapting to different tasks. Given the query $\boldsymbol{h}$, the task-specific representations $\boldsymbol{h}_s^t$ are generated from $\boldsymbol{S}$ via a Dynamic Task Gating mechanism, which is detailed in the following section.

Dynamic Task Gating. The skill matrix $\boldsymbol{S}$ is designed to explicitly capture and store task-shared adaptation patterns during training. To enable task-specific skill utilization, we propose Dynamic Task Gating as an addressing mechanism that activates the relevant skills for a given task t. Specifically, each task is assigned two learnable linear layers, $W_\gamma^t \in \mathbb{R}^{R\times M}$ and $W_\beta^t \in \mathbb{R}^{R\times M}$, which are used to compute two sets of allocation coefficients, $\gamma^t(\boldsymbol{h}) \in \mathbb{R}^{L\times M}$ and $\beta^t(\boldsymbol{h}) \in \mathbb{R}^{L\times M}$, respectively, as follows:

$$\gamma^t(\boldsymbol{h}) = \boldsymbol{h}\cdot W_\gamma^t, \beta^t(\boldsymbol{h}) = \boldsymbol{h}\cdot W_\beta^t. \tag{3}$$

To avoid negative skill activation, we employ the softmax operation to normalize the allocation coefficients as:

$$\hat{\gamma}_i^t(\boldsymbol{h}) = \frac{\exp(\gamma_i^t(\boldsymbol{h}))}{\sum_{j=1}^M \exp(\gamma_j^t(\boldsymbol{h}))}, \hat{\beta}_i^t(\boldsymbol{h}) = \frac{\exp(\beta_i^t(\boldsymbol{h}))}{\sum_{j=1}^M \exp(\beta_j^t(\boldsymbol{h}))}, \tag{4}$$

where the $\hat{\gamma}_i^t(\boldsymbol{h})$, $\hat{\beta}_i^t(\boldsymbol{h})$ denote the i-th entry of normalized coefficients. Let vector $\mathbf{s}_i$ represents the i-th row of $\boldsymbol{S}$, the skill-augmented scaling and shifting parameters $\gamma^t(\boldsymbol{h},\boldsymbol{S})$ and $\beta^t(\boldsymbol{h},\boldsymbol{S})$ can be obtained via a weighted sum as:

$$\gamma^t(\boldsymbol{h},\boldsymbol{S}) = \textstyle\sum_{i=1}^M \hat{\gamma}_i^t(\boldsymbol{h})\,\mathbf{s}_i, \beta^t(\boldsymbol{h},\boldsymbol{S}) = \sum_{i=1}^M \hat{\beta}_i^t(\boldsymbol{h})\,\mathbf{s}_i. \tag{5}$$

These task-aware parameters are then used as scaling and shifting factors in Feature Transform Tuning FT to modulate the low-rank representations:

$$\boldsymbol{h}_s^t = \gamma^t(\boldsymbol{h},\boldsymbol{S}) \odot \boldsymbol{h} + \beta^t(\boldsymbol{h},\boldsymbol{S}). \tag{6}$$

The task-specific transformed low-rank representation $\boldsymbol{h}_s^t$ is then activated by a GELU nonlinearity Φ and passed through a shared up-projection layer $W_u^s \in \mathbb{R}^{R\times C}$ to produce the task-specific fine-tuned feature:

$$\boldsymbol{x}^t = \Phi\left(\boldsymbol{h}_s^t\right)\cdot W_u^s. \tag{7}$$

Finally, the lateral features for each task are processed through pre-trained downsampling layers (*e.g.*, Patch Merging layers) before being passed to the next ViT stage, and produce multi-scale task features.

Relation with Adapter and Feature Transformation. It is worth noting that the generation of $\boldsymbol{x}^t$ is jointly conditioned on the skill representation $\boldsymbol{S}$, the input feature $\boldsymbol{x}$, and the task identity t. Among them, $\boldsymbol{S}$ is input-independent and is optimized during training to encode the prototypical transformation patterns required across multiple tasks over the entire dataset. At the same time, $\boldsymbol{x}^t$ is also dynamically modulated by both the input feature $\boldsymbol{x}$ and the task t. This design ensures that the resulting fine-tuned representation captures both dataset-level generalization and instance-specific

nuances, thereby seamlessly integrating the adaptation principles of Adapter tuning (Houlsby et al. 2019) and Feature Transform tuning (Lian et al. 2022).

4.3 Task-Synergistic Adaptation Interaction

To leverage the adapted features for downstream dense prediction, we introduce Task-Synergistic Adaptation Interaction (TSAI), comprising scale fusion and branch synergy (Fig. 2(d)). The former integrates multi-scale features per task, while the latter promotes cross-task adaptation by aligning task-specific and generic representations.

Following (Agiza, Neseem, and Reda 2024), we extract intermediate features from the 1st to 4th Transformer stages of each task branch. For scale fusion, we adopt the MLP-based design in (Xie et al. 2021), using task-specific fusion modules to retain unique characteristics. The fused outputs are denoted as task-specific representations $\boldsymbol{X}^t \in \mathbb{R}^{\frac{HW}{4^2} \times C}$.

The global representation $\boldsymbol{X}^g = \sum \boldsymbol{X}^t$ is then obtained through task aggregation, serving as a bridge (Zhang et al. 2025) to facilitate inter-task interaction. Branch synergy is achieved by injecting $\boldsymbol{X}^g$ into each $\boldsymbol{X}^t$ via:

$$\begin{aligned} \boldsymbol{X}_s^{t'} &= \mathrm{Attention}\left(\mathrm{LN}(\boldsymbol{X}^t), \mathrm{LN}(\boldsymbol{X}^g)\right) + \boldsymbol{X}^t, \\ \boldsymbol{X}_s^t &= \mathrm{FFN}\left(\mathrm{LN}\left(\boldsymbol{X}_s^{t'}\right)\right) + \boldsymbol{X}_s^{t'}, \end{aligned} \tag{8}$$

where $\mathrm{LN}(\cdot)$ is LayerNorm, and $\mathrm{Attention}(\cdot)$ uses sparse attention (Zhu et al. 2021) for computational efficiency. $\boldsymbol{X}_s^t$ denotes the synergistic representation for task t. This interaction process is repeated for K layers to further enhance synergy across tasks. Finally, for each task, we apply a prediction head composed of convolutional layers to perform channel projection and produce the final output.

4.4 Theoretical Analysis

Backbone Efficiency: Existing methods such as Polyhistor (Liu et al. 2022) and Hyperformer (Karimi Mahabadi et al. 2021) embed task-specific adapters sequentially into the backbone, requiring T backbone passes for T tasks, yielding an $O(T)$ complexity. In contrast, TDSA integrates lateral branches to produce task-specific features, ensuring constant backbone execution, *i.e.*, $O(1)$ *w.r.t.* task count T.

Parameter Scalability: As detailed in Section 4.2, TDSA shares W_d^s, W_u^s, and the skill matrix $\boldsymbol{S}$ across tasks. Only the gating layers $W_\gamma^t, W_\beta^t \in \mathbb{R}^{R \times M}$ are task-specific, with small M (default 10). TSAI's scale fusion grows linearly with T, whereas branch synergy remains task-independent. Thus, *TDSS* introduces minimal task-wise parameter overhead. As shown in Fig. 2(e), it uses significantly fewer trainable parameters than full fine-tuning and MTLoRA (Agiza, Neseem, and Reda 2024).

5 Experiment

5.1 Experimental Setups

Dataset: We conduct benchmark experiments on the PASCAL-Context (Everingham et al. 2010) and NYUD (Silberman et al. 2012) datasets. PASCAL-Context contains 4,998 training and 5,105 testing images, covering five tasks: 21-class semantic segmentation, 7-class human part segmentation, surface normal estimation, edge detection, and saliency estimation. NYUD includes 765 training and 654 testing samples, involving 19-class semantic segmentation, surface normal estimation, and depth estimation.

Metrics: Following prior works (Liu et al. 2022; Agiza, Neseem, and Reda 2024), we use mean IoU (mIoU) for semantic segmentation, human part segmentation, and saliency estimation, and mean error (mErr) for surface normal and depth estimation. We also report $\Delta\mathbf{m}$ (Agiza, Neseem, and Reda 2024) to compare PEFT-based methods with single-task full fine-tuning (Liu et al. 2022).

Implementation Details: For a fair comparison, we conduct all experiments on the same device using PyTorch. we adopt the loss setup from (Lu et al. 2024). In default settings, we set the number of skills to $M{=}10$, the low-rank dimension to $R{=}C/4$, and stack $K{=}4$ synergy modules.

5.2 Baselines

We compare baseline methods across three categories:

A). **Fine-tuning baselines:** (1) *Single-task full fine-tuning*, with separate encoders and decoders per task; (2) *Multi-task full fine-tuning*, with a shared encoder and task-specific decoders; and (3) *Decoder-only fine-tuning*, where only decoders are updated.

B). **PEFT-based methods:** Adapter (He et al. 2022), LoRA (Hu et al. 2022), BitFit (Zaken, Goldberg, and Ravfogel 2022), VL-Adapter (Yi-Lin Sung 2022), and VPT (Jia et al. 2022) (shallow/deep). Originally for single-task setups, they are adapted to multi-task settings via shared or task-dedicated modules.

C). **Advanced multi-tasking PEFT methods:** HyperFormer (Karimi Mahabadi et al. 2021), Polyhistor (Liu et al. 2022), VMT-Adapter (Xin et al. 2024a), MTLoRA (Agiza, Neseem, and Reda 2024), DiTASK (Mantri et al. 2025), and TADFormer (Baek et al. 2025), specifically designed for multi-task learning.

For fair comparison, we use the Swin-Tiny backbone (Liu et al. 2021) pre-trained on *Imagenet-1K* dataset for both competing methods and our approach.

5.3 Main Results

We first evaluate on the PASCAL-Context dataset, with results shown in Tab. 1. Multi-task full fine-tuning outperforms single-task fine-tuning by $+2.23\%$ in $\Delta\mathbf{m}$ due to cross-task complementarity, but still requires a large number of trainable parameters (30.06M). PEFT-based methods reduce parameters but yield negative $\Delta\mathbf{m}$, indicating inferior performance to single-task baselines. The "sha" (shared) setting outperforms the "ded." (dedicated) one by leveraging task-sharing, while the latter decouples adaptations and ignores task interactions—consistent with our earlier observations. Though multi-task PEFT methods achieve $\Delta\mathbf{m} > 0$, some like HyperFormer demand 75.32M parameters—more than the backbone itself. In contrast, our proposed *TDSS* achieves the highest $\Delta\mathbf{m}$ ($+6.13\%$), surpassing the previous best TAD-Former ($+4.24\%$) with far fewer trainable parameters (3.52M vs. 7.38M). Notably, only *TDSS*, DiTask, and VMT-Adapter consider both backbone efficiency and

Method	Taxonomy	SemSeg (mIoU ↑)	Human Parts (mIoU ↑)	Saliency (mIoU ↑)	Normals (mErr ↓)	$\Delta\mathbf{m}(\%)$	#TP (M)	BEE	PTS
BASELINE FINE-TUNING									
Single-task full fine-tuning	–	67.21	61.93	62.35	17.97	0	112.62	×	×
Multi-task full fine-tuning	–	68.71	62.13	64.18	17.35	+2.23	30.06	✓	×
Decoder-only fine-tuning	–	63.14	52.37	58.39	20.89	-11.02	2.55	✓	✓
PEFT-BASED									
Adapter(sha.) (He et al. 2022)	*ICLR 22*	70.21	59.15	62.29	19.26	-0.64	4.71	✓	✓
LoRA(sha.) (Hu et al. 2022)	*ICLR 22*	68.96	56.71	61.07	17.92	-1.90	2.68	✓	✓
BitFit(sha.) (Zaken, Goldberg, and Ravfogel 2022)	*ACL 22*	67.99	56.23	60.96	18.63	-3.49	2.63	✓	✓
VL-Adapter(sha.) (Yi-Lin Sung 2022)	*CVPR 22*	66.84	55.52	60.21	18.51	-4.33	2.62	✓	✓
VPT-shallow(sha.) (Jia et al. 2022)	*ECCV 22*	62.96	52.27	58.31	20.90	-11.18	0.63	✓	✓
VPT-deep(sha.) (Jia et al. 2022)	*ECCV 22*	64.35	52.54	58.15	21.07	-10.85	1.49	✓	✓
Adapter(ded.) (He et al. 2022)	*ICLR 22*	69.21	57.38	61.28	18.83	-2.71	11.24	×	×
LoRA(ded.) (Hu et al. 2022)	*ICLR 22*	70.12	57.73	61.90	18.96	-2.17	2.87	×	✓
BitFit(ded.) (Zaken, Goldberg, and Ravfogel 2022)	*ACL 22*	68.57	55.99	60.64	19.42	-4.60	2.85	×	✓
MULTI-TASK PEFT-BASED									
HyperFormer (Karimi Mahabadi et al. 2021)	*ACL 21*	71.43	<u>60.73</u>	<u>65.54</u>	17.77	+2.64	75.32	×	×
Polyhistor (Liu et al. 2022)	*NIPS 22*	70.87	59.54	65.47	17.47	+2.34	8.96	×	×
VMT-Adapter (Xin et al. 2024a)	*AAAI 24*	<u>71.60</u>	60.67	64.02	<u>16.41</u>	+3.96	3.68	✓	✓
MTLoRA (Agiza, Neseem, and Reda 2024)	*CVPR 24*	67.90	59.84	65.40	16.60	+2.55	8.34	✓	×
DiTASK (Mantri et al. 2025)	*CVPR 25*	70.09	59.03	64.55	17.47	+1.45	1.61	✓	✓
TADFormer (Baek et al. 2025)	*CVPR 25*	70.82	60.45	**65.88**	16.48	+4.24	7.38	✓	×
***TDSS*(Ours)**	*AAAI 26*	**73.14**	**63.06**	64.86	**14.59**	**+8.37**	**3.52**	✓	✓

Table 1: Comparison of the proposed *TDSS* with existing fine-tuning methods on PASCAL-Context dataset, including quantitative performance on four tasks, improvement over single-task full fine-tuning ($\Delta\mathbf{m}$), number of trainable parameters (#TP), backbone execution efficiency (BFE), and parameter task scalability (PTS).

Method	NORMALS (mErr ↓)	DEPTH (mErr ↓)	SemSeg (mIoU ↑)	#TP (M)	$\Delta\mathbf{m}(\%)$
Single-task	22.43	0.814	36.73	82.85	0
Multi-task	22.86	0.689	33.77	28.29	+1.79
Decoder-only	30.03	0.858	27.78	0.30	-21.22
MTLoRA	26.62	0.626	34.92	7.81	-0.17
TADFormer	26.47	**0.593**	**42.15**	6.47	<u>+7.09</u>
***TDSS*(Ours)**	**20.72**	<u>0.598</u>	<u>40.75</u>	**3.27**	**+15.03**

Table 2: Comparison results on the NYUD dataset.

parameter scalability.Similarly, as shown in Tab. 2, the supplementary results on the NYUD dataset further confirm that *TDSS* achieves the best overall performance, notably outperforming MTLoRA and TADFormer in normal estimation.

As shown in Tab. 3, under the same computational settings, our *TDSS* model demonstrates significantly better efficiency than the strongest baselines, MTLORA and TADFormer. It requires fewer trainable parameters and, due to the lateral topology of TDSA that avoids gradient propagation through the backbone, also consumes less GPU memory. While *TDSS* has a slightly longer training time than MTLORA and TADFormer, it yields significantly higher $\Delta\mathbf{m}$ performance than both.

Method	#TP (M)	Memory(GB)	Time(s/batch)	$\Delta\mathbf{m}(\%)$
Single-task	112.62	21.15	0.667	0
MTLORA	8.34	<u>18.22</u>	**0.232**	+2.55
TADFormer	<u>7.38</u>	20.30	<u>0.376</u>	<u>+4.24</u>
***TDSS*(Ours)**	**3.52**	**14.65**	0.380	**+8.37**

Table 3: Comparison of model efficiency of TDSS, MTLORA, and TADFormer on the PASCAL-Context dataset.

5.4 Ablation Study

We conducted ablation studies to evaluate the effectiveness and design choices of the proposed method in this section.

Ablation on TDSA: The proposed TDSA module enables task-specific adaptation of backbone features, with corresponding ablation results shown in Tab. 5. We first evaluate its integration position within the backbone. Variants inserting TDSA only after attention or FFN layers reduce $\Delta\mathbf{m}$ from $+8.37\%$ to $+7.41\%$ and $+7.19\%$, respectively. Removing the skill-based adaptation components—Skill Representation Experts (SRE) and Dynamic Task Gating (DTG)—also reduces performance (from $+8.37\%$ to $+7.85\%$), underscoring the effectiveness of our TDSA.

Ablation on TSAI: We separately investigate the effects of scale fusion and branch synergy in TSAI, as shown in Fig. 3. Specifically, we explore variants with shared vs. task-specific scale fusion modules, as well as the interaction

Module	#TP (M)	$\Delta\mathbf{m}(\%)$
Baseline	112.62	0
+TDSA	5.56	+0.94
+TSAI	1.15	+5.76
TDSS	3.52	**+8.37**

(a) Overall ablation.

Skills	#TP (M)	$\Delta\mathbf{m}(\%)$
$M = 5$	3.42	+8.12
$M = 10$	3.52	**+8.37**
$M = 15$	3.62	+8.13
$M = 20$	3.72	+8.23

(b) Skill number M.

Scale R	#TP (M)	$\Delta\mathbf{m}(\%)$
C	10.61	+8.67
$C/2$	5.89	+8.76
$C/4$	3.52	**+8.37**
$C/8$	2.34	+7.61

(c) Low-rank dimension R.

Layers	#TP (M)	$\Delta\mathbf{m}(\%)$
$K = 1$	3.39	+5.29
$K = 2$	3.43	+7.05
$K = 4$	3.52	**+8.37**
$K = 6$	3.61	+9.18

(d) Synergistic modules K.

Table 4: Ablation studies of *TDSS* on the PASCAL-Context Dataset. Trainable parameters (#TP) and $\Delta\mathbf{m}(\%)$ are reported.

Attention	FFN	SRE	DTG	#TP (M)	$\Delta\mathbf{m}(\%)$
✔		✔	✔	2.34	+7.41
	✔	✔	✔	2.34	+7.19
✔	✔			3.32	+7.85
✔	✔	✔	✔	3.52	**+8.37**

Table 5: Ablation experiment of TDSA module.

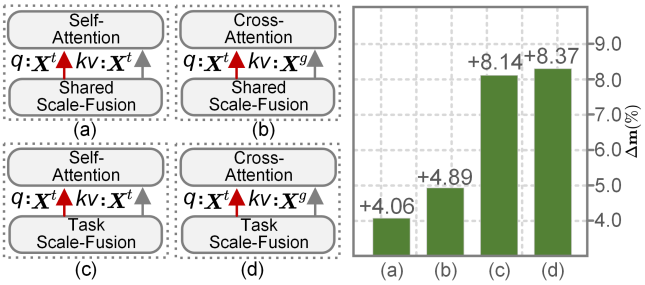

Figure 3: Ablation experiment of TSAI module.

between task-specific and global generic features. Results show that using task-specific scale fusion along with interaction with the global features yields the best performance.

Overall ablation: We progressively integrate TDSA and TSAI into the baseline to assess their effectiveness. As shown in Table 4(a), applying TDSA alone to adapt the pretrained backbone with the vanilla SegFormer (Xie et al. 2021) decoder yields a $\Delta\mathbf{m}$ improvement of $+0.94\%$. In contrast, feeding pretrained features directly into TSAI without TDSA achieves a larger gain of $+5.76\%$. When both components are combined, the performance boost reaches $+8.37\%$. These results highlight the importance of both modules and their complementarity.

Hyper-parameter analysis: We first investigate the number of skills M. Too few skills cannot support effective multi-task adaptation, while too many introduce redundancy. As shown in Tab. 4(b), performance increases with M and peaks at $M = 10$. Next, we study the compressed feature dimension R, which controls the low-rank representation and reduces trainable parameters. Tab. 4(c) shows that setting $R = C/4$ offers a good trade-off between performance and efficiency. Finally, we evaluate the stacking times K in TSAI, which influence the interaction between task-generic and task-specific features. A larger K improves synergy but increases computation. We set $K = 4$ as the optimal balance, as shown in Tab. 4(d).

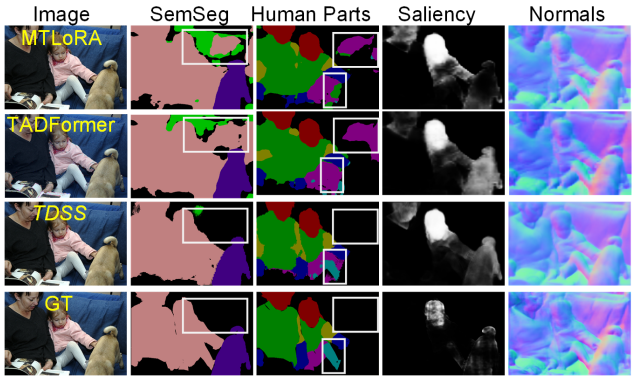

Figure 4: Qualitative comparison of multi-task prediction results of MTLoRA, TADFormer and *TDSS*.

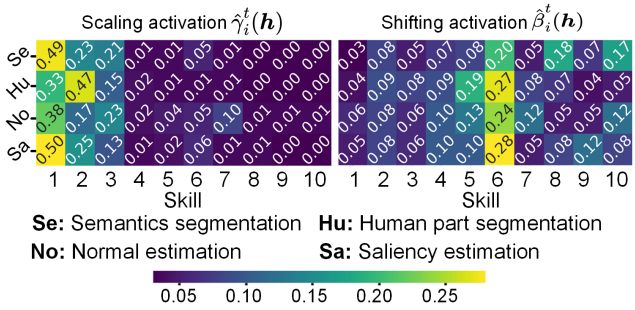

Figure 5: The averaged skill activation of different tasks.

5.5 Visualization

Fig. 4 presents a qualitative comparison between TDSS, MT-LoRA (Agiza, Neseem, and Reda 2024), TADFormer (Baek et al. 2025), and the ground truth. Our method clearly produces more accurate and reasonable predictions across different tasks. Fig. 5 visualizes the learned relationships between tasks and skills within TDSA. Specifically, we compute the average activation of each skill across tasks using the TDSA module in the attention layer of the deepest Swin block. The results show that certain skills—such as the first three scale-aware skills and the sixth shift-aware skill—are shared across tasks, reflecting task-generic adaptability. Meanwhile, each task shows distinct activation patterns, indicating task-specific skill usage. This balance of shared and specialized skills aligns well with our intuition.

6 Conclusion

In this paper, we introduce *TDSS*, a novel PEFT-based multi-task dense prediction framework. To address the limitations of existing PEFT/multi-task PEFT methods in terms of backbone execution efficiency, task parameter scalability, and cross-task fine-tuning interactions, we propose two key components: Task-Dynamic Skill Adapters (TDSA) and Task-Synergistic Adaptation Interaction (TSAI). Extensive experiments demonstrate that our method outperforms existing approaches and provides an efficient and scalable solution. More broadly, TDSA can serve as a plug-and-play PEFT module for multi-task learning beyond vision, while TSAI offers a promising approach for multimodal feature fusion. A potential limitation is that the model's performance on heterogeneous tasks (*e.g.*, dense prediction and object detection) remains unclear due to the lack of suitable datasets. We leave this for future work.

References

Agiza, A.; Neseem, M.; and Reda, S. 2024. MTLoRA: Low-Rank Adaptation Approach for Efficient Multi-Task Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 16196–16205.

Baek, S.; Lee, S.; Jo, H.; Choi, H.; and Min, D. 2025. TADFormer: Task-Adaptive Dynamic Transformer for Efficient Multi-Task Learning. *arXiv preprint arXiv:2501.04293*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Cai, W.; Jiang, J.; Wang, F.; Tang, J.; Kim, S.; and Huang, J. 2024. A survey on mixture of experts. *arXiv preprint arXiv:2407.06204*.

Chotard, P.; Touvron, H.; Caron, M.; Lacroix, J.; Bojanowski, P.; Douze, M.; Jégou, H.; and Misra, I. 2023. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research (TMLR)*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Ding, N.; Qin, Y.; Yang, G.; Wei, F.; Yang, Z.; Su, Y.; Hu, S.; Chen, Y.; Chan, C.-M.; Chen, W.; et al. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3): 220–235.

Everingham, M.; Van Gool, L.; Williams, C. K.; Winn, J.; and Zisserman, A. 2010. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision (IJCV)*.

Fifty, C.; Amid, E.; Zhao, Z.; Yu, T.; Anil, R.; and Finn, C. 2021. Efficiently identifying task groupings for multi-task learning. *Advances in Neural Information Processing Systems*, 34: 27503–27516.

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.

He, J.; Zhou, C.; Ma, X.; Berg-Kirkpatrick, T.; and Neubig, G. 2022. Towards a Unified View of Parameter-Efficient Transfer Learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

He, K.; Fan, H.; Wu, Y.; Xie, S.; and Girshick, R. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 9729–9738.

Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; De Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; and Gelly, S. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2790–2799. PMLR.

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

Jia, M.; Tang, L.; Chen, B.-C.; Cardie, C.; Belongie, S.; Hariharan, B.; and Lim, S.-N. 2022. Visual Prompt Tuning. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Karimi Mahabadi, R.; Ruder, S.; Dehghani, M.; and Henderson, J. 2021. Parameter-efficient Multi-task Fine-tuning for Transformers via Shared Hypernetworks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Lian, D.; Zhou, D.; Feng, J.; and Wang, X. 2022. Scaling & shifting your features: A new baseline for efficient model tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Liu, Y.-C.; Ma, C.-Y.; Tian, J.; He, Z.; and Kira, Z. 2022. Polyhistor: Parameter-Efficient Multi-Task Adaptation for Dense Vision Tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

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 International Conference on Computer Vision (ICCV)*.

Lu, Y.; Sirejiding, S.; Bayramli, B.; Huang, S.; Ding, Y.; and Lu, H. 2024. Task Indicating Transformer for Task-Conditional Dense Predictions. In *ICASSP*, 3625–3629. IEEE.

Mantri, K. S. I.; Schönlieb, C.-B.; Ribeiro, B.; Baskin, C.; and Eliasof, M. 2025. DiTASK: Multi-Task Fine-Tuning with Diffeomorphic Transformations. *arXiv preprint arXiv:2502.06029*.

Silberman, N.; Hoiem, D.; Kohli, P.; and Fergus, R. 2012. Indoor segmentation and support inference from rgbd images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 746–760. Springer.

Sung, Y.-L.; Cho, J.; and Bansal, M. 2022. Lst: Ladder side-tuning for parameter and memory efficient transfer learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 35: 12991–13005.

Vandenhende, S.; Georgoulis, S.; Gansbeke, W. V.; Proesmans, M.; Dai, D.; and Gool, L. V. 2021a. Multi-Task Learning for Dense Prediction Tasks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Vandenhende, S.; Georgoulis, S.; Van Gansbeke, W.; Proesmans, M.; Dai, D.; and Van Gool, L. 2021b. Multi-task learning for dense prediction tasks: A survey. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Xie, E.; Wang, W.; Yu, Z.; Anandkumar, A.; Alvarez, J. M.; and Luo, P. 2021. SegFormer: Simple and efficient design for semantic segmentation with transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Xin, Y.; Du, J.; Wang, Q.; Lin, Z.; and Yan, K. 2024a. Vmt-adapter: Parameter-efficient transfer learning for multi-task dense scene understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 38, 16085–16093.

Xin, Z.; Sirejiding, S.; Lu, Y.; Ding, Y.; Wang, C.; Alsarhan, T.; and Lu, H. 2024b. TFUT: Task fusion upward transformer model for multi-task learning on dense prediction. *Computer Vision and Image Understanding*, 244: 104014.

Xu, Y.; Li, X.; Yuan, H.; Yang, Y.; and Zhang, L. 2023. Multi-task learning with multi-query transformer for dense prediction. *IEEE Transactions on Circuits and Systems for Video Technology*.

Xu, Y.; Yang, Y.; and Zhang, L. 2023. DeMT: Deformable mixer transformer for multi-task learning of dense prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 37, 3072–3080.

Ye, H.; and Xu, D. 2024. InvPT++: Inverted Pyramid Multi-Task Transformer for Visual Scene Understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Yi-Lin Sung, M. B., Jaemin Cho. 2022. VL-Adapter: Parameter-Efficient Transfer Learning for Vision-and-Language Tasks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yoo, M.; Cho, S.; and Woo, H. 2022. Skills regularized task decomposition for multi-task offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35: 37432–37444.

Yu, B. X.; Chang, J.; Liu, L.; Tian, Q.; and Chen, C. W. 2022. Towards a Unified View on Visual Parameter-Efficient Transfer Learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Zaken, E. B.; Goldberg, Y.; and Ravfogel, S. 2022. BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Zhang, J.; Fan, J.; Ye, P.; Zhang, B.; Ye, H.; Li, B.; Cai, Y.; and Chen, T. 2025. BridgeNet: Comprehensive and Effective Feature Interactions via Bridge Feature for Multi-Task Dense Predictions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhang, Y.; and Yang, Q. 2018. An overview of multi-task learning. *National Science Review*, 5(1): 30–43.

Zhang, Y.; and Yang, Q. 2021. A survey on multi-task learning. *IEEE transactions on knowledge and data engineering*, 34(12): 5586–5609.

Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; and Dai, J. 2021. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *Proceedings of the International Conference on Learning Representations (ICLR)*.