VMT-Adapter: Parameter-Efficient Transfer Learning for Multi-Task Dense Scene Understanding

Yi Xin[1,2], Junlong Du[2], Qiang Wang[2], Zhiwen Lin[2], Ke Yan[2*]

[1]State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
[2]Youtu Lab, Tencent
xinyi@smail.nju.edu.cn, {jeffdu, albertqwang, xavierzwlin, kerwinyan}@tencent.com

Abstract

Large-scale pre-trained models have achieved remarkable success in various computer vision tasks. A standard approach to leverage these models is to fine-tune all model parameters for downstream tasks, which poses challenges in terms of computational and storage costs. Recently, inspired by Natural Language Processing (NLP), parameter-efficient transfer learning has been successfully applied to vision tasks. However, most existing techniques primarily focus on single-task adaptation, and despite limited research on multi-task adaptation, these methods often exhibit suboptimal training and inference efficiency. In this paper, we first propose an once-for-all *Vision Multi-Task Adapter (VMT-Adapter)*, which strikes approximately $O(1)$ training and inference efficiency w.r.t task number. Concretely, *VMT-Adapter* shares the knowledge from multiple tasks to enhance cross-task interaction while preserves task-specific knowledge via independent knowledge extraction modules. Notably, since task-specific modules require few parameters, *VMT-Adapter* can handle an arbitrary number of tasks with a negligible increase of trainable parameters. We also propose *VMT-Adapter-Lite*, which further reduces the trainable parameters by learning shared parameters between down- and up-projections. Extensive experiments on four dense scene understanding tasks demonstrate the superiority of *VMT-Adapter(-Lite)*, achieving a 3.96% (1.34%) relative improvement compared to single-task full fine-tuning, while utilizing merely $\sim$ 1% (0.36%) trainable parameters of the pre-trained model.

1 Introduction

The pretrain-finetune paradigm has made significant strides in Natural Language Processing (NLP) (Devlin et al. 2018; Brown et al. 2020), Computer Vision (CV) (He et al. 2022b; Xin et al. 2023), and various other domains. Typically, given a pre-trained model, the conventional fine-tuning approach involves adjusting the entire model, that is, performing full fine-tuning for downstream tasks. However, as state-of-the-art pre-trained models expand to encompass billions or even trillions of parameters, the traditional method of full fine-tuning becomes increasingly untenable due to the immense computational and storage resource demands.

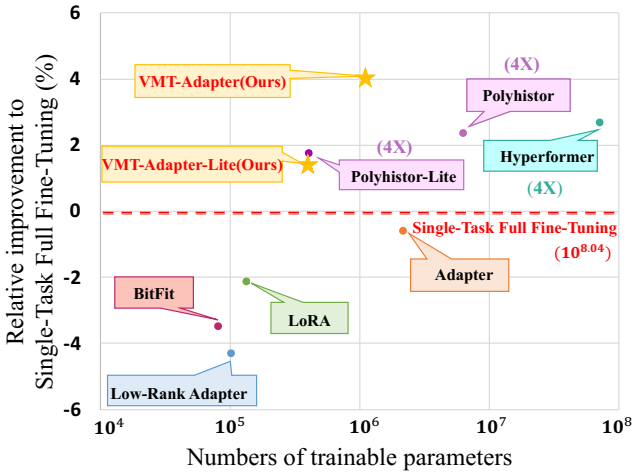


Figure 1: The trade-off between performance and trainable parameters of various parameter-efficient tuning methods. The result is the average performance on four dense tasks. $4X$ means that training and inference costs is four times.

To address these challenges, researchers have extensively investigated parameter-efficient transfer learning approaches (Houlsby et al. 2019; Hu et al. 2022; Zaken, Goldberg, and Ravfogel 2022), which strive to achieve an optimal balance between trainable parameters and performance on downstream tasks. Among these methods, the Adapter (Houlsby et al. 2019) and its variants (Karimi Mahabadi et al. 2021; mahabadi, Henderson, and Ruder 2021) have gained widespread adoption in the NLP domain and have been integrated into various architectures. The Adapter is a compact module incorporated into the intermediate layers of the model (as depicted in Figure 2), enabling comparable performance to full fine-tuning while only training a limited set of parameters. This innovative approach significantly reduces the computational and storage overhead, making it a more practical solution for large-scale models and diverse applications. Recently, Adaptformer (Chen et al. 2022) introduced adapter into computer vision, focusing on visual recognition tasks. Owing to its success, parameter-

*Corresponding author.

efficient transfer learning for vision has garnered considerable interest. While some subsequent studies (He et al. 2022c; Yu et al. 2022) have demonstrated promising results, these approaches primarily concentrate on single-task adaptation. For multiple downstream tasks, the individual training and storage of task-specific parameters prove to be inefficient, as the trainable parameters grows in proportion to the number of tasks. Consequently, further exploration is warranted to determine how pre-trained models can be transferred in a parameter-efficient manner to tackle more challenging, simultaneous multi-task vision problems, such as semantic segmentation, surface normal estimation, and saliency detection.

In multi-task parameter-efficient transfer learning, one intuitive approach is to incorporate adapters (or other single-task adaptation methods) for each task, referred to as *Multiple Adapter*, as illustrated in Figure 2a. However, this method results in a linear increase in the number of adapters with respect to the task number, leading to relatively high parameters. Moreover, the lack of task interactions among these adapters may lead to suboptimal performance. An alternative approach is to employ a shared adapter (or other single-task adaptation methods) across tasks, known as *Shared Adapter*, as illustrated in Figure 2b. While this method reduces trainable parameters and promotes interaction between tasks, it may lack task-specific knowledge for each individual task. Polyhistor (Liu et al. 2022) is the first to address parameter-efficient multi-task transfer learning, building upon the *Multiple Adapter*. It reduces trainable parameters by sharing adapter parameters across tasks and layers. However, there are still some limitations. Due to the design of task-independent adapters, the number of adapters, as well as training and inference costs, grow linearly with task number. Furthermore, Polyhistor involves numerous hyperparameters, which may hinder rapid application and deployment.

To overcome the aforementioned limitations, we propose a novel parameter-efficient *VMT-Adapter*, which comprises shared projections and knowledge extraction modules for adapting to multiple vision tasks. The *VMT-Adapter* unifies the learning of task-generic and task-specific representations within a single framework. Notably, the shared projections parameters are independent of task number, while the knowledge extraction modules merely contains extremely few parameters. As a result, *VMT-Adapter* is well-suited for multi-task transfer learning and can be seamlessly integrated into any transformer intermediate layer. Furthermore, drawing inspiration from Low-Rank Adapter (Yi-Lin Sung 2022) and Compactor (mahabadi, Henderson, and Ruder 2021), we develop the *VMT-Adapter-Lite*, which reduces the parameters to $\frac{1}{m}$ of the original (where m represents the dimension of the shared matrices, as detailed in Section 4.2). This is achieved by sharing learnable parameters between the down- and up-projections within the *VMT-Adapter*.

To establish a multi-task parameter-efficient transfer learning benchmark, Liu et al. (Liu et al. 2022) adapt various single-task adaptation methods from NLP to vision multi-task settings. Building upon their work, we extend these baselines and compare them with state-of-the-art approaches. Our experiments are conducted on the PASCAL-Context dataset, focusing on dense scene understanding tasks. As depicted in Figure 1, the results highlight the effectiveness of *VMT-Adapter(-Lite)*, striking an optimal balance between performance and trainable parameters.

Our main contributions are as follows:

- We propose *VMT-Adapter*, a novel parameter-efficient adapter for multi-task dense scene understanding, which learns task-generic and task-specific representations in an unified framework. To the best of our knowledge, *VMT-Adapter* is the first once-for-all adapter for vision multi-task learning with approximately $O(1)$ training and inference efficiency w.r.t task number.
- We propose a light-weight version termed *VMT-Adapter-Lite* via a parameter-sharing strategy to meet more efficient requirements.
- Experimental results on dense scene understanding tasks show that *VMT-Adapter(-Lite)* achieves competitive performance compared to the single-task full fine-tuning leveraging merely $\sim$ 1% (0.36%) of the pre-trained model parameters, as shown in Figure 1.

2 Related Work

Transformer in Vision. Transformer was initially introduced for Natural Language Processing (NLP) tasks, such as machine translation (Vaswani et al. 2017) and text generation (Devlin et al. 2018). Its remarkable success in these domains has inspired a shift in computer vision research towards Transformer-based models, beginning with the introduction of the Vision Transformer (ViT) (Dosovitskiy et al. 2020). Since then, a variety of Transformer-based models (Liu et al. 2021; Xie et al. 2021) have demonstrated impressive performance across a wide range of vision tasks, including image classification, semantic segmentation, object detection, etc. To further enhance the performance of downstream tasks and reduce the consumption of training resources, researchers have provided pre-trained ViT-based models on large-scale datasets, such as ImageNet (Russakovsky et al. 2015). By fine-tuning these pre-trained models on downstream tasks, researchers have consistently achieved faster convergence(He, Girshick, and Dollár 2019) and improved performance, showcasing the potential of Transformer-based models in computer vision.

Multi-Task Dense Scene Understanding. Multi-Task Learning (MTL) aims to simultaneously learn multiple tasks by sharing knowledge and computation. Numerous studies (Li, Liu, and Bilen 2022; Xu, Yang, and Zhang 2023) have demonstrated that various dense vision tasks, which are more challenging than image classification, can benefit from multi-task learning. Current research on multi-task dense scene understanding primarily focuses on model architecture design, which can be categorized into encoder-based and decoder-based methods. Encoder-based methods (Gao et al. 2019; Liu, Johns, and Davison 2019) are dedicated to designing task interaction modules embedded within the encoder, while decoder-based methods (Brüggemann et al. 2021; Zhang et al. 2021) focus on module design at the

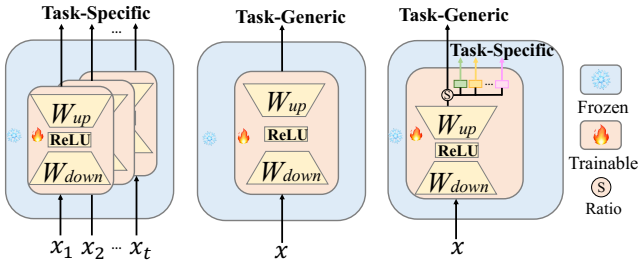


Figure 2: Illustration of (a) Multiple Adapter: inject separate adapters for each task and (b) Shared Adapter: t tasks share one adapter. (c) VMT-Adapter (Ours): t tasks share the down- and up-projections to enhance cross-task interaction and split the cross-task representations into task-generic and the input of task-specific knowledge extraction modules.

decoder stage. As vision pre-trained models become increasingly powerful, encoders directly adopt models such as the Vision Transformer (ViT) (Dosovitskiy et al. 2020) and Swin Transformer (Liu et al. 2021), which are pre-trained on large-scale image datasets. This approach leads to significant performance improvements and has become the mainstream for multi-task. However, the pretrain-finetune paradigm has two notable drawbacks. First, fine-tuning the encoder results in an unavoidable computational cost, as all model parameters must be adjusted. Second, a shared encoder across multiple tasks primarily extracts task-generic knowledge, while task-specific knowledge is neglected.

Parameter-Efficient Transfer Learning. Parameter Efficient Transfer Learning aims to adapt a pre-trained model to downstream tasks by training only a small number of parameters. The most straightforward approach involves freezing the pre-trained encoder and fine-tuning the last layer. However, this method often falls short of full fine-tuning in terms of downstream task accuracy. Several studies (Zaken, Goldberg, and Ravfogel 2022) have attempted to improve linear probing performance by updating the bias term in all layers. In contrast, other works (Chen et al. 2022; He et al. 2022a) have proposed to insert adapters into transformer layers for fine-tuning in a parameter-efficient manner. More recently, LoRA (Hu et al. 2022) introduced a method that generates two low-rank matrices, which are then multiplied and serve as a residual of attention weight matrices. While these methods have demonstrated satisfactory performance with few trainable parameters, they primarily focus on single-task transfer learning. For multi-task parameter-efficient transfer learning, Polyhistor (Liu et al. 2022) incorporated task-independent adapters into each transformer layer and further reduced trainable parameters through a parameter-sharing strategy between adapters across different layers and tasks. However, the multi-task adapter structure design itself has not been thoroughly explored, leading to separate adapters for each task and increased training and inference costs. Consequently, there is a linear relationship between the number of adapters and tasks.

3 Background

Hierarchical Vision Transformer. The hierarchical vision transformer, a variant of ViT (Dosovitskiy et al. 2020), is more effective on dense scene understanding tasks. In this work, we adopt SwinTransformer (Liu et al. 2021), which consists of four hierarchical blocks. Each block has several transformer layers, and each transformer layer comprises a Shifted Window-based Multi-head Self-Attention (SW-MSA) module and a MLP layer. Layer Normalization (LN) and residual connection are performed before and after the MLP and SW-MSA modules, as shown in Figure 3.

Adapter. Adapter (Houlsby et al. 2019) is a bottleneck-like architecture that consists of a down-projection layer $W_{down} \in R^{d\times k}$ and an up-projection layer $W_{up} \in R^{k\times d}$, where k reduces the dimension of representation d into low-rank ($k << d$). Additionally, there is a ReLU layer between the two layers for non-linear projection. In general, Adapter is injected into transformer layers and updated during training, while the parameters of the transformer are frozen. Given a specific input feature $x_\ell \in R^d$, Adapter generates the adapted features with a residual connection:

$$\tilde{x}_\ell = \mathrm{ReLU}\left(x_\ell \cdot W_{down}\right) \cdot W_{up} + x_\ell, \tag{1}$$

where $\tilde{x}_\ell$ is the output, and $W = [W_{down}; W_{up}^T] \in R^{d\times 2k}$ denotes all trainable parameters.

Kronecker Product. The Kronecker Product between matrix $\mathbf{A} \in R^{a_1\times a_2}$ and $\mathbf{B} \in R^{b_1\times b_2}$ yields a block matrix $W \in R^{w_1\times w_2}$, where $w_1 = a_1 \times b_1$ and $w_2 = a_2 \times b_2$. In W, each block (i, j) is the result of multiplying the element a_{ij} with matrix $\mathbf{B}$, which is defined as:

$$\mathbf{W} = \mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} a_{11}\mathbf{B} & \cdots & a_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathbf{B} & \cdots & a_{mn}\mathbf{B} \end{pmatrix}. \tag{2}$$

4 Method

Overall. Our goal is to efficiently adapt a large-scale pre-trained model for multi-task scenarios by introducing only a few additional trainable parameters. We enhance the multi-task parameter-efficient transfer learning process in two significant ways: (1) We design VMT-Adapter tailored for multi-task characteristics, considering both task-generic and task-specific representations while dramatically reducing the number of trainable parameters. To the best of our knowledge, this is the first once-for-all adapter structure designed for multi-task dense scene understanding. (2) A parameter-sharing method between down-projection and up-projection inside VMT-Adapters is proposed to further reduce the trainable parameters.

4.1 VMT-Adapter

The Architecture of VMT-Adapter. The core of VMT-Adapter is to share the knowledge from multiple tasks to enhance cross-task interaction meanwhile remain specific knowledge for each task with minimal additional trainable parameters. VMT-Adapter comprises a shared down-projection layer $W_{down} \in R^{d\times k}$, a nonlinear activation

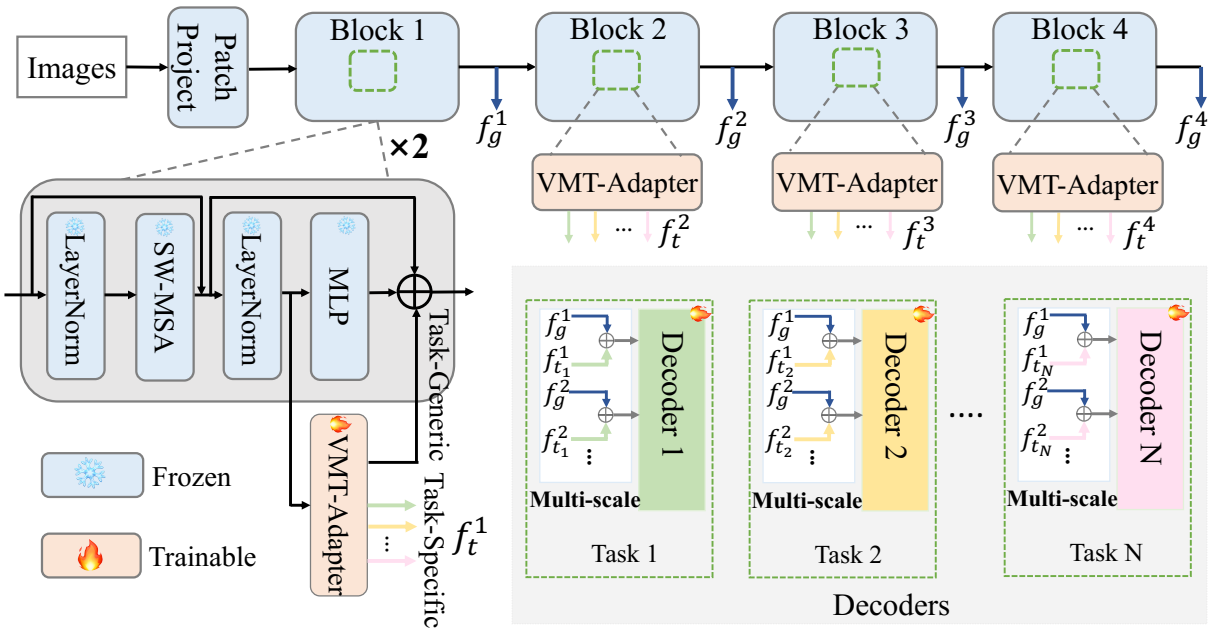


Figure 3: Illustration of SwinTransformer architecture and how to use VMT-Adapter. We insert VMT-Adapter in parallel to the MLP layer. The decoder of each task receives multi-scale information from different transformer blocks, including task-generic and task-specific representations. 🔥 represents trainable parameters, ❄ represents frozen parameters.

function ReLU, a shared up-projection $W_{up} \in R^{k \times d}$, and independent task-specific knowledge extraction modules. The shared projections facilitate interaction across multiple tasks, resulting in cross-task latent features F, as follows:

$$F = \text{ReLU}\left(x_\ell \cdot W_{down}\right) \cdot W_{up}, \tag{3}$$

where x_ℓ represents the input of ℓ-th layer. Subsequently, we partition the cross-task features into two components using a gating ratio. The first component represents the task-generic knowledge, while the second component is fed into the knowledge extraction modules. These modules perform scaling and shifting operations via dot product to obtain task-specific representations for each task, as follows:

$$f_{t_i} = \alpha_i \odot (s \cdot F) + \gamma_i, \tag{4}$$

where s is the gating ratio, $\alpha_i \in R^d$ and $\gamma_i \in R^d$ denote the scale and shift factors for i-th task. $\odot$ is the dot product.

How to use VMT-Adapter. For multi-task dense scene understanding, the decoder of each task receives multi-scale information from different transformer blocks. Therefore, task-specific representations generated by VMT-Adapter in each layer is directly transmitted to the decoder as a part of the multi-scale feature. The task-generic representations of the VMT-Adapter is added to the encoder and transferred to the decoder after each block. For the decoder of i-th task, the received multi-scale features from encoder are expressed as

$$F_i^{ms} = [(f_g^1 + f_{t_i}^1),\ (f_g^2 + f_{t_i}^2),\ (f_g^3 + f_{t_i}^3),\ (f_g^4 + f_{t_i}^4)], \tag{5}$$

where f_g^j and $f_{t_i}^j$ denote task-generic and task-specific representations from the j-th block.

4.2 A More Lightweight VMT-Adapter

In accordance with the VMT-Adapter architecture, the trainable parameters of the task-specific knowledge extraction module are almost negligible compared to the two shared projections. This implies that adjusting W_{down} and W_{up} can further minimize the additional parameters, thereby meeting more stringent efficiency requirements. Therefore, we propose a parameter-sharing method to further decrease the number of trainable parameters.

We generate W_{down} and W_{up} using a set of shared matrices $\mathbf{A} = \{A^i | 1 \le i \le m\}$, down-specific matrices $\mathbf{B}_{down} = \{B_{down}^i | 1 \le i \le m\}$, and up-specific matrices $\mathbf{B}_{up} = \{B_{up}^i | 1 \le i \le m\}$, by computing the sum of Kronecker products as follows:

$$W_{down} = \sum_{i=1}^{m} A^i \otimes B_{down}^i\,;\ W_{up} = \sum_{i=1}^{m} A^i \otimes B_{up}^i, \tag{6}$$

where $A^i \in R^{m \times m}$, $B_{down}^i \in R^{\frac{d}{m} \times \frac{k}{m}}$ and $B_{up}^i \in R^{\frac{k}{m} \times \frac{d}{m}}$. This shared strategy reduces the parameters of the VMT-Adapter to $\frac{1}{m}$ of the original, yielding a more light-weight Adapter termed *VMT-Adapter-Lite*.

4.3 Discussion

Trainable Parameter. We compare the trainable parameters of Multiple Adapter, Shared Adapter, and VMT-Adapter(-Lite). Taking SwinTransformer as an example, in which the dimension is d and the number of layers is L. Assuming that Adapter projects features from d-dim to k-dim, where $k = \frac{d}{\rho}$ and ρ is the down-projection ratio.

Method	# Trainable Params	# T&I Efficiency
Multiple Adapter	$\frac{2TL}{\rho}d^2$	$O(T)$
Shared Adapter	$\frac{2L}{\rho}d^2$	$O(1)$
VMT-Adapter	$\frac{2L}{\rho}d^2 + 2Td$	$O(1)$
VMT-Adapter-Lite	$m^3 + \frac{2L}{m\rho}d^2 + 2Td$	$O(1)$

Table 1: The trainable parameters and training/inference (T&I) efficiency comparisons.

Given T tasks, Multiple Adapter inserts T adapters in each transformer layer. Each adapter consists of $2kd$ parameters for the down- and up-projections. Therefore, the total number of trainable parameters for the SwinTransformer model with L layers is $TL \cdot 2kd$. Shared Adapter inserts single adapter in each transformer layer with $L \cdot 2kd$ parameters. VMT-Adapter introduces independent knowledge extraction module on the basis of Shared Adapter. For T tasks, the parameter number of this module is $2Td$. Consequently, the total parameters is $L \cdot 2kd + 2Td$, where $T << k$. VMT-Adapter-Lite shares the trainable parameter $\{A^i\}_{i=1}^m$ of m^3. The parameters for down- and up-projections are reduced to $\frac{kd}{m}$. For a SwinTransformer model with L layers, the total number of parameters of $m^3 + L \cdot \frac{2kd}{m} + 2Td$. Finally, k is replaced by $\frac{d}{\rho}$, and the results are presented in Table 1.

Training & Inference Efficiency. Since the Multiple Adapter establishes separate paths for each task during both training and inference, each sample must pass through the encoder T times to obtain predictions for T tasks. Consequently, the training and inference efficiency of the Multiple Adapter, including Polyhistor (Liu et al. 2022) and Hyperformer (Karimi Mahabadi et al. 2021), is $O(T)$. In contrast, the VMT-Adapter architecture allows only the task-generic representations to pass through the encoder, while the task-specific representations are computed in parallel. This results in an approximately $O(1)$ training and inference efficiency. In summary, our method not only strikes a balance between trainable parameters and performance but also achieves optimal training and inference efficiency.

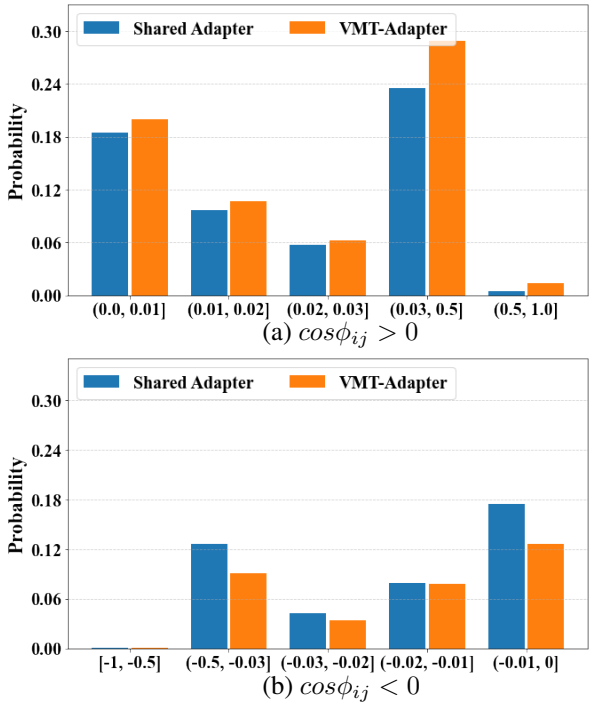
Gradient Analysis. For task t_i, the loss function is $\mathcal{L}_i(\theta_{sh}, \theta^i_{sp})$, where θ_{sh} are down- and up-projection parameters shared among all tasks and θ^i_{sp} are task-specific parameters. We denote the gradient of task t_i with respect to the shared parameters as $\mathbf{g}_i = \nabla_{\theta_{\text{sh}}} \mathcal{L}_i \left(\theta_{\text{sh}}, \theta^i_{sp}\right)$. The effect of this change on another task t_j is measured by:

$$\begin{aligned} \Delta\mathcal{L}_j &= \mathcal{L}_j \left(\theta_{\text{sh}} - \eta \mathbf{g}_i, \theta^j_{sp}\right) - \mathcal{L}_j \left(\theta_{\text{sh}}, \theta^j_{sp}\right) \\ &= -\eta \mathbf{g}_i \cdot \mathbf{g}_j + o(\eta), \end{aligned} \tag{7}$$

where η is a sufficiently small step size and the second equality is obtained by first order Taylor approximation. The model update for task t_i is considered to have a positive effect on task t_j when $\mathbf{g}_i \cdot \mathbf{g}_j > 0$. Therefore, the cosine similarity can be used to represent the relationship between tasks, specifically expressed as:

$$cos\phi_{ij} = \frac{\mathbf{g}_i \cdot \mathbf{g}_j}{\| \mathbf{g}_i \| \| \mathbf{g}_j \|}, \tag{8}$$

where ϕ_{ij} is the angle between $\mathbf{g}_i$ and $\mathbf{g}_j$. We then compute conflict angle for the shared parameters between any two tasks (four tasks on PASCAL-Context) in terms of $cos\phi_{ij}$. We then count and draw the distribution of $cos\phi_{ij}$ in all training iterations. As shown in Figure 4, it can be seen that (a) the probability of $cos\phi_{ij} > 0$ in VMT-Adapter has been improved, increasing the synergy between tasks and (b) the probability of $cos\phi_{ij} < 0$ in VMT-Adapter is reduced, reducing conflict between tasks.



(a) $cos\phi_{ij} > 0$

(b) $cos\phi_{ij} < 0$

Figure 4: The gradient conflicts distributions of the Shared Adapter and VMT-Adapter. (a) $cos\phi_{ij} > 0$ represents positive effect and (b) $cos\phi_{ij} < 0$ represents negative effect.

5 Experiment

5.1 Experimental Settings

Datasets and Downstream Tasks. To evaluate our proposed approach for multi-task dense scene understanding, we follow the prior works (Vandenhende et al. 2021; Liu et al. 2022) and conduct experiments on the PASCAL-Context (Vandenhende, Georgoulis, and Van Gool 2020) dataset. PASCAL-Context comprises 4,998 and 5,105 images in the training and testing splits, respectively. We evaluate on four dense prediction tasks, including 21-class semantic segmentation, 7-class human part segmentation, surface normals estimation, and saliency detection. We use the mean Intersection-over-Union (mIoU) metric to evaluate the semantic segmentation, human part segmentation and saliency detection tasks, while the mean error (mErr) metric is used for the surface normals estimation task.

	Trainable Parameters Encoder/All(M)	Performance of Each Downstream Task Seg. ↑	H.Part ↑	Sal. ↑	Normals ↓	Averaged Δ_{up}
Single-task Full Fine-tuning	110.07 / 112.62	67.21	61.93	62.35	17.97	0.00%
Fine-tuning Decoders	0.00 / 2.55	63.14	52.37	58.39	20.89	-11.02%
Multi-task Full Fine-tuning	27.51 / 30.06	68.71	62.13	64.18	17.35	2.23%
Multiple Bitfit (Zaken, Goldberg, and Ravfogel 2022)	0.30 / 2.85	68.57	55.99	60.64	19.42	-4.60%
Multiple Relative bias (Liu et al. 2021)	0.09 / 2.64	63.51	52.35	57.74	21.07	-11.40%
Multiple LoRA (Hu et al. 2022)	0.32 / 2.87	70.12	57.73	61.90	18.96	-2.17%
Multiple Adapter (He et al. 2022a)	8.69 / 11.24	69.21	57.38	61.28	18.83	-2.71%
Multiple Low-rank adapter (Yi-Lin Sung 2022)	0.34 / 2.89	68.31	56.53	60.29	19.36	-4.54%
Shared BitFit (Zaken, Goldberg, and Ravfogel 2022)	0.08 / 2.63	67.99	56.23	60.96	18.63	-3.49%
Shared Relative bias (Liu et al. 2021)	0.03 / 2.58	65.55	54.44	59.14	19.52	-9.56%
Shared LoRA (Hu et al. 2022)	0.13 / 2.68	68.96	56.71	61.07	17.92	-1.90%
Shared Adapter (He et al. 2022a)	2.17 / 4.71	70.21	59.15	62.29	19.26	-0.64%
Shared Low-rank adapter (Yi-Lin Sung 2022)	0.10 / 2.62	66.84	55.52	60.21	18.51	-4.33%
Hyperformer (Karimi Mahabadi et al. 2021)	72.77 / 75.32	**71.43**	**60.73**	**65.54**	17.77	**2.64**%
Polyhistor (Liu et al. 2022)	6.41 / 8.96	70.87	59.54	65.47	17.47	2.34%
Polyhistor-lite (Liu et al. 2022)	0.41 / 2.96	70.24	59.12	64.75	**17.40**	1.74%
VMT-Adapter	1.13 / 3.68	**71.60**	**60.67**	64.02	**16.41**	**3.96**%
VMT-Adapter-Lite	0.40 / 2.95	70.03	59.51	62.45	17.09	1.34%

Table 2: Experimental results on Multi-Task Transfer Learning. We use SwinTransformer-Tiny as the encoder. Δ_{up} represents relative improvement against the Single-task Full Fine-tuning. Results with the symbol ↑ / ↓ indicate higher/lower is better.

Model Architecture. For the encoder, we adopt the Swin-Transformer architecture and initialize it with the parameters pre-trained on ImageNet. For the decoder, we use the All-MLP decoder of Segformer (Xie et al. 2021) and set up a decode structure for each task. This structure consists of linear layers and bilinear upsampling layers, which enable efficient performance of dense scene understanding tasks. To ensure a fair comparison, we use the same encoder and decoder for all methods, and our experimental settings are consistent with the previous work Polyhistor(Liu et al. 2022).

Implementation Details. We conduct all experiments using the PyTorch toolkit on 4 NVIDIA V100 GPUs. For fair comparison, the hyper-parameters of all methods are the same. Specifically, we use batch size 12 and train for 60 epochs for each task. We employ the Adam optimizer with a learning rate $1e^{-4}$ and a weight decay $1e^{-4}$, and the learning rate is linearly decreased with respect to the iteration.

5.2 Baselines

Single-task Full Fine-tuning uses independent pre-trained encoder and decoder for each task, and updates the entire model. **Fine-tuning Decoders** updates the decoder parameters. **Multi-task Full Fine-tuning** uses a shared pre-trained encoder and separate decoder, then updates the entire model.

For single-task parameter-efficient transfer learning methods (*i.e.*, BitFit, Relative bias, LoRA, Adapter, and Low-rank Adapter), Polyhistor proposed a **Multiple** benchmark, which involves using a separate adaptation method for each task. In contrast, we extend a **Shared** benchmark, which involves sharing the adaptation method across multiple tasks.

For multi-task settings with multiple adapters, **Hyperformer** applies a hyper-network to produce weights for the adapters. **Polyhistor** further uses a scaling module to adapt the shared weights to hierarchical vision transformers.

5.3 Performance Comparisons

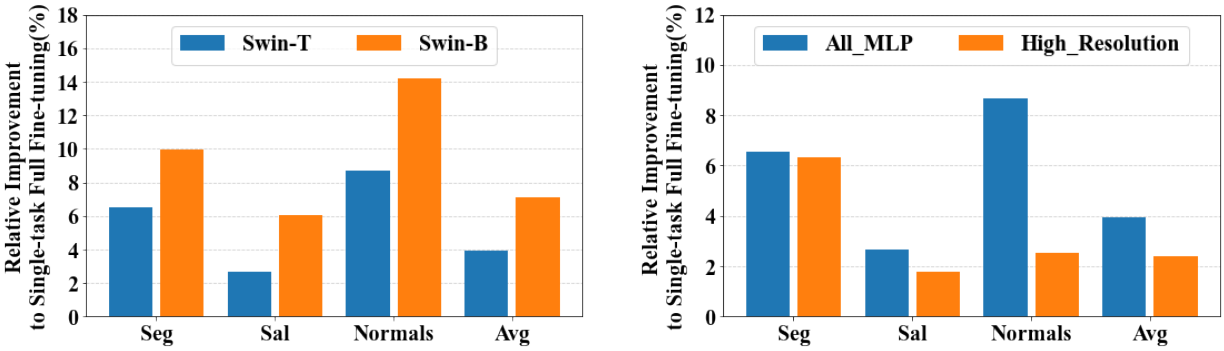
We evaluate all methods on four dense scene understanding tasks. In addition to showing the performance metrics of each task, we also evaluate the relative improvement against Single-Task Full Fine-tuning. At last, the number of trainable parameters is also reported.

As presented in Table 2 and Figure 1, *VMT-Adapter* outperforms all other methods with an average improvement of +3.96% on four downstream tasks against Single-task Full Fine-tuning, while leveraging only 1.13M trainable parameters. VMT-Adapter-Lite reduces the trainable parameters to 0.40M and still achieves a +1.34% improvement. Hyperformer performs the best besides our method with +2.64% improvement. However, Hyperformer introduces considerable additional parameters, i.e., 72.77M that even exceeds the encoder with 27.51M parameters, violating the principle of parameter-efficient transfer learning. Polyhistor performs well with a +2.34% average improvement on four downstream tasks, leveraging 6.41M trainable parameters (about 6 times as much as ours). Polyhistor-Lite further reduces the number of trainable parameters while ensuring performance. However, both methods use task-independent adapters, leading to $O(T)$ training and inference efficiency.

For single-task adaptation methods (*e.g.*, BitFit, LoRA, Adapter, and Low-rank Adapter), despite some methods having fewer trainable parameters, they cannot relieve the performance drop, resulting in a significant gap from -0.64% to -11.40% against Single-Task Full Fine-tuning. Additionally, it is worth noting that the Shared Adapter performs better than the Multiple Adapter, with even competitive results on par with Single-Task Full Fine-tuning, indicating that col-

	Ratio ρ	Trainable Parameters Encoder / All (M)	Performance of Each Downstream Task Seg. ↑	H.Seg. ↑	Sal. ↑	Normals ↓	Averaged Δ_{up}
Single-task Full Fine-tuning	-	110.07 / 112.62	67.21	61.93	62.35	17.97	0.00%
Fine-tuning Decoders	-	0.00 / 2.55	63.14	52.37	58.39	20.89	-11.02%
VMT-Adapter	$\rho = 1$	4.36 / 6.91	71.26	61.01	63.76	16.12	**4.27**%
VMT-Adapter	$\rho = 2$	2.21 / 4.56	70.96	60.82	64.01	16.39	**3.81**%
VMT-Adapter	$\rho = 4$	1.13 / 3.68	71.60	60.67	64.02	16.41	**3.96**%
VMT-Adapter	$\rho = 8$	0.59 / 3.14	71.34	59.95	63.48	16.82	**3.02**%
VMT-Adapter-Lite	$m = 3$	0.40 / 2.95	70.03	59.51	62.45	17.09	**1.34** %
VMT-Adapter-Lite	$m = 6$	0.22 / 2.77	69.85	59.38	61.95	17.53	**0.61** %
VMT-Adapter-Lite	$m = 12$	0.15 / 2.70	69.33	59.15	61.88	17.55	**0.06** %

Table 3: Ablation study on the ratio of down-projection of VMT-Adapter and parameter-sharing matrix dimension m of VMT-Adapter-Lite. We vary ratio ρ from 1 to 8 and m from 3 to 12 ($\rho = 4$).



(a) Different Encoder (b) Different Decoder

Figure 5: Ablation study on (a) different encoder and (b) different deocder. All results are produced by VMT-Adapter.

laboration between tasks can improve the performance.

5.4 Ablation Studies

Different Pre-trained Encoders. To verify the generalization of our method, we conduct experiments on a larger backbone, SwinTransformer-Base, pre-trained on ImageNet. The results are shown in Figure 5a. According to the results, the performance of four tasks is significantly improved compared with the Single-task Full Fine-tuning (+3.96% average improvement of SwinTransformer-Tiny and +7.10% of SwinTransformer-Base). We found that *VMT-Adapter* works better on larger backbone, showing that our method is applicable to various backbones for multi-task dense scene understanding.

Different Decoders. The decoder is an essential part for dense scene understanding tasks. In order to prove that our method is not benefiting from a specific decoder structure, we further conduct experiments on the high-resolution decoder of HRNet-V2 (Sun et al. 2019), which aggregates the representations at different resolutions. For a fair comparison, we use SwinTransformer-Tiny as the encoder structure. As shown in Figure 5b, *VMT-Adapter* achieves +3.96% and +2.41% relative improvements using All-MLP and High-Resolution decoders, respectively. Therefore, *VMT-Adapter* is flexible and can be adapted to various decoders.

Down-Projection Ratio. The down-projection ratio of our *VMT-Adapter* is a crucial hyper-parameter, thus we try different ratios $\rho = \frac{d}{k}$. As presented in Table 3, we vary the ratio ρ from 1 to 8 based on SwinTransformer-Tiny. The experimental results show that different ρ improves performance from 3.02% to 4.27%, using only 0.5% to 3.9% trainable parameters of Single-task Full Fine-tuning.

Parameter-Sharing Matrix Dimension. We also examine how our proposed *VMT-Adapter-Lite* performs when different parameter-sharing matrix dimensions are used. From Table 3, we observe that with the increase of m, the number of parameters gradually decreases, but the performance of multiple tasks also decreases. Therefore, in practical applications, we recommend setting m to 3, which can achieve a trade-off between performance and trainable parameters.

6 Conclusion

In this work, we propose *VMT-Adapter(-Lite)* for vision multi-task parameter-efficient transfer learning, which can simultaneously learn task-generic and task-specific representations from different transformer blocks. Compared with Single-task Full Fine-tuning and other parameter-efficient transfer learning methods, *VMT-Adapter(-Lite)* achieves favorable results while using a limited number of tunable parameters. In addition, *VMT-Adapter(-Lite)* is also optimal regarding training/inference costs. The potential limitation of *VMT-Adapter* is that the task-specific module parameters will be higher than the shared parameters when the number of tasks reaches thousands.

References

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)*.

Brüggemann, D.; Kanakis, M.; Obukhov, A.; Georgoulis, S.; and Van Gool, L. 2021. Exploring relational context for multi-task dense prediction. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Chen, S.; Ge, C.; Tong, Z.; Wang, J.; Song, Y.; Wang, J.; and Luo, P. 2022. Adaptformer: Adapting vision transformers for scalable visual recognition. In *Advances in Neural Information Processing Systems (NeurIPS)*.

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)*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Gao, Y.; Ma, J.; Zhao, M.; Liu, W.; and Yuille, A. L. 2019. Nddr-cnn: Layerwise feature fusing in multi-task cnns by neural discriminative dimensionality reduction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

He, K.; Chen, X.; Xie, S.; Li, Y.; Dollár, P.; and Girshick, R. 2022b. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

He, K.; Girshick, R.; and Dollár, P. 2019. Rethinking imagenet pre-training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

He, X.; Li, C.; Zhang, P.; Yang, J.; and Wang, X. E. 2022c. Parameter-efficient fine-tuning for vision transformers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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)*.

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)*.

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)*.

Li, W.-H.; Liu, X.; and Bilen, H. 2022. Learning multiple dense prediction tasks from partially annotated data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Liu, S.; Johns, E.; and Davison, A. J. 2019. End-to-end multi-task learning with attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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)*.

mahabadi, R. K.; Henderson, J.; and Ruder, S. 2021. Compacter: Efficient Low-Rank Hypercomplex Adapter Layers. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. In *International Journal of Computer Vision (IJCV)*.

Sun, K.; Zhao, Y.; Jiang, B.; Cheng, T.; Xiao, B.; Liu, D.; Mu, Y.; Wang, X.; Liu, W.; and Wang, J. 2019. High-resolution representations for labeling pixels and regions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

Vandenhende, S.; Georgoulis, S.; and Van Gool, L. 2020. MTI-Net: Multi-Scale Task Interaction Networks for Multi-Task Learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*.

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.; Yan, K.; and Ding, S. 2023. MmAP : Multi-modal Alignment Prompt for Cross-domain Multi-task Learning. In *arXiv preprint arXiv:2312.08636*.

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)*.

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)*.

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, X.; Zhou, L.; Li, Y.; Cui, Z.; Xie, J.; and Yang, J. 2021. Transfer vision patterns for multi-task pixel learning. In *Proceedings of the ACM Conference on Multimedia (MM)*.