PURDUE UNIVERSITY

UNIVERSITY OF CAMBRIDGE

INSIGHT Intelligent Systems Geometric Computing & Perception Lab

DiTask: Multi-Task Fine-Tuning with Diffeomorphic Transformations

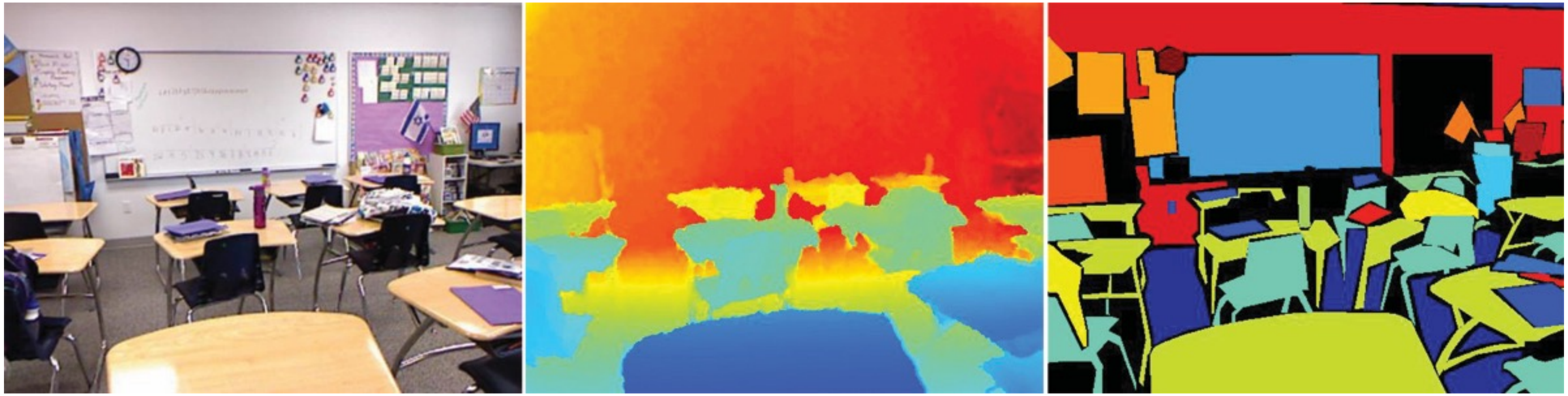
Krishna Sri Ipsit Mantri[1], Carola-Bibiane Schönlieb[2], Bruno Ribeiro[1], Chaim Baskin[3], Moshe Eliasof[2]
mantrik@purdue.edu, cbs31@cam.ac.uk, ribeirob@purdue.edu, chaimbaskin@bgu.ac.il, me532@cam.ac.uk

CVPR 2025

[1]Purdue University [2]University of Cambridge [3]Ben-Gurion University of the Negev

BGU Ben-Gurion University of the Negev

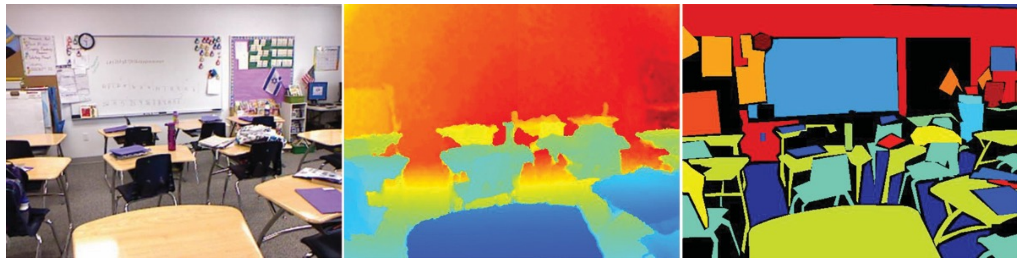
Multi-Task Learning (MTL) in Vision



https://cs.nyu.edu/~fergus/datasets/nyu_depth_v2.html

MTL in Vision: Fundamentals

- **Definition:** Learning multiple vision tasks simultaneously with shared representations
- Common Vision Tasks
  - Dense Prediction – Semantic segmentation, depth estimation, surface normal
  - Classification, detection, etc
- Benefits
  - Reduced computational resources
  - Improved generalization through shared knowledge
  - Single model for multiple downstream applications



MTL Architectures: Parameter Sharing

Hard Parameter Sharing

- Shared encoder backbone with task-specific decoders
- Most parameters are shared across tasks
- Efficient but prone to task interference

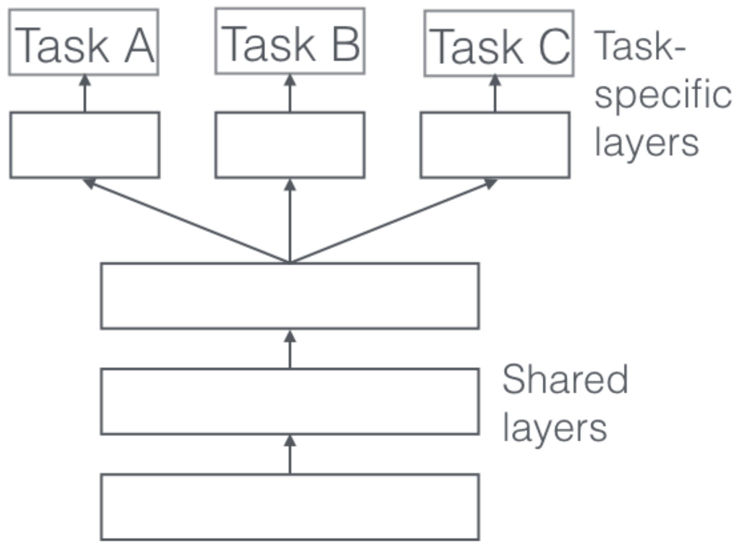
Soft Parameter Sharing

- Separate models with mechanisms to share information
- Task-specific backbones with regularization between parameters
- More flexible but less parameter efficient

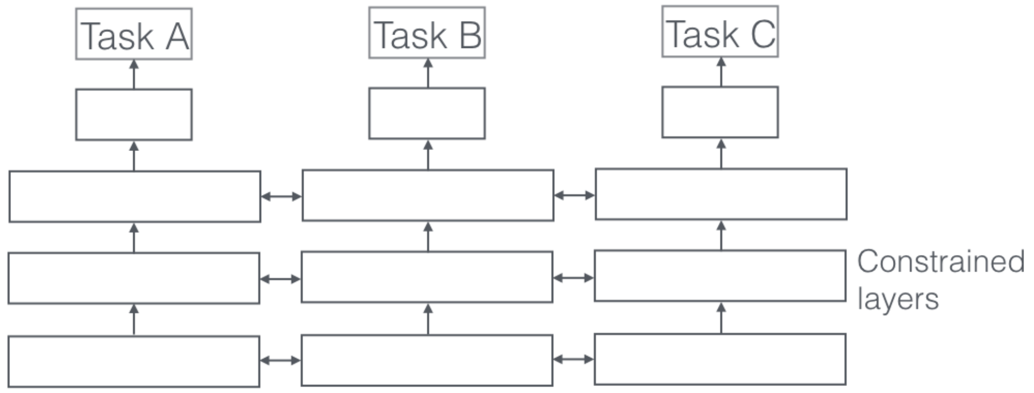
Sebastian Ruder, An Overview of Multi-Task Learning in Deep Neural Networks, arXiv preprint 1706.05098v1, 2017

MTL Architectures: Parameter Sharing

Hard Parameter Sharing



Soft Parameter Sharing



Sebastian Ruder, An Overview of Multi-Task Learning in Deep Neural Networks, arXiv preprint 1706.05098v1, 2017

Pre-trained Models & Fine-tuning for MTL

- **Why Pre-trained models?**
  - Learn general visual representations from large-scale datasets
  - Transfer learning enables adaptation to specialized tasks
  - Foundation models like ViTs encode rich visual knowledge
- **Why fine-tune instead of training from scratch?**
  - Vastly reduced computational cost
  - Fewer labeled examples needed
  - Better performance on downstream tasks
  - Preservation of learned visual features

Dosovitskiy et al, An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, ICLR 2021.

Parameter-Efficient Fine-Tuning Methods

- **Adapter-based Methods:** Insert trainable modules between frozen layers. Examples: Adapter, BitFit, VPT
- **Low-Rank Methods:** Constrain weight updates to low-dimensional subspaces. Examples: LoRA, DoRA, SVFT
- **Hypernetwork Approaches:** Generate task-specific parameters. Examples: HyperFormer, Polyhistor

Poth et al, Adapters: A Unified Library for Parameter-Efficient and Modular Transfer Learning, EMNLP 2023

Low-Rank Adaptation (LoRA)

- Constrain weight updates using low-rank decomposition
- Parameter reduction: $O(2rd) \ll O(d^2)$
- Memory and computation efficiency, no additional inference latency
- **Limitations in MTL:**
  - **Fixed subspace** constrains adaptations
  - **Competing task gradients** within limited space

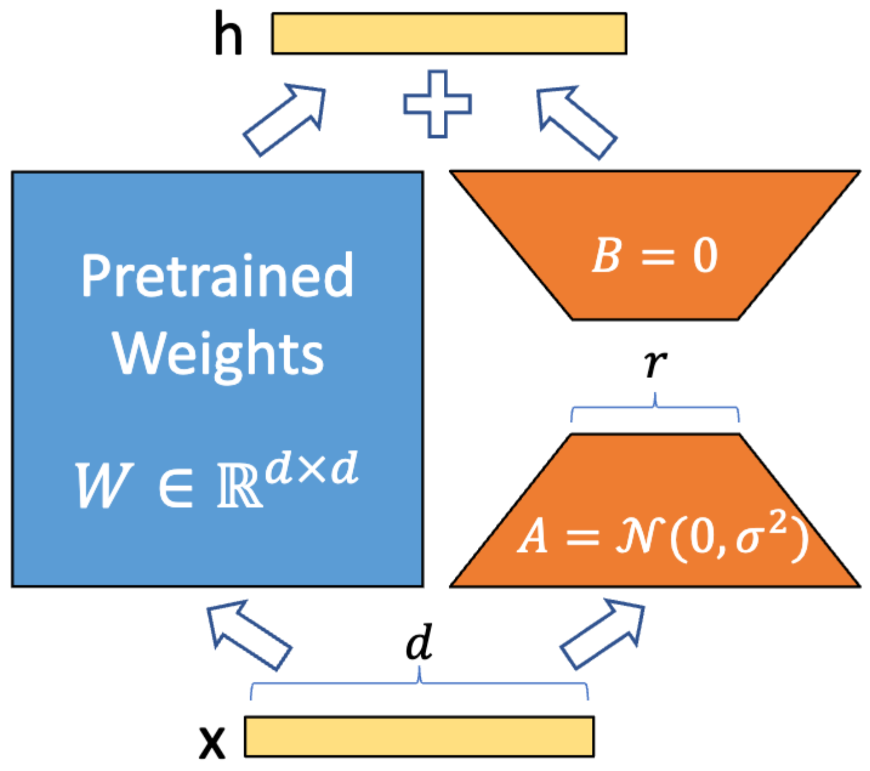
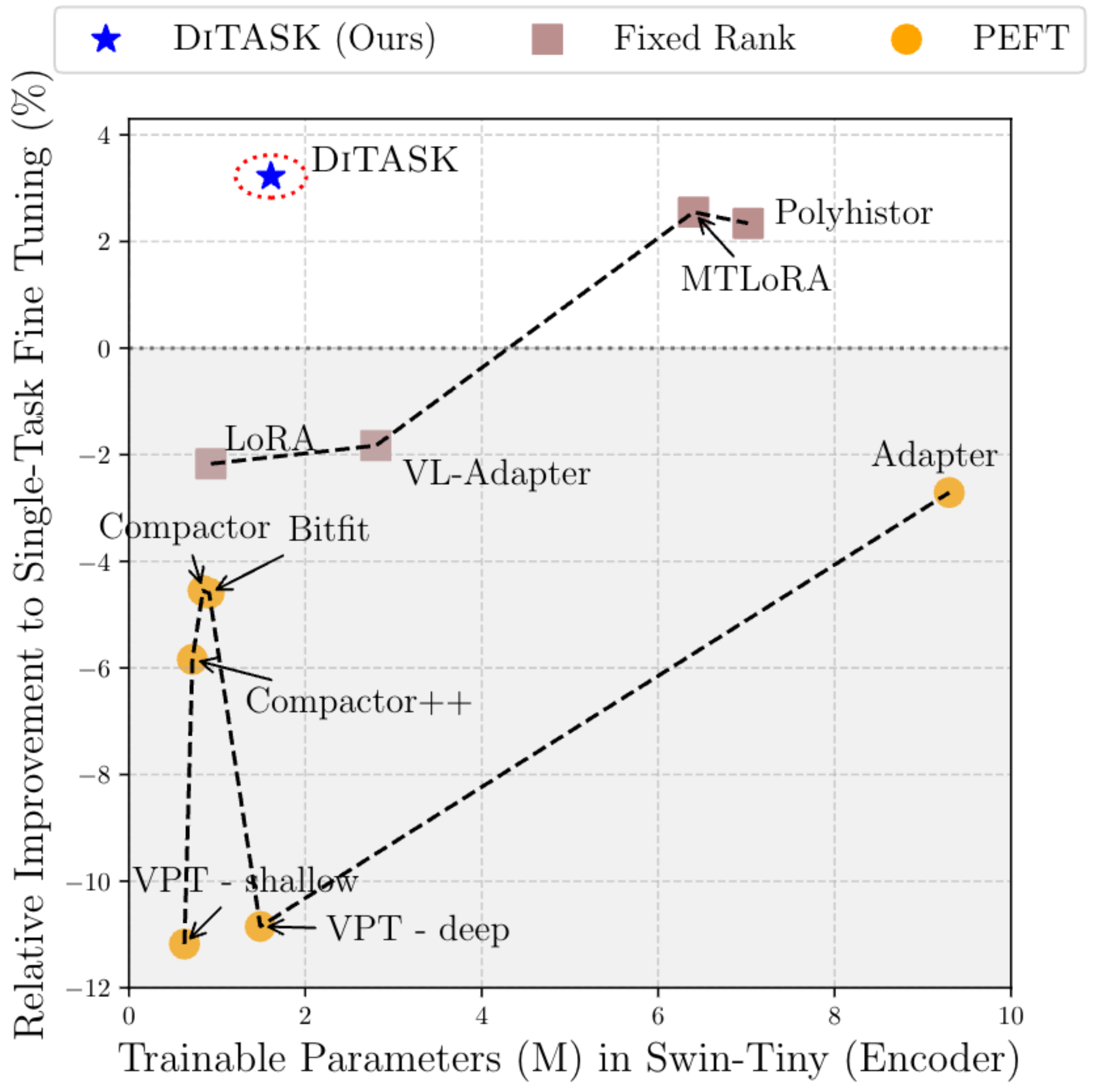

h
Pretrained Weights
$W \in \mathbb{R}^{d\times d}$
$B = 0$
r
$A = \mathcal{N}(0, \sigma^2)$
d
x

Figure 1: Our reparametrization. We only train A and B.

Hu et al. "LoRA: Low-rank adaptation of large language models.", ICLR 2022

Challenges in MTL Fine-tuning

- **Task Interference:** Competing gradient updates from different tasks
- **Negative Transfer:** When sharing hurts rather than helping
- **Optimization Difficulties:** Finding a single set of parameters optimal for all tasks
- **Efficiency-Performance Tradeoff:** Most methods require more parameters to match single-task performance



Yu et al, Gradient Surgery for Multi-Task Learning, NeurIPS 2020

How can we efficiently adapt pre-trained ViTs for MTL while preserving rich learned representations?

- ✓ Scalability with tasks
- ✓ Computational Efficiency
- ✓ Lower task interference

Our Key Insight

"Preserving the structure of pre-trained models is crucial"

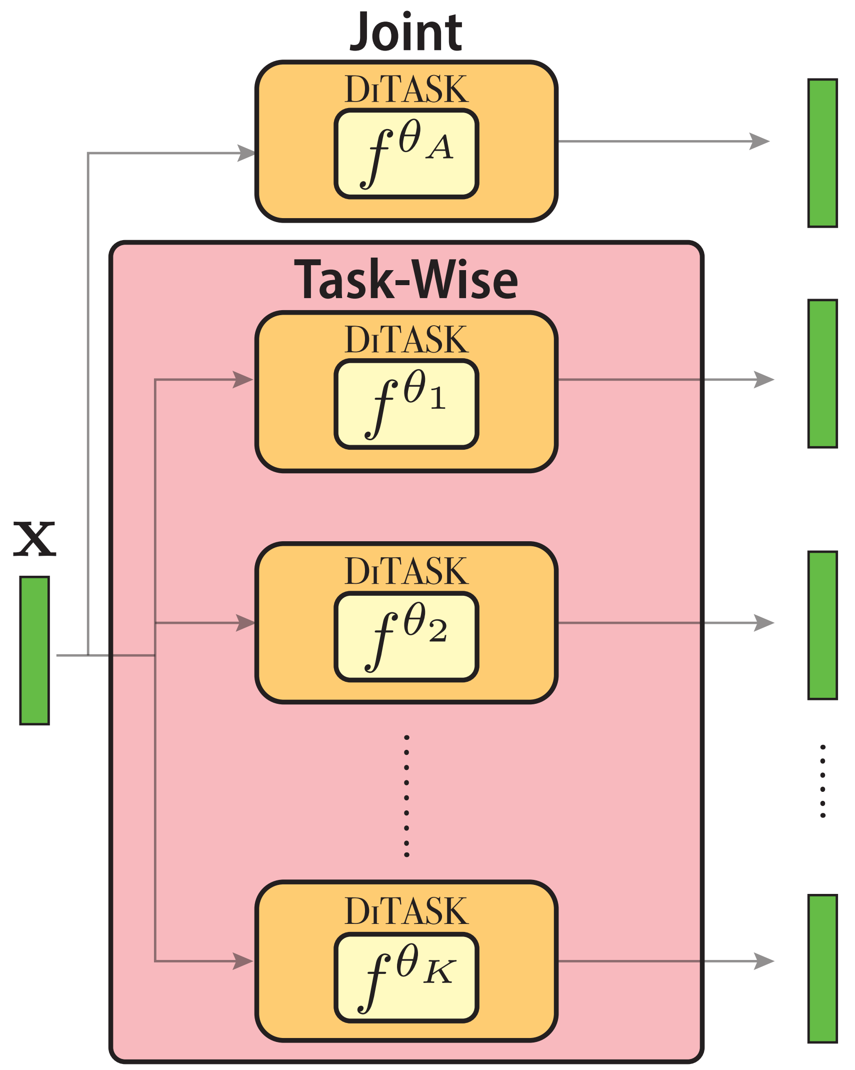
- Pre-trained weight matrices W encode essential visual knowledge through:
  - Singular Vectors (U, V) : Feature orientations/directions
  - Singular Values (Σ) : Feature importance

Conventional methods (e.g., LoRA) modify both, causing task interference

Our Solution: Preserve singular vectors while allowing flexible adaptation of singular values

DiTASK: Diffeomorphic Multi-TASK Learning

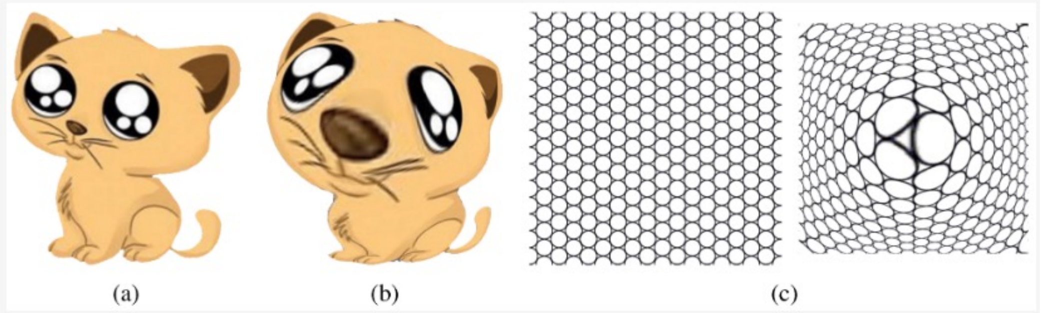
- Preserves pre-trained singular vectors (U, V) to maintain learned feature directions
- Transforms singular values (Σ) using learnable diffeomorphic transformations
  - Uses CPAB diffeomorphisms (parameter efficient)
- Enables both shared and task-specific adaptations



What are Diffeomorphisms?

- Smooth and invertible functions (e.g. Tanh, splines)
- Some of these functions are learnable

Figure 1. A diffeomorphism of an image caused by a quasiconformal mapping. The distortion of circles demonstrates the diffeomorphism of the original image. (**a**) Original image; (**b**) deformed image; (**c**) quasiconformal deformation.

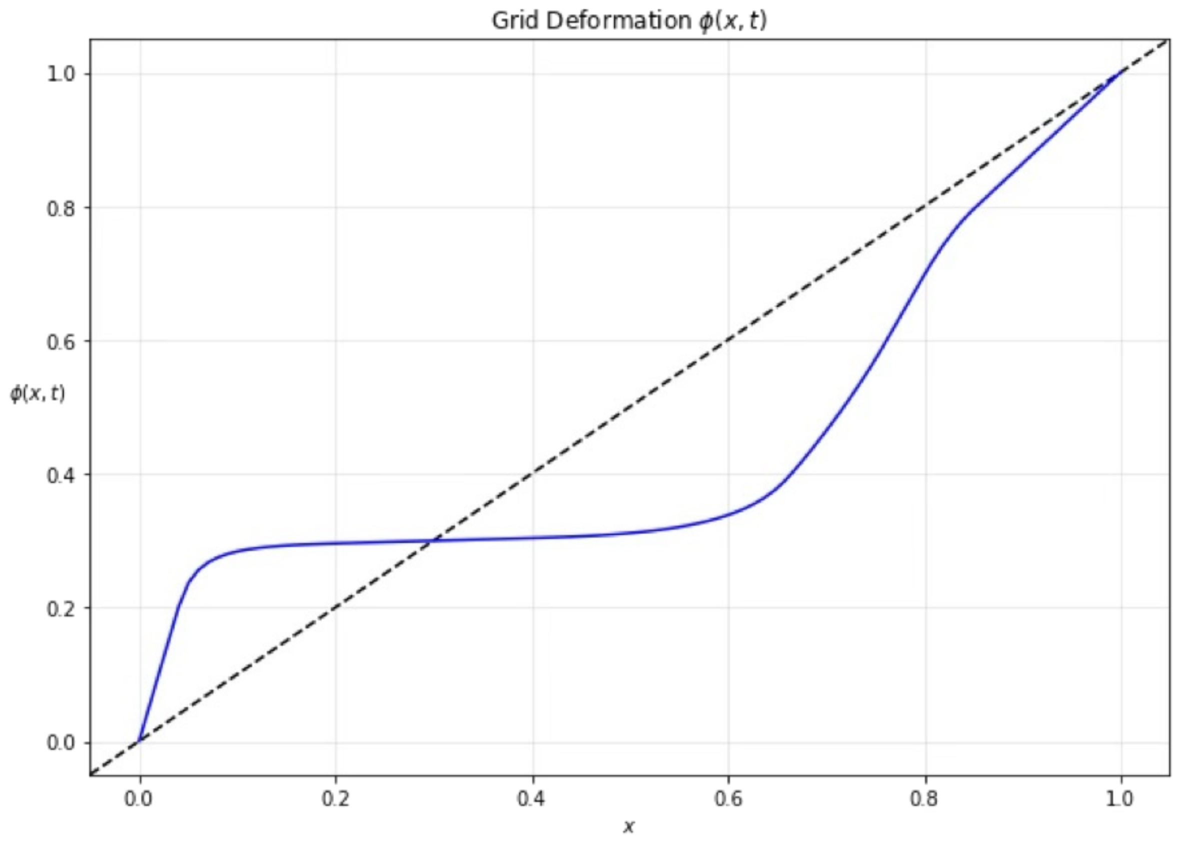


Why Diffeomorphisms?

- **Smooth**: We be used in gradient descent/end-to-end training
- **Invertible**: Well-behaved transformations, bounded, monotone
- **Flexible**: Good function approximators
- In DiTASK, we use **CPAB diffeomorphisms!**

Mantri et al, DiGRAF: Diffeomorphic Graph-Adaptive Activation Function, NeurIPS 2024

CPAB Diffeomorphisms



- **Diffeomorphism**: On a closed interval $\Omega = [a,b]$, any function $f : \Omega \to \Omega$ is a diffeomorphism if it is (1) bijective, (2) smooth and (3) admits a smooth inverse f^{-1}
- **CPAB Diffeomorphism**: A velocity field v^{θ} parameterized by $\theta \in \mathbb{R}^{N_P}$ that is **C**ontinuous and **P**iecewise **A**ffine on $N_P + 1$ subintervals (typically 32) of Ω, yields a diffeomorphism f^{θ} given by

$$f^{\theta}(x) = x + \int_{0}^{1} v^{\theta}\left(f^{\theta}(x,t)\right)\,dt$$

Freifeld et al. "Highly-expressive spaces of well-behaved transformations: Keeping it simple.", ICCV 2015

Freifeld et al. "Transformations based on continuous piecewise-affine velocity fields.", IEEE PAMI 2017

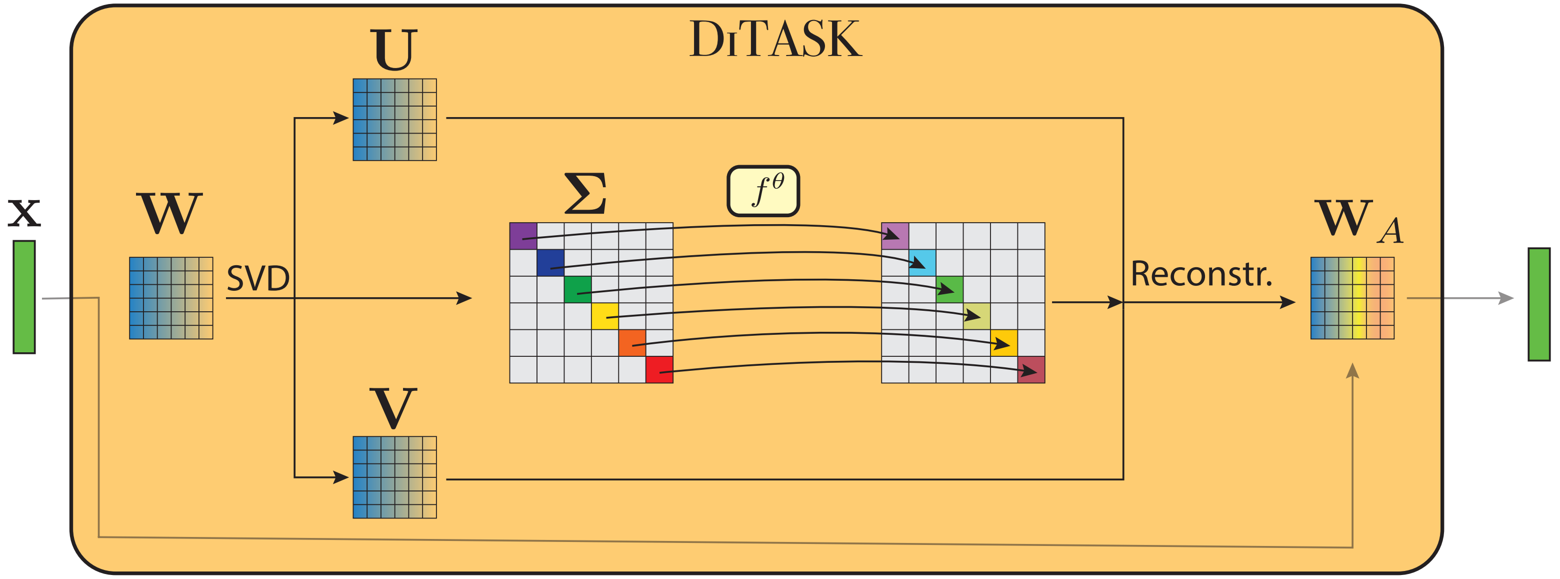
DiTASK Mechanism

For a pre-trained weight matrix

$$W = U\,\Sigma\,V^\top$$

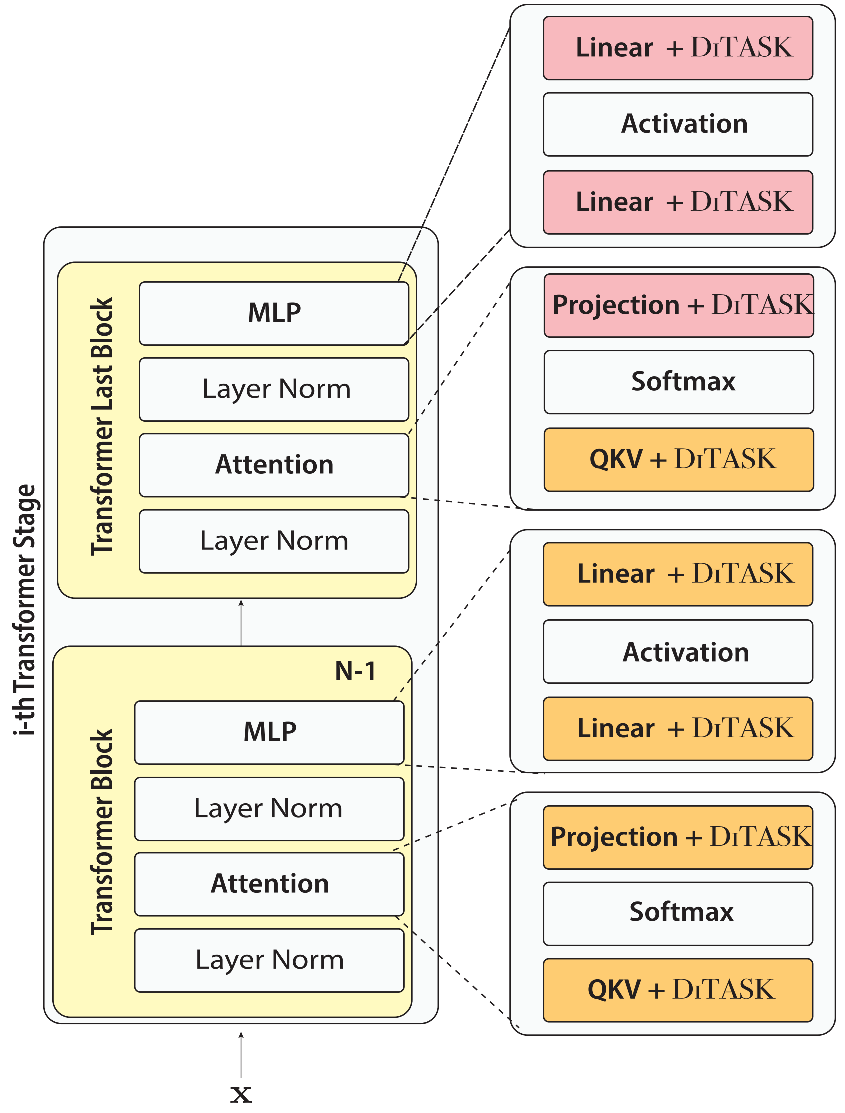
Advantages:

- ✓ Maintains geometric structure of pre-trained features
- ✓ Achieves full-rank gradient updates with minimal parameters
- ✓ Preserves feature importance hierarchies
- ✓ θ Learned end-to-end



$$W_A = U \begin{bmatrix} f^\theta(\sigma_1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & f^\theta(\sigma_d) \end{bmatrix} V^\top$$

DiTASK in Transformer Architecture



Experimental Setup

- **Architecture**: Shared Encoder + Task-specific decoder
- **Datasets**: PASCAL MTL, NYUDv2, VTAB
- **Backbones**: Swin family, Pyramid ViT
- **Baselines**: Full fine-tuning, Adapters, and Fixed-Rank Methods
- **Evaluation Metric for MTL**:

$$\Delta m = \frac{1}{T}\sum_{t=1}^{T}(-1)^{l_t}\frac{M_t - M_{s,t}}{M_{s,t}}$$

Where $l_t = 1$ for RMSE and $l_t = 0$ for mIOU, M_t is performance on task t and $M_{s,t}$ is the single-task baseline

Main Results

- Superior Performance-Efficiency Tradeoff
  - **+3.22%** improvement over single-task baselines
  - Uses **75% fewer parameters** than state-of-the-art methods
  - Achieves state-of-the-art on PASCAL MTL benchmark
- Memory and Computational Efficiency
  - **22%** faster inference
  - **500x** fewer adaptation parameters than MTLoRA

Agiza et al., MTLoRA: Low Rank Adaptation Approach for Efficient Multi-Task Learning, CVPR 2024

Performance Comparison

- DiTASK outperforms all adapter-based and most fixed-rank methods
- Achieves positive improvement over single-task baseline with the fewest trainable parameters
- Requires only **1.61M** trainable parameters vs **6.40M** for MTLoRA

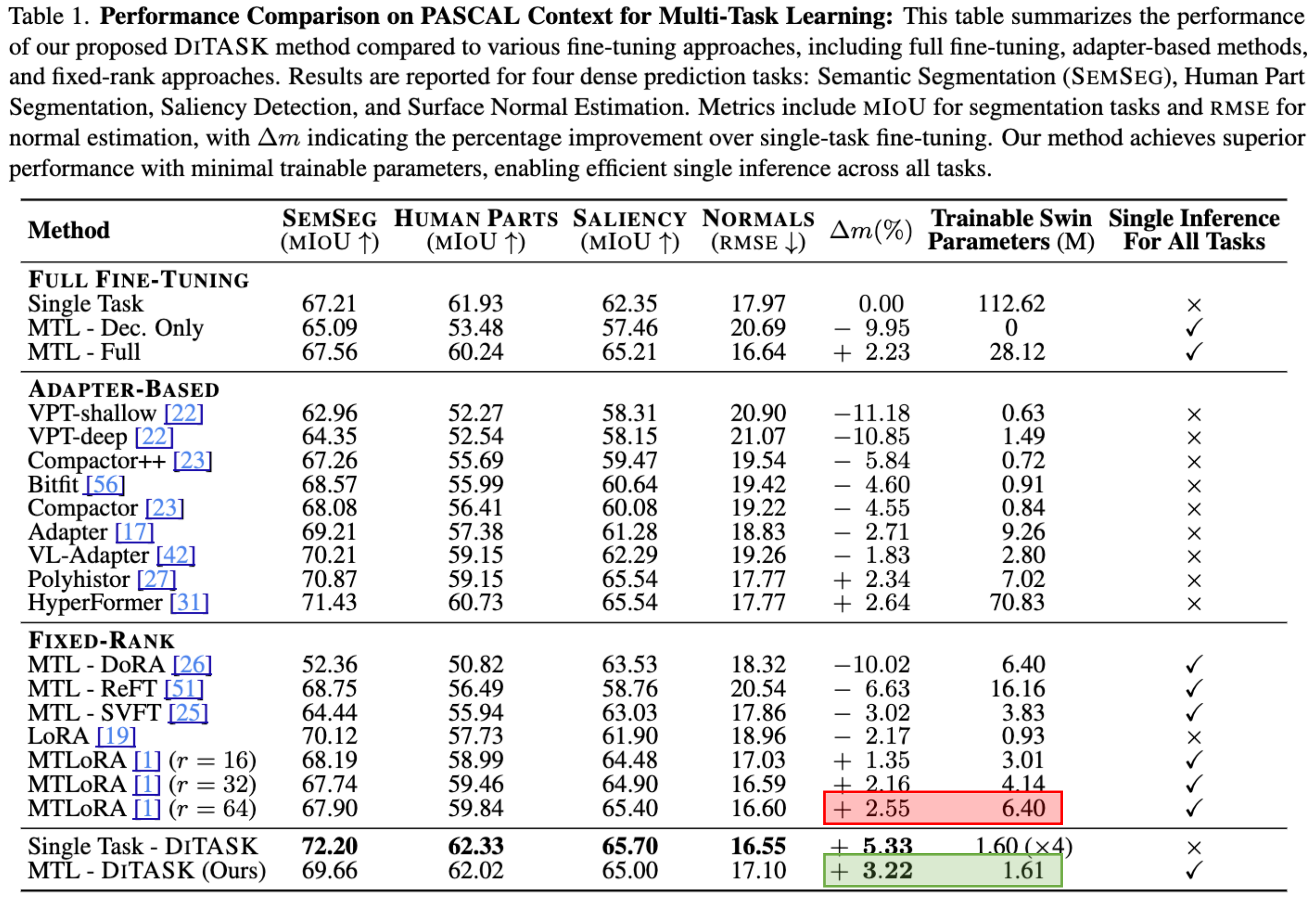


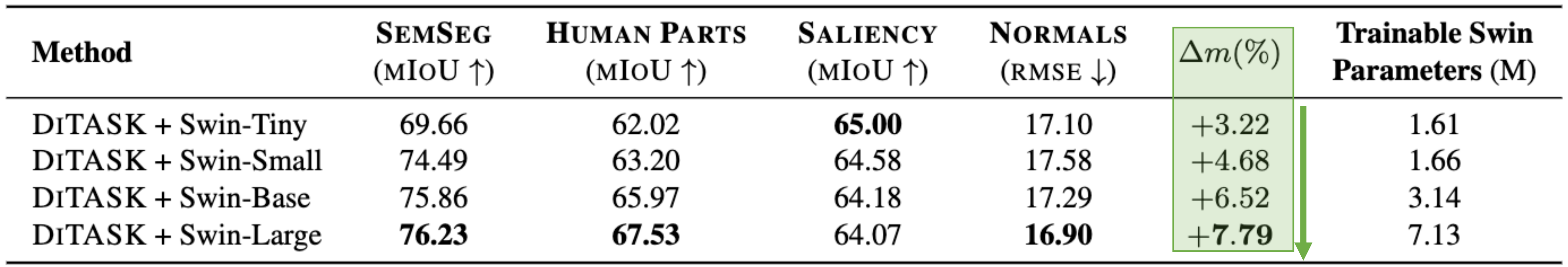
Table 1. **Performance Comparison on PASCAL Context for Multi-Task Learning:** This table summarizes the performance of our proposed DITASK method compared to various fine-tuning approaches, including full fine-tuning, adapter-based methods, and fixed-rank approaches. Results are reported for four dense prediction tasks: Semantic Segmentation (SEMSEG), Human Part Segmentation, Saliency Detection, and Surface Normal Estimation. Metrics include MIOU for segmentation tasks and RMSE for normal estimation, with Δm indicating the percentage improvement over single-task fine-tuning. Our method achieves superior performance with minimal trainable parameters, enabling efficient single inference across all tasks.

Method	SEMSEG (MIOU ↑)	HUMAN PARTS (MIOU ↑)	SALIENCY (MIOU ↑)	NORMALS (RMSE ↓)	$\Delta m(\%)$	Trainable Swin Parameters (M)	Single Inference For All Tasks
FULL FINE-TUNING							
Single Task	67.21	61.93	62.35	17.97	0.00	112.62	×
MTL - Dec. Only	65.09	53.48	57.46	20.69	− 9.95	0	✓
MTL - Full	67.56	60.24	65.21	16.64	+ 2.23	28.12	✓
ADAPTER-BASED							
VPT-shallow [22]	62.96	52.27	58.31	20.90	−11.18	0.63	×
VPT-deep [22]	64.35	52.54	58.15	21.07	−10.85	1.49	×
Compactor++ [23]	67.26	55.69	59.47	19.54	− 5.84	0.72	×
Bitfit [56]	68.57	55.99	60.64	19.42	− 4.60	0.91	×
Compactor [23]	68.08	56.41	60.08	19.22	− 4.55	0.84	×
Adapter [17]	69.21	57.38	61.28	18.83	− 2.71	9.26	×
VL-Adapter [42]	70.21	59.15	62.29	19.26	− 1.83	2.80	×
Polyhistor [27]	70.87	59.15	65.54	17.77	+ 2.34	7.02	×
HyperFormer [31]	71.43	60.73	65.54	17.77	+ 2.64	70.83	×
FIXED-RANK							
MTL - DoRA [26]	52.36	50.82	63.53	18.32	−10.02	6.40	✓
MTL - ReFT [51]	68.75	56.49	58.76	20.54	− 6.63	16.16	✓
MTL - SVFT [25]	64.44	55.94	63.03	17.86	− 3.02	3.83	✓
LoRA [19]	70.12	57.73	61.90	18.96	− 2.17	0.93	×
MTLoRA [1] ($r = 16$)	68.19	58.99	64.48	17.03	+ 1.35	3.01	✓
MTLoRA [1] ($r = 32$)	67.74	59.46	64.90	16.59	+ 2.16	4.14	✓
MTLoRA [1] ($r = 64$)	67.90	59.84	65.40	16.60	+ 2.55	6.40	✓
Single Task - DITASK	**72.20**	**62.33**	**65.70**	**16.55**	+ **5.33**	1.60 (×4)	×
MTL - DITASK (Ours)	69.66	62.02	65.00	17.10	+ **3.22**	1.61	✓

Mottaghi et al., The Role of Context for Object Detection and Semantic Segmentation in the Wild, CVPR 2014

DiTASK is more effective for larger models

Table 2. Effect of Backbone Size – Relative improvement increases with increasing capacity of the backbone



Method	SemSeg (mIoU ↑)	Human Parts (mIoU ↑)	Saliency (mIoU ↑)	Normals (RMSE ↓)	$\Delta m(\%)$	Trainable Swin Parameters (M)
DiTASK + Swin-Tiny	69.66	62.02	**65.00**	17.10	+3.22	1.61
DiTASK + Swin-Small	74.49	63.20	64.58	17.58	+4.68	1.66
DiTASK + Swin-Base	75.86	65.97	64.18	17.29	+6.52	3.14
DiTASK + Swin-Large	**76.23**	**67.53**	64.07	**16.90**	+**7.79**	7.13

Faster Inference, Lower Memory Footprint

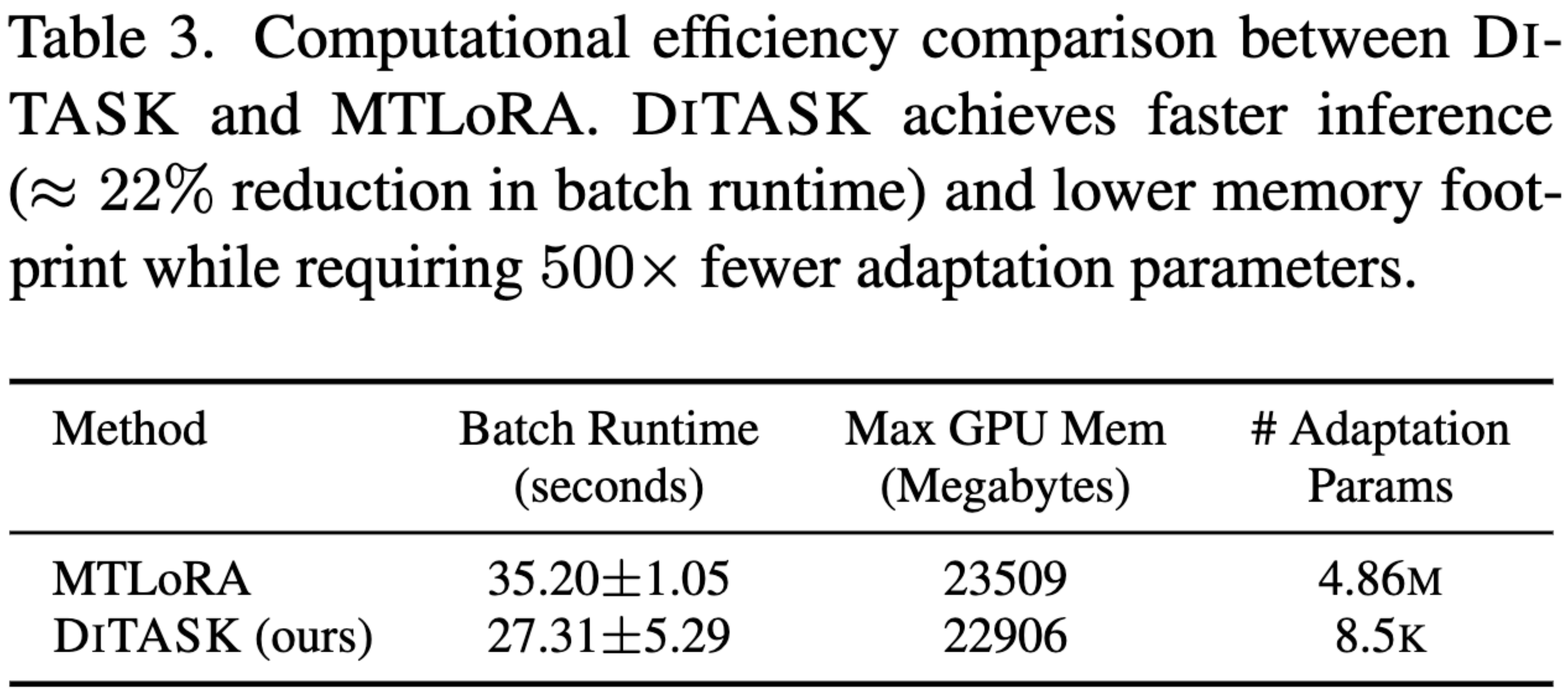
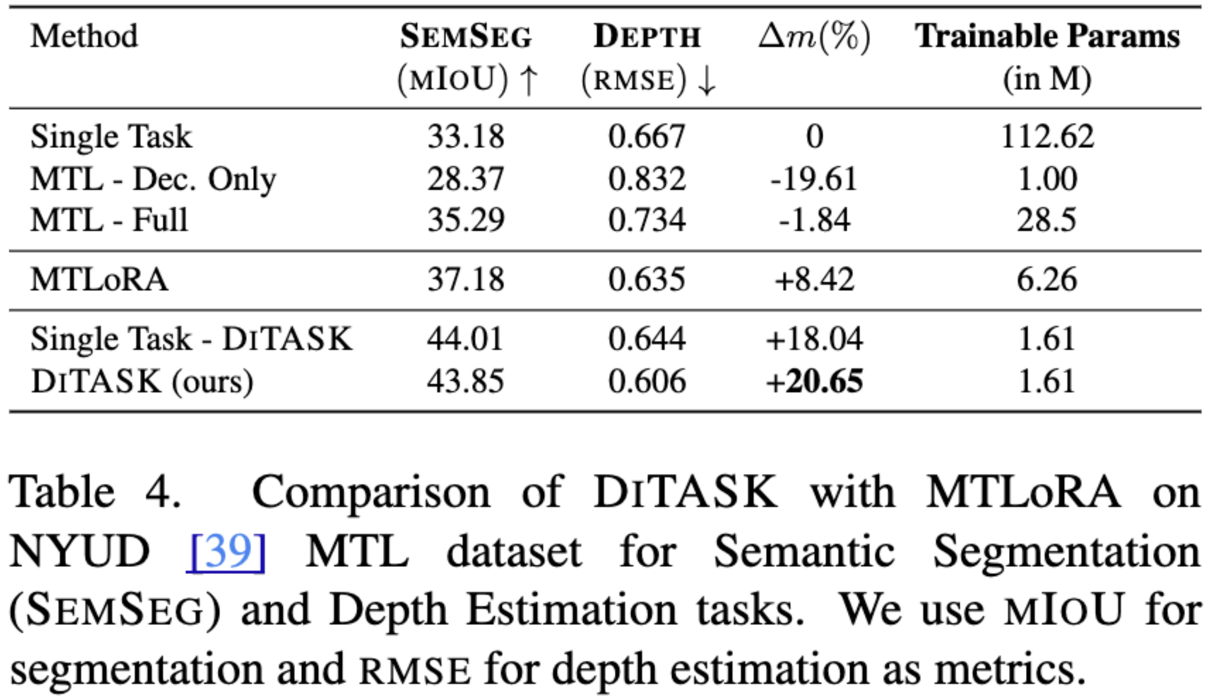


Table 3. Computational efficiency comparison between DiTASK and MTLoRA. DiTASK achieves faster inference ($\approx$ 22% reduction in batch runtime) and lower memory footprint while requiring 500$\times$ fewer adaptation parameters.

Method	Batch Runtime (seconds)	Max GPU Mem (Megabytes)	# Adaptation Params
MTLoRA	35.20$\pm$1.05	23509	4.86M
DiTASK (ours)	27.31$\pm$5.29	22906	8.5K

Results on NYUDv2

- > **2x** improvement over previous S.O.T.A
- **75% fewer** trainable parameters



Method	SEMSEG (MIOU) ↑	DEPTH (RMSE) ↓	$\Delta m(\%)$	**Trainable Params** (in M)
Single Task	33.18	0.667	0	112.62
MTL - Dec. Only	28.37	0.832	-19.61	1.00
MTL - Full	35.29	0.734	-1.84	28.5
MTLoRA	37.18	0.635	+8.42	6.26
Single Task - DITASK	44.01	0.644	+18.04	1.61
DITASK (ours)	43.85	0.606	**+20.65**	1.61

Table 4. Comparison of DITASK with MTLoRA on NYUD [39] MTL dataset for Semantic Segmentation (SEMSEG) and Depth Estimation tasks. We use MIOU for segmentation and RMSE for depth estimation as metrics.

Silberman et al., Indoor Segmentation and Support Inference from RGBD Images, ECCV 2012

Single-Task Generalization on VTAB

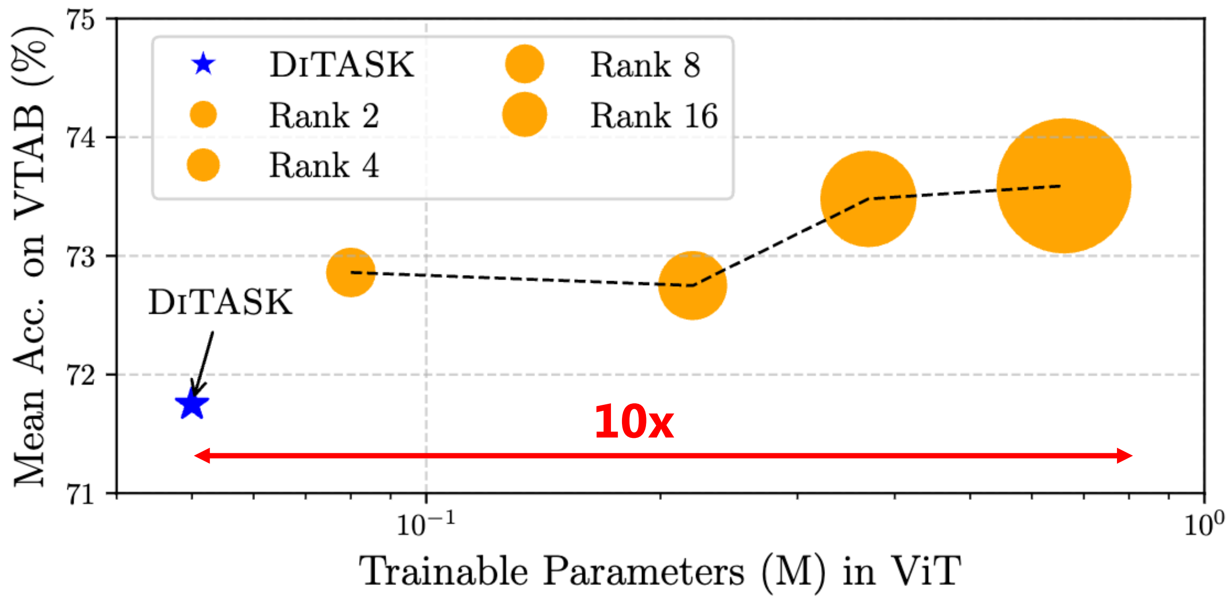
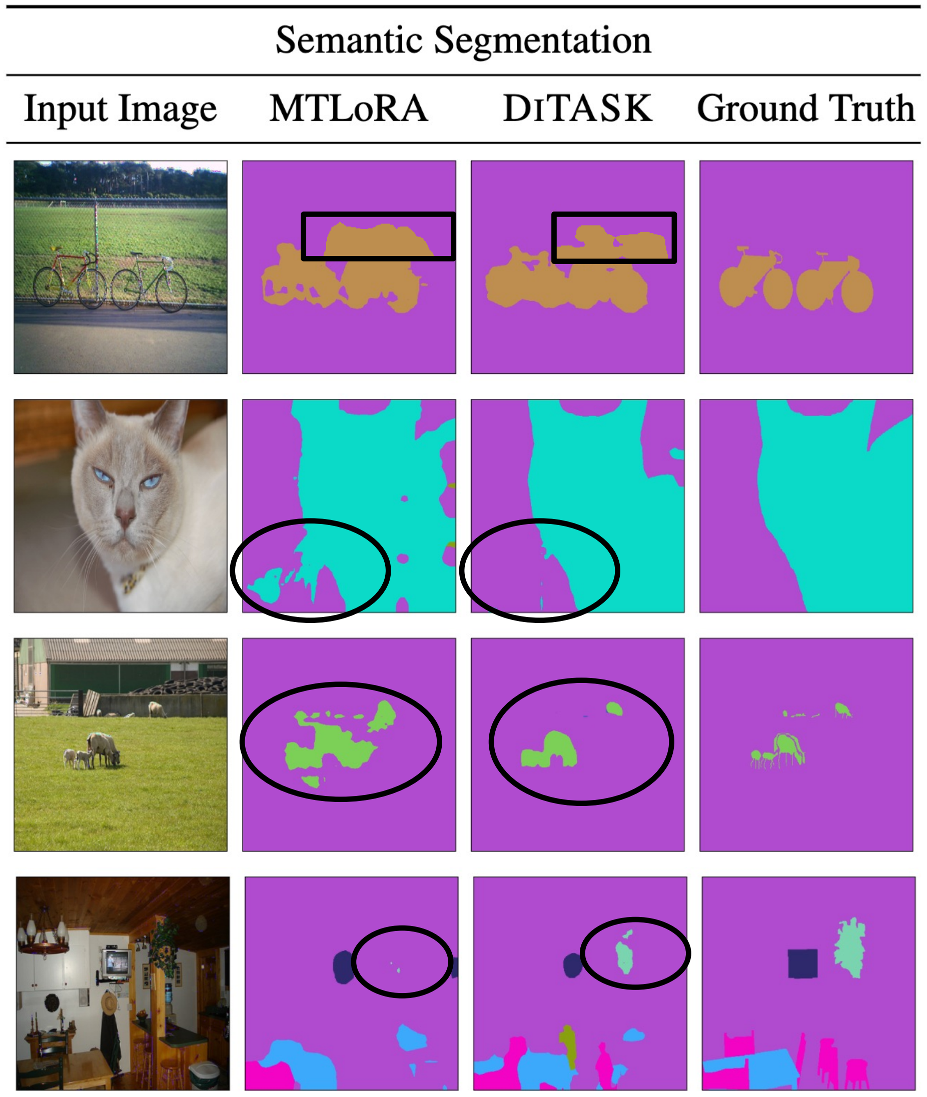


Figure 9. Pareto optimal curve on VTAB benchmark using DiTASK and LoRA. DiTASK achieves competitive performance using $\sim 10\times$ fewer parameters.

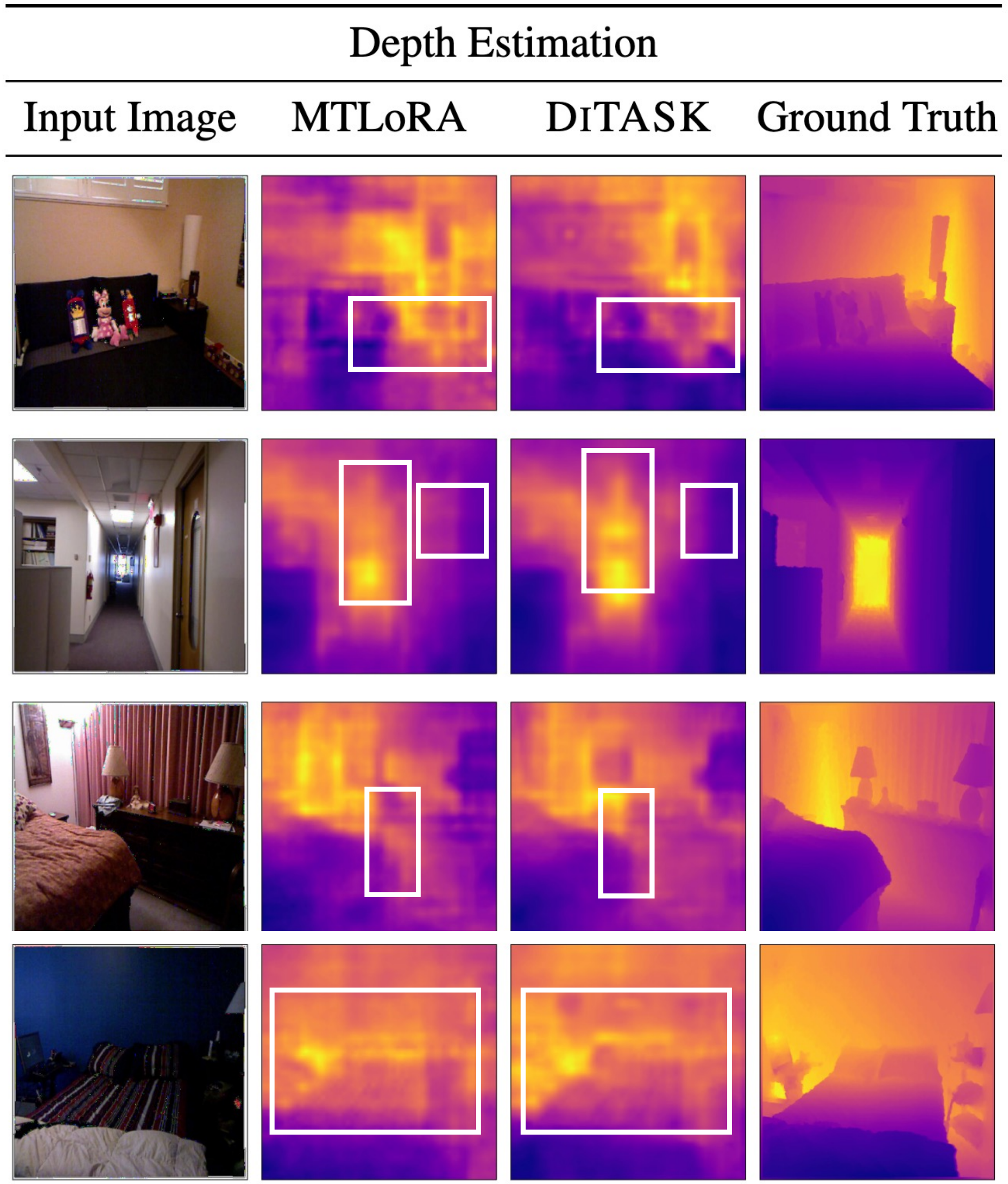
Qualitative Results

- Sharper boundaries
- More consistent segmentation
- Better preservation of fine details
- Improved handling of complex scenes



Qualitative Results

- Better depth discontinuities
- Improved structure preservation
- More accurate foreground-background separation
- Finer depth variations



Joint vs. Task-Wise Singular Value Modulation

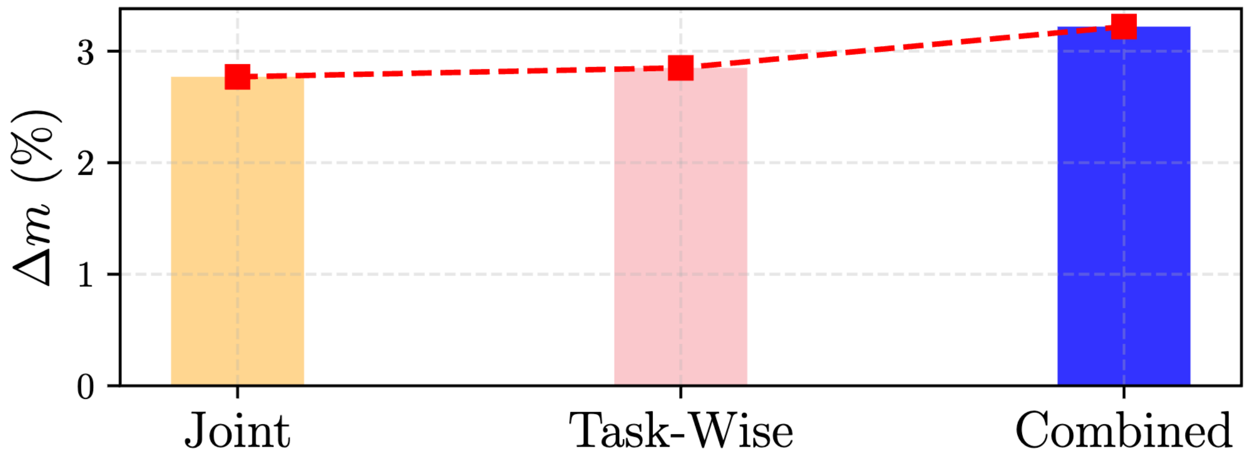
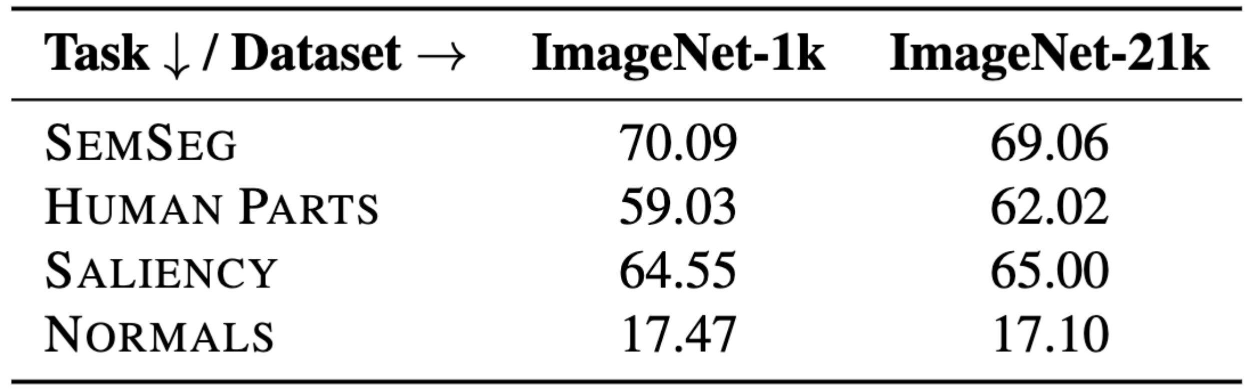


Figure 6. Effect of task-specific and task-agnostic components

Impact of Pre-training dataset scale

- Pre-training on larger datasets benefits more from DiTASK's approach of preserving learned representations
- Overall better performance with ImageNet-21k

Table 5. MTL Performance using DITASK on PASCAL for varying pre-training dataset scale



Task ↓ / Dataset →	ImageNet-1k	ImageNet-21k
SEMSEG	70.09	69.06
HUMAN PARTS	59.03	62.02
SALIENCY	64.55	65.00
NORMALS	17.47	17.10

DiTASK is effective across different backbone architectures

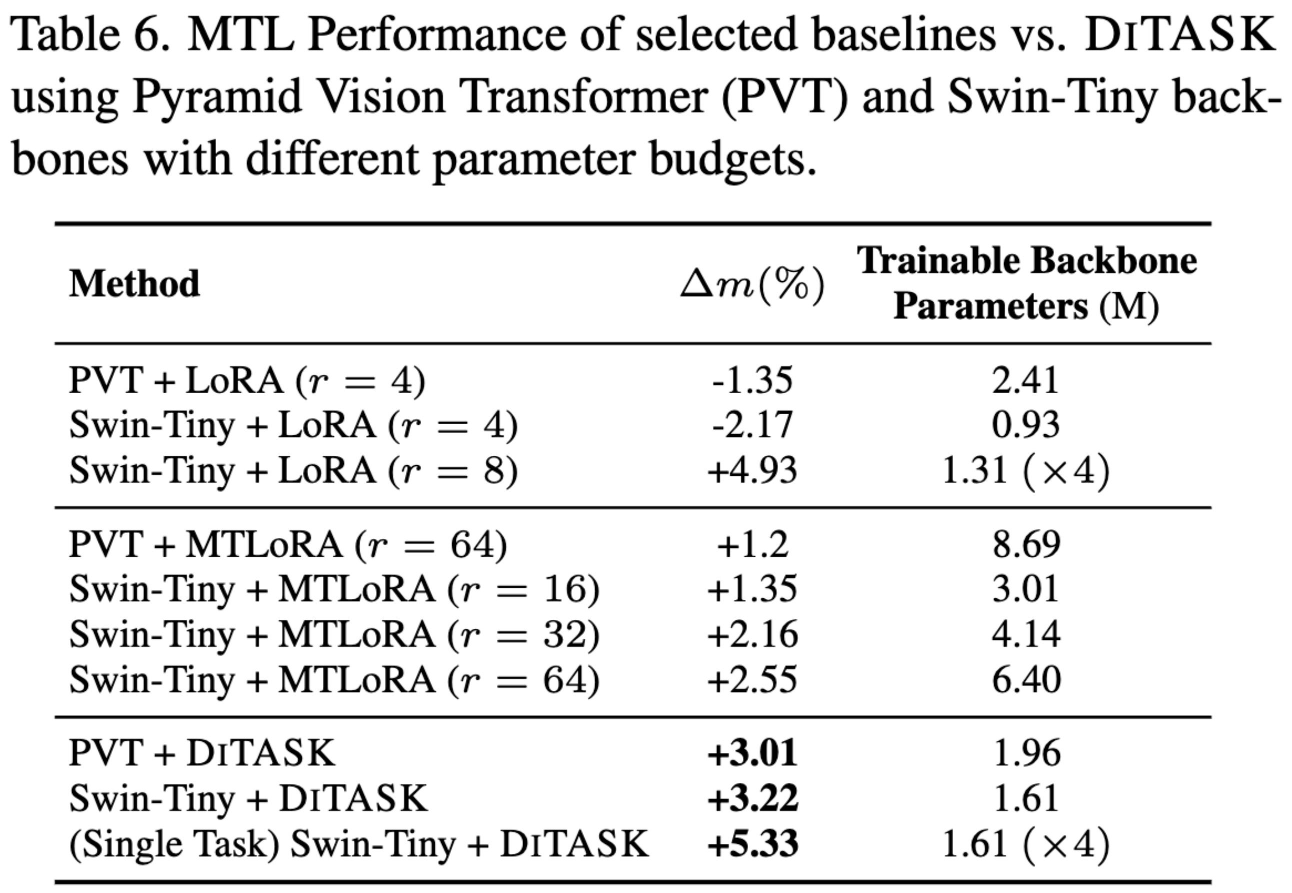


Table 6. MTL Performance of selected baselines vs. DiTASK using Pyramid Vision Transformer (PVT) and Swin-Tiny backbones with different parameter budgets.

Method	$\Delta m(\%)$	**Trainable Backbone Parameters** (M)
PVT + LoRA ($r = 4$)	-1.35	2.41
Swin-Tiny + LoRA ($r = 4$)	-2.17	0.93
Swin-Tiny + LoRA ($r = 8$)	+4.93	1.31 ($\times 4$)
PVT + MTLoRA ($r = 64$)	+1.2	8.69
Swin-Tiny + MTLoRA ($r = 16$)	+1.35	3.01
Swin-Tiny + MTLoRA ($r = 32$)	+2.16	4.14
Swin-Tiny + MTLoRA ($r = 64$)	+2.55	6.40
PVT + DiTASK	**+3.01**	1.96
Swin-Tiny + DiTASK	**+3.22**	1.61
(Single Task) Swin-Tiny + DiTASK	**+5.33**	1.61 ($\times 4$)

Summary

- DiTASK achieves S.O.T.A performance with **75%** fewer parameters
- Preserving singular vectors while modulating singular values is key to MTL
- Benefits increase with model scale and pre-training data scale
- Consistent improvements across datasets, tasks and architectures