ENHANCED CONTINUAL LEARNING OF VISION-LANGUAGE MODELS WITH MODEL FUSION

Haoyuan Gao[1,*], **Zicong Zhang**[1,*], **Yuqi Wei**[1], **Linglan Zhao**[4]
Guilin Li[4], **Yexin Li**[3], **Linghe Kong**[1], **Weiran Huang**[1,2,3,†]

[1] Shanghai Jiao Tong University [2] Shanghai Innovation Institute
[3] State Key Laboratory of General Artificial Intelligence, BIGAI [4] Tencent

ABSTRACT

Vision-Language Models (VLMs) represent a breakthrough in artificial intelligence by integrating visual and textual modalities to achieve impressive zero-shot capabilities. However, VLMs are susceptible to catastrophic forgetting when sequentially fine-tuned on multiple downstream tasks. Existing continual learning methods for VLMs often rely heavily on additional reference datasets, compromise zero-shot performance, or are limited to parameter-efficient fine-tuning scenarios. In this paper, we propose Continual Decoupling-Unifying (ConDU), a novel approach, by introducing model fusion into continual learning for VLMs. ConDU maintains a unified model along with task triggers and prototype sets, employing an iterative process of decoupling task-specific models for previous tasks and unifying them with the model for the newly learned task. Additionally, we introduce an inference strategy for zero-shot scenarios by aggregating predictions from multiple decoupled task-specific models. Extensive experiments across various settings show that ConDU achieves up to a 2% improvement in average performance across all seen tasks compared to state-of-the-art baselines, while also enhancing zero-shot capabilities relative to the original VLM.

1 INTRODUCTION

The human brain demonstrates remarkable plasticity, seamlessly learning new tasks while preserving the ability to perform previously acquired ones. In contrast, Artificial Neural Networks (ANNs) often suffer a significant performance drop on earlier tasks when learning sequentially. This issue, known as catastrophic forgetting (McCloskey & Cohen, 1989; Ramasesh et al., 2020), limits the adaptability of ANNs in dynamic environments. To overcome this challenge, continual learning (also referred to as lifelong learning) (Wang et al., 2024; Verwimp et al., 2023; Shi et al., 2024) has been developed. This paradigm aims to enable machine learning models to acquire new knowledge over time while preserving previously learned information, thus mimicking the adaptability of the human brain.

Recently, Vision-Language Models (VLMs) such as CLIP (Radford et al., 2021) represent a breakthrough in artificial intelligence by integrating visual and textual modalities to achieve impressive zero-shot capabilities. However, despite their demonstrated success (Shen et al., 2021; Zhao et al., 2023b; Fan et al., 2024), VLMs remain susceptible to catastrophic forgetting when fine-tuned for multiple downstream tasks. Conventional continual learning approaches are insufficient for VLM fine-tuning, as they struggle to maintain the crucial zero-shot capabilities that make these models valuable (Zheng et al., 2023). Alternative approaches, such as prompt learning (Liu et al., 2021), which preserve model parameters by updating only learnable prompts, face their own limitations due to constraints imposed by the restricted length of the prompts (Wang et al., 2022).

In contrast to the extensive research on conventional continual learning, relatively few methods (Zheng et al., 2023; Yu et al., 2025; Park, 2024; Xu et al., 2024) have been proposed for continual

[*]Haoyuan (gaohaoyuan@sjtu.edu.cn) and Zicong contributed equally to this work. This work was conducted at MIFA Lab, School of Computer Science, Shanghai Jiao Tong University.
[†]Corresponding author.

learning of VLMs. Some methods, such as (Zheng et al., 2023) and (Yu et al., 2025), require additional reference datasets for distillation from pre-trained models, and their performance is highly sensitive to the choice of the dataset (Zheng et al., 2023). Moreover, these methods require careful tuning of multiple handcrafted hyperparameters to balance different optimization objectives: mitigating catastrophic forgetting, preserving zero-shot capabilities, and optimizing performance on the current task. Alternative methods (Yu et al., 2024; Park, 2024; Xu et al., 2024) focus exclusively on parameter-efficient fine-tuning (Ding et al., 2023) employing modules such as adapters or LoRA, but struggle to adapt to full fine-tuning scenarios, which often offer superior performance.

To overcome these limitations, we propose leveraging model fusion as a novel paradigm for advancing the continual learning of VLMs. Model fusion (Ilharco et al., 2022; Yang et al., 2023; Huang et al., 2024) combines multiple models into a single unified model without requiring training data, enabling the unified model to retain the strengths of its constituent models. In this paper, we propose the *Continual Decoupling-Unifying* (ConDU), a novel continual learning approach for VLMs by introducing model fusion. Notably, ConDU is inherently compatible with both parameter-efficient and full fine-tuning paradigms, offering a flexible solution for diverse continual learning scenarios.

In particular, ConDU maintains a unified model, a set of task triggers, and a series of prototype sets throughout the continual learning process (see Figure 1). In each session t, a new task t is introduced (where the same notation t is used for both session and task without causing ambiguity). ConDU begins by fine-tuning the given pre-trained VLM to obtain the task-specific model for the current task t via parameter-efficient fine-tuning (*e.g.*, LoRA) or full fine-tuning. Subsequently, ConDU *decouples* the latest unified model using all stored $t-1$ task triggers to approximately reconstruct task-specific models for all previous tasks $1, 2, \ldots, t-1$. Next, ConDU *unifies* these $t-1$ models and the task-specific model for the current task t to produce a new unified model, and calculate t updated task triggers to replace the old ones based on the new unified model and all t task-specific models. Meanwhile, ConDU computes the prototypes of the current task t using the pre-trained VLM to form the prototype set of task t and append it to the existing prototype sets. This concludes the current session, with the updated unified model, task triggers, and prototype sets prepared for use in the next session $t+1$. In practice, we operate on incremental models (referred to as delta models) instead of full task-specific models for simplicity. We remark that the decoupling and unifying procedures introduced in ConDU are *training-free*, and thus their running time is much shorter than the time required for model fine-tuning. Moreover, compared to previous continual learning methods for VLMs mentioned earlier, ConDU eliminates the need for adjusting trade-off hyperparameters, incorporating reference datasets, and maintaining replay memory.

After the above continual learning process, our method supports multiple inference scenarios. If the test sample belongs to a previously seen task and its task ID is known, we can directly reconstruct the corresponding task-specific model based on the unified model and use it for prediction. When the task ID is unknown or the test sample comes from an unseen task (*i.e.*, the zero-shot scenario), we can instead reconstruct multiple task-specific models relevant to the test sample's domain and make a prediction by aggregating their results. Evaluated on widely used benchmarks across diverse settings, including Multi-domain Task Incremental Learning (MTIL), few-shot MTIL, and task-agnostic MTIL. ConDU achieves up to a 2% improvement in average performance across all seen tasks compared to state-of-the-art baselines, demonstrating the effectiveness of incorporating model fusion. Moreover, ConDU exhibits strong zero-shot capabilities of VLMs, outperforming the original pre-trained VLM and other state-of-the-art continual learning methods.

The contributions of this work are summarized as follows:

- We introduce model fusion into continual learning for VLMs and propose a novel Continual Decoupling-Unifying (ConDU) framework, which is compatible with both parameter-efficient and full fine-tuning paradigms.
- We propose aggregating the results of multiple decoupled task-specific models for prediction in zero-shot scenarios.
- Through extensive experiments on multiple benchmarks, we demonstrate that ConDU effectively learns new knowledge while preserving previously acquired knowledge and enhancing zero-shot capabilities.

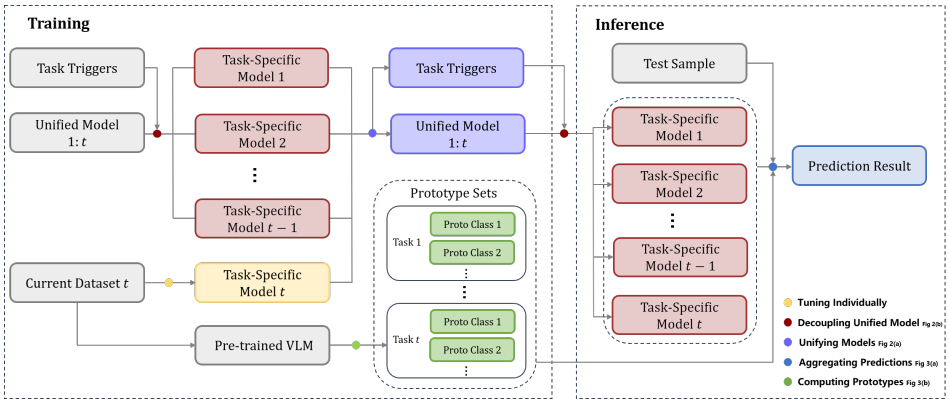


Figure 1: Overall framework of the proposed method. The colored points in the process denote modules of our method, including Tuning Individually, Decoupling Unified Model (see Figure 2a), Unifying Models (see Figure 2b), Aggregating Prediction (see Figure 3a), and Computing Prototypes (see Figure 3b).

2 ConDU: Continual Decoupling-Unifying

We propose *Continual Decoupling-Unifying* (ConDU), a novel continual learning approach for VLMs that leverages model fusion. Figure 1 shows the overall framework of ConDU. ConDU maintains a unified model, a set of task triggers, and a series of prototype sets throughout the continual learning process. Our framework includes five modules (denoted as colored points in Figure 1). We will introduce three modules for training in section 2.1 and two modules for inference in section 2.2.

2.1 Delta Models Continually Fusion at Training Stage

At each session t of the continual learning process, ConDU implements three steps: Tuning Individually, Decoupling Unified Model, and Unifying Models. Since the process of Decoupling Unified Model relies on the task triggers produced during Unifying Models, we first introduce the process of Unifying Models before detailing process of Decoupling Unified Model.

Tuning Individually. We denote a VLM as $f(\cdot;\theta)$, where θ represents only the learnable parameters of the VLM, excluding the frozen parameters for clarity. At session t, by fine-tuning the pre-trained VLM θ^0 on task t, we obtain a task-specific model θ^t. We defined delta model t, the parameter offsets of task-specific model t relative to the pre-trained VLM, as $\delta^t = \theta^t - \theta^0$. We will unify delta models instead of directly unifying task-specific models following the setting of advanced model fusion methods (Ilharco et al., 2022; Yadav et al., 2024; Yang et al., 2023; Huang et al., 2024).

Unifying Models. The process of Unifying Models is illustrated in Figure 2 a). When task t arrives, we first decouple the current unified delta model $\delta^{1:t-1}$ to obtain the approximation of δ^i denoted as $\tilde{\delta}^i$ for $i = 1, 2, \ldots, t-1$ (this process will be introduced in the paragraph of Decoupling Unified Model). Then let $\delta^i \leftarrow \tilde{\delta}^i, i = 1, 2, \ldots, t-1$, the calculation of unified model is $\delta^{1:t} = \text{unify}(\{\delta^1, \delta^2, \ldots, \delta^t\})$, where $\delta^{1:t}$ are the unified delta model, and the jth dimension of it is calculated as

$$\delta_j^{1:t} = \begin{cases} \max_i(\delta_j^i) & \text{if } \sum_{i=1}^t \delta_j^i > 0 \\ \min_i(\delta_j^i) & \text{if } \sum_{i=1}^t \delta_j^i < 0 \end{cases}$$

This process means choosing the jth parameter of all 1 to t delta models with the largest absolute value and has the same sign as $\sum_{i=1}^t \delta_j^i$, retaining the largest magnitude and consistent sign information shared across the delta models.

Then the unified delta model is added to the pre-trained VLM to construct the unified model $\theta^{1:t} = \theta^0 + \delta^{1:t}$.

Except unified model, other productions of Unifying Models at session t is t task triggers. For $i = 1, 2, \ldots, t$, each task trigger i will be used on the unified model to reconstruct delta model i in the future, composed of a mask M^i with the same dimension as the delta model, and a rescaling scalar λ^i. The binary number of M^i at position j indicates whether the delta model i has the same sign as the unified delta model at position j, that is $M^i_j = \begin{cases} 1 & \text{if } \delta^i_j \cdot \delta^{1:t}_j > 0 \\ 0 & \text{if } \delta^i_j \cdot \delta^{1:t}_j < 0. \end{cases}$ The rescaler is to preserve the average magnitude of elements in δ^i and $M^i \odot \delta^{1:t}$, defined as $\lambda^i = \frac{\text{sum}(\text{abs}(\delta^i))}{\text{sum}(\text{abs}(M^i \odot \delta^{1:t}))}$.

The final productions of Unifying Model at session t is a unified model and t task triggers. Then we introduce how to use task triggers to decouple the unified model.

Decoupling Unified Model. The process of Decoupling Unified Model is illustrated in 2 b). This process is both needed in the beginning of a training session and the inference stage. If there are t seen tasks, t task triggers are applied to the unified delta model to obtain t delta models $\tilde{\delta}^i = \lambda^i \cdot M^i \odot \delta^{1:t}$, which are then added to the pre-trained VLM θ^0 to obtain t task-specific models $\tilde{\theta}^i = \theta^0 + \tilde{\delta}^i$. At a training session, $\tilde{\delta}^i$ will attend the unifying instead of δ^i. At inference stage, the output logits of all reconstructed task-specific model $f(\cdot, \tilde{\theta}^i)$ will aggregated to predict the test samples. We will introduce this Aggregating Predictions in the next section.

2.2 Semantic-based Aggregating Mechanism at Inference Stage

During inference, we propose a Semantic-based Aggregating Mechanism to predict a sample without task ID or a sample from unseen tasks. Specifically, the unified model is decoupled into task-specific models by task triggers at the inference stage. For a test sample from a seen task with a known task ID, we choose the corresponding task-specific model to predict. For a test sample from an unseen task or without task ID, we input the sample into all task-specific models, and the output logits are added with weights calculated by semantic matching, using memory-stored prototypes computed during the training stage as the next paragraph shown.

Computing Prototypes. The process of Computing Prototypes is illustrated in 3 b). For each category in each task, we save its prototype during training. The prototype of the kth category in the ith task is the mean of the image feature vectors plus the text feature vector for that category, extracted by the pre-trained VLM, that is $P^i_k = f(y, \theta^0) + \frac{1}{|\mathcal{D}^t_k|} \sum_{m=1}^{|\mathcal{D}^t_k|} f(x_m, \theta^0)$, where $\mathcal{D}^t_k$ is the dataset of the kth category in the ith task, y is the text of this category, and x_m is the mth image of $\mathcal{D}^t_k$. Then we will introduce how the aggregating weights is calculated by these prototypes and how they are utilized to aggregating predictions.

Aggregating Predictions. The process of Aggregating Predictions is illustrated in Figure 3 a). For a test image x, we use $f(\cdot, \theta^0)$ to extract the image feature. We then calculate the cosine similarity between the test image feature and the learned prototypes of different tasks. For each task, we select the highest similarity score and then compare the similarity scores across different tasks to choose the K-highest tasks. The weights of the K-highest tasks are 1 and the others are 0. The outputs logits of all task-specific models are added with these weights and determine the final prediction result.

3 Experiments

The detailed experiment setting is shown in Appendix B. We report two results of ConDU. ConDU(FT) denotes ConDU with full fine-tuning, and ConDU(LoRA) denotes ConDU with LoRA-based fine-tuning. The experimental results on task-agnostic MTIL is shown in I.

3.1 Comparison with State-of-the-art Methods

Multi-domain Task Incremental Learning. Table 1 presents the detailed comparison results of our proposed ConDU and the baselines on the MTIL benchmark. As seen, our method outperforms all baseline methods across all three metrics. The "Transfer" metric of our method is 70.8% for FT and 70.3% for LoRA, which exceeds the best baseline by 0.7%. The "Average" metric of our method

Table 1: Comparison with SOTA methods on MTIL benchmark in terms of "Transfer", "Average.", and "Last" scores (%). We label the best methods on average of all datasets with **bold** styles. The results of more baselines can be found in Appendix I.

	Method	Aircraft	Caltech101	CIFAR100	DTD	EuroSAT	Flowers	Food	MNIST	OxfordPet	Cars	SUN397	*Average*
	Zero-shot	24.3	88.4	68.2	44.6	54.9	71.0	88.5	59.4	89.0	64.7	65.2	65.3
	Individual FT	62.0	95.1	89.6	79.5	98.9	97.5	92.7	99.6	94.7	89.6	81.8	89.2
Transfer	ZSCL	-	86.0	67.4	45.4	50.4	69.1	87.6	61.8	86.8	60.1	66.8	68.1
	Dual-RAIL	-	88.4	68.2	44.6	54.9	71.0	88.5	59.6	89.0	64.7	65.2	69.4
	DPeCLIP	-	88.2	67.2	44.7	54.0	70.6	88.2	59.5	89.0	64.7	64.8	69.1
	MulKI	-	87.8	69.0	46.7	51.8	71.3	88.3	64.7	89.7	63.4	68.1	70.1
	ConDU(FT)	-	88.1	68.9	46.4	57.1	71.4	88.7	65.5	89.3	65.0	67.8	**70.8**
	ConDU(LoRA)	-	88.1	68.9	45.7	57.0	71.3	88.8	61.2	89.3	65.1	67.8	70.3
Average	ZSCL	45.1	92.0	80.1	64.3	79.5	81.6	89.6	75.2	88.9	64.7	68.0	75.4
	Dual-RAIL	52.5	96.0	80.6	70.4	81.3	86.3	89.1	73.9	90.2	68.5	66.5	77.8
	DPeCLIP	49.9	94.9	82.4	69.4	82.2	84.3	90.0	74.0	90.4	68.3	66.3	77.5
	MulKI	52.5	93.6	79.4	67.0	79.8	83.9	89.6	77.1	91.2	67.1	69.1	77.3
	ConDU(FT)	59.6	93.4	83.7	68.1	83.4	83.7	90.1	76.7	90.6	68.6	68.6	**78.8**
	ConDU(LoRA)	51.9	94.9	84.4	69.8	81.1	84.4	90.0	77.3	89.5	69.0	69.3	78.3
Last	ZSCL	40.6	92.2	81.3	70.5	94.8	90.5	91.9	98.7	93.9	85.3	80.2	83.6
	Dual-RAIL	52.5	96.8	83.3	80.1	96.4	99.0	89.9	98.8	93.5	85.5	79.2	86.8
	DPeCLIP	49.9	95.6	85.8	78.6	98.4	95.8	92.1	99.4	94.0	84.5	81.7	86.9
	MulKI	49.7	93.0	82.8	73.7	96.2	92.3	90.4	99.0	94.8	85.2	78.9	85.1
	ConDU(FT)	58.6	93.7	86.6	76.1	98.2	93.4	91.9	99.6	94.8	84.9	80.5	**87.1**
	ConDU(LoRA)	48.9	95.2	87.8	78.5	96.3	95.2	91.7	97.6	93.0	85.3	78.8	86.2

Table 2: Comparison with SOTA methods on few-shot MTIL benchmark in terms of "Transfer", "Average.", and "Last" scores (%). We label the best methods on average of all datasets with **bold** styles. The results of more baselines can be found in Appendix I.

	Method	Aircraft	Caltech101	CIFAR100	DTD	EuroSAT	Flowers	Food	MNIST	OxfordPet	Cars	SUN397	*Average*
	Zero-shot	24.3	88.4	68.2	44.6	54.9	71.0	88.5	59.6	89.0	64.7	65.2	65.3
	Individual FT	30.6	93.5	76.8	65.1	91.7	92.9	83.3	96.6	84.9	65.4	71.3	77.5
Transfer	Continual FT	-	72.8	53.0	36.4	35.4	43.3	68.4	47.4	72.6	30.0	52.7	51.2
	WiSE-FT	-	77.6	60.0	41.3	39.4	53.0	76.6	58.1	75.5	37.3	58.2	57.7
	ZSCL	-	84.0	68.1	44.8	46.8	63.6	84.9	61.4	81.4	55.5	62.2	65.3
	MoE	-	87.9	68.2	44.1	48.1	64.7	88.8	69.0	89.1	64.5	65.1	68.9
	Dual-RAIL	-	88.4	68.2	44.6	54.9	71.0	88.5	59.6	89.0	64.7	65.2	69.4
	ConDU(FT)	-	88.0	69.5	45.6	54.4	71.1	88.7	62.2	88.9	64.4	66.6	70.0
	ConDU(LoRA)	-	88.1	68.5	45.6	56.4	71.2	89.0	64.0	88.8	64.9	66.4	**70.3**
Average	Continual FT	28.1	86.4	59.1	52.8	55.8	62.0	70.2	64.7	75.5	35.0	54.0	58.5
	WiSE-FT	32.0	87.7	61.0	55.8	68.1	69.3	76.8	71.5	77.6	42.0	59.3	63.7
	ZSCL	28.2	88.6	66.5	53.5	56.3	73.4	83.1	56.4	82.4	57.5	62.9	64.4
	MoE	30.0	89.6	73.9	58.7	69.3	79.3	88.1	76.5	89.1	65.3	65.8	71.4
	Dual-RAIL	36.0	94.2	70.9	58.8	70.6	84.3	88.5	70.3	89.7	66.5	65.8	72.3
	ConDU(FT)	33.1	90.5	74.1	58.3	76.2	81.0	87.9	73.4	88.0	64.8	67.1	72.3
	ConDU(LoRA)	32.4	92.1	75.4	58.8	75.1	82.9	87.3	74.0	89.3	65.1	67.0	**72.7**
Last	Continual FT	27.8	86.9	60.1	58.4	56.6	75.7	73.8	93.1	82.5	57.0	66.8	67.1
	WiSE-FT	30.8	88.9	59.6	60.3	80.9	81.7	77.1	94.9	83.2	62.8	70.0	71.9
	ZSCL	26.8	88.5	63.7	55.7	60.2	82.1	82.6	58.6	85.9	66.7	70.4	67.4
	MoE	30.1	89.3	74.9	64.0	82.3	89.4	87.1	89.0	89.1	69.5	72.5	76.1
	Dual-RAIL	36.0	94.8	71.5	64.1	79.5	95.3	88.5	89.4	91.5	74.6	71.3	**77.9**
	ConDU(FT)	33.3	90.7	75.0	63.1	88.8	88.6	87.0	91.8	85.6	66.5	71.9	76.6
	ConDU(LoRA)	31.8	92.4	76.7	63.4	86.8	91.8	85.6	93.9	90.3	68.1	70.9	77.4

is 78.8% for FT and 78.3% for LoRA, which exceeds the best baseline by 1.5%. The "Last" metric of our method is 87.1% for FT and 86.2% for LoRA, which exceeds the best baseline by 0.2%.

Few-shot MTIL. Table 6 presents the detailed comparison results of our proposed ConDU and the baselines on the few-shot MTIL benchmark. The "Transfer" metric of our method is 70.0% for FT and 70.3% for LoRA, which exceeds the best baseline by 0.6%. The "Average" metric of our method is 72.3% for FT and 72.7% for LoRA, which exceeds the best baseline by 0.4%. The "Last" metric of our method is 76.6% for FT and 77.4% for LoRA, which has the second-best performance.

4 Conclusion

In this paper, we introduce Continual Decoupling-Unifying (ConDU), a novel continual learning framework for VLMs, by introducing model fusion. ConDU is not dependent on reference datasets, or complex trade-off hyperparameters, and ConDU is inherently compatible with both parameter-efficient and full fine-tuning paradigms. Evaluated on widely used benchmarks across diverse settings, ConDU achieves up to a 2% improvement in average performance across all seen tasks compared to state-of-the-art baselines, while exhibiting strong zero-shot capabilities of VLMs.

Acknowledgments

This project is supported by the National Natural Science Foundation of China (No. 62406192), Opening Project of the State Key Laboratory of General Artificial Intelligence (No. SKLAGI2024OP12), Tencent WeChat Rhino-Bird Focused Research Program, and Doubao LLM Fund.

References

Matteo Boschini, Lorenzo Bonicelli, Pietro Buzzega, Angelo Porrello, and Simone Calderara. Class-incremental continual learning into the extended der-verse. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5), 2022.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part VI 13*, pp. 446–461. Springer, 2014.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems*, 33:15920–15930, 2020.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3606–3613, 2014.

Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142, 2012.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, 2023.

Yuxuan Ding, Lingqiao Liu, Chunna Tian, Jingyuan Yang, and Haoxuan Ding. Don't stop learning: Towards continual learning for the clip model. *arXiv preprint arXiv:2207.09248*, 2022.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Lijie Fan, Dilip Krishnan, Phillip Isola, Dina Katabi, and Yonglong Tian. Improving clip training with language rewrites. *Advances in Neural Information Processing Systems*, 36, 2024.

Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pp. 178–178. IEEE, 2004.

Ronald A Fisher. On the mathematical foundations of theoretical statistics. *Philosophical transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character*, 222(594-604):309–368, 1922.

Rui Gao and Weiwei Liu. Ddgr: Continual learning with deep diffusion-based generative replay. In *International Conference on Machine Learning*, pp. 10744–10763. PMLR, 2023.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.*, 12(7):2217–2226, 2019.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Chenyu Huang, Peng Ye, Tao Chen, Tong He, Xiangyu Yue, and Wanli Ouyang. Emr-merging: Tuning-free high-performance model merging. *arXiv preprint arXiv:2405.17461*, 2024.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. In *The Eleventh International Conference on Learning Representations*, 2022.

Junsu Kim, Hoseong Cho, Jihyeon Kim, Yihalem Yimolal Tiruneh, and Seungryul Baek. Sddgr: Stable diffusion-based deep generative replay for class incremental object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28772–28781, 2024.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proc. Nat. Acad. Sci. USA*, 114(13):3521–3526, 2017.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pp. 554–561, 2013.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Univ. Toronto, Toronto, ON, Canada, Tech. Rep.*, 2009.

Yukun Li, Guansong Pang, Wei Suo, Chenchen Jing, Yuling Xi, Lingqiao Liu, Hao Chen, Guoqiang Liang, and Peng Wang. Coleclip: Open-domain continual learning via joint task prompt and vocabulary learning. *arXiv preprint arXiv:2403.10245*, 2024.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Junxin Lu and Shiliang Sun. Pamk: Prototype augmented multi-teacher knowledge transfer network for continual zero-shot learning. *IEEE Trans. on Image Process.*, 2024. doi:10.1109/TIP.2024.3403053.

Yadong Lu, Shitian Zhao, Boxiang Yun, Dongsheng Jiang, Yin Li, Qingli Li, and Yan Wang. Boosting open-domain continual learning via leveraging intra-domain category-aware prototype. *arXiv preprint arXiv:2408.09984*, 2024.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 7765–7773, 2018.

Michael S Matena and Colin A Raffel. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716, 2022.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

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

Sejik Park. Learning more generalized experts by merging experts in mixture-of-experts. *arXiv preprint arXiv:2405.11530*, 2024.

Omkar M Parkhi, Andrea Vedaldi, CV Jawahar, and Andrew Zisserman. The truth about cats and dogs. In *2011 International Conference on Computer Vision*, pp. 1427–1434. IEEE, 2011.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Vinay V. Ramasesh, Ethan Dyer, and Maithra Raghu. Anatomy of catastrophic forgetting: Hidden representations and task semantics. In *Advances in Neural Information Processing Systems*, 2020.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2018.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International conference on machine learning*, pp. 4548–4557. PMLR, 2018.

Sheng Shen, Liunian Harold Li, Hao Tan, Mohit Bansal, Anna Rohrbach, Kai-Wei Chang, Zhewei Yao, and Kurt Keutzer. How much can clip benefit vision-and-language tasks? *arXiv preprint arXiv:2107.06383*, 2021.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, and Hao Wang. Continual learning of large language models: A comprehensive survey. *arXiv preprint arXiv:2404.16789*, 2024.

Eli Verwimp, Rahaf Aljundi, Shai Ben-David, Matthias Bethge, Andrea Cossu, Alexander Gepperth, Tyler L. Hayes, Eyke Hüllermeier, Christopher Kanan, Dhireesha Kudithipudi, Christoph H. Lampert, Martin Mundt, Razvan Pascanu, Adrian Popescu, Andreas S. Tolias, Joost van de Weijer, Bing Liu, Vincenzo Lomonaco, Tinne Tuytelaars, and Gido M. van de Ven. Continual learning: Applications and the road forward. *arXiv preprint arXiv:2311.11908*, 2023.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

Yabin Wang, Zhiwu Huang, and Xiaopeng Hong. S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. *Advances in Neural Information Processing Systems*, 35:5682–5695, 2022.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 7959–7971, 2022.

Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pp. 3485–3492. IEEE, 2010.

Yicheng Xu, Yuxin Chen, Jiahao Nie, Yusong Wang, Huiping Zhuang, and Manabu Okumura. Advancing cross-domain discriminability in continual learning of vision-language models. *arXiv preprint arXiv:2406.18868*, 2024.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36, 2024.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. *arXiv preprint arXiv:2310.02575*, 2023.

Jiazuo Yu, Yunzhi Zhuge, Lu Zhang, Ping Hu, Dong Wang, Huchuan Lu, and You He. Boosting continual learning of vision-language models via mixture-of-experts adapters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23219–23230, 2024.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. *arXiv preprint arXiv:2311.03099*, 2023.

Yu-Chu Yu, Chi-Pin Huang, Jr-Jen Chen, Kai-Po Chang, Yung-Hsuan Lai, Fu-En Yang, and Yu-Chiang Frank Wang. Select and distill: Selective dual-teacher knowledge transfer for continual learning on vision-language models. In *European Conference on Computer Vision*, pp. 219–236. Springer, 2025.

Hongsheng Zhang, Zhong Ji, Jingren Liu, Yanwei Pang, and Jungong Han. Multi-stage knowledge integration of vision-language models for continual learning. *arXiv preprint arXiv:2411.06764*, 2024.

Linglan Zhao, Jing Lu, Yunlu Xu, Zhanzhan Cheng, Dashan Guo, Yi Niu, and Xiangzhong Fang. Few-shot class-incremental learning via class-aware bilateral distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11838–11847, 2023a.

Linglan Zhao, Xuerui Zhang, Ke Yan, Shouhong Ding, and Weiran Huang. Safe: Slow and fast parameter-efficient tuning for continual learning with pre-trained models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Zihao Zhao, Yuxiao Liu, Han Wu, Mei Wang, Yonghao Li, Sheng Wang, Lin Teng, Disheng Liu, Zhiming Cui, Qian Wang, et al. Clip in medical imaging: A comprehensive survey. *arXiv preprint arXiv:2312.07353*, 2023b.

Zangwei Zheng, Mingyuan Ma, Kai Wang, Ziheng Qin, Xiangyu Yue, and Yang You. Preventing zero-shot transfer degradation in continual learning of vision-language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19125–19136, 2023.

APPENDIX

A PROBLEM FORMULATION

In this paper, we focus on continual learning for Vision-Language Models (VLMs). Given a pre-trained VLM (*e.g.*, CLIP (Radford et al., 2021)), a sequence of T tasks arrives incrementally, where each task t is associated with a training dataset $\mathcal{D}^t$. These tasks may involve distinct classes, different domains, or exhibit significant variation in sample sizes. After seeing each task, the VLM can be updated with access to a limited memory storing essential information (*e.g.*, selected past data or parameters). The goal is to develop a method that incrementally updates the VLM while achieving high performance on all previously encountered tasks and retaining its zero-shot capabilities for unseen tasks. Additionally, we aim for the proposed continual learning method to support both parameter-efficient fine-tuning (*e.g.*, LoRA or adapters) and full fine-tuning.

Under the problem definition of continual learning, only a single VLM is allowed to be maintained throughout the learning process. However, if multiple models could instead be maintained, each individually fine-tuned from the pre-trained VLM using task-specific data, the corresponding model could be selected for prediction when a test sample comes from a known task. Furthermore, a key characteristic of VLMs is their zero-shot capability, which should be preserved or even enhanced after continual learning. For test samples from unseen tasks (*i.e.*, the zero-shot scenario for VLMs), using multiple specialized models fine-tuned on diverse domains for prediction is expected to outperform a single VLM, such as by aggregating predictions from the specialized models.

Inspired by this, if the shared components of these individual models could be extracted and fused into a single maintained VLM, while the task-specific variations are stored in the limited memory, it would be possible to effectively emulate the performance of multiple models with only one main VLM and small memory. Moreover, such an approach is inherently compatible with both parameter-efficient and full fine-tuning paradigms.

B EXPERIMENT SETTING

Dataset. We test our method on three benchmarks, including Multi-domain Task Incremental Learning (MTIL) (Zheng et al., 2023), task-agnostic MTIL, and few-shot MTIL.

The MTIL (Zheng et al., 2023) extends task incremental learning to a cross-domain setting, where each task is derived from a distinct domain. The MTIL framework comprises 11 individual tasks, each associated with a separate dataset, collectively representing a total of 1201 classes. In alignment with previous works, we adopt the following datasets: Aircraft (Maji et al., 2013), Caltech101 (Fei-Fei et al., 2004), CIFAR100 (Krizhevsky et al., 2009), DTD (Cimpoi et al., 2014), EuroSAT (Helber et al., 2019), Flowers (Nilsback & Zisserman, 2008), Food (Bossard et al., 2014), MNIST (Deng, 2012), OxfordPet (Parkhi et al., 2011), StanfordCars (Krause et al., 2013), and SUN397 (Xiao et al., 2010). The task-agnostic MTIL is the variant of the MTIL benchmark, where the task ID is unknown during inference for each test sample. The few-shot MTIL variant involves training with only five train samples per category for each task.

Protocol. In our experiments, all evaluation protocols follow the existing works (Zheng et al., 2023; Park, 2024; Yu et al., 2024; Li et al., 2024) for fair comparison. We utilize a pre-trained CLIP model with a ViT-B/16 (Dosovitskiy et al., 2020) image encoder. We perform 1000 iterations of training for each task in both MTIL and task-agnostic MTIL. For few-shot MTIL, we train each task for 500 iterations. We use AdamW (Loshchilov & Hutter, 2017) as the optimizer and set the batch size to 32 across all experiments.

Metric. For evaluating the MTIL, task-agnostic MTIL, and few-shot MTIL, we follow the existing works (Zheng et al., 2023; Park, 2024; Yu et al., 2024; Li et al., 2024) to use three key metrics: "Average", "Last", and "Transfer". The "Average" metric computes the average accuracy across all seen tasks. The "Transfer" metric evaluates the model's zero-shot transfer performance on subsequent tasks. The "Last" metric reflects the model's average performance at the end of the continual learning process. In the task-agnostic MTIL setting, we omit the "Transfer" metric and focus solely on the "Average" and "Last" metrics.

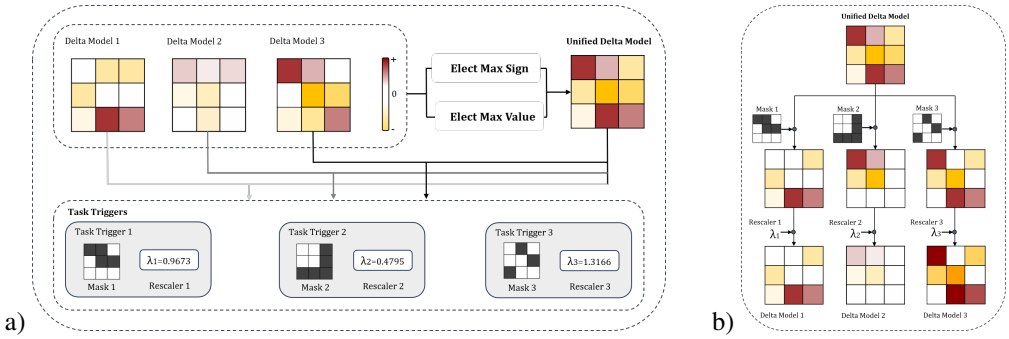


Figure 2: The process of Unifying Models (a) and Decoupling Unified Model (b) is transformed to unifying delta models (a) and decoupling unified delta model (b), respectively. A delta model represents the parameter offsets of a task-specific model relative to the pre-trained VLM. a) When unifying delta models, the unified model is obtained by an election process. Each task's task trigger is calculated according to the difference between the delta model and the unified delta model. b) When decoupling the unified delta model, we use the task trigger i on the unified delta model to reconstruct the delta model i.

Baseline. We compare our method with several state-of-the-art (SOTA) approaches, including:

(1) Zero-shot, (2) Individual FT, (3) Continual FT, (4) LwF (Li & Hoiem, 2017), (5) iCaRL (Rebuffi et al., 2017), (6) LwF-VR (Ding et al., 2022), (7) WiSE-FT (Wortsman et al., 2022) (8) ZSCL (Zheng et al., 2023), (9) MoE (Park, 2024), (10) MA (Yu et al., 2024), (11) Primal-RAIL (Xu et al., 2024), (12) Dual-RAIL (Xu et al., 2024), (13) CoLeCLIP (Li et al., 2024), (14) DPeCLIP (Lu et al., 2024), (15) MulKI (Zhang et al., 2024).

Zero-shot denotes directly using the pre-trained VLM for prediction on each task without additional fine-tuning. The results of Individual FT represent the performance of using a fully fine-tuned model, trained independently on each task based on the pre-trained VLM, for prediction. Continual FT refers to incrementally fine-tuning the VLM on new tasks without employing any forgetting mitigation strategies.

C Modules Illustration

Figure 2 illustrates the process of Unifying Models and Decoupling Unified Model, and Figure 3 illustrates the process of Aggregating Prediction and Computing Prototypes.

D Related Work

Continual Learning for VLMs. Conventional continual learning has been extensively studied, including architecture-based methods (Rusu et al., 2016; Mallya & Lazebnik, 2018; Serra et al., 2018), replay-based methods (Riemer et al., 2018; Buzzega et al., 2020; Boschini et al., 2022; Gao & Liu, 2023; Kim et al., 2024), and regularization-based methods (Kirkpatrick et al., 2017; Li & Hoiem, 2017; Zhao et al., 2023a; Lu & Sun, 2024; Zhao et al., 2024). However, these methods cannot be directly applied to recently developed Vision-Language Models (VLMs), as they struggle to maintain the crucial zero-shot capabilities (Zheng et al., 2023).

Recently, continual learning methods specifically designed for VLMs have been introduced. These methods can be broadly classified into parameter-efficient fine-tuning based approaches (Wang et al., 2022; Yu et al., 2024; Park, 2024; Li et al., 2024; Xu et al., 2024) and distillation-based methods (Ding et al., 2022; Zheng et al., 2023; Yu et al., 2025). However, these methods either require reference datasets or the careful adjustment of trade-off hyperparameters, or they are not suitable for full fine-tuning. In contrast, our method is compatible with both parameter-efficient and full fine-tuning paradigms, without the need for reference datasets, replay memory, or tuning trade-off hyperparameters.

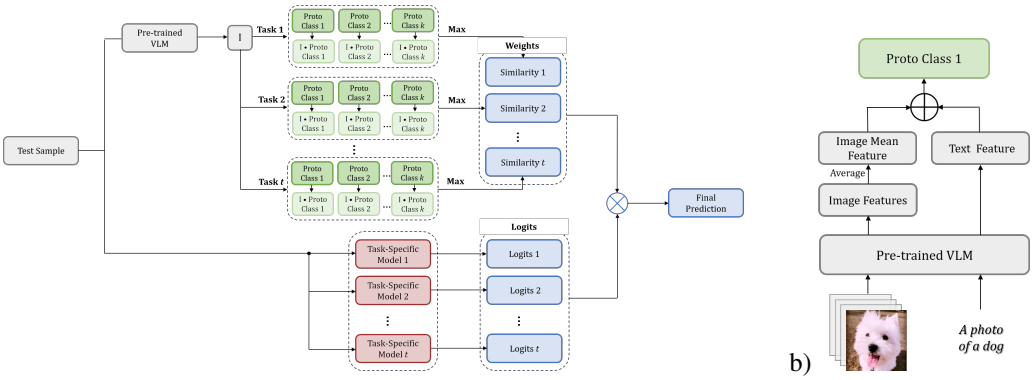


Figure 3: a) The process of Aggregating Prediction: we calculate the cosine similarity between the image embedding of the test sample and prototypes of each category in the feature space of pre-trained VLM, then choose the maximum similarity in each task as the weight of the corresponding task-specific model. b) The process of Computing Prototypes: The prototype of each category is the mean of the image feature vectors plus the text feature vector for that category, all extracted by the original pre-trained VLM.

Model Fusion. Model fusion combines multiple models into a single unified model, retaining the strengths of its constituent models without requiring additional training data. Fisher Merging (Matena & Raffel, 2022) and RegMean (Jin et al., 2022) use Fisher information matrices (Fisher, 1922) and inner-product matrices (Jin et al., 2022), respectively, to compute fusion coefficients for weighted model fusion. Task Arithmetic (Ilharco et al., 2022) introduces a fusion technique that combines models by summing delta models, where a delta model is defined as the difference between the parameters of a fine-tuned model and its pre-trained counterpart. Other approaches, such as TIES Merging (Yadav et al., 2024), Ada Merging (Yang et al., 2023), DARE (Yu et al., 2023), and EMR Merging (Huang et al., 2024), focus on enhancing delta model-based fusion in various ways.

E Introduction of LoRA

Below we revisit LoRA, in which g is the module that PEFT attached to, $\mathbf{e}$ and $\mathbf{h}$ are input and output of the original g and $\mathbf{h}'$ is output of g attached with PEFT.

LoRA (Hu et al., 2021) assumes the change of parameters is in a low-rank space when tuning the pre-trained model on a downstream task. For a linear layer with weight $\mathbf{W} \in \mathbb{R}^{d \times d'}$, the weight updates $\Delta\mathbf{W}$ can be decomposed into the multiplication of two small matrices:

$$\Delta\mathbf{W} = \mathbf{W}_{down}\mathbf{W}_{up},$$

where $\mathbf{W}_{down} \in \mathbb{R}^{d \times r}$ and $\mathbf{W}_{up} \in \mathbb{R}^{r \times d'}$. For the convolution layer, the updates can be reshaped into the kernel shape. Finally, LoRA modifies the forward pass of the adapted layer into the following form:

$$\mathbf{h}' = \mathbf{h} + \mathbf{e} * (\mathbf{W}_{down}\mathbf{W}_{up}),$$

where $*$ is matrix multiplication or convolution operation, the bias and reshape operation are omitted for conciseness. Since LoRA adapts the weight of g, the weight updates can be merged into g to reduce the inference latency.

F Storage Analysis

The masks align the unified delta model's direction with that of each delta model, and the rescalers ensure that the unified model's parameter magnitude matches that of each delta model. Although the masks share the structure of the delta model, their binary nature ensures they require much less storage than the delta models, and the rescalers are t scalars whose storage is negligible. We compare the storage size of model parameter files saved in Python between our method and Individual FT,

The results of Individual FT represent the performance of using a fully fine-tuned model, trained independently on each task based on the pre-trained VLM, for prediction.

First, we look at the comparison in the full fine-tuning scenario. After training all 11 tasks, the storage size for CUD is as follows: Clip Model (570.86 MB) + Unified Delta Model (570.86 MB) + Masks (196.20 MB) + Rescalers (747 KB) = 1377.92 MB. In contrast, Individual FT's storage size is Task-specific Model (570.86 MB) * 11 = 6279.46 MB, saving a total of 4901.54 MB in storage.

Next, we compare in the LoRA (Rank=64) scenario. After training all 11 tasks, the storage size for CUD is as follows: Clip Model (570.86 MB) + Unified Delta Model of LoRA (37.53 MB) + Masks of LoRA (12.89 MB) + Rescalers (747 KB) = 621.28 MB. For Individual FT, the storage size is Clip Model (570.86 MB) + LoRA (37.53 MB) * 11 = 983.51 MB, saving a total of 362.23 MB in storage.

As shown, our method significantly alleviates the excessive storage requirement of Individual FT, and the storage reduction is more pronounced as the proportion of fine-tunable parameters increases and the number of tasks grows.

G Theoretical Analysis

G.1 Definitions

For a given task T^i, where $i \in [1, \ldots, N]$, the corresponding delta model is defined as $\delta^i = \theta^i - \theta^0$, where $\delta^i \in \mathbb{R}^d$. We define one iteration as consisting of the following steps:

1. Compute the sign vector γ_{uni} using the expression $\gamma_{uni} = sgn\left(\sum_{t=1}^{n} \delta^i\right)$.
2. At each position, select the maximum absolute value among all delta models that share the same sign as γ_{uni} to form the absolute value vector $\epsilon_{uni} \in \mathbb{R}^d$.
3. Define $\odot$ as element-wise multiplication. The unified delta model δ is then defined as $\delta = \gamma_{uni} \odot \epsilon_{uni}$.
4. For task i, define a task-specific mask $M^i = (\delta^i \odot \delta)$, where elements with signs inconsistent with δ are set to 0, and the rest are set to 1.
5. Define a scaling factor $\lambda^i = \frac{\sum \|\delta^i\|_1}{\sum \|(M^i \odot \delta)\|_1}$, where $\|\cdot\|_1$ represents the L_1 norm.
6. The iteration result is $\hat{\delta}^i = \lambda^i \cdot (M^i \odot \delta)$.

Define $\delta^i(j)$ as the result of the i-th delta model after j iterations. Here, the subscript (j) indicates the result after j iterations, assuming a fixed set of n initial delta models without any additions during the iterative process. Additionally, $\epsilon_{uni,k}$ represents the k-th element of the unified delta model ϵ_{uni}.

G.2 Related Lemmas

Lemma G.1 (Sign Preservation of δ^i)**.** *For a task δ^i in a continual learning session, the parameter at any position is guaranteed to preserve its sign. Specifically, if a position in δ^i (denoted as $a^i(1)$) is positive (or negative) after an iteration, then $\forall j, a^i(j) \geq 0$ (or ≤ 0). Moreover, if $a^i_k = 0$, then $\forall j > k$, $a^i(j) = 0$.*

Proof. We observe that $M^i(j) = (\delta^i(j-1) \odot \delta(j))$. Therefore, if the signs remain consistent after an iteration, then $M^i(j) = 1$, and if the signs change, then $M^i(j) = 0$. As a result, positive (or negative) values in δ^i during the iteration process will either retain their signs or become zero.From the definition of $M^i(j)$, if $\delta^i(j-1) = 0$, it implies $M^i(j) = 0$, and thus $\delta^i(j) = 0$.Hence, the latter part of the lemma is also proved. □

Lemma G.2 (Preservation of L_1 Norm)**.** *The L_1 norm of δ^i remains invariant under the iteration process as defined, satisfying $\left\|\delta^i(j)\right\|_1 = \left\|\delta^i\right\|_1$ for all iterations j.*

Proof. This follows directly from the definition of λ^i. □

G.3 Main Theorems

Theorem G.3 (Convergence of Iteration). *Given n initial delta models δ^i, where $i \in [1, \ldots, n]$, after infinitely many iterations, if the relative order of λ^i values remains unchanged and $\forall i \neq j$, $\{k \mid M_k^i = 1 \text{ and } M_k^j = 1\} \neq \varnothing$, then these n delta models will converge to a uniquely determined set of n delta models.*

Proof. Initially, $\{\delta^i\}_{i=1}^n$ transforms into $\{\delta^i(1)\}_{i=1}^n$. First, we show that $\forall i \in [1, \ldots, n], j > 0$, $M^i(1) = M^i(j)$.

Indeed, from the proof of Lemma G.1, we know that the number of zeros in M^i does not decrease during iterations. Since no new delta models are introduced, the number of zeros in M^i cannot increase either. Thus, the sign of each position remains fixed during iterations.

Let $\delta_k^i(j)$ denote the value at the k-th position of the i-th delta model after j iterations. Note that $\delta^i(1)$ is obtained by scaling ϵ_{uni}^1 with $\lambda^i(1)$ and setting certain positions to 0:

$$\epsilon_{uni,k}(2) = \epsilon_{uni,k}(1) \cdot \max\{\lambda^i(1) \mid M_k^i = 1\}. \tag{1}$$

From Eq. (1), it follows that the delta model $\delta^i(1)$ with the largest $\lambda^i(1)$ contributes all its non-zero values to $\epsilon_{uni}(2)$. Let i_k denote the index of the k-th largest scaling factor $\lambda^i(1)$.

For $j \geq 2$, we show that $\lambda^{i_1}(j) = 1$. By assumption, $\lambda^{i_1}(j) = \max\{\lambda^i(j) \mid i = 1, \ldots, n\}$ for all j. Consider the part of $\delta^{i_1}(j)$ where $M_k^{i_1}(j) = 1$. These positions remain unchanged in $\epsilon_{uni,k}(j)$ during iterations. Hence, $\delta^{i_1}(j) = M^{i_1}(j) \odot \epsilon_{uni}(j) = \delta^{i_1}(j+1)$, and $\lambda^{i_1}(j) = 1$.

Next, we consider i_2 and divide δ^{i_2} into two parts. Let x_t denote the sum of absolute values in δ^{i_2} at positions where $M^{i_1} = 1$ during the t-th iteration, and let y_t represent the sum of absolute values at positions where $M^{i_1} = 0$. By Lemma G.2, we know that $x_t + y_t = c$ for all t, where c is a constant. During each iteration, the values in y_t are incorporated into ϵ_{uni}. Let s denote the sum of absolute values in δ^{i_1} at positions where $M^{i_2} = 1$, which is also a constant.Then, we have:

$$\begin{cases} \lambda^{i_2}(t) = \frac{x_t + y_t}{s + y_t} = \frac{c}{s + y_t}, \\ x_{t+1} = \lambda^{i_2}(t) \cdot s = \frac{cs}{s + y_t}, \\ y_{t+1} = \lambda^{i_2}(t) \cdot y_t = \frac{cy_t}{s + y_t}. \end{cases}$$

Since iterations do not alter the relative order of λ^i, we have $\lambda^{i_2} < \lambda^{i_1} = 1$. Consequently, $0 < y_{t+1} < y_t$, which implies that both x_t and y_t converge, and δ^{i_2} monotonically converges to a stable solution.

Using a similar analysis for $i^3, \ldots, i^n$, it can be shown that all δ^i eventually converge to unique solutions. □

Corollary G.4. *Under the same conditions as Theorem G.3, the following holds:*

$$\lim_{t \to \infty} \frac{1}{n} \sum_{i=1}^{n} \left\| \delta^i(t+1) - \delta^i(t) \right\|_1 = 0.$$

Theorem G.5. *If an initial delta model δ^1 is given, and during the n-th operation, a new delta model δ^n is added, and the current set of delta models $\{\delta^i(n) \mid i \in \{1, \ldots, n+1\}\}$ undergoes one iteration, then under the same conditions as Theorem G.3, and assuming all δ^i are independent and identically distributed, we have:*

1. *The probability of any $M_k^i(j)$ changing becomes negligible as n increases.*
2. *For each position in $\epsilon_{uni}(j)$, the probability of selecting a different corresponding delta model is small, and even if changes occur, their impact is minimal.*

Proof. For (1): From Lemma G.1, if $M_k^i(j)$ changes, it must transition from 1 to 0. This implies that the sign of δ_k^n is different from that of δ_k^i, and $|\delta_k^n| \geq \sum_{i=1}^{n-1} \left|\delta_k^i(n-1)\right|$. Under the assumption of independence and identical distribution, this probability approaches 0 as n increases.

Table 3: Comparison of "Transfer" metric on MTIL with different choice of K.

K	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	Avg
$K=1$		88.42	68.22	44.68	55.26	71.04	88.52	59.45	89.04	64.71	65.39	69.47
$K=2$		88.08	68.5	45.32	57.00	71.17	88.77	62.09	89.23	64.68	66.50	70.13
$K=3$		88.08	68.88	45.74	57.39	71.17	88.74	61.31	89.42	64.90	67.16	70.28
$K=4$		88.08	68.88	45.74	57.02	71.26	88.84	61.23	89.34	65.05	67.82	70.33
$K=5$		88.08	68.88	45.74	55.56	71.04	88.71	61.68	89.02	65.04	67.90	70.17
$K=6$		88.08	68.88	45.74	55.56	70.21	88.51	60.44	88.61	64.99	68.09	69.91
$K=7$		88.08	68.88	45.74	55.56	70.21	88.12	60.91	88.28	64.84	67.94	69.86
$K=8$		88.08	68.88	45.74	55.56	70.21	88.12	59.24	87.71	64.92	67.86	69.63
$K=9$		88.08	68.88	45.74	55.56	70.21	88.12	59.24	87.63	64.83	67.85	69.61
$K=10$		88.08	68.88	45.74	55.56	70.21	88.12	59.24	87.63	64.66	67.79	69.59
$K=11$		88.08	68.88	45.74	55.56	70.21	88.12	59.24	87.63	64.66	67.70	69.58

For (2): Based on the proof of Theorem G.3, if ϵ_{uni} changes its selected corresponding delta model at any position during an iteration, this change can only be caused by one of the two most recently added delta models δ^n. There are only two possible cases:

1. If δ^n affects ϵ_{uni} without undergoing iteration, this is improbable because $\mathbb{E}(|\delta^i|) < \mathbb{E}(|\delta^n|)$ for $i < n$, given that $\delta^i(n-1)$ has been masked.
2. If $\delta^n(1)$ changes the value delta model of ϵ_{uni}, then $\lambda^n(1) > 1$. However, the deviation from 1 is small (since $\delta^n \in \mathbb{R}^d$, where d is large), so the impact remains minimal.

□

Corollary G.6. *Under the same conditions as Theorem G.5, the following holds:*

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \left\| \delta^i(n) - \delta^i(n-1) \right\|_1 = 0.$$

Proof. From Theorem G.3, after one iteration, all delta models gradually stabilize. From Theorem G.5, the impact of newly added delta models on the iteration process is negligible. Thus, Corollary G.4 also implies Corollary G.6. □

H Ablation Study of K of Aggregating

Table 3 presents the inference performance of the "Transfer" metric of CUD(FT) on the MTIL benchmark with different values of K. T1-T11 represent the 11 tasks, AirCraft-SUN397, ordered alphabetically, while Avg denotes the average result. From the table, we can observe that when K is set to 1, the Avg performance is the lowest at 69.47%. As K increases, the Avg performance gradually improves, reaching the best result of 70.33% when K is 4. However, after this point, the Avg performance begins to decrease as K increases further. Overall, the performance of the aggregating mechanism is relatively insensitive to the choice of K, with the best and worst Avg values differing by only 0.86%.

I More Results of Comparison with State-of-the-art Methods

MTIL. Table 4 presents the full version of Table 1.

Few-shot MTIL. Table 6 presents the full version of Table 2.

Task-Agnostic MTIL. Table 5 presents the detailed comparison results of our proposed ConDU and the baselines on the task-agnostic MTIL benchmark. As seen, our method outperforms all baseline methods across all two metrics. The "Average" metric of our method is 78.1% for FT and 78.0% for LoRA, which exceeds the best baseline by 2% and surpasses pre-trained VLM by 20.3%. The "Last" metric of our method is 86.4% for FT and 85.1% for LoRA, which exceeds the best baseline by 1.8% and surpasses the pre-trained VLM by 28.6%. These results highlight our approach's effectiveness in mitigating generalization and catastrophic forgetting while progressively incorporating new knowledge even without a task ID.

Table 4: Comparison with SOTA methods on MTIL benchmark in terms of "Transfer", "Average.", and "Last" scores (%). We label the best methods on average of all datasets with **bold** styles.

	Method	Aircraft	Caltech101	CIFAR100	DTD	EuroSAT	Flowers	Food	MNIST	OxfordPet	Cars	SUN397	*Average*
	Zero-shot	24.3	88.4	68.2	44.6	54.9	71.0	88.5	59.4	89.0	64.7	65.2	65.3
	Individual FT	62.0	95.1	89.6	79.5	98.9	97.5	92.7	99.6	94.7	89.6	81.8	89.2
Transfer	Continual-FT	-	67.1	46.0	32.1	35.6	35.0	57.7	44.1	60.8	20.5	46.6	44.6
	LwF	-	74.5	56.9	39.1	51.1	52.6	72.8	60.6	75.1	30.3	55.9	58.9
	iCaRL	-	56.6	44.6	32.7	39.3	46.6	68.0	46.0	77.4	31.9	60.5	50.4
	LwF-VR	-	77.1	61.0	40.5	45.3	54.4	74.6	47.9	76.7	36.3	58.6	57.2
	WiSE-FT	-	73.5	55.6	35.6	41.5	47.0	68.3	53.9	69.3	26.8	51.9	52.3
	ZSCL	-	86.0	67.4	45.4	50.4	69.1	87.6	61.8	86.8	60.1	66.8	68.1
	MoE	-	88.2	66.9	44.7	54.1	70.6	88.4	59.5	89.0	64.7	65.0	69.1
	MA	-	87.9	68.2	44.4	49.9	70.7	88.7	59.7	89.1	64.5	65.5	68.9
	Primal-RAIL	-	88.4	68.2	44.6	54.9	71.0	88.5	59.6	89.0	64.7	65.2	69.4
	Dual-RAIL	-	88.4	68.2	44.6	54.9	71.0	88.5	59.6	89.0	64.7	65.2	69.4
	CoLeCLIP	-	88.2	65.1	44.7	54.1	68.8	88.5	59.5	89.0	64.7	65.1	68.8
	DPeCLIP	-	88.2	67.2	44.7	54.0	70.6	88.2	59.5	89.0	64.7	64.8	69.1
	MulKI	-	87.8	69.0	46.7	51.8	71.3	88.3	64.7	89.7	63.4	68.1	70.1
	CUD(FT)	-	88.1	68.9	46.4	57.1	71.4	88.7	65.5	89.3	65.0	67.8	**70.8**
	CUD(LoRA)	-	88.1	68.9	45.7	57.0	71.3	88.8	61.2	89.3	65.1	67.8	70.3
Average	Continual-FT	25.5	81.5	59.1	53.2	64.7	51.8	63.2	64.3	69.7	31.8	49.7	55.9
	LwF	36.3	86.9	72.0	59.0	73.7	60.0	73.6	74.8	80.0	37.3	58.1	64.7
	iCaRL	35.5	89.2	72.2	60.6	68.8	70.0	78.2	62.3	81.8	41.2	62.5	65.7
	LwF-VR	29.6	87.7	74.4	59.5	72.4	63.6	77.0	66.7	81.2	43.7	60.7	65.1
	WiSE-FT	26.7	86.5	64.3	57.1	65.7	58.7	71.1	70.5	75.8	36.9	54.6	60.7
	ZSCL	45.1	92.0	80.1	64.3	79.5	81.6	89.6	75.2	88.9	64.7	68.0	75.4
	MoE	37.4	93.9	80.5	68.3	81.9	84.1	90.0	74.0	90.6	67.7	66.4	75.9
	MA	50.2	91.9	83.1	69.4	78.9	84.0	89.1	73.7	89.3	67.7	66.9	76.7
	Primal-RAIL	51.9	95.8	80.1	70.3	81.1	86.1	89.0	73.9	90.2	68.4	66.4	77.6
	Dual-RAIL	52.5	96.0	80.6	70.4	81.3	86.3	89.1	73.9	90.2	68.5	66.5	77.8
	CoLeCLIP	48.7	94.3	76.6	69.2	79.0	83.8	89.7	73.3	90.5	68.0	66.5	76.3
	DPeCLIP	49.9	94.9	82.4	69.4	82.2	84.3	90.0	74.0	90.4	68.3	66.3	77.5
	MulKI	52.5	93.6	79.4	67.0	79.8	83.9	89.6	77.1	91.2	67.1	69.1	77.3
	CUD(FT)	59.6	93.4	83.7	68.1	83.4	83.7	90.1	76.7	90.6	68.6	68.6	**78.8**
	CUD(LoRA)	51.9	94.9	84.4	69.8	81.1	84.4	90.0	77.3	89.5	69.0	69.3	78.3
Last	Continual-FT	31.0	89.3	65.8	67.3	88.9	71.1	85.6	99.6	92.9	77.3	81.1	77.3
	LwF	26.3	87.5	71.9	66.6	79.9	66.9	83.8	99.6	92.1	66.1	80.4	74.6
	iCaRL	35.8	93.0	77.0	70.2	83.3	88.5	90.4	86.7	93.2	81.2	81.9	80.1
	LwF-VR	20.5	89.8	72.3	67.6	85.5	73.8	85.7	99.6	93.1	73.3	80.9	76.6
	WiSE-FT	27.2	90.8	68.0	68.9	86.9	74.0	87.6	99.6	92.6	77.8	81.3	77.7
	ZSCL	40.6	92.2	81.3	70.5	94.8	90.5	91.9	98.7	93.9	85.3	80.2	83.6
	MoE	34.6	94.7	82.7	76.9	97.7	94.8	91.9	99.4	94.7	80.9	80.5	84.4
	MA	49.8	92.2	86.1	78.1	95.7	94.3	89.5	98.1	89.9	81.6	80.0	85.0
	Primal-RAIL	51.9	96.5	82.8	80.0	96.0	98.7	89.7	98.8	93.3	84.8	78.7	86.5
	Dual-RAIL	52.5	96.8	83.3	80.1	96.4	99.0	89.9	98.8	93.5	85.5	79.2	86.8
	CoLeCLIP	48.7	94.9	78.8	78.4	88.9	96.3	91.1	97.6	94.4	82.7	80.2	84.7
	DPeCLIP	49.9	95.6	85.8	78.6	98.4	95.8	92.1	99.4	94.0	84.5	81.7	86.9
	MulKI	49.7	93.0	82.8	73.7	96.2	92.3	90.4	99.0	94.8	85.2	78.9	85.1
	CUD(FT)	58.6	93.7	86.6	76.1	98.2	93.4	91.9	99.6	94.8	84.9	80.5	**87.1**
	CUD(LoRA)	48.9	95.2	87.8	78.5	96.3	95.2	91.7	97.6	93.0	85.3	78.8	86.2

J OTHER ORDER OF MTIL

To further validate the effectiveness of our approach, we refer to existing works (Zheng et al., 2023; Zhang et al., 2024) and present the MTIL experimental results based on another task order in Table 7, with all other experimental settings unchanged. Order II is StanfordCars, Food, MNIST, OxfordPet, Flowers, SUN397, Aircraft, Caltech101, DTD, EuroSAT, CIFAR100. Only a few baselines have reported results for this task order, and we present the results of these baselines in the table. Even with a different task order, we still achieve performance beyond the state-of-the-art.

Table 5: Comparison with SOTA methods on task-agnostic MTIL benchmark in terms of "Transfer", "Average.", and "Last" scores (%). We label the best methods on average of all datasets with **bold** styles. Individual FT can not be utilized on task-agnostic MTIL, so the Individual FT results here is the prediction with task ID while other methods cannot know the task ID.

	Method	Aircraft	Caltech101	CIFAR100	DTD	EuroSAT	Flowers	Food	MNIST	OxfordPet	Cars	SUN397	*Average*
	Zero-shot	24.4	63.7	41.0	39.3	53.0	70.0	88.4	39.6	88.9	64.5	63.3	57.8
	Individual FT	62.0	95.1	89.6	79.5	98.9	97.5	92.7	99.6	94.7	89.6	81.8	89.2
Average	Continual-FT	25.5	81.5	59.1	53.2	64.7	51.8	63.2	64.3	69.7	31.8	49.7	55.9
	ZSCL	46.3	68.3	74.3	56.3	79.1	81.4	89.5	74.0	89.0	64.4	67.5	71.8
	MoE	37.2	65.3	79.5	67.6	19.7	83.1	80.5	74.0	88.5	67.5	65.3	66.2
	Primal-RAIL	42.4	88.5	57.1	55.7	64.7	80.7	83.0	62.9	84.8	68.7	63.7	68.4
	Dual-RAIL	45.0	88.8	57.8	56.8	66.2	81.0	85.2	63.4	87.8	68.9	64.7	69.6
	CoLeCLIP	48.2	77.8	71.7	65.7	76.8	83.8	89.6	72.2	90.3	68.0	66.4	73.7
	DPeCLIP	49.9	85.3	81.5	65.3	81.6	84.3	89.9	74.0	90.4	68.3	66.2	76.1
	ConDU(FT)	59.7	90.4	83.6	67.0	81.8	83.6	90.2	75.0	90.8	68.7	68.4	**78.1**
	ConDU(LoRA)	51.8	94.4	84.2	68.8	80.0	84.1	90.0	77.1	88.9	68.8	69.3	78.0
Last	Continual-FT	31.0	89.3	65.8	67.3	88.9	71.1	85.6	99.6	92.9	77.3	81.1	77.3
	ZSCL	42.5	64.4	67.2	54.8	89.7	90.4	91.7	95.8	93.4	85.2	78.3	77.6
	MoE	34.1	47.6	80.9	75.5	0.0	93.0	70.8	99.4	86.4	79.8	68.9	66.9
	Primal-RAIL	41.9	94.0	73.7	67.8	84.4	97.0	83.4	92.6	86.9	75.7	71.4	79.0
	Dual-RAIL	45.2	94.4	74.7	70.7	87.3	97.9	86.5	92.8	91.9	81.7	76.7	81.8
	CoLeCLIP	48.1	73.1	65.2	69.6	84.0	96.2	90.9	94.6	93.5	82.6	79.3	79.7
	DPeCLIP	49.9	84.21	83.2	71.1	97.0	95.8	92.0	99.4	93.9	84.5	80.2	84.6
	ConDU(FT)	58.6	90.8	86.3	74.0	96.3	93.4	91.9	99.6	94.7	84.9	80.1	**86.4**
	ConDU(LoRA)	48.4	94.4	87.3	77.1	94.1	94.3	90.8	96.2	90.8	84.3	78.1	85.1

Table 6: Comparison with SOTA methods on few-shot MTIL benchmark in terms of "Transfer", "Average.", and "Last" scores (%). We label the best methods on average of all datasets with **bold** styles.

	Method	Aircraft	Caltech101	CIFAR100	DTD	EuroSAT	Flowers	Food	MNIST	OxfordPet	Cars	SUN397	*Average*
	Zero-shot	24.3	88.4	68.2	44.6	54.9	71.0	88.5	59.6	89.0	64.7	65.2	65.3
	Individual FT	30.6	93.5	76.8	65.1	91.7	92.9	83.3	96.6	84.9	65.4	71.3	77.5
Transfer	Continual FT	-	72.8	53.0	36.4	35.4	43.3	68.4	47.4	72.6	30.0	52.7	51.2
	LwF	-	72.1	49.2	35.9	44.5	41.1	66.6	50.5	69.0	19.0	51.7	50.0
	LwF-VR	-	82.2	62.5	40.1	40.1	56.3	80.0	60.9	77.6	40.5	60.8	60.1
	WiSE-FT	-	77.6	60.0	41.3	39.4	53.0	76.6	58.1	75.5	37.3	58.2	57.7
	ZSCL	-	84.0	68.1	44.8	46.8	63.6	84.9	61.4	81.4	55.5	62.2	65.3
	MoE	-	87.9	68.2	44.1	48.1	64.7	88.8	69.0	89.1	64.5	65.1	68.9
	Primal-RAIL	-	88.4	68.2	44.6	54.9	71.0	88.5	59.6	89.0	64.7	65.2	69.4
	Dual-RAIL	-	88.4	68.2	44.6	54.9	71.0	88.5	59.6	89.0	64.7	65.2	69.4
	CUD(FT)	-	88.0	69.5	45.6	54.4	71.1	88.7	62.2	88.9	64.4	66.6	70.0
	CUD(LoRA)	-	88.1	68.5	45.6	56.4	71.2	89.0	64.0	88.8	64.9	66.4	**70.3**
Average	Continual FT	28.1	86.4	59.1	52.8	55.8	62.0	70.2	64.7	75.5	35.0	54.0	58.5
	LwF	23.5	77.4	43.5	41.7	43.5	52.2	54.6	63.4	68.0	21.3	52.6	49.2
	LwF-VR	24.9	89.1	64.2	53.4	54.3	70.8	79.2	66.5	79.2	44.1	61.6	62.5
	WiSE-FT	32.0	87.7	61.0	55.8	68.1	69.3	76.8	71.5	77.6	42.0	59.3	63.7
	ZSCL	28.2	88.6	66.5	53.5	56.3	73.4	83.1	56.4	82.4	57.5	62.9	64.4
	MoE	30.0	89.6	73.9	58.7	69.3	79.3	88.1	76.5	89.1	65.3	65.8	71.4
	Primal-RAIL	32.9	94.5	69.9	58.1	71.8	84.4	88.5	70.4	89.0	66.1	65.7	71.9
	Dual-RAIL	36.0	94.2	70.9	58.8	70.6	84.3	88.5	70.3	89.7	66.5	65.8	72.3
	CUD(FT)	33.1	90.5	74.1	58.3	76.2	81.0	87.9	73.4	88.0	64.8	67.1	72.3
	CUD(LoRA)	32.4	92.1	75.4	58.8	75.1	82.9	87.3	74.0	89.3	65.1	67.0	**72.7**
Last	Continual FT	27.8	86.9	60.1	58.4	56.6	75.7	73.8	93.1	82.5	57.0	66.8	67.1
	LwF	22.1	58.2	17.9	32.1	28.1	66.7	46.0	84.3	64.1	31.5	60.1	46.5
	LwF-VR	22.9	89.9	59.3	57.1	57.6	79.2	78.3	77.7	83.6	60.1	69.8	66.9
	WiSE-FT	30.8	88.9	59.6	60.3	80.9	81.7	77.1	94.9	83.2	62.8	70.0	71.9
	ZSCL	26.8	88.5	63.7	55.7	60.2	82.1	82.6	58.6	85.9	66.7	70.4	67.4
	MoE	30.1	89.3	74.9	64.0	82.3	89.4	87.1	89.0	89.1	69.5	72.5	76.1
	Primal-RAIL	32.9	95.1	70.3	63.2	81.5	95.6	88.5	89.7	89.0	72.5	71.0	77.2
	Dual-RAIL	36.0	94.8	71.5	64.1	79.5	95.3	88.5	89.4	91.5	74.6	71.3	**77.9**
	CUD(FT)	33.3	90.7	75.0	63.1	88.8	88.6	87.0	91.8	85.6	66.5	71.9	76.6
	CUD(LoRA)	31.8	92.4	76.7	63.4	86.8	91.8	85.6	93.9	90.3	68.1	70.9	77.4

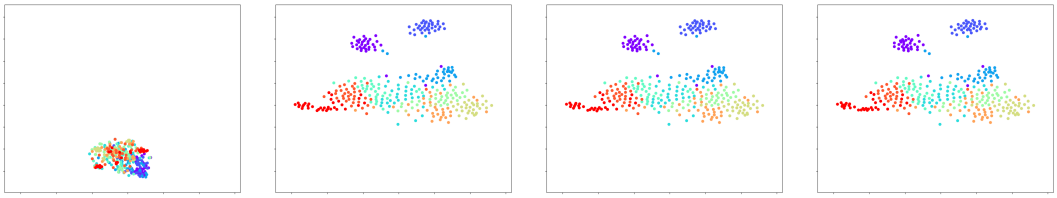
K t-SNE VISUALIZATION OF FEATURE SPACE

To compare the changes in the feature space of task-specific models from initial fine-tuning to the end of all sessions, we perform t-SNE visualization of the features extracted from the training data of Task 1 (AirCraft). For the training data, we sample 10 categories to make the results clearer.

Table 7: Performance (%) Comparison of of state-of-the-art CL methods on MTIL benchmark in Order II

Method	Transfer	Δ	Average	Δ	Last	Δ
CLIP Zero-shot (Radford et al., 2021)	65.4	0.0	65.3	0.0	65.3	0.0
Continual FT	46.6	-18.8	56.2	-9.1	67.4	2.1
LwF (Li & Hoiem, 2017)	53.2	-12.2	62.2	-3.1	71.9	6.6
iCaRL (Rebuffi et al., 2017)	50.9	-14.5	56.9	-8.4	71.6	6.3
LwF-VR (Ding et al., 2022)	53.1	-12.3	60.6	-4.7	68.3	3.0
WiSE-FT (Wortsman et al., 2022)	51.0	-14.4	61.5	-3.8	72.2	6.9
ZSCL (Zheng et al., 2023)	64.2	-1.2	74.5	9.2	83.4	18.1
MoE (Park, 2024)	64.3	-1.1	74.7	9.4	84.1	18.8
MulKI (Zhang et al., 2024)	65.6	0.2	75.0	9.7	84.2	18.9
CUD(FT)	66.5	1.1	75.9	10.6	85.6	20.3

We show the zero-shot results of the pre-trained VLM and the task-specific model 1 at the end of sessions 1, 6, and 11. Figure 4 illustrates that after session 1, the fine-tuned task-specific model 1 shows significantly better data discrimination on Task 1 compared to the pre-trained VLM. Moreover, throughout the continual learning process, task-specific model 1 undergoes multiple rounds of unifying and decoupling, but its feature space changes very little, almost undetectable by t-SNE visualization. This indicates that the task-specific model reconstructed by ConDU closely matches the representation ability of the model obtained through initial fine-tuning.



Pre-Trained Session 1 Session 6 Session 11

Figure 4: t-SNE Visualization of Feature Space.