Model-Adaptive Component Designs for Efficient Deep Neural Network Training

MENGYU ZHENG

Doctor of Philosophy

Supervisor: Dr. Chang Xu

A thesis submitted in fulfilment of
the requirements for the degree of
Doctor of Philosophy

School of Computer Science
Faculty of Engineering
The University of Sydney
Australia

9 December 2025

Statement of Originality

This is to certify that to the best of my knowledge, the content of this thesis is my own work. This thesis has not been submitted for any degree or other purposes.

I certify that the intellectual content of this thesis is the product of my own work and that all the assistance received in preparing this thesis and sources have been acknowledged.

Student Name Signature

Mengyu Zheng

Authorship Attribution Statement

I designed the study and wrote the drafts of the papers that constitute some parts of the thesis. Chapter 3 was published as (Zheng et al. 2024a), Chapter 4 of the thesis was (Zheng et al. 2025b), Chapter 5 was accepted as (Zheng et al. 2024b), and Chapter 6 was accepted as (Zheng et al. 2024c). In addition to the statements above, in cases where I am not the corresponding author of a published item, permission to include the published material has been granted by the corresponding author.

Student Name Signature
Mengyu Zheng

As the supervisor of the candidature upon which this thesis is based, I can confirm that the authorship attribution statements above are correct.

Supervisor Name Signature
Chang Xu

List of Research Outcome

Mengyu Zheng, Yehui Tang, Zhiwei Hao, Kai Han, Yunhe Wang and Chang Xu. Adapt without Forgetting: Distill Proximity from Dual Teachers in Vision-Language Models. ***European Conference on Computer Vision 2024 (ECCV 2024)***, Milano, Italy, Sepetember 29 - Otcober 4 2024.

Mengyu Zheng, Zhiwei Hao, Yeihui Tang and Chang Xu. Visual Prompting via Partial Optimal Transport. ***European Conference on Computer Vision 2024 (ECCV 2024)***, Milano, Italy, Sepetember 29 - Otcober 4 2024.

Mengyu Zheng, Hanting Chen, Tianyu Guo, Chong Zhu, Binfan Zheng, Chang Xu and Yunhe Wang. Enhancing Large Language Models through Adaptive Tokenizers. ***Conference on Neural Information Processing Systems 2024 (NeurIPS 2024)***, Vancouver, Canada, December 12-15 2024.

Mengyu Zheng, Yehui Tang, Yuchuan Tian, Tianyu Guo, Chang Xu and Yunhe Wang. Multimodal Mixture of Experts for Continual Learning in Vision-Language Models. Under review at International Conference on Computer Vision, 2025.

Generative AI attribution statement

The student used ChatGPT for the purposes of text enhancement. The use of this generative AI tool includes spelling corrections, minor sentence restructuring and clarity enhancement. The author confirms that where text was modified by generative AI, the content was reviewed for possible errors, inaccuracies, and bias. The author takes full responsibility for the submitted thesis, confirms the work is their own, and has used generative AI in accordance with University guidelines and policies.

Student Name	Signature
Mengyu Zheng	

Abstract

With the advent of the Transformer architecture, large-scale models have rapidly advanced across various domains such as vision, language, and multimodal learning. Increasing model parameters and enlarging training datasets have become common strategies for improving model performance. However, blindly scaling up model size no longer guarantees significant gains and leads to higher training and deployment costs. As a result, reducing training costs without further increasing model parameters has become a critical challenge, both during model training and in adapting pre-trained models to new tasks.

Although numerous studies have been devoted to improving model training efficiency, most existing approaches focus on general-purpose designs that are broadly applicable across various model architectures and scales. However, they rarely perform adaptive customization of such mechanisms in conjunction with model components (e.g., tokenizer of LLMs, Experts or router mechanisms in MoE LoRA, and label mapping). As is well known, different components and modalities often require distinct learning rate schedules or optimization algorithms to resolve compatibility issues during training. This naturally raises an important question: should training efficiency strategies also be adapted to the interactions of model components? While prior work has largely overlooked the collaborative properties of components and their task-specific connections, we take a step further by exploring component-collaborative training efficiency strategies. We investigate how different implementation strategies enhance adaptability through component coordination across model modalities.

Our study begins by investigating the integration of adaptive components within vision-language models (VLMs), aiming to enhance training efficiency by explicitly exploiting the unique cross-modal interactions between visual and textual modalities. First, building on existing distillation-based continual learning methods, we introduce a sample re-weighting mechanism that dynamically adjusts the influence of each teacher model for each sample. The use of dual teachers ensures that each sample receives appropriate learning guidance.

Moreover, we propose a novel graph-based multimodal proximity distillation approach, which preserves both intra-modal and inter-modal relationships across visual and textual features, leading to improving training efficiency. Second, to further enhance the fine-tuning efficiency of vision-language models (VLMs) for the same task, we investigate whether full parameter updates can be avoided. We design a multimodal Mixture-of-Experts (MoE) architecture that jointly exploits both image and text inputs. This structure mitigates the risk of textual representation collapse while promoting better cross-modal alignment. For each task, only a subset of experts needs to be updated, significantly reducing training time and GPU burden and thereby improving overall efficiency. For unimodal models such as CV backbones or LLMs, which involve relatively simpler information structures, our study focuses on lightweight, input/output-level adaptive components that are decoupled from the backbone. This design reduces training costs and enables flexible, model-aware integration without modifying the original architecture. Third, for large language models (LLMs), we investigate the tokenizer, which serves as a critical interface between raw linguistic data and the model. Since textual inputs must be tokenized before being processed by the model, we propose a simple yet effective tokenizer learning method tailored to each LLM. Starting from a broad initial vocabulary, we refine the tokenizer by tracking perplexity changes during training, allowing us to select a tokenizer that aligns closely with the model's evolving behavior. Compared to training large-scale model parameters, customizing a tokenizer for each model incurs much lower computational costs, significantly reducing training costs. Last, in the context of large-scale vision models, we explore visual prompts as a lightweight and efficient adaptation strategy that only modifies the input layer. We propose an Optimal Transport-based Label Mapping (OTLM) strategy to minimize distribution shifts and reduce the adaptation burden on visual prompts. Unlike existing approaches, our method explicitly models the relationship between the pre-trained model and the downstream task to construct an optimal label alignment, which facilitates better prompt transfer.

Acknowledgements

It has been my great honor and good fortune to receive tremendous help and support throughout my Ph.D. journey. These generous contributions—whether academic, emotional, or logistical—have enabled me to persevere on this long road of research. I would like to express my heartfelt gratitude to everyone who has helped me along the way.

First and foremost, I would like to express my deepest appreciation to my advisor, Dr. Chang Xu. He has provided invaluable guidance and support throughout my research. In the early stages of my Ph.D., he patiently taught me the foundations of computer vision and deep learning. During moments of confusion or stagnation, he offered direction and insight, often reigniting my motivation with his thoughtful advice. I am truly honored to have been his student and to have received his mentorship over the past years. His patience and encouragement when I encountered setbacks, as well as his rigorous academic attitude and self-discipline, have served as a beacon and role model in my research career.

I would also like to extend my sincere thanks to Dr. Yunhe Wang and Dr. Tianyu Guo. In our collaborative research on efficient neural networks, I greatly benefited from their academic insights and valuable discussions, which broadened my horizons and enhanced my research methodologies.

I am also grateful to my research collaborators, including Hanting Chen, Zhiwei Hao, Yehui Tang, Kai Han, Xinghao Chen, Chong Zhu and Zheyuan Bai for their inspiring perspectives, fresh ideas, and strong commitment to scientific inquiry. Our pleasant and productive collaborations have laid a solid foundation for the progress of my research.

My heartfelt thanks also go to my labmates and friends, Linwei Tao, Yuemin Wu, Jianyuan Guo, Qianhan Feng, Xikun Zhang, Kaining Zhang and JiaJun Huang, whose encouragement and companionship made this journey both meaningful and enjoyable.

I deeply appreciate the financial support provided by the university. This support has been essential during my time in Sydney, allowing me to fully dedicate myself to my research and studies without the burden of financial concerns.

Finally, I would like to thank my family, especially my husband. The solitude and challenges of scientific research have been made endurable—and even beautiful—because of your unwavering support, understanding, and love.

Mengyu Zheng

Sydney, Australia, 2025

Contents

Statement of Originality ii

Authorship Attribution Statement iii

List of Research Outcome iv

Generative AI attribution statement v

Abstract vi

Acknowledgements viii

Contents x

List of Figures xvi

List of Tables xviii

Chapter 1 Introduction 1

1.1 Thesis Contributions 5

1.1.1 Contributions of AwoF 5

1.1.2 Contributions of MM-MoE 6

1.1.3 Contributions of ADAT 7

1.1.4 Contributions of OTLM 7

1.2 Thesis Outline 8

Chapter 2 Literature Review 10

2.1 Vision Language Models. 10

2.2 Continual Learning. 11

Chapter 3 Model-Adaptive Distillation Strategy for Vision-Language Models 13

3.1 Related Work 13

3.1.1 Knowledge Distillation. 13
3.2 Motivation 13
3.3 Preliminary 15
3.3.1 CLIP on Continual Learning. 15
3.3.2 Knowledge Distillation in Data-Unknown Scenarios. 16
3.4 Methodology 17
3.4.1 Graph-Based Multi-modal Representation 17
3.4.1.1 First-Order Proximity. 18
3.4.1.2 Second-Order Proximity. 19
3.4.1.3 Comparison with existing knowledge distillation methods. 20
3.4.2 Dual Distillation with Samplewise Balance 21
3.4.3 Algorithm of Method. 23
3.5 Experiment 23
3.5.1 Experimental setup 23
3.5.1.1 Continual learning setting. 23
3.5.1.2 Metric. 25
3.5.1.3 Model. 25
3.5.1.4 Baseline. 25
3.5.1.5 Dataset. 25
3.5.2 Results on MTIL 26
3.5.3 Details of Experimental Results. 27
3.5.4 Ablation study 29
3.5.4.1 Impact of proposed proximities. 29
3.5.4.2 Adaptive weighting. 30
3.5.5 Convergence performance 31
3.5.6 Class Incremental Learning 32
3.6 Conclusion 34

Chapter 4 Model-Adaptive Mixture-of-Experts Architecture for Vision-Language Models 35

4.1 Related Work 35

4.1.1 Mixture of Experts. 35
4.2 Motivation 35
4.3 Preliminary 37
4.3.1 CLIP 38
4.3.2 Mixture-of-Experts in Continual Learning. 38
4.4 Methodology 39
4.4.1 Problem Analysis 39
4.4.1.1 Diversity Collapse in Text Modality 40
4.4.1.2 Disruption of alignment. 40
4.4.2 Multi-Modality Router 41
4.4.3 Dynamic Expert Expansion Mechanism 42
4.4.4 Algorithm of the proposed MM-MoE 43
4.5 Experiments 44
4.5.1 Implementation Details 44
4.5.1.1 Continual learning setting. 44
4.5.1.2 Datasets. 45
4.5.1.3 Model. 45
4.5.1.4 Baselines. 46
4.5.2 Results of Multi-domain Task Incremental Learning 46
4.5.2.1 Main Results 46
4.5.2.2 Details of Experimental Results under MTIL benchmark 48
4.5.3 Results of Class Incremental Learning 49
4.5.3.1 Main Results 49
4.5.3.2 More Results under CIL benchmark 53
4.5.4 Analysis of the Compute Costs 53
4.5.5 Ablation Study 54
4.5.5.1 Impact of Multi-Modality Router and Dynamic Expert Expansion Mechanisms. 54
4.5.5.2 Analysis of incremental MoE. 55
4.5.5.3 Impact of λ. 56

4.5.6 Discussion 56
4.6 Conclusion 57
4.6.1 Limitations and Future Work 57

Chapter 5 Adaptive Input-Side Components for Large Language Models 58
5.1 Related Work 58
5.1.1 Subword tokenizer 58
5.1.2 Learnable tokenizer 59
5.2 Motivation 60
5.3 Adaptive Tokenizers 61
5.3.1 Unigram Model 63
5.3.2 LLM-Enhanced Tokenization 64
5.3.2.1 Random sampling 65
5.3.2.2 Loss momentum 65
5.4 Experiments 66
5.4.1 Experimental Setup 66
5.4.1.1 Model Framework 66
5.4.1.2 Data Corpus 66
5.4.1.3 Baseline Methods 67
5.4.1.4 Evaluation Metrics 67
5.4.2 Tokenization Methods Evaluation 68
5.4.3 Scalability 69
5.4.4 Cross-Model Adaptability 71
5.4.5 Model and Vocabulary Size 72
5.4.6 Ablation Study 73
5.4.6.1 Corpus Size used in Loss Calculation 73
5.4.6.2 Initial Vocabulary Size 74
5.4.6.3 Momentum Strategy 75
5.4.6.4 Balance Strategy 75
5.4.7 Additional analysis experiments 76
5.4.7.1 Analysis of the Compute Costs 76

5.4.7.2 Compression Rate results. 77
5.4.7.3 Analysis of Differences Between Vocabularies 77
5.4.7.4 Impact of Training Epochs 78
5.4.7.5 Expanded Analysis on Infer Data Volume Tokens and Initial Vocabulary Size 79
5.5 Conclusion 79

Chapter 6 Adaptive Output-Side Components for Vision Transformers 81
6.1 Related Work 81
6.1.1 Natural language prompting and visual prompting 81
6.1.2 Label mapping for visual prompting 81
6.1.3 Optimal transport in machine learning 82
6.2 Motivation 82
6.3 Preliminaries 85
6.3.1 Visual Prompting (VP) Framework. 85
6.4 Methodology 86
6.4.1 Label Mapping from an Optimal Transport Perspective 86
6.4.2 Cost Matrix Definition 88
6.4.2.1 Inter-sample Cost Definition. 88
6.4.2.2 Intra-sample Cost Definition. 89
6.4.3 Solution of the Optimal Transport Plan 91
6.4.4 Analysis 91
6.5 Experiment 92
6.5.1 Setup 92
6.5.1.1 Frameworks. 92
6.5.1.2 Baselines. 93
6.5.1.3 Backbones Specifications 93
6.5.1.4 Downstream Datasets. 93
6.5.2 Performance Under the VP Framework 94
6.5.2.1 Main results 94
6.5.2.2 More Results on Other Backbones 96

6.5.3 Performance Under the VPT Framework 97
6.5.4 Data Efficiency in Downstream Task Transfer 99
6.5.5 Scalability 99
6.5.6 Ablation Study 101
6.5.7 Data Efficiency 102
6.5.8 Discussion on Tunable Parameters and Running Speed 102
6.5.8.1 Tunable parameters. 102
6.5.9 Visualization 103
6.5.9.1 Visualize mapping results. 103
6.5.9.2 Explanation of Different Label Mapping Strategies. 103
6.6 Conclusion 104

Chapter 7 Conclusion **105**
7.1 Future Work 105

Appendix A Appendix for Chapter 5 **108**
A1 Results of ADAT in the 1B model. 108
A2 Details of Model Parameters 108
A3 List of Training Dataset 109

Appendix B Appendix for Chapter 6 **113**
B1 Running speed. 113

Appendix Bibliography **114**

List of Figures

1.1 Algorithms for model adaptive for training efficiency. 5

3.1 Overview of the proposed continual learning framework for vision-language models. (a) The dual-teacher distillation with CLIP models C_0 and C_{i-1} using reference data. (b) Cross-modal graph construction for first-order and second-order proximity modeling. (c) The adaptive re-weighting mechanism for balanced knowledge distillation from different teacher. 18

3.2 Convergence performance. (a) Convergence performance of ZSCL and our proposed method on Aircraft. Result achieved by Fine-tune performs as the upper bound. (b) Immediate accuracy measured after training on each task during continual learning. 33

4.1 Illustrates the overall framework of the proposed Multimodal MoE. (a) Lora-MoE Integration with Pre-training Module: A multimodal router allocates experts based on image and text features. For each task, a dynamic router and m experts are introduced, with the router accessing all previous experts but only the parameters associated with the current task are updated. (b) The structure of the proposed Multimodal Router, where the gating function integrates image and text modalities to determine the activated experts. 40

4.2 Impact of λ 56

5.1 Illustration of the proposed ADAT pipeline. (a) The traditional tokenizer algorithm that directly extracts vocabulary from data. (b) The framework of the LLM-enhanced tokenizer, iteratively refining vocabulary based on model feedback. (c) Overview of ADAT, encompassing initial tokenizer acquisition, training and inference to derive token losses, token pruning based on scores and losses. 62

6.1 Overview of OTLM: (a) VP pipeline entails transforming input by incorporating prompts with images and transforming output by mapping source labels to

target labels; (b) Label mapping within the Partial Optimal Transportation (POT) framework involves computing the transportation cost as the sum of the pointwise multiplication between the mapping matrix and the cost matrix; (C) The design of the cost matrix involves obtaining predictions of the entire target dataset and then aggregating them by class through grouped summation. In FLM/ILM, the cost at each position is defined by transporting the intra-class distribution into the selected position, while OTLM defines it by transporting the inter-class distribution outside the selected position. 87

6.2 Data efficiency of variant label mapping approaches. Compared to FLM and ILM, our method demonstrates significantly higher accuracy gains over VP, particularly evident when the downstream dataset has fewer averaged samples per class. This highlights exceptional data efficiency of our method. 100

6.3 Comparison of different source datasets by employing Vit-B source models pretrained on either ImageNet-1K or ImageNet-22K. The performance improvement over VP and average cost under the CUB200 downstream dataset is reported. 101

6.4 Visualize mapping results of three label mapping methods on CIFAR10 dataset under the VP framework. 103

6.5 Explanation of the difference between frequency-based LM and OTLM. (a) The blue line represents the mapping of the ‘gray jay’ selected from the source labels by FLM. The green line represents the mapping of the ‘gray’ jay selected from the source labels by OTLM. (b) The orange line indicates the logit for the class ‘gray catbird’ on the selected label of FLM for ‘gray jay’. (c) and (d) report the logits of ‘gray jay’ on labels selection results by frequency-based LM and OTLM, respectively. 104

List of Tables

3.1 Detailed description of the 11 datasets in the multi-domain task incremental learning benchmark. 26

3.2 Comparison results on the MTIL benchmark under task order setting I. Δ represents the performance gap between a CLIP model after continual learning and the original CLIP model (Zero-shot). 27

3.3 Comparison results on the MTIL benchmark under task order setting II. Δ represents the performance gap between a CLIP model after continual learning and the original CLIP model (Zero-shot). 27

3.4 Performance of trained model on each task in the MTIL benchmark under task order setting I. 28

3.5 Accuracy(%) on every dataset after training on each task in the MTIL benchmark under task order setting I. 29

3.6 Accuracy(%) on every dataset after training on each task in the MTIL benchmark under task order setting II. 30

3.7 Performance of trained model on each task in the MTIL benchmark under task order setting II. 31

3.8 Impact of the first-order and second-order proximity on continual learning performance under Order I and Order II settings. 32

3.9 Comparison of different weighting schemes under Order I and Order II evaluation settings. 32

3.10 Results on the CIFAR-100 dataset under the class-incremental learning (CIL) benchmark with 10, 20, and 50 steps. 33

3.1 Results on the TinyImageNet dataset under the class-incremental learning (CIL) benchmark with 10, 20, and 50 steps. 34

4.1 Results comparison on the MTIL benchmark under task order setting I. Δ indicates the performance difference between each continual learning method and the original CLIP model (Zero-shot). 46

4.2 Results comparison on the MTIL benchmark under task order setting II. Δ indicates the performance difference between each continual learning method and the original CLIP model (Zero-shot). 47

4.3 Performance of the proposed method and baselines on each task in the MTIL benchmark for order I. The best result is highlighted in **bold**, and the second-best is underlined. 48

4.4 Task-wise accuracy (%) across all datasets in the MTIL benchmark, based on task order setting I, following the completion of training on each task. 49

4.5 Task-wise accuracy (%) across all datasets in the MTIL benchmark, based on task order setting II, following the completion of training on each task. 50

4.6 Performance of the proposed method and baselines on each task in the MTIL benchmark for order I. The best result is highlighted in **bold**, and the second-best is underlined. 51

4.7 Performance of the trained model at each step in the CIL benchmark on CIFAR-100. The “Average Accuracy" represents the mean accuracy across all previous tasks.“MM-MoE*" indicates the proposed method without Multi-Modality Router for the two modalities. The best result is highlighted in **bold**. 52

4.8 Performance of the trained model at each step in the CIL benchmark on TinyImageNet. The “Average Accuracy" represents the mean accuracy across all previous tasks. “MM-MoE*" indicates the proposed method without Multi-Modality Router for the two modalities. The best result is highlighted in **bold**. 52

4.9 Results on the CIFAR100 dataset across different steps under CIL benchmark. 53

4.10 Comparison of the number of computational cost during the training process of each task between MM-MoE and others. 54

4.11 Impact of Multi-Modality Router and Dynamic Expert Expansion Mechanism. 54

4.12 Ablation study of Mixture-of-Experts on the number of experts per task and the top-k of router gates. 55

5.1 Comparison of Different Tokenization Methods in Terms of PPL and Accuracy Across Multiple Datasets. 69

5.2 Evaluation on Three Different Model Scales: 70M, 160M, and 410M. 70

5.3 Cross-Model Adaptability of Vocabularies. Performance of vocabularies obtained from different models when applied to models of other sizes. 71

5.4 Impact of Vocabulary Size on Model Performance Across Different Model Sizes. 73

5.5 Ablation Study Results on Inference Data Tokens, Initial Vocabulary Size, Momentum, and different choice of Balance $F(a, b)$. 74

5.6 Comparison of ADAT Optimization Runtime and Training Time Across Different Model Sizes. 76

5.7 Performance comparison of different tokenization methods. 77

5.8 Comparison between Vocabulary of ADAT and Unigram. 78

5.9 Comparison with the Initial Vocabulary (100k size): Percentage of Tokens Falling into Each Score Interval. 78

5.10 Impact of Training Epochs on Model Performance. 79

5.11 Results for different Infer Data Volume Tokens and Initial Vocabulary Sizes. 79

6.1 Specifications of six pre-trained backbones employed in this paper. All of these backbones are pre-trained on ImageNet (Deng et al. 2009). 93

6.2 Description of datasets evaluated under the Visual Prompting and Visual Prompt Tuning frameworks. 94

6.3 Comparison of performance on 10 downstream datasets under the VP framework. The utilized source model is ViT-B pretrained on ImageNet-22K. The highest performance achieved among prompt-based methods is highlighted in **bold**. 95

6.4 Comparison of performance on 10 downstream datasets under the VP framework. The utilized source model is ResNet50 pretrained on ImageNet-1K. The highest performance achieved among prompt-based methods is highlighted in **bold**. 96

6.5 Comparison of performance on 10 downstream datasets under the VP framework. The utilized source model is Swin-B pretrained on ImageNet-22K. The highest performance achieved among prompt-based methods is highlighted in **bold**. 97

6.6 Comparison of performance on 10 downstream datasets under the VP framework. The utilized source model is ViT-B pretrained on ImageNet-1K. The highest performance achieved among prompt-based methods is highlighted in **bold**. 97

6.7 Comparison of performance on 3 downstream datasets under the VP framework. The utilized source models are ViT-L pretrained on ImageNet-1K and ConvNeXt-B pretrained on ImageNet-22K. The highest performance achieved among prompt-based methods is highlighted in **bold**. 98

6.8 Comparison of performance on five datasets under the VPT framework using model Swin-B pretrained on ImageNet-22K. The highest performance achieved among label mapping methods without additional heads is highlighted in **bold**. 98

6.9 Ablation study investigating the effects of various solving algorithms and cost matrix designs on model performance. 101

6.10 Comparison of the efficiency between the proposed model and other models under different proportions of training samples. 102

6.11 Total trainable parameters in the input prompt or model finetuning for all 10 downstream datasets under VP framework on Vit-B pretrained on ImageNet-22K. '$\times$' denotes the multiple of the tunable parameter amount relative to the total amount of pre-trained Vit-B pretrained on ImageNet-22K encoder parameters (85.8M). 103

A.1 Results in the 1B model. 108

A.2 Specifications of different LLMs used in the paper. 108

B.1 Running time(s/epoch) under VP framework on DTD dataset with source model Vit-B pretrained on ImageNet-22K. 113

CHAPTER 1

Introduction

Since the introduction of the Transformer architecture, transformer-based models have become the dominant paradigm across a wide range of research fields, consistently achieving state-of-the-art (SOTA) performance in computer vision (CV), natural language processing (NLP), and multimodal learning. However, the increasing scale of foundation models is evident in recent developments such as LLaVA-OneVision, with 72 billion parameters (Li et al. 2024), and GPT-3, with 175 billion parameters (Brown et al. 2020). While these large-scale models deliver impressive performance, they also impose significant computational and time costs on both academic research and industrial deployment. As a result, improving training efficiency and reducing computational overhead—during both pretraining and fine-tuning phases—has emerged as a key research focus.

Improving the training efficiency of large-scale neural networks has become a central focus in contemporary deep learning research (Shen et al. 2023a). One mainstream line of work involves modifying the model architecture to reduce unnecessary computation. The motivation behind this direction lies in the observation that, while large models are composed of billions of parameters, not all of them are essential during training. To address this, various lightweight architectures have been proposed. For instance, models such as ALBERT (Lan et al. 2019) and MobileBERT (Sun et al. 2020) leverage parameter sharing and layer reduction to decrease model size, while pruning techniques eliminate less important weights or structural components (e.g., Transformer layers or CNN filters). Additionally, Mixture-of-Experts (MoE) architectures adopt sparsely activated expert modules, enabling partial parameter activation during training and inference, thus significantly reducing computational overhead. These architectural interventions offer fine-grained control over internal computation pathways,

allowing more efficient modeling of attention patterns and feature interactions. That said, such methods often require direct modification of the backbone, making them more suitable for improving open-source models with accessible architectures.

Alternatively, for models whose architectures cannot be modified due to cross-task parameter sharing, task-module preservation, catastrophic forgetting avoidance, or deployment constraints, decoupled approaches have gained increasing attention. These methods optimize the input or output interfaces to achieve efficiency gains without altering the backbone. For example, visual prompt tuning introduces a small set of learnable embeddings at the input level to enable efficient fine-tuning on downstream tasks. Similarly, dynamic input resizing allows trade-offs between accuracy and computation by adjusting the image resolution. These approaches are highly flexible and portable, making them well-suited for integration into existing training pipelines and applicable across different models and tasks.

Together, these two classes of methods, architecture-level and external components, constitute the dominant paradigms for improving training efficiency. However, despite their effectiveness, most existing methods adopt model-agnostic designs that do not consider the alignment between added components and the model. For instance, visual prompt methods typically rely on fixed frequency-based label mappings across different architectures, pretraining datasets, and model sizes. For instance, Visual prompt methods typically rely on fixed or frequency-based label mappings across different architectures and pretraining datasets. However, such simple frequency-based mapping between the model and the target task labels does not contribute to improving training efficiency — it merely serves the basic purpose of achieving mapping. Currently, widely used tokenizers such as BPE and Unigram are model-agnostic and do not take the characteristics or performance of the LLM itself into account during training. As a result, they offer little to no benefit in improving training efficiency. In vision-language models (VLMs), MoE and knowledge distillation rarely consider the inter-modal coordination between vision and language representations. While generality facilitates reusability and deployment, it also leads to missed opportunities for model adaptation—especially in multimodal models. Designing adaptive components based

on the characteristics of multimodality and leveraging cross-modal interactions to improve model efficiency is a promising direction worth exploring.

Although existing training acceleration methods generally bring performance gains across a wide range of models, they are often designed as "one-size-fits-all" solutions, overlooking the intrinsic characteristics of individual models. Just as researchers select different activation functions, optimization algorithms, and objective functions tailored to specific architectures, efficiency improvement strategies should also be adapted to the unique traits of each model. To address the above mentioned limitation, we propose a framework of model adaptive efficiency strategies, which explicitly aligns component design with the model's architecture, scale, and modality characteristics. Specifically, we explore adaptive components inside the model for complex VLMs, leveraging the unique cross-modal interactions between vision and language to improve training efficiency. For unimodal models such as CV backbones or LLMs, where the information structure is relatively simpler, we focus on input/output-level adaptive components, aiming to reduce training costs by optimizing external mechanisms in a model-aware fashion. Next, we mainly summarize the specific challenges in the following four aspects.

- Firstly, the proposed adaptive distillation strategy developed for vision-language models. Multi-modal models such as CLIP possess remarkable zero-shot transfer capabilities, making them highly effective in continual learning tasks. However, this advantage is severely compromised by catastrophic forgetting, which undermines the valuable zero-shot learning abilities of these models. Existing methods based on the teacher-student paradigm primarily focus on preserving zero-shot capabilities. While effective in retaining the original performance, they often fail to fully exploit the rich multi-modal information inherent in VLMs, instead distilling the two modalities separately.
- Secondly, the specialized MoE architecture for vision-language models enables fine-tuning without touching the backbone. To further improve fine-tuning efficiency, we explore whether it is possible to freeze all backbone parameters and introduce only a small number of task-specific parameters for each fine-tuning task—a property that

aligns well with the Mixture of Experts (MoE) paradigm. However, adapting MoE to multimodal models such as CLIP poses significant challenges. Existing MoE-CLIP architectures often lack effective cross-modal interaction, resulting in a misalignment between image and text experts. Moreover, this deficiency in interaction limits the diversity of learned representations, preventing the development of text-specific experts. Evidently, simply combining the MoE architecture with the image encoder and text encoder in VLMs is a straightforward yet suboptimal approach. Such a rigid integration limits both the efficiency and effectiveness of model fine-tuning.

- Thirdly, the exploration of adaptive input-side components for large language models. Tokenizers serve as crucial interfaces between models and linguistic data, substantially influencing the efficacy and precision of large language models (LLMs). Traditional tokenization methods often rely on static frequency-based statistics and are not inherently synchronized with LLM architectures, which may limit the efficency of the model.
- Lastly, the investigation into adaptive output mechanisms for vision transformers. Visual prompts provide a lightweight approach for adapting pre-trained models to downstream tasks without modifying model parameters. This adaptation is typically achieved by modifying the input through prompt engineering and the output through label mapping. While existing studies have largely focused on refining prompts, label mapping strategies remain predominantly frequency-based and have received limited attention. However, such approaches clearly overlook the compatibility between the mapping strategy and the model or prompt training dynamics. We argue that an ideal label mapping approach should minimize the extent of output adjustments required by the model to align with the target labels, thereby improving the overall efficiency of prompt-based adaptation.

Building upon the four identified issues and underexplored potentials, this thesis presents four corresponding algorithms, each introduced in a dedicated chapter to address these gaps, including Distill Proximity from Dual Teachers in Vision-Language Models (AwoF), Multimodal Mixture of Experts for Vision-Language Models (MM-MoE), Adaptive Tokenizers for large language models (ADAT), Optimal Transport-based Label Mapping strategy (OTLM).

1.1 Thesis Contributions

Chapters 3 to 6 present four model-specifc adaptive components designed to enhance model efficiency. An overview of these algorithms is provided in Figure 1.1. Each proposed algorithm is developed to address a specific challenge or untapped potential identified earlier. This section summarizes the main contributions of each algorithm.

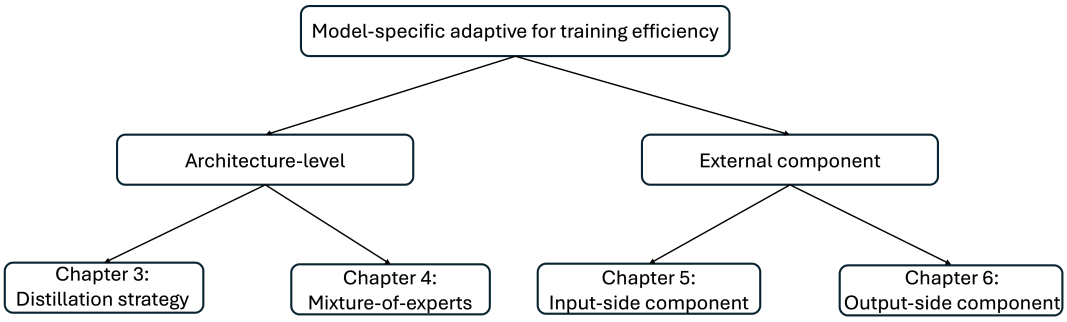


FIGURE 1.1. Algorithms for model adaptive for training efficiency.

1.1.1 Contributions of AwoF

In addressing the identified shortcomings of existing continual learning strategies for multi-modal models, our research emphasizes alleviating the conflict between maintaining zero-shot transfer abilities and adapting to new datasets by proposing a model-specific adaptive knowledge distillation strategy. This challenge is particularly significant, as preserving zero-shot capabilities often hinders the model's plasticity to new data. Our study introduces an innovative dual-teacher distillation strategy, which focuses on maintaining the relative proximity between samples rather than directly modeling individual outputs. Relative relationships are more conducive to preserving a model's inherent capabilities while ensuring its adaptability. To effectively model these proximity, we introduce a sophisticated graph-based approach that captures both intra- and inter-modal interactions. Then a proximity distillation method is conducted with this graph to learn and preserve relationships from teachers. Additionally, we propose a sample re-weighting mechanism designed to dynamically balance the insights from the two teacher models from sample aspect, which allows us to further address the tension between model stability and plasticity.

In summary, our approach revolutionizes continual learning for multi-modal models by introducing a graph-based multi-modal proximity distillation coupled with an innovative sample re-weighting mechanism. A key contribution of our work is the novel use of a graph structure to reveal both inter- and intra-modal relative relationships for CLIP models. This methodology not only resolves the inherent conflict between maintaining zero-shot capabilities and adapting to new datasets but also significantly enhances the adaptability and stability of multi-modal models in continue learning learning environments.

1.1.2 Contributions of MM-MoE

In order to address the adaptation challenges of integrating the MoE architecture with vision-language models, we propose a Multimodal MoE architecture that optimizes the interaction between image and text modalities while preserving their alignment and addressing the modality imbalance. Rather than independently altering text features, our architecture refines task-specific label representations in the context of their associated images. Specifically, textual embeddings are dynamically adjusted during their interaction with image embeddings, ensuring that modifications remain contextual and localized to specific image-text pairs. This approach preserves the distribution of textual features while enabling fine-grained, task-specific adjustments. By anchoring textual updates to the visual context, we balance the learning dynamics between the modalities, ensuring both contribute equally to the shared representation space. Experimental results demonstrate a considerable improvement over existing methodologies, which illustrate the effectiveness of the proposed method in the field of continual learning.

This method can be considered as a context-aware adaptation not only maintains semantic consistency but also improves the utilization of textual experts, allowing them to handle more diverse and meaningful interactions, effectively mitigating the imbalance between modalities. Additionally, we introduce a dynamic expert expansion mechanism that scales the expert pool as new tasks emerge, ensuring sufficient capacity for both old and new tasks. Our architecture leads to significant performance gains across various benchmarks, demonstrating its effectiveness in multimodal continual learning. Extensive experiments demonstrate that

our approach significantly enhances CLIP's performance in continual learning, achieving superior results across diverse benchmarks.

1.1.3 Contributions of ADAT

While traditional tokenization methods have been instrumental in enhancing the efficiency and effectiveness of large language models (LLMs), they are typically decoupled from the model's learning mechanisms. This means they do not adapt or evolve based on the model's performance or the specific requirements of the tasks being addressed. Instead, these methods prioritize compressing the vocabulary size, which can sometimes lead to suboptimal performance in complex language tasks where adaptability and contextual understanding are crucial. Recent advances in end-to-end learnable tokenization [17, 16, 38] aim to address these deficiencies by more closely integrating tokenization with the model's learning processes. However, these systems, while innovative, introduce significant computational overhead (e.g., gradient-based tokenization and pooling modules) and lack the flexibility of traditional tokenization methods, which can be easily transferred across different models.

In this study, we address the limitations of traditional tokenization methods by creating a system where the tokenizer's development is coupled with the performance of the LLM itself. Specifically, we introduce an adaptive tokenizer that begins with a comprehensive initial vocabulary. As training progresses, we fine-tune this tokenizer by closely monitoring the model's perplexity. This ongoing adjustment allows the tokenizer to evolve in tandem with the LLM, ensuring that the tokenization process remains optimally aligned with the model's dynamic learning patterns. Our empirical results confirm that this adaptive approach markedly improves accuracy over traditional methods, demonstrating its potential to significantly enhance LLM functionality.

1.1.4 Contributions of OTLM

The selection of label mapping strategies significantly impacts the effectiveness of visual prompting. A carefully designed label mapping can significantly reduce the alterations

required to adjust the output distribution to match the target labels. Less modification implies a more efficient utilization of prompt optimization capabilities, leading to improved outcomes. Existing label mapping methods have not enhanced the effectiveness from this perspective.

To address this, we propose an **O**ptimal **T**ransport-based **L**abel **M**apping strategy (**OTLM**) that effectively reduces distribution migration and lessens the modifications required by the visual prompts. Specifically, we reconceptualize label mapping as a partial optimal transport problem, and introduce a novel transport cost matrix. Through the optimal transport framework, we establish a connection between output-side label mapping and input-side visual prompting. Additionally, we analyze frequency-based label mapping methods within this framework. We also offer an analysis of frequency-based label mapping techniques and demonstrate the superiority of our OTLM method. Our experiments across multiple datasets and various model architectures demonstrate significant performance improvements, which prove the effectiveness of the proposed method.

1.2 Thesis Outline

As outlined above, this thesis explores solutions to existing challenges and untapped potentials through the lens of component-model adaptability. The structure of this thesis is organized as follows:

Chapter 1 reviews existing approaches to improving model training efficiency, which can be broadly categorized into two types: modifying the model architecture and modifying input/output components. We further identify an underexplored potential in these approaches—their insufficient consideration of the compatibility between the components and the underlying model architecture. In addition, we outline the main contributions of this thesis, which include four proposed algorithms designed to address this gap.

Chapter 2 reviews the foundational literature relevant to our work, providing a general overview of the research landscape. More focused discussions of related work are provided in later chapters where appropriate.

In Chapter 3, we investigate an adaptive distillation strategy tailored for vision-language models. We propose a novel graph-based multi-modal proximity distillation method that effectively preserves both intra- and inter-modal information across visual and textual modalities. To further improve its effectiveness, a sample re-weighting mechanism is introduced to dynamically adjust the contribution of teacher models on a per-sample basis.

Chapter 4 extends our investigation to the possibility of freezing the backbone in order to further reduce the training time cost. In this context, we introduce a multimodal MoE architecture that jointly leverages image and text inputs, effectively mitigating textual diversity collapse and improving cross-modal alignment.

In Chapter 5, we explore adaptive input-side components for large language models. We propose a simple yet effective approach for learning tokenizers that are specifically designed to integrate seamlessly with LLMs. This method, referred to as Adaptive Tokenizer (ADAT), dynamically aligns tokenization with the model's training dynamics.

In Chapter 6, we investigate adaptive output-side components for vision transformers (ViTs). We introduce an optimal transport-based label mapping strategy (OTLM) that effectively mitigates distributional shift and reduces the extent of modifications required in visual prompt design.

In Chapter 7, we conclude the thesis by summarizing the key findings and outlining potential directions for future research.

CHAPTER 2

Literature Review

2.1 Vision Language Models.

Inspired by the human multi-modal learning process, pretrained vision-language models (VLMs) have been proposed and are rapidly gaining attention for their ability to handle the complexities inherent in multimodal tasks (Li et al. 2022a; Liu et al. 2024). Unlike traditional vision models such as ViT (Dosovitskiy et al. 2020a; Jia et al. 2021; Yao et al. 2021), VLMs demonstrate impressive zero-shot generalization (**zhang2023vision**), enabling their application across a broader range of downstream tasks (Zhou et al. 2022; Gao et al. 2023b), including image captioning, visual question answering, and text-to-image generation (Li et al. 2023a). A representative model in this field is CLIP (Radford et al. 2021), which consists of an image encoder and a text encoder trained with contrastive learning (Chen et al. 2020b) to align the embeddings of paired image-text samples while separating mismatched pairs. Thanks to pretraining on massive image-text data, CLIP not only achieves remarkable zero-shot performance on classification tasks without fine-tuning, but also serves as a backbone for various vision and multimodal benchmarks (Zhang et al. 2024b; Geng et al. 2023). Maintaining such zero-shot capability is thus essential for downstream performance. Building upon CLIP, recent methods like ADEM-VL (Hao et al. 2024) further enhance training efficiency by introducing parameter-free similarity metrics and reformulated cross-attention mechanisms, significantly reducing the number of trainable parameters. Meanwhile, some studies focus on scaling up CLIP and continuously improving its zero-shot performance. EVA-CLIP-18B (Sun et al. 2024) expands the pretraining to an 18B-parameter scale and achieves strong results on more than 20 benchmarks. Meanwhile, SigLIP-2 (Tschannen et al.

2025) enhances the extraction of dense features, enabling better zero-shot, retrieval, and feature representation capabilities, and achieves breakthroughs in segmentation and depth estimation tasks.

2.2 Continual Learning.

Continual Learning (CL) aims to train models sequentially on a long series of tasks while maintaining performance across all previously learned tasks without suffering from catastrophic forgetting (Li and Hoiem 2017; Wang et al. 2024). Existing approaches to mitigating forgetting can be broadly categorized into three classes: replay-based, regularization-based (Li and Hoiem 2017; Kirkpatrick et al. 2017; Aljundi et al. 2018), and parameter isolation methods (Mallya and Lazebnik 2018; Fernando et al. 2017). Replay-based methods (Rebuffi et al. 2017; Buzzega et al. 2020; Kamra et al. 2017) store and reuse past data during training on new tasks, but face scalability issues and data privacy constraints, especially in vision-language models (VLMs), where data access is often restricted. Parameter isolation methods (Li et al. 2019; Yoon et al. 2017; Wang et al. 2020) assign a dedicated sub-network or parameters to each task, effectively avoiding interference but leading to rapid model size growth when applied to long task sequences. Regularization-based methods instead constrain model updates to retain prior knowledge, often via knowledge distillation.

In the context of CLIP-based continual learning (Thengane et al. 2022), such methods are particularly promising due to CLIP's inherent zero-shot capabilities. Techniques like LWF-VR (Ding et al. 2022) distill generated sentences from the vocabulary to simulate both zero-shot and seen classes, while ZSCL (Zheng et al. 2023a) introduces reference images and sentences to compute KD-divergence from CLIP across tasks. However, these approaches often focus on incorporating auxiliary data, without jointly optimizing for both zero-shot transfer and the retention of previously acquired knowledge. Models like DyTox (Douillard et al. 2022) further address practical deployment concerns, such as task identifier dependency, by introducing task-attention modules, underscoring the importance of scalability and flexibility in real-world continual learning scenarios.

On the other hand, some existing continual learning methods are specifically designed for vision-language models (Liu et al. 2023d; Zheng et al. 2023b; Zheng et al. 2025a; Yu et al. 2024). Continual-CLIP (**thengane2022clip**) experimentally demonstrated that CLIP can achieve impressive continual learning performance without any finetuning. ZSCL (Zheng et al. 2023b) helps maintain zero-shot capabilities by introducing reference data and utilizing knowledge distillation. Adapt without Forgetting (Zheng et al. 2025a) employs first-order and second-order proximities, along with dual distillation, to prevent catastrophic forgetting. MoE-adapters (Yu et al. 2024) integrate MoE into the CLIP model and leverage DDAS to significantly enhance learning efficiency. However, MoE-adapters do not consider the dual-modal nature of multimodal models. In contrast, our approach fully leverages the relationship between images and text to guide MoE.

CHAPTER 3

Model-Adaptive Distillation Strategy for Vision-Language Models

---

3.1 Related Work

3.1.1 Knowledge Distillation.

Knowledge distillation was proposed for model compression initially (Sau and Balasubramanian 2016; Lopes et al. 2017; Yim et al. 2017), which is also an efficient method for preventing catastrophic forgetting in Continual Learning (Ding et al. 2022). Hinton *et al*. (Hinton et al. 2015) presents a teacher-student framework to allow knowledge transformed from pretrained large model to student model. Different from some works only concentrating on transforming instance features, such as final predicted probabilities (Ding et al. 2022), and features of intermediate layers (Zagoruyko and Komodakis 2016; Romero et al. 2014). Several existing works pay attention to the images relationships. (Liu et al. 2019) distilled an instance relationship graph from some intermediate layers. To transform local relationship of the features, (Chen et al. 2020a) distilled the relationship between one feature and its k-nearest neighbors. Unlike such distilling relationships approaches, our methods are able to maintain first-order proximity and second-order proximity to capture intra- and inter-modal interactions.

3.2 Motivation

Recently, multi-modal pretrained models, particularly exemplified by CLIP (Radford et al. 2021), have emerged as a cornerstone in generalization capabilities (Yao et al. 2021). These

models, benefiting from extensive pre-training on large-scale text-image datasets, exhibit two distinct advantages (Yu et al. 2022; Schuhmann et al. 2022; Nie et al. 2023). Firstly, their zero-shot transfer ability enables them to maintain considerable accuracy on previously unseen data, a crucial feature for models expected to adapt to ever-changing real-world scenarios (Li et al. 2022b; Yao et al. 2022). Secondly, they demonstrate remarkable efficiency in adapting to new downstream tasks with limited training data (Zang et al. 2022; Xing et al. 2022). The potential of these multi-modal pretrained models in continual learning (CL) scenarios, where adaptability and long-term learning are essential, appears immense. However, recent research shows that fine-tuning on downstream tasks could damage to the generalization ability of the pretrained models. This decline not only impairs the zero-shot capabilities but also leads to a gradual loss in performance gains on continual learning progress. Moveover, such phenomenon aligns closely with the notorious issue of "catastrophic forgetting" in continual learning (Aljundi et al. 2018; Shmelkov et al. 2017; Rannen et al. 2017), posing a significant hurdle in leveraging the full potential of multi-modal models in dynamic and evolving real-world applications.

While replay-based methods and the teacher-student learning paradigm have made significant progress in addressing the decline in generalization capabilities of pretrained models within a continual learning framework (Radford et al. 2021; Ding et al. 2022), they face limitations in balancing knowledge retention and adaptation to new data. Replay-based methods depend on pre-training phase data and labels, struggling to integrate this knowledge with new domain data (Rebuffi et al. 2017; Lopez-Paz and Ranzato 2017). Conversely, the teacher-student paradigm, though effective in preserving original capabilities, often fails to exploit the full potential of rich multi-modal information, particularly in adapting to diverse and evolving datasets. Approaches like ZSCL (Zheng et al. 2023a) within the teacher-student framework introduce reference data to mitigate forgetting, providing valuable insights but not fully addressing the dynamic complexity of multi-modal information.

To address the limitations of current continual learning strategies for multi-modal models, our research focuses on mitigating the conflict between preserving zero-shot transferability

and adapting to new datasets. This tension is particularly critical, as maintaining zero-shot capabilities often restricts a model's flexibility in learning from novel data.

We propose a novel dual-teacher distillation framework that shifts the focus from directly matching output predictions to preserving the relative proximity between samples. Modeling these relative relationships better supports the retention of the model's core capabilities while enabling adaptation. To capture such proximity effectively, we design a graph-based mechanism that encodes both intra- and inter-modal interactions. Using this structure, we perform a proximity-based distillation process to transfer relational knowledge from the teacher models.

Furthermore, we introduce a sample-level re-weighting strategy that dynamically adjusts the influence of the two teacher models based on individual sample characteristics. This mechanism allows for an adaptive trade-off between model stability and plasticity.

In summary, our approach redefines continual learning for multi-modal models by introducing a graph-based multi-modal proximity distillation method, coupled with an adaptive sample re-weighting strategy. A key innovation lies in leveraging graph structures to explicitly model relative relationships across and within modalities for CLIP-like architectures. This methodology not only resolves the trade-off between zero-shot retention and adaptability but also significantly boosts the robustness and flexibility of multi-modal models in continual learning settings. Experimental results demonstrate a considerable improvement over existing methodologies, which illustrate the effectiveness of the proposed method in the field of continual learning.

3.3 Preliminary

3.3.1 CLIP on Continual Learning.

CLIP (Radford et al. 2021) is renowned as a multimodal model, distinguished by its exceptional zero-shot transfer ability for various downstream tasks. Unlike traditional computer

vision systems, such as ResNet, which categorize images using predefined labels, CLIP integrates a text encoder. This encoder effectively translates labels described in natural language into text features, aligning them with corresponding visual representations. The CLIP model undergoes pre-training on a vast corpus of over 400 million image-text pairs, leveraging a strategy that synergizes these two modalities for improved performance and versatility. Specifically, the training objective is defined as:

$$L_{\mathrm{C}} = -\sum_{i=1}^{N} \log \frac{\exp(\mathrm{sim}(E_{\mathrm{img}}(\mathbf{x}_i), E_{\mathrm{txt}}(\boldsymbol{y})/\tau)}{\sum_{j=1}^{N} \exp(\mathrm{sim}(E_{\mathrm{img}}(\mathbf{x}_i), E_{\mathrm{txt}}(\boldsymbol{t}_j))/\tau)}, \tag{3.1}$$

where $\mathrm{sim}(u, v) = u^T v/\|u\| \cdot \|v\|$ is the cosine similarity function, τ is a temperature parameter, $\boldsymbol{y}$ is the text that correctly pairs with image $\mathbf{x}$, and $E_{\mathrm{img}}(\mathbf{x}_i)$ and $E_{\mathrm{txt}}(\boldsymbol{t}_j)$ are the embeddings from the image and text encoders for the i-th and j-th elements in a batch, respectively. This approach enables the model to effectively bridge the semantic gap between visual and textual data, fostering a robust multimodal understanding.

In the context of continual learning, the CLIP model is sequentially fine-tuned across a series of tasks, each associated with distinct datasets denoted as $\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_N$. A crucial objective in this process is for CLIP to preserve its inherent zero-shot capabilities and simultaneously mitigate catastrophic forgetting. In this paper, the fine-tuned model for each specific task $\mathcal{T}_i$ is represented by C_i, which is iteratively built upon the foundation of the preceding model C_{i-1}. For reference, the original pre-trained CLIP model is designated as C_0.

3.3.2 Knowledge Distillation in Data-Unknown Scenarios.

The concept of conducting knowledge distillation in scenarios where the training data of teacher model is unknown, as highlighted in works like wild distillation (Chen et al. 2021a) and the ZSCL (Zheng et al. 2023a) framework, is of notable relevance to the field. This approach gains significance in the context of models like CLIP, characterized by extensive yet undefined pre-training datasets. Here, the student network $\mathcal{N}_S$ is tailored to leverage 'wild' data, often unaligned with the teacher's training regime. The distillation process adapts to this

complexity as:

$$L_{\mathrm{KD}} = \sum_{i=0} [L_{\mathrm{CE}} \left(\mathcal{N}_T(\mathbf{x}_i, \boldsymbol{t}), \mathcal{N}_S(\mathbf{x}_i, \boldsymbol{t})\right)], \tag{3.2}$$

where $\mathbf{x} \sim p_{\mathrm{wild}}(\mathbf{x})$ and $\boldsymbol{t} \sim p_{\mathrm{wild}}(\boldsymbol{t})$ denote the diverse, unstructured data samples and L_{CE} represents the cross-entropy loss function. This paradigm, resonating with ZSCL's emphasis on leveraging varied data for continual learning, showcases the necessity of refining student models to function effectively in data-ambiguous environments, underscoring a nuanced approach to knowledge distillation.

3.4 Methodology

In this section, we outline our proximity distillation strategy to learn from two teacher models. We construct a graph utilizing multi-modal information and define both first-order and second-order proximity. Then we employ a teacher-student paradigm, focusing on distilling the proximity defined on this graph from both the original CLIP model and a model fine-tuned on previous datasets. To balance the above two teachers and further alleviate the conflict between stability and plasticity, a sample re-weighting mechanism is incorporated to our method.We conclude the framework of the proposed method in Figure 3.1.

3.4.1 Graph-Based Multi-modal Representation

As mentioned above, preserving relationships between samples rather than their individual features, offers a more flexible constraint, allowing the model ample freedom to adapt to varying data distributions while maintaining its inherent capabilities. In this part, we develop a cross-modal graph where each node represents a sample, and the edges reflect the model's depiction of relationships between these samples across different modalities.

We define this graph as $G = (V_I, V_T, E)$, where V_I and V_T respectively represent sets of from image and text modalities. Each vertex is categorized either as an image vertex $v_I \in V_I$ or a text vertex $v_T \in V_T$, with the edges $e \in E$ being undirected to illustrating the mutual

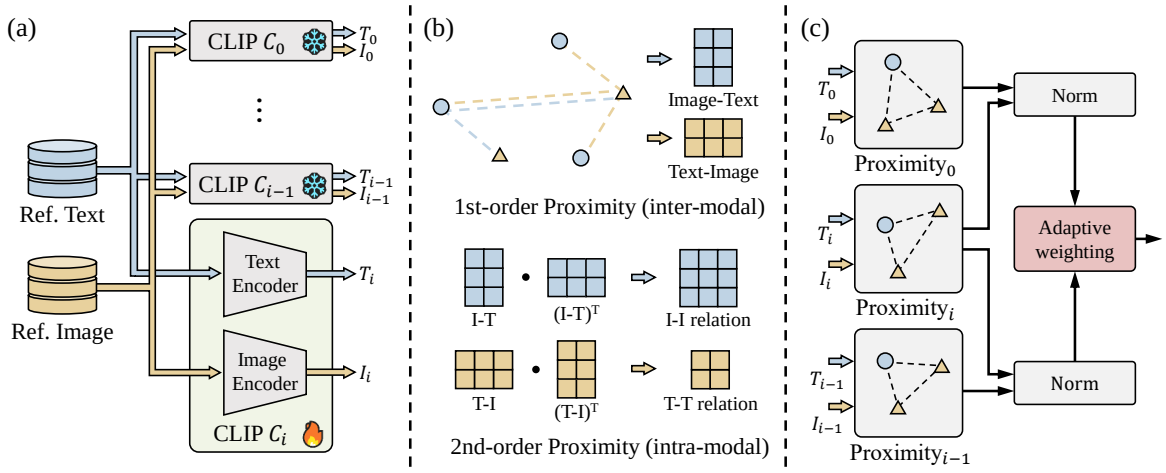


FIGURE 3.1. Overview of the proposed continual learning framework for vision-language models. (a) The dual-teacher distillation with CLIP models C_0 and C_{i-1} using reference data. (b) Cross-modal graph construction for first-order and second-order proximity modeling. (c) The adaptive re-weighting mechanism for balanced knowledge distillation from different teacher.

interaction that exist between the visual and textual components. This graph is constructed to encapsulate the relationships inherent in multi-modal model. It serves as a foundation for our advanced manifold distillation technique, which we elaborate on in the following sections.

3.4.1.1 First-Order Proximity.

A fundamental component of our methodology is the integration of first-order proximity within the framework of the CLIP model to optimize its performance in continual learning contexts. We utilize a graph $G = (V_T, V_I, E)$ to capture the direct pairwise relationships between the visual and textual modalities. The concept of first-order proximity is essential in quantifying the immediate similarity between two vertices, thereby facilitating a understanding of these modality interactions.

In graph embedding learning (Tang et al. 2015), first-order proximity is typically defined by the probability of two vertices sharing similarities. It can be mathematically represented as:

$$\hat{p}_1(v_i, v_j) = \text{sim}(\boldsymbol{u}_i, \boldsymbol{u}_j), \quad \forall (i,j) \in E, \tag{3.3}$$

where $\boldsymbol{u}_i$ and $\boldsymbol{u}_j$ are the embeddings of the respective vertices, and $\text{sim}(u, v) = \frac{u^T v}{\|u\| \cdot \|v\|}$ is the cosine similarity function as illustrated above. The first-order proximity is defined as zero for any pair of vertices that are not connected by an edge.

It is important to note that within the CLIP model, direct connections only occur between vertices from two distinct modalities. Consequently, the first-order proximity effectively aligns with the classification mechanism of the CLIP model. This is achieved by extracting embeddings for each modality using their respective encoders, such as $\boldsymbol{u}_I = \text{Enc}_I(v_I)$ for image vertices and $\boldsymbol{u}_T = \text{Enc}_T(v_T)$ for textual vertices. In this context, a higher value of $\hat{p}_1(v_i, v_j)$ denotes a stronger correlation between the vertices, indicating a higher likelihood of the image sample v_i being associated with the label represented by text vertex v_j.

Through the lens of first-order proximity, we define the relationships between image and text modalities within our graph-based framework. This approach not only characterizes the classification mechanism of the model but also serves as a flexible regularization. This form of proximity is particularly informative in preserving the capability of a model, making it a valuable asset in our continual learning strategy.

3.4.1.2 Second-Order Proximity.

Building on the first-order proximity that focuses on immediate and inter-modal proximity in our multi-modal framework, we integrate second-order proximity (Tang et al. 2015) to model a broader and global perspective. This broader view enables us to delve into intra-modal proximity within the CLIP model, going beyond direct image-text connections.

Second-order proximity assesses vertex similarity by examining neighborhood structures. We initially define the proximity between two directly connected vertices v_i and v_j through first-order proximity in eq. (3.3). From this foundation, the conditional distribution for a vertex v_i with a neighboring vertex v_n is derived as:

$$p_1(v_n|v_i) = \frac{\text{sim}(\boldsymbol{u}_i, \boldsymbol{u}_n)}{\sum_{(i,k)\in E} \text{sim}(\boldsymbol{u}_i, \boldsymbol{u}_k)}, \quad \forall (i, n) \in E. \tag{3.4}$$

Then we can model the second-order proximity between two vertices without direct connections by measuring the similarity in their conditional distributions across respective neighborhoods. Formally, we define the second-order proximity between two vertices v_i and v_j as:

$$\hat{p}_2(v_j, v_i) = \mathrm{sim}(p_1(\cdot|v_i), p_1(\cdot|v_j)), \quad \forall (i,j) \notin E. \tag{3.5}$$

eq. (3.5) highlights that second-order proximity identifies similarities between unconnected vertices based on neighborhood likeness. In the CLIP model, connections are established between every image-text vertex pair. Consequently, unconnected vertices typically belong to the same modality, making second-order proximity a measure of intra-modal proximity.

Our approach leverages the multi-modal proximity established by the CLIP model to construct intra-modal connections. Specifically, in eq. (3.5), we define the proximity between vertices of the same modality, such as images, utilizing the contextual associations provided by the textual modality. This strategy differs from the standard pairwise similarity often calculated as $\boldsymbol{u}_i \cdot \boldsymbol{u}_j^T$, which typically does not incorporate multi-modal information. Through this, we effectively utilize the unique strengths of the CLIP model in capturing cross-modal interactions to enhance the modeling of same-modality proximity.

3.4.1.3 Comparison with existing knowledge distillation methods.

To elucidate the distinctions between our method and existing knowledge distillation approaches, we primarily focus on the calculation of cosine similarity between image nodes, particularly emphasizing the relationships defined within our framework. Our method diverges from the standard normalization process to streamline the explanation and clarity of the proximity representation. Consequently, we can reformulate eq. (3.5) as follows,

$$p_2'(v_j, v_i) = \boldsymbol{u}_i^T \cdot \boldsymbol{U}_T \boldsymbol{U}_T^T \cdot \boldsymbol{u}_j, \quad \forall (i,j) \notin E, \tag{3.6}$$

where $\boldsymbol{U}_T = [\boldsymbol{u}_{T_0}, \boldsymbol{u}_{T_1}, ..., \boldsymbol{u}_{T_N}]$ donates the embeddings for all text vertices v_T extracted by the corresponding encoder. eq. (3.6) demonstrates that our methodology for second-order proximity extends beyond conventional pairwise comparisons, which typically achieved through cosine similarity and denoted as $\boldsymbol{u}_i \cdot \boldsymbol{u}_j^T$. By integrating weighted similarities,

where the weights are informed by shared connections with text vertices, we provide a more comprehensive analysis of relationships. This method distinctly differs from and surpasses the conventional $\boldsymbol{u}_i \cdot \boldsymbol{u}_j^T$ approach by incorporating multi-modal information.

In summary, The foundational distinction of our method from existing knowledge distillation techniques lies in its dual-focus on inter-modal and intra-modal relations. Firstly, our approach introduces the concept of first-order proximity to encapsulate inter-modal relations. Secondly, it explores second-order proximity by examining the similarity of neighborhood structures, rather than direct feature similarity, thereby providing a more granular understanding of intra-modal relations. This exploration of neighborhood structures represents a significant departure from existing methods, which primarily focus on direct feature comparisons. In the distillation phase, we utilize a conditional distribution model for second-order proximity, as shown below,

$$p_2(v_j|v_i) = \frac{\exp(p_2(v_i, v_j))}{\sum_{(i,k)\notin E} \exp(p_2(v_i, v_k))}, \quad \forall (i,j) \notin E. \tag{3.7}$$

This refined approach, grounded in a comprehensive examination of both inter-modal and intra-modal relationships, underscores our method's innovation and its departure from traditional knowledge distillation frameworks.

3.4.2 Dual Distillation with Samplewise Balance

In the domain of continual learning, maintaining the zero-shot transfer capabilities of the CLIP model while adapting to novel data distributions is of paramount importance. Zero-shot transfer encapsulates the model's ability to generalize to untrained data, whereas adaptation evaluates its performance post fine-tuning on novel datasets. This dual requirement often leads to a dichotomy, where fine-tuning to optimize for a specific dataset might compromise the model's generalization ability.

To address this, we introduce a dual-teacher distillation framework that capitalizes on the distinct advantages of the pre-trained CLIP model C_0 and its preceding iteration C_{i-1}. This approach is anchored in leveraging unpaired image and text data for distillation in the absence

of original pre-training data, inspired by wild distillation (Chen et al. 2021a) and ZSCL (Zheng et al. 2023a) methodologies. This approach enables effective knowledge transfer even without access to the original multi-modal training dataset.

In our approach, a key objective is for the student model to preserve both first and second order proximities as exhibited by the teacher model, using a set of reference data. This can be achieved through the following objective function:

$$\overline{\ell}_{\mathrm{pd}}(v_k) = L_{\mathrm{CE}}\left(\overline{p}_1(\cdot|v_k), p_1(\cdot|v_k)\right) + L_{\mathrm{CE}}\left(\overline{p}_2(\cdot|v_k), p_2(\cdot|v_k)\right), \tag{3.8}$$

where $\overline{p}_1(\cdot|v_k)$ and $\overline{p}_2(\cdot|v_k)$ represent the first-order and second-order proximity of the original CLIP model, respectively. In parallel, we utilize $\tilde{p}_1$ and $\tilde{p}_2$ to denote the first-order and second-order proximity within the model C_{i-1}. It is straightforward to distill proximity from two teachers as following objective function,

$$L_{\mathrm{PD}} = \frac{1}{N}\sum_{k=1}^{N}[\overline{\ell}_{\mathrm{pd}}(v_k) + \lambda \cdot \tilde{\ell}_{\mathrm{pd}}(v_k)]. \tag{3.9}$$

Eq.(3.9) tries to balance the knowledge from two teachers applying a uniform weighting across different samples, which does not account for the individual learning needs or the diversity of the dataset, potentially limiting the efficacy of the distillation process. To address this, we introduces a dynamic adjustment to the samples, towards a effective knowledge transfer tailored to the specific needs of each sample. Specifically, for a sample v_k, the distance between teacher models C_0 and student C_i is assessed as follows,

$$\overline{d}_p(v_k) = d\left(\overline{p}_1(v_k), p_1(v_k)\right), \tag{3.10}$$

where $d(\cdot,\cdot)$ denotes the distance metric. The distance $\tilde{d}_p(v_k)$ between model C_{i-1} and C_i is similarly defined. Then the samplewise weights for the k-th sample are normalized accordingly,

$$\overline{w}_k = \frac{\exp(\overline{d}_p(v_k))}{\exp(\overline{d}_p(v_k)) + \exp(\tilde{d}_p(v_k))}. \tag{3.11}$$

The weight $\tilde{w}_k$ for C_{i-1} model is also similarly defined. This strategy tailors the learning focus for each sample in the reference dataset, determining whether to align with C_0 for

zero-shot capabilities or C_{i-1} for the adaptability. Then the proximity distillation loss can be reformulated as

$$L_{\mathrm{PD}} = \frac{1}{N}\sum_{k=1}^{N}[\overline{w}_k \cdot \overline{\ell}_{\mathrm{pd}}(v_k) + (1-\overline{w}_k)\cdot \tilde{\ell}_{\mathrm{pd}}(v_k)]. \tag{3.12}$$

This adaptive sample-wise methodology not only preserves zero-shot abilities acquired during pre-training but also allows for effective adaptation to new data distribution, crucial for performance in dynamic learning environments. The final objective function encapsulates our approach:

$$L = L_{\mathrm{PD}} + L_{\mathrm{CE}}. \tag{3.13}$$

The second term L_{CE} serves as the original objective function for fine-tuning the CLIP model with a new dataset. Paired with our introduced first term L_{PD}, this combined methodology aims to better maintain the zero-shot learning capabilities of CLIP while it undergoes fine-tuning on novel datasets. We have chosen to apply equal weighting to both loss components because experimental results demonstrates low sensitivity to variations in weighting.

3.4.3 Algorithm of Method.

To help understand the training process of our method, we now provide the whole training algorithm of our method in Algorithm 1.

3.5 Experiment

3.5.1 Experimental setup

3.5.1.1 Continual learning setting.

We mainly evaluate the proposed method on two continental learning settings. They are multi-domain task incremental learning (MTIL) and class incremental learning (CIL).

MTIL (Zheng et al. 2023b) is a benchmark of cross-domain version of task incremental learning, where different tasks are sourced from different datasets. This benchmark demands

Algorithm 1 Implementing Graph Distillation in CLIP

Input: A series of task datasets $\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_N$, wild images and text datasets $p_{\mathrm{w}}(\mathbf{x})$ and $p_{\mathrm{w}}(\boldsymbol{t})$, pretrained CLIP model C_0.

Input: Initializing the model fine-tuned on previous dataset C_{i-1} by C_0, the number of iterations $iter$ trained on each tasks, the batch size m for task dataset, and n for wild data, temperature τ.

for $i = 1, ..., N$ **do**

while iterations $< iter$ **do**

Randomly sample $\{\mathbf{x}^{(k)}, \boldsymbol{y}^{(k)}\}_{k=1}^{m}$ from $\mathcal{T}_i$

Randomly sample $\{\mathbf{x}_w^{(j)}\}_{j=1}^{m} \sim p_{\mathrm{w}}(\mathbf{x})$, $\{\boldsymbol{t}_w^{(j)}\}_{j=1}^{n} \sim p_{\mathrm{w}}(\boldsymbol{t})$, donate as $\{v^k\}_{k=1}^{m+n}$

Calculate $L_{\mathrm{CE}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} CE(\mathbf{x}^{(i)}, \boldsymbol{y}^{(i)}, \tau)$

Calculate 1st-order proximity $p_1(\cdot|v^k)$, $\tilde{p}_1(\cdot|v^k)$, $\overline{p}_1(\cdot|v^k)$.

Calculate 2nd-order proximity $p_2(\cdot|v^k)$, $\tilde{p}_2(\cdot|v^k)$, $\overline{p}_2(\cdot|v^k)$.

Calculate sample-wise distillate weights $\overline{w}_k$ and $\tilde{w}_k$ of each vertex for C_0 and C_{i-1}.

Calculate $L \leftarrow L_{\mathrm{CE}} + \frac{1}{m+n}\sum_{k=1}^{m+n}[\overline{w}_k \cdot \overline{\ell}_{\mathrm{pd}}(v_k) + \tilde{w}_k \cdot \tilde{\ell}_{\mathrm{pd}}(v_k)$

Update the parameters θ of C_i with optimizer.

end while

Update the previous model $C_{i-1} = C_i$.

end for

Output: A continual learning CLIP model C_N.

models to learn knowledge from different domains to achieve promising performance. Specifically, MTIL benchmark consists of 11 datasets, which are Aircraft, Caltech101, CIFAR100, DTD, EuroSAT, Flowers, Food, MNIST, OxfordPet, StanfordCars, and SUN397. Each dataset serves as an individual task for continental learning. We follow the setting in (Zheng et al. 2023b) and utilize two orders of tasks for evaluation. The first one (Order I) follows the alphabetical order, while the second one (Order II) follows a random order.

CIL is a essential setting for continual learning, requiring the model to be updated incrementally with only new class instances (Masana et al. 2022). We evaluate our method on CIFAR100 (Krizhevsky, Hinton et al. 2009) and TinyImageNet (Le and Yang 2015) datasets by following the same evaluation setting in (Douillard et al. 2022). Specifically, with a t-step setting, classes in CIFAR100 are split into t groups evenly and each group serves as a task. We set $t = \{10, 20, 50\}$. Besides, 100 classes are learned in TinyImageNet at a base step and the remaining 100 classes are split into t groups and we set $t = \{5, 10, 20\}$ for TinyImageNet.

3.5.1.2 Metric.

Three metrics are adopted in our experiments, which are "Transfer", "Avg." and "Last". If there are N continual learning tasks in total, and the model achieves an accuracy of $\mathcal{A}(i|j)$ on task i after training on task j, then we can formulate these metrics as:

$$\begin{aligned} \text{Transfer} &= \frac{1}{N-1}\sum_{j=2}^{N}\frac{1}{j-1}\sum_{i=1}^{j}\mathcal{A}(i|j), \\ \text{Avg.} = \frac{1}{N^2}\sum_{j=1}^{N}\sum_{i=1}^{N}\mathcal{A}(i|j), &\quad \text{Last} = \frac{1}{N}\sum_{i=1}^{N}\mathcal{A}(i|N). \end{aligned} \tag{3.14}$$

The Avg. and the Last metrics are adopted across all our experiments. On the MTIL benchmark, we also adopt the Transfer metric to evaluate zero-shot transfer ability of a trained model.

3.5.1.3 Model.

We employ a pretrained CLIP (Radford et al. 2021) model with a ViT-B/16 image encoder in our experiments. Input resolution and patch size are 224 and 16.

3.5.1.4 Baseline.

Since the CLIP model can adapt to new data distributions without re-training, we adopt zero-shot transfer as one of the baseline (Thengane et al. 2022). Direct fine-tuning the CLIP model on each task is also invited, which is condidered as an upper bound. Besides, we compare the proposed method with not only conventional continual learning methods, but also recent proposed method WiSE-FT (Wortsman et al. 2022), LwF-VR (Ding et al. 2022) and ZSCL (Zheng et al. 2023b), which are specifically designed for CLIP.

3.5.1.5 Dataset.

The detailed characteristics of the 11 datasets applied in the MTIL benchmark are shown in Table 3.1.

TABLE 3.1. Detailed description of the 11 datasets in the multi-domain task incremental learning benchmark.

Dataset	# classes	# train	# test	Description
Aircraft (Maji et al. 2013)	100	3334	3333	aircraft model
Caltech101 (Fei-Fei et al. 2004)	101	6941	1736	real-life object
CIFAR100 (Krizhevsky, Hinton et al. 2009)	100	50000	10000	real-life object
DTD (Cimpoi et al. 2014)	47	1880	1880	texture database
EuroSAT (Helber et al. 2019)	10	21600	5300	satellite images
Flowers (Nilsback and Zisserman 2008)	102	1020	6149	images of flowers
Food (Bossard et al. 2014)	101	75750	25250	images of food
MNIST (Deng 2012)	10	60000	10000	handwritten digits
OxfordPet (Parkhi et al. 2012)	37	3680	3669	pet image dataset
Cars (Krause et al. 2013)	196	7144	8041	car images
SUN397 (Xiao et al. 2010)	397	87003	21751	scene categories
Total	1201	319352	97109	

3.5.2 Results on MTIL

We compare the proposed method with baselines on the MTIL benchmark. The CLIP model is trained on each task for 1000 iterations with a batch size of 64. The learning rate is configured at 3e-5 for the initial task and 1e-5 for all subsequent tasks. Adam is adopted as the optimizer.

Table 3.2 and Table 3.3 showcases the results under both task order settings I and II. As the results show, the proposed method outperforms all other baselines under every metric. Notably, when task order settings I is adopted, the Transfer performance of our proposed method is 69.8%, which is 0.4% higher than even the original CLIP model. Meanwhile all existing methods demonstrate declined Transfer performance. This result shows the superiority of our method in preserving the zero-shot performance of the CLIP model while achieving preferable fine-tuning performance during continual learning. The results obtained under task order settings II demonstrate a similar trend.

Table 4.3 provides more details about the performance of each method on each task under task order setting I. It presents performance of final trained models on each task in the MTIL benchmark. In terms of both the Transfer and the Avg. performance, the proposed method achieved consistent performance improvement over existing baselines on all tasks. Even under the Last metric, our method also achieves competitive performance, with only a marginal gap

TABLE 3.2. Comparison results on the MTIL benchmark under task order setting I. Δ represents the performance gap between a CLIP model after continual learning and the original CLIP model (Zero-shot).

Method	**Transfer**	Δ	**Avg.**	Δ	**Last**	Δ
Zero-shot	69.4	-	65.3	-	65.3	-
Continual FT	44.6	-24.8	55.9	-9.4	77.3	+12.0
LwF (Li and Hoiem 2017)	56.9	-12.5	64.7	-0.6	74.6	+9.0
iCaRL (Rebuffi et al. 2017)	50.4	-19.0	65.7	+0.4	80.1	+14.8
LwF-VR (Ding et al. 2022)	57.2	-12.2	65.1	-0.2	76.6	+11.3
WiSE-FT (Wortsman et al. 2022)	52.3	-17.1	60.7	-4.6	77.7	+12.4
ZSCL (Zheng et al. 2023b)	68.1	-1.3	75.4	+10.1	83.6	+18.3
Ours	**69.8**	**+0.4**	**76.9**	**+11.6**	**85.1**	**+19.8**

TABLE 3.3. Comparison results on the MTIL benchmark under task order setting II. Δ represents the performance gap between a CLIP model after continual learning and the original CLIP model (Zero-shot).

Method	**Transfer**	Δ	**Avg.**	Δ	**Last**	Δ
Zero-shot	65.4	-	65.3	-	65.3	-
Continual FT	46.6	-18.8	56.2	-9.1	67.4	+2.1
LwF (Li and Hoiem 2017)	53.2	-12.2	62.2	-5.2	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)	50.9	-14.5	56.9	-8.4	71.6	+6.3
WiSE-FT (Wortsman et al. 2022)	51.0	-14.4	61.5	-5.9	72.2	+6.9
ZSCL (Zheng et al. 2023b)	64.2	-1.2	74.5	+9.2	83.4	+18.1
Ours	**65.4**	**+0.0**	**75.9**	**+10.6**	**85.4**	**+20.1**

on two datasets. More results on the MTIL benchmark can be found in the supplementary material.

3.5.3 Details of Experimental Results.

Table 4.4 shows the accuracy results on all datasets after training on each task in the MTIL benchmark under task order setting I. Each row represents the accuracy results of our model on each dataset after completing training for the current task.

TABLE 3.4. Performance of trained model on each task in the MTIL benchmark under task order setting I.

Method	Aircraft	Caltech101	CIFAR100	DTD	EuroSAT	Flowers	Food	MNIST	OxfordPet	Cars	SUN397
Zero-shot	24.3	88.4	68.2	44.6	54.9	71.0	88.5	59.4	89.0	64.7	65.2
Fine-tuning	62.0	95.1	89.6	79.5	98.9	97.5	92.7	99.6	94.7	89.6	81.8
Transfer											
Continual FT		67.1	46.0	32.1	35.6	35.0	57.7	44.1	60.8	20.5	46.6
LwF		74.5	56.9	39.1	51.1	52.6	72.8	60.6	75.1	30.3	55.9
iCaRL		56.6	44.6	32.7	39.3	46.6	68.0	46.0	77.4	31.9	60.5
LwF-VR		77.1	61.0	40.5	45.3	54.4	74.6	47.9	76.7	36.3	58.6
WiSE-FT		73.5	55.6	35.6	41.5	47.0	68.3	53.9	69.3	26.8	51.9
ZSCL		86.0	67.4	45.4	50.4	69.1	87.6	61.8	86.8	60.1	66.8
Ours		**88.8**	**68.4**	**46.1**	**56.2**	**70.6**	**87.9**	**62.4**	**88.1**	**62.2**	**67.0**
Avg.											
Continual FT	25.5	81.5	59.1	53.2	64.7	51.8	63.2	64.3	69.7	31.8	49.7
LwF	36.3	86.9	72.0	59.0	73.7	60.0	73.6	74.8	80.0	37.3	58.1
iCaRL	35.5	89.2	72.2	60.6	68.8	70.0	78.2	62.3	81.8	41.2	62.5
LwF-VR	29.6	87.7	74.4	59.5	72.4	63.6	77.0	66.7	81.2	43.7	60.7
WiSE-FT	26.7	86.5	64.3	57.1	65.7	58.7	71.1	70.5	75.8	36.9	54.6
ZSCL	45.1	92.0	80.1	64.3	79.5	81.6	89.6	75.2	88.9	64.7	68.0
Ours	**46.6**	**93.4**	**82.0**	**67.4**	**82.6**	**83.7**	**90.0**	**75.7**	**89.9**	**66.8**	**68.4**
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
LwF	26.3	87.5	71.9	66.6	79.9	66.9	83.8	**99.6**	92.1	66.1	80.4
iCaRL	35.8	**93.0**	77.0	70.2	83.3	88.5	90.4	86.7	93.2	81.2	81.9
LwF-VR	20.5	89.8	72.3	67.6	85.5	73.8	85.7	**99.6**	93.1	73.3	80.9
WiSE-FT	27.2	90.8	68.0	68.9	86.9	74.0	87.6	**99.6**	92.6	77.8	81.3
ZSCL	40.6	92.2	81.3	70.5	94.8	90.5	91.9	98.7	**93.9**	85.3	80.2
Ours	**42.4**	92.7	**83.2**	**73.2**	**97.0**	**91.8**	**92.2**	99.1	**93.9**	**87.4**	**82.6**

Meanwhile, we provide additional experiments details about order setting II. Firstly, the sequence of order setting II is StanfordCars, Food, MNIST, OxfordPet, Flowers, SUN397, Aircraft, Caltech101, DTD, EuroSAT, and CIFAR100. Moverover, the complete experiment result of our method are shown in Table 4.5. Lastly, Table 4.6 displays the experimental results, showcasing a comparative results of our method against others using the Transfer,

Avg., and Last metrics under task order setting II. Despite varying order settings, our method consistently demonstrates significant improvements across nearly all metrics.

TABLE 3.5. Accuracy(%) on every dataset after training on each task in the MTIL benchmark under task order setting I.

	Aircraft	Caltech101	CIFAR100	DTD	EuroSAT	Flowers	Food	MNIST	OxfordPet	Cars	SUN397
Aircraft	59.3	88.8	67.7	46.0	54.6	70.6	88.5	56.4	88.6	62.8	65.8
Caltech101	50.0	94.8	69.1	45.8	59.0	68.9	87.7	58.7	86.2	62.3	66.0
CIFAR100	48.3	94.1	87.5	46.6	55.2	70.3	88.0	63.0	87.7	61.8	67.0
DTD	48.2	94.5	86.6	77.6	55.8	71.5	87.8	64.7	89.1	62.1	67.2
EuroSAT	47.8	94.4	85.6	76.9	98.4	71.5	87.9	67.0	89.1	62.3	67.5
Flowers	45.5	94.1	85.3	76.8	98.3	97.2	87.3	63.1	88.0	61.6	67.5
Food	44.8	93.7	84.8	75.1	97.9	95.5	92.6	63.6	88.2	62.6	67.8
MNIST	43.0	93.4	84.0	74.6	97.7	94.8	92.5	99.2	88.1	62.1	66.7
OxfordPet	42.9	93.7	83.9	74.6	97.5	94.6	92.5	99.2	95.3	62.0	67.0
Cars	40.0	93.3	84.0	73.8	97.3	94.1	92.3	99.1	94.5	88.1	67.0
SUN397	42.4	92.7	83.2	73.2	97.0	91.9	92.2	99.1	94.0	87.4	82.6
Transfer		88.8	68.4	46.1	56.2	70.6	87.9	62.4	88.1	62.2	67.0
Avg.	46.6	93.4	82.0	67.4	82.6	83.7	90.0	75.7	89.9	66.8	68.4
Last	42.4	92.7	83.2	73.2	97.0	91.8	92.2	99.1	93.9	87.4	82.6

3.5.4 Ablation study

3.5.4.1 Impact of proposed proximities.

We conduct experiments on the MTIL benchmark to study the impact of these the 1st-order and the 2nd-order proximities on continental learning performance. We perform 5 experiments and report the mean and standard deviation of the results in Table 3.8. The results indicate that the model exhibits the poorest performance across all three metrics when neither proximity is preserved. Introducing either the 1st-order or the 2nd-order proximity independently enhances the model's performance. Furthermore, the best result is achieved when both the two proximities are adopted, with the maximum performance improvements being 25.2% for Transfer, 21.0% for Avg., and 7.8% for Last metrics.

TABLE 3.6. Accuracy(%) on every dataset after training on each task in the MTIL benchmark under task order setting II.

	Cars	Food	MNIST	OxfordPet	Flowers	SUN397	Aircraft	Caltech101	DTD	EuroSAT	CIFAR100
Cars	88.7	88.6	59.3	87.7	71.2	65.6	22.6	89.0	45.7	54.4	68.6
Food	88.4	92.9	64.1	87.6	70.6	67.2	22.0	89.3	45.0	55.5	68.8
MNIST	88.0	92.8	99.2	87.1	70.3	66.8	22.2	89.3	45.0	55.6	69.0
OxfordPet	87.5	92.6	99.1	95.2	70.7	66.6	22.7	89.2	45.1	54.1	68.3
Flowers	86.9	92.4	99.0	94.5	97.0	66.4	22.0	89.6	45.3	51.9	68.5
SUN397	86.1	92.2	99.0	93.9	95.1	82.8	22.3	89.8	47.1	53.6	69.0
Aircraft	84.3	91.9	98.9	93.2	94.1	81.9	60.3	88.7	45.8	49.8	67.8
Caltech101	84.4	91.9	98.8	93.0	92.7	81.4	53.4	94.0	45.4	52.9	68.2
DTD	84.5	91.8	98.8	92.6	93.2	81.3	52.9	93.8	77.9	55.1	69.0
EuroSAT	84.7	91.7	98.8	92.7	92.8	81.2	51.8	93.8	77.2	98.5	70.2
CIFAR100	83.9	91.5	97.6	92.7	89.2	80.7	50.6	92.9	75.3	97.0	87.8
Transfer		88.6	61.7	87.5	70.7	66.5	22.3	89.3	45.5	53.6	68.7
Avg.	86.1	91.8	92.0	91.8	85.2	74.7	36.6	90.9	54.1	61.7	70.5
Last	83.9	91.5	97.6	92.7	89.2	80.7	50.6	92.9	75.3	97.0	87.8

3.5.4.2 Adaptive weighting.

To assess the effectiveness of the proposed sample re-weighting mechanism, we conduct experiments to compare it with several other weighting schemes on the MTIL benchmark, which are 1-0 weighting (Only C_0), 0-1 weighting (Only C_{i-1}), and average weighting. Table 3.9 illustrates the results. With the proposed method using C_{i-1} as the teacher only incurs a slight decline in the Transfer performance, but achieving remarkable performance improvement under the other two metrics, compared to using C_0 as the teacher. This demonstrates the effectiveness of the proposed method in preserving the original structure of out-domain feature space. Combining both teacher C_0 and C_{i-1} by average weighting further leads to improved downstream performance, while it also suffers from lower zero-shot performance. In contrast, adopting the proposed adaptive weighting scheme achieves competitive Avg. and Last performance and significantly improves the Transfer performance. The experiment results demonstrate that our adaptive weighting scheme effectively balances the stability and plasticity of the CLIP model during continual learning.

TABLE 3.7. Performance of trained model on each task in the MTIL benchmark under task order setting II.

Method	Cars	Food	MNIST	OxfordPet	Flowers	SUN397	Aircraft	Caltech101	DTD	EuroSAT	CIFAR100
Zero-shot	64.7	88.5	59.4	89.0	71.0	65.2	24.3	88.4	44.6	54.9	68.2
Fine-tuning	89.6	92.7	99.6	94.7	97.5	81.8	62.0	95.1	79.5	98.9	89.6
Transfer											
Continual FT		85.9	59.6	57.9	40.0	46.7	11.1	70.0	30.5	26.6	37.7
LwF Li and Hoiem 2017		87.8	58.5	71.9	46.6	57.3	12.8	81.4	34.5	33.7	46.8
iCaRL Rebuffi et al. 2017		86.1	51.8	67.6	50.4	57.9	11.0	72.3	31.2	32.7	48.1
LwF-VR Ding et al. 2022		88.2	57.0	71.4	50.0	58.0	13.0	82.0	34.4	29.3	47.6
WiSE-FT Wortsman et al. 2022		87.2	57.6	67.0	45.0	54.0	12.9	78.6	35.5	28.4	44.3
ZSCL Zheng et al. 2023b		88.3	57.5	84.7	68.1	64.8	21.1	88.2	45.3	**55.2**	68.2
Ours		**88.6**	**61.7**	**87.5**	**70.7**	**66.5**	**22.3**	**89.3**	**45.5**	53.6	**68.7**
Avg.											
Continual FT	42.1	70.5	92.2	80.1	54.5	59.1	19.8	78.3	41.0	38.1	42.3
LwF Li and Hoiem 2017	49.0	77.0	92.1	85.9	66.5	67.2	20.9	84.7	44.6	45.5	50.5
iCaRL Rebuffi et al. 2017	52.0	75.9	77.4	74.6	58.4	59.3	11.7	79.6	42.1	43.2	51.7
LwF-VR Ding et al. 2022	44.9	75.8	91.8	85.3	63.5	67.6	16.9	84.9	44.0	40.6	51.3
WiSE-FT Wortsman et al. 2022	52.6	79.3	91.9	83.9	63.4	65.2	23.3	83.7	45.4	40.0	48.2
ZSCL Zheng et al. 2023b	81.7	91.3	91.1	91.0	82.9	72.5	33.6	89.7	53.3	**62.8**	69.9
Ours	**86.1**	**91.8**	**92.0**	**91.8**	**85.2**	**74.7**	**36.6**	**90.9**	**54.1**	61.7	**70.5**
Last											
Continual FT	24.0	67.3	99.1	87.4	44.3	67.0	29.5	92.3	61.3	81.0	**88.1**
LwF Li and Hoiem 2017	34.6	69.6	99.3	88.7	61.1	72.5	32.5	88.1	65.6	90.9	87.9
iCaRL Rebuffi et al. 2017	46.0	81.5	91.3	82.8	66.5	72.2	16.3	91.6	68.1	83.2	87.8
LwF-VR Ding et al. 2022	27.4	61.2	99.4	86.3	60.6	70.7	23.4	88.0	61.3	84.3	**88.1**
WiSE-FT Wortsman et al. 2022	35.6	76.9	99.5	89.1	62.1	71.8	27.8	90.8	67.0	85.6	87.6
ZSCL Zheng et al. 2023b	78.2	91.1	**97.6**	92.5	87.4	78.2	45.0	92.3	72.7	96.2	86.3
Ours	**83.9**	**91.5**	**97.6**	**92.7**	**89.2**	**80.7**	**50.6**	**92.9**	**75.3**	**97.0**	87.8

3.5.5 Convergence performance

We evaluate the accuracy of the model at different training iterations on Aircraft. As illustrated in Figure 3.2a, our method exhibits significantly faster convergence compared to ZSCL. Particularly, the accuracy of our proposed model after training for 500 iterations is higher than that achieved by ZSCL after 1000 iterations.

TABLE 3.8. Impact of the first-order and second-order proximity on continual learning performance under Order I and Order II settings.

		Order I			Order II		
1st-order	2nd-order	Transfer	Avg.	Last	Transfer	Avg.	Last
✗	✗	44.6±0.3	55.9±0.2	77.3±0.4	46.6±0.3	56.2±0.3	67.4±0.6
✓	✗	68.9±0.2	76.2±0.3	84.5±0.4	63.8±0.2	75.1±0.2	84.2±0.3
✗	✓	67.6±0.3	73.9±0.1	83.4±0.5	62.9±0.3	73.0±0.4	81.6±0.5
✓	✓	**69.8**±0.1	**76.9**±0.1	**85.1**±0.2	**69.8**±0.1	**76.9**±0.2	**85.1**±0.2

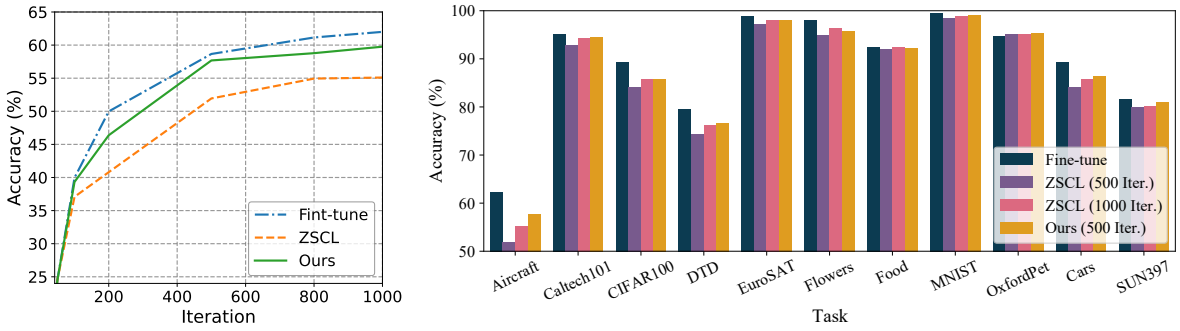
TABLE 3.9. Comparison of different weighting schemes under Order I and Order II evaluation settings.

	Order I			Order II		
Scheme	Transfer	Avg.	Last	Transfer	Avg.	Last
Only C_0	69.3±0.2	76.4±0.3	84.0±0.4	65.2±0.2	75.6±0.3	84.7±0.5
Only C_{i-1}	69.2±0.1	76.7±0.1	84.8±0.5	65.1±0.2	75.8±0.2	85.2±0.5
Average weighting	69.1±0.1	76.6±0.2	84.6±0.5	65.0±0.1	75.5±0.3	84.8±0.4
Adaptive weighting	**69.8**±0.1	**76.9**±0.1	**85.1**±0.2	**65.4**±0.1	**75.9**±0.2	**85.4**±0.2

We also present the immediate accuracy on each task during continual learning in Figure 3.2b, where the accuracy on each task is measured after completing the training on this task. Similarly, our proposed method outperforms ZSCL across all tasks with only a half iterations. It is noteworthy that on the OxfordPet, our method even outperforms direct fine-tuning of the original CLIP model, which can be considerated as upper bound. This phenomenon can be attributed to capability of our method to retain knowledge from previous tasks and apply it to the learning of current tasks. The better convergence performance of the proposed method shows that our trained model is adapted to new tasks easier.

3.5.6 Class Incremental Learning

To evaluate the proposed methods under the CIL setting, we conduct experiments on both CIFAR100 and TinyImageNet datasets. Results achieved on CIFAR100 and TinyImageNet are presented in Table 4.9 and Table 3.11, respectively. On both datasets, our method outperforms existing approaches in terms of both Avg. and Last performance. The learning



(A) (B)

FIGURE 3.2. Convergence performance. (a) Convergence performance of ZSCL and our proposed method on Aircraft. Result achieved by Fine-tune performs as the upper bound. (b) Immediate accuracy measured after training on each task during continual learning.

task becomes more challenge with the value CIL steps increases, but the propose method still surpass the others by a significant margin. Specifically, when training on CIFAR100 with 50 steps, our method achieves 81.91% Avg. accuracy and 70.17% Last accuracy, which are 1.98% and 2.81% higher than that achieved by the second best baseline, respectively. When training on TinyImageNet with 20 steps, the gaps even increase to 3.50% and 7.18%, respectively. The results in CIL benchmark further demonstrate our method is capable of not only preserving zero-shot transfer but also retaining knowledge acquired from previous tasks, which contributes to mitigating catastrophic forgetting in continual learning.

TABLE 3.10. Results on the CIFAR-100 dataset under the class-incremental learning (CIL) benchmark with 10, 20, and 50 steps.

	10 steps		20 steps		50 steps	
Method	**Avg.**	**Last**	**Avg.**	**Last**	**Avg.**	**Last**
UCIR (Hou et al. 2019)	58.66	43.39	58.17	40.63	56.86	37.09
DER (Yan et al. 2021)	74.64	64.35	73.98	62.55	72.05	59.76
DyTox+ (Douillard et al. 2022)	74.10	62.34	71.62	57.43	68.90	51.09
Zero-shot	74.57	65.98	75.34	65.98	75.88	65.98
Continual FT	65.46	53.23	59.69	43.13	39.23	18.89
LwF (Li and Hoiem 2017)	65.86	48.04	60.64	40.56	47.69	32.90
iCaRL (Rebuffi et al. 2017)	79.35	70.97	73.32	64.55	71.28	59.07
LwF-VR (Ding et al. 2022)	78.81	70.75	74.54	63.54	71.02	59.45
Cont.-CLIP (Thengane et al. 2022)	75.17	66.72	75.95	66.72	76.49	66.72
ZSCL (Radford et al. 2021)	82.15	73.65	80.39	69.58	79.92	67.36
Ours	**86.15**	**79.75**	**85.38**	**77.68**	**81.91**	**70.17**

TABLE 3.11. Results on the TinyImageNet dataset under the class-incremental learning (CIL) benchmark with 10, 20, and 50 steps.

	5 steps		10 steps		20 steps	
Method	**Avg.**	**Last**	**Avg.**	**Last**	**Avg.**	**Last**
EWC (Kirkpatrick et al. 2017)	19.01	6.00	15.82	3.79	12.35	4.73
UCIR (Hou et al. 2019)	50.30	39.42	48.58	37.29	42.84	30.85
DyTox (Douillard et al. 2022)	55.58	47.23	52.26	42.79	46.18	36.21
Zero-shot	69.68	65.47	69.62	65.47	69.56	65.47
Continual FT	61.54	46.66	57.05	41.54	54.62	44.55
LwF (Li and Hoiem 2017)	60.97	48.77	57.60	44.00	54.79	42.26
iCaRL (Rebuffi et al. 2017)	77.02	70.39	73.48	65.97	69.65	64.68
LwF-VR (Ding et al. 2022)	77.56	70.89	74.12	67.05	69.94	63.89
Cont.-CLIP (Thengane et al. 2022)	70.49	66.43	70.55	66.43	70.51	66.43
ZSCL	80.27	73.57	78.61	71.62	77.18	68.30
Ours	**82.43**	**77.53**	**81.74**	**76.51**	**80.68**	**75.48**

3.6 Conclusion

In this paper, we present a novel strategy for improving continual learning in vision-language models while preserving zero-shot capabilities. Specifically, Our approach focuses on balancing the model's zero-shot transfer capabilities with its adaptability to new data distributions. By constructing a cross-modal graph, we explore both inter- and intra-modal proximity. A distillation paradigm is introduced to preserve these proximity from two teachers, incorporated with a strategic sample re-weighting mechanism. This method effectively mitigates the potential conflict between model stability and plasticity, allowing both of these abilities to improve simultaneously. The results of the experiment, conducted on several benchmarks, demonstrate the effectiveness of the proposed approach.

CHAPTER 4

Model-Adaptive Mixture-of-Experts Architecture for Vision-Language Models

---

4.1 Related Work

4.1.1 Mixture of Experts.

The mixture of experts (MoE) has proven to be an efficient approach for significantly increasing model capacity while keeping computational overhead minimal (Rajbhandari et al. 2022; Lewis et al. 2021; Liu et al. 2023b). The MoE consists of multiple experts and a gating network (Shazeer et al. 2017). Several existing works aim to enhance model efficiency and capability by designing the structure of experts and routers, as well as incorporating regulation losses to reduce computational costs (He et al. 2021; He et al. 2022; Dai et al. 2024). Other studies have explored applying MoE to multimodal models to address existing challenges (Shen et al. 2023b). For example, LIMoE (Mustafa et al. 2022) is applied to the training of image and text data, and leverages contrastive loss and entropy-based regularization to address the load balancing issues typical of MoE systems. MoCLE (**MOCLE**) utilizes LoRA within the MoE and introduces a distinct universal expert to activate parameters of task-specific models according to the instruction clusters.

4.2 Motivation

In recent years, vision-language models like CLIP (Radford et al. 2021) have demonstrated exceptional performance across tasks such as image classification (Dong et al. 2022) and

text-based retrieval by aligning image and text representations within a shared embedding space. This cross-modal alignment enables CLIP to generalize to new tasks with minimal fine-tuning (Zhang et al. 2022), making it an ideal candidate for continual learning (CL) (Thengane et al. 2022), where models must adapt incrementally to new tasks while retaining previously learned knowledge. Similarly, Mixture of Experts (MoE) (Shazeer et al. 2017) architectures, which dynamically allocate computational resources, excel in CL by routing task-specific inputs to specialized experts, helping to mitigate catastrophic forgetting and enhancing adaptability to new tasks. However, integrating MoE with CLIP presents challenges, particularly in maintaining cross-modal alignment and balancing the learning dynamics between modalities, which can limit their effectiveness when combined.

Recently, continual learning approaches have focused on regularization, knowledge distillation, and MoE. Regularization techniques, such as EWC (Kirkpatrick et al. 2017) and LwF (Li and Hoiem 2017), constrain parameter updates to prevent forgetting but struggle with scaling to pretrained models. Knowledge distillation methods, like ZSCL (Zheng et al. 2023b) utilize reference data to maintain the relationship between modalities, balancing the model's zero-shot capability with adaptability to new tasks. MoE-enabled CLIP (Yu et al. 2024), enables separately expert selection to handle multiple tasks, but faces challenges in maintaining the cross-modality alignment and diverse task coverage.

While effective in certain contexts, these methods fall short in addressing the unique challenges of applying CLIP to continual learning. First, existing MoE implementations typically operate within single modalities, optimizing experts for either image or text inputs independently. This disrupts the cross-modal alignment central to CLIP's success, altering the representation distributions of image and text embeddings separately, which can degrade the shared semantic space. Second, this modality-specific design leads to an imbalance between the two modalities. For tasks like image classification, the text modality, typically corresponding to a fixed and limited set of labels, interacts with far fewer and less diverse samples than the image modality. This imbalance reduces the effectiveness of MoE in the text encoder, as its capacity cannot be fully utilized, limiting the model's ability to learn robust textual representations.

To tackle the aforementioned challenges, we propose a multimodal Mixture-of-Experts (MoE) architecture designed to enhance interaction between image and text modalities, while maintaining cross-modal alignment and mitigating modality imbalance. Instead of modifying textual features in isolation, our approach refines task-specific label representations in conjunction with their corresponding visual context. Specifically, textual embeddings are adaptively adjusted through interaction with image embeddings, ensuring that the updates are context-dependent and localized to specific image-text pairs. This design preserves the overall distribution of textual representations while allowing precise, task-aware refinements.

Our method serves as a context-aware adaptation mechanism that not only maintains semantic consistency but also improves the effectiveness of textual experts, enabling them to engage in richer and more diverse multimodal interactions. This, in turn, addresses the issue of modality imbalance. Moreover, we introduce a dynamic expert expansion mechanism that incrementally grows the expert pool as new tasks arrive, thereby ensuring adequate capacity for both previously seen and novel tasks. Extensive empirical results show that our model-adaptive strategy substantially enhances CLIP's training efficiency in continual learning settings, consistently achieving strong performance across a variety of benchmarks.

4.3 Preliminary

In the context of continual learning, the model is tasked with learning from a sequence of tasks $T = \{T_1, T_2, \ldots, T_N\}$, where each task T_i consists of a dataset $D_i = \{(x_{ij}, y_{ij})\}_{j=1}^{n_i}$. Here, x_{ij} represents an image sample, and y_{ij} is its corresponding textual label. The goal is to optimize the model such that it can effectively handle new tasks without degrading performance on previously learned ones, a phenomenon commonly referred to as catastrophic forgetting. Let M_t denote the model trained on task T_t, and $P(M_t, T_i)$ represent its performance on task T_i. The continual learning objective is to maximize the cumulative performance over all tasks seen so far:

$$\max_{M_t} \sum_{i=1}^{t} P(M_t, T_i). \tag{4.1}$$

This requires the model to balance adapting to new tasks while retaining knowledge from previous tasks, a central challenge in continual learning scenario.

4.3.1 CLIP

CLIP (Radford et al. 2021) is a vision-language model that aligns image and text representations in a shared embedding space. It consists of an image encoder E_{img} and a text encoder E_{txt}. Given an image x and its corresponding textual label y, their embeddings are computed as:

$$z_{img} = E_{img}(x), \quad z_{txt} = E_{txt}(y). \tag{4.2}$$

The alignment between z_{img} and z_{txt} is optimized using a contrastive loss:

$$-\mathbb{E}_{(x,y)\in D_i}\left[\log \frac{\exp(\text{sim}(z_{img}, z_{txt})/\tau)}{\sum_{y'\in Y}\exp(\text{sim}(z_{img}, E_{txt}(y'))/\tau)}\right] \tag{4.3}$$

where $\text{sim}(\cdot,\cdot)$ denotes cosine similarity, τ is a temperature parameter, and Y is the set of all labels in the current task. This loss ensures that the image and its corresponding textual label are closely aligned in the shared space.

4.3.2 Mixture-of-Experts in Continual Learning.

In continual learning, the Mixture-of-Experts framework facilitates the dynamic selection of experts to efficiently adapt to new tasks while mitigating catastrophic forgetting. The core idea of MoE is to route input z through a subset of experts, where each expert specializes in different aspects of the task. The output of the MoE model is defined as:

$$h(z) = z + \sum_{k=1}^{K} g_k(z) E_k(z), \tag{4.4}$$

where $g_k(z)$ is the gating function that determines the importance of each expert $E_k(z)$, based on the input z. The experts $E_k(z)$ are typically modified by task-specific parameters, allowing the model to adapt to new tasks without forgetting previously learned knowledge.

In CL tasks, MoE is often combined with low-rank adaptations, like LoRA (Hu et al. 2021), to efficiently handle task-specific updates. For each expert k, the adaptation is computed as:

$$E_k(z) = f(A_k B_k z), \tag{4.5}$$

where $A_k \in \mathbb{R}^{d \times r}$ and $B_k \in \mathbb{R}^{r \times d}$ are low-rank matrices, and $f(\cdot)$ represents a non-linear activation function. This low-rank formulation allows MoE to efficiently adapt to new tasks while minimizing the computational overhead.

In the CL scenario, the gating mechanism is also task-specific, dynamically selecting which experts to activate based on the input z and task t. The output for a specific task t is computed as:

$$h_t(z) = z + \sum_{k=1}^{K} g_k^t(z) E_k(z), \tag{4.6}$$

where $g_k^t(z)$ is the gating function that depends on the input z for the task t. This enables the model to selectively update experts that are relevant to the current task, avoiding catastrophic forgetting of previously learned tasks and enabling task-specific adaptation. Thus, MoE in CL effectively balances the need for task-specific knowledge with the ability to preserve previously learned knowledge, making it an efficient and scalable solution for continual learning in vision-language models.

4.4 Methodology

4.4.1 Problem Analysis

While Mixture of Experts (MoE) architectures have demonstrated strong performance in single-modality tasks, there is still room for improvement in adapting them to multimodal learning, particularly in continual learning settings.

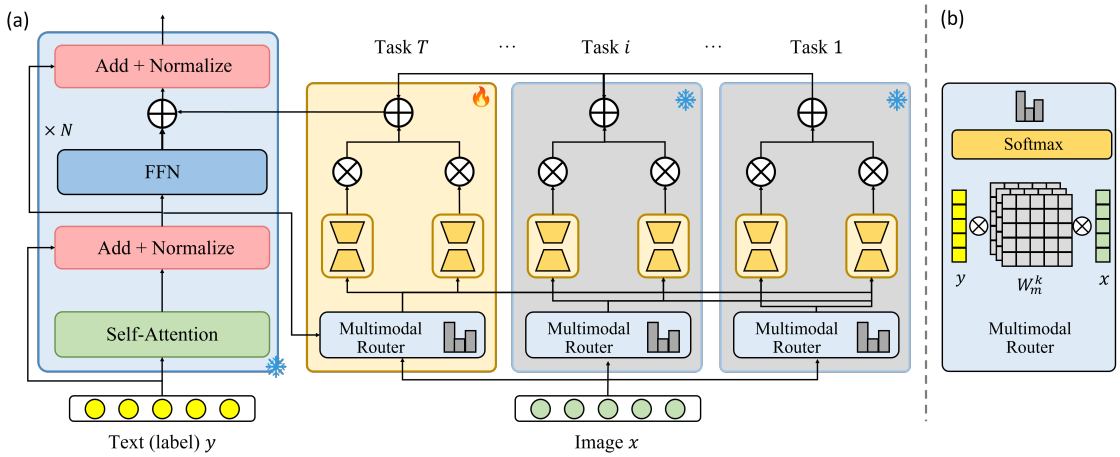


FIGURE 4.1. Illustrates the overall framework of the proposed Multimodal MoE. (a) Lora-MoE Integration with Pre-training Module: A multimodal router allocates experts based on image and text features. For each task, a dynamic router and m experts are introduced, with the router accessing all previous experts but only the parameters associated with the current task are updated. (b) The structure of the proposed Multimodal Router, where the gating function integrates image and text modalities to determine the activated experts.

4.4.1.1 Diversity Collapse in Text Modality

Continual learning in CLIP faces challenges in modality coordination, particularly in the text encoder, which suffers from semantic collapse when incremental tasks provide only sparse label texts (e.g., “dog” or “cat”). Although CLIP aligns visual and textual spaces through contrastive learning (Radford et al. 2021), its MoE-based text encoder relies on diverse inputs to effectively activate specialized experts (Fedus et al. 2022). However, low-diversity textual inputs in continual learning result in expert under-utilization, constraining both adaptability and generalization. Traditional MoE in CLIP relies solely on textual features, overlooking not only CLIP’s cross-modal nature but also the semantic collapse in the text modality, which ultimately limits the effectiveness of MoE.

4.4.1.2 Disruption of alignment.

Another limitation of existing CLIP-based MoE methods is the lack of inter-modal alignment in expert design. For instance, recent works (Zhang et al. 2024a; Yu et al. 2024) independently

adjust encodings for single modality without considering their cross-modal relationships. As discussed earlier, in continual learning, both text and image features are closely tied to domain-specific information. For example, in a task involving an image of a “cat" and its corresponding description, the routing of image features may differ from that in a “dog" task. As more tasks are introduced, the model may gradually misalign image-text pairs, leading to performance degradation. Therefore, we argue that existing approaches hinder efficient task transfer and struggle to maintain consistent image-text alignment, ultimately affecting the model's ability to generalize across tasks.

4.4.2 Multi-Modality Router

To address these limitations, we propose a multimodal MoE (MM-MoE) architecture, where the gating mechanism incorporates both image and text features. Specifically, given an image x and its corresponding text labels y, we extract the image and text embeddings z_{img} and z_{txt} with the CLIP encoder integrated with MoE components adapted to each layer of CLIP. The gating function then decides which experts should be activated based on both modalities. The multi-modal MoE features are computed as follows:

$$h_t(z_{\text{txt}}) = z_{\text{txt}} + \sum_{k=1}^{K} g_k^t(z_{\text{txt}}, z_{\text{img}}) E_k(z_{\text{txt}}). \tag{4.7}$$

Subsequently, we will delve into the architecture of the router, elaborating on its design and functionality. In a typical MoE architecture, the router computes gating weights $g_k(z)$ for each expert based on the input feature z. These weights determine the contribution of each expert to the final output, computed using a softmax function over the raw outputs $f_k(z)$:

$$g_k(z) = \frac{\exp(f_k(z))}{\sum_{k=1}^{K} \exp(f_k(z))}, \tag{4.8}$$

where $f_k(z) = zW_k$ is the linear transformation of the input feature z, and W_k is the learned weight matrix for expert k. The softmax ensures the gating weights sum to 1, assigning relative importance to each expert for the input z.

To extend this to multimodal inputs, such as image and text, we integrate both modalities into the gating computation. Specifically, we define the gating function as:

$$f(z_{\text{img}}, z_{\text{txt}}) = z_{\text{img}} W_M^k z_{\text{txt}}^T, \tag{4.9}$$

where $z \in \mathbb{R}^D$ is the input feature vector, and $W_M^k \in \mathbb{R}^{D \times D}$ is the learned weight matrix for the k-th expert. These weights $g_k(z)$ are then used to route the input to the appropriate experts. This formulation computes pairwise interactions between each image token and each text token, effectively capturing cross-modal dependencies. The output of the gating function is then used to compute the gating weights, which determine the influence of each expert based on both image and text features.

Our approach jointly utilizes both image and text features to guide the routing of text-specific MoE components. This is based on two key observations: First, text features, as labels, have discrete and enumerable properties, which align well with task partitioning. Second, text samples are less diverse than images, making them more suited for dynamic task-based routing. Consequently, the multi-modality routing mechanism is exclusively applied to the text domain, enabling fine-grained adjustments to text representations while maintaining the existing routing strategy for image extraction. In summary, For the same text, the model generates distinct routing decisions based on different images, ensuring that the text representation aligns more effectively with the context of each image. This adjustment prevents disruptions to image-text alignment, preserving the matching probability of images across tasks in the shared semantic space.

4.4.3 Dynamic Expert Expansion Mechanism

To further address the challenges of continual learning, our multi-modal MoE architecture introduces an expert and router expansion mechanism. This design enables the model to effectively scale as new tasks are introduced, ensuring that each task is assigned dedicated routers and experts, which minimizes interference between tasks. By preserving task-specific knowledge and allowing for independent adaptation, this mechanism ensures that the model

can continue to learn new tasks without compromising previously learned representations. We now describe the specific mechanisms for both the image and text modalities.

Image Modality. For each new task T_{t+1}, we introduce n new experts and a dedicated router for the image modality. The expert set $\mathcal{E}^{(t+1)}$ grows to $(t+1) \times n$.

$$\mathcal{E}^{(t+1)} = \{E_1, E_2, \ldots, E_{t\times n}, \ldots, E_{(t+1)\times n}\}. \tag{4.10}$$

The gating function $G_{\text{img}}^{(t+1)}$ computes routing weights $g_k(x_{\text{img}})$ for each expert.

$$G^{(t+1)}(x) = \{g_1(x), g_2(x), \ldots, g_{t+1}(x)\}, \quad \sum_{k=1}^{t+1} g_k(x) = 1.$$

Text Modality. Similarly, for the text modality, each new task T_{t+1} introduces a new router and m new experts. The gating function $G_{\text{txt}}^{(t+1)}$ restricts routing to the new experts.

Task-Adaptive Expert Assignment. To ensure task-specific knowledge is retained within each expert while promoting cross-task generalization, we optimize a task-adaptive loss function that balances specialization and knowledge sharing. The optimization objective for task T_{t+1} is formulated as:

$$\mathcal{L}_{T_t} = \mathcal{L}_{CLIP}(T_t; \Theta) + \lambda \sum_{k=1}^{t\cdot n} \Omega(E_k), \tag{4.11}$$

where Θ includes all trainable parameters for routers and experts, λ is a hyperparameter controlling the trade-off between task-specific learning and knowledge retention across tasks. The regularization term $\Omega(E_k)$ is introduced to encourage selective expert utilization by reducing the number of active experts for each task. This can be achieved through sparsity constraints, such as L1 regularization, which penalize excessive expert activation.

4.4.4 Algorithm of the proposed MM-MoE

To provide a clearer understanding of the training process for the proposed method, we present the complete training algorithm in Algorithm 2. This detailed algorithm outlines the step-by-step procedure.

Algorithm 2 Dynamic Expert Expansion for continual Learning

Input: Initial expert pool $\mathcal{E}^{(0)} = \{E_1, E_2, \ldots, E_N\}$, initial router $G^{(0)}$, task sequence $T = \{T_1, T_2, \ldots, T_M\}$, expansion size m.
Output: Trained model Θ with dynamic experts and routers.

for each task $T_t \in T$ **do**
  Load task dataset $D_t = \{(x_{ij}, y_{ij})\}_{j=1}^{n_t}$
  if $t > 1$ **then**
    Expand expert pool: Add m new experts $\{E_{N+1}, \ldots, E_{N+m}\}$ to $\mathcal{E}^{(t)}$
    Expand router: Update gating network $G^{(t)}$ to handle $N + m$ experts
  end if
  Optimize task-specific parameters by minimizing: $\mathcal{L}_{T_t} = \mathcal{L}_{MoE-CLIP}(T_t; \Theta) + \lambda \sum_{k=1}^{N+m} \Omega(E_k)$
  Update model parameters: $\Theta \leftarrow \Theta - \eta \nabla \mathcal{L}_{T_t}$
  Increment expert counter: $N \leftarrow N + m$
end for
Trained model Θ

4.5 Experiments

In this section, the implementation details required for experiments are introduced first. Next, we present and analyze the results of the proposed MM-MOE compared to state-of-the-art approaches under the MTIL and CIL experimental settings. Following this, an analysis and comparison of compute costs are provided. Then, we conduct an ablation study to explore the impact of various factors on the proposed method. Finally, we also present a comparison between our method and other approaches on the ADEM-VL model.

4.5.1 Implementation Details

4.5.1.1 Continual learning setting.

We primarily evaluate the proposed method on two continual learning settings: multi-domain task incremental learning (MTIL) and class incremental learning (CIL).

MTIL is a widely-used benchmark specifically designed for cross-domain task incremental learning, where each task originates from a distinct dataset.

CIL is a fundamental setting in continual learning, in which the model is progressively updated with only new class instances. During inference, the model needs to distinguish all previously learned classes but does not know the task ID of the sample.

4.5.1.2 Datasets.

In the MTIL benchmark, we use 11 datasets: Aircraft (Maji et al. 2013), Caltech101 (Fei-Fei et al. 2004), CIFAR100 (Krizhevsky, Hinton et al. 2009), DTD (Cimpoi et al. 2014), EuroSAT (Helber et al. 2019), Flowers (Nilsback and Zisserman 2008), Food (Bossard et al. 2014), MNIST (Deng 2012), OxfordPet (Parkhi et al. 2012), StanfordCars (Krause et al. 2013), and SUN397 (Xiao et al. 2010). Additionally, we followed the settings in (Zheng et al. 2023b) and conducted experiments on two different task orders. In the CIL benchmark, we employ CIFAR-100 (Krizhevsky, Hinton et al. 2009), where the 100 classes are evenly partitioned into 10 steps, and TinyImageNet (Le and Yang 2015), where the base step comprises 100 classes, while the remaining classes are uniformly distributed across 10 subsequent steps, with each step constituting a task. we use CIFAR100 (Krizhevsky, Hinton et al. 2009) and TinyImageNet (Le and Yang 2015) datasets, where the classes in each dataset are evenly split into 10 groups, with each group serving as a task.

4.5.1.3 Model.

A pretrained CLIP model with a ViT-B/16 image encoder is employed for the MTIL and CIL experiments. Additionally, we use a pretrained ADEM-VL model in the MTIL setting. The input resolution is 224, and the patch size is 16.

Metric. In the MTIL setting, we utilize metrics “Transfer", “Average" and “Last" as used in (Zheng et al. 2023b). “Transfer" metric evaluates the model’s zero-shot transfer capability on unseen data, while “Last" measures the model’s ability to retain historical knowledge. The “Average" metric is a composite measure representing the mean performance across “Transfer" and “Last". In the CIL experimental setting, we assess the model’s performance using the average accuracy at each step, where average accuracy represents the average performance across all previous tasks.

4.5.1.4 Baselines.

Given that the CLIP model can adapt to new data distributions without requiring retraining, we use zero-shot transfer as one of our baselines. We also explore direct fine-tuning of the CLIP model as upper bound. MM-MoE is compared with fully trainable methods, including typical continual learning approaches as well as existing CLIP-based methods. Additionally, we include a method that utilizes the mixture of experts on CLIP: MoE-adapters.

4.5.2 Results of Multi-domain Task Incremental Learning

4.5.2.1 Main Results

We evaluate MM-MoE against baselines using the MTIL benchmark. The backbone utilized is the CLIP model with ViT-B/16. Each task undergoes training for 1000 iterations using a batch size of 64. LoRA (Hu et al. 2021) is employed as the experts, with the number of experts per task set to 2. A single MLP serves as the router, combining experts based on the top-3 gating scores. AdamW (Loshchilov 2017) is used as the optimizer. The parameter settings for DDAS follow those in (Yu et al. 2024).

TABLE 4.1. Results comparison on the MTIL benchmark under task order setting I. Δ indicates the performance difference between each continual learning method and the original CLIP model (Zero-shot).

Method	**Transfer**	Δ	**Avg.**	Δ	**Last**	Δ
Zero-shot	69.4	-	65.3	-	65.3	-
Continual FT	44.6	-24.8	55.9	-9.4	77.3	+12.0
LwF (Li and Hoiem 2017)	56.9	-12.5	64.7	-0.6	74.6	+9.0
iCaRL (Rebuffi et al. 2017)	50.4	-19.0	65.7	+0.4	80.1	+14.8
LwF-VR (Ding et al. 2022)	57.2	-12.2	65.1	-0.2	76.6	+11.3
WiSE-FT (Wortsman et al. 2022)	52.3	-17.1	60.7	-4.6	77.7	+12.4
ZSCL (Zheng et al. 2023b)	68.1	-1.3	75.4	+10.1	83.6	+18.3
MoE-Adapters (Yu et al. 2024)	68.9	-0.5	76.7	+11.4	85.0	+19.7
MM-MoE	**69.4**	**+0.0**	**78.2**	**+12.9**	**87.1**	**+21.8**

Table 4.1 and Table 4.2 present the results of continual learning after sequentially completing all 11 tasks, covering both task order settings I and II. As shown, MM-MoE outperforms

TABLE 4.2. Results comparison on the MTIL benchmark under task order setting II. Δ indicates the performance difference between each continual learning method and the original CLIP model (Zero-shot).

Method	**Transfer**	Δ	**Avg.**	Δ	**Last**	Δ
Zero-shot	65.4	-	65.3	-	65.3	-
Continual FT	46.6	-18.8	56.2	-9.1	67.4	+2.1
LwF (Li and Hoiem 2017)	53.2	-12.2	62.2	-5.2	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)	50.9	-14.5	56.9	-8.4	71.6	+6.3
WiSE-FT (Wortsman et al. 2022)	51.0	-14.4	61.5	-5.9	72.2	+6.9
ZSCL (Zheng et al. 2023b)	64.2	-1.2	74.5	+9.2	83.4	+18.1
MoE-Adapters (Yu et al. 2024)	64.3	-1.1	74.7	+9.4	84.1	+18.8
MM-MoE	**65.1**	**-0.3**	**75.5**	**+10.2**	**86.7**	**+21.4**

all other baselines across every metric. Regarding the Transfer metric, continual fine-tuning on the CLIP model leads to a significant decline in Transfer performance, whereas ZSCL and MoE-Adapters show a minimal decrease. Notably, in task order setting I, the Transfer performance of the proposed method matches that of the original CLIP model. Since MM-MoE isolates parameters for each task as much as possible, it effectively mitigates catastrophic forgetting, resulting in significant improvements on both the Avg. and Last metrics, with increases of 12.9% and 21.8%, respectively. The results observed under task order settings II show a similar pattern, with improvements of 10.2% and 21.4% in “Avg." and “Last" respectively.

Table 4.3 presents detailed results for each method on individual tasks under task order setting I, showcasing the performance of the final trained models on each task in the MTIL benchmark. MM-MoE still consistently outperforms the baselines in the “Transfer", “Avg.," and “Last" metrics. For “Last" metric, Fully parameter training methods, such as ZSCL and WiSE-FT, perform well for certain datasets due to their significantly larger number of parameters compared to the proposed mothod. However, MM-MoE achieves the best results across most datasets. Overall, MM-MoE demonstrates superior performance across three metrics, highlighting its robustness and effectiveness in continual learning compared to existing baselines.

TABLE 4.3. Performance of the proposed method and baselines on each task in the MTIL benchmark for order I. The best result is highlighted in **bold**, and the second-best is underlined.

Method	Aircraft	Caltech101	CIFAR100	DTD	EuroSAT	Flowers	Food	MNIST	OxfordPet	Cars	SUN397
Zero-shot	24.3	88.4	68.2	44.6	54.9	71.0	88.5	59.4	89.0	64.7	65.2
Fine-tuning	62.0	95.1	89.6	79.5	98.9	97.5	92.7	99.6	94.7	89.6	81.8
						Transfer					
Continual FT		67.1	46.0	32.1	35.6	35.0	57.7	44.1	60.8	20.5	46.6
LwF		74.5	56.9	39.1	51.1	52.6	72.8	60.6	75.1	30.3	55.9
iCaRL		56.6	44.6	32.7	39.3	46.6	68.0	46.0	77.4	31.9	60.5
LwF-VR		77.1	61.0	40.5	45.3	54.4	74.6	47.9	76.7	36.3	58.6
WiSE-FT		73.5	55.6	35.6	41.5	47.0	68.3	53.9	69.3	26.8	51.9
ZSCL		86.0	67.4	**45.4**	50.4	69.1	87.6	**61.8**	86.8	60.1	**66.8**
MoE-Adapters		87.9	**68.2**	44.4	49.9	**70.7**	**88.7**	59.7	**89.1**	64.5	65.5
MM-MoE		**88.4**	**68.2**	43.9	**55.3**	**70.7**	88.5	59.5	**89.1**	**64.7**	65.6
						Avg.					
Continual FT	25.5	81.5	59.1	53.2	64.7	51.8	63.2	64.3	69.7	31.8	49.7
LwF	36.3	86.9	72.0	59.0	73.7	60.0	73.6	74.8	80.0	37.3	58.1
iCaRL	35.5	89.2	72.2	60.6	68.8	70.0	78.2	62.3	81.8	41.2	62.5
LwF-VR	29.6	87.7	74.4	59.5	72.4	63.6	77.0	66.7	81.2	43.7	60.7
WiSE-FT	26.7	86.5	64.3	57.1	65.7	58.7	71.1	70.5	75.8	36.9	54.6
ZSCL	45.1	92.0	80.1	64.3	79.5	81.6	**89.6**	**75.2**	88.9	64.7	**68.0**
MoE-Adapters	50.2	91.9	83.1	69.4	78.9	84.0	89.1	73.7	**89.3**	67.7	66.9
MM-MoE	**56.3**	**94.3**	**84.2**	**69.8**	**82.3**	**84.9**	88.6	73.9	89.1	**68.6**	67.0
						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
LwF	26.3	87.5	71.9	66.6	79.9	66.9	83.8	**99.6**	92.1	66.1	80.4
iCaRL	35.8	93.0	77.0	70.2	83.3	88.5	90.4	86.7	93.2	81.2	81.9
LwF-VR	20.5	89.8	72.3	67.6	85.5	73.8	85.7	**99.6**	93.1	73.3	80.9
WiSE-FT	27.2	90.8	68.0	68.9	86.9	74.0	87.6	**99.6**	92.6	77.8	**81.3**
ZSCL	40.6	92.2	81.3	70.5	94.8	90.5	**91.9**	98.7	**93.9**	85.3	80.2
MoE-Adapters	49.8	92.2	86.1	78.1	95.7	94.3	89.5	98.1	89.9	81.6	80.0
MM-MoE	**56.3**	**95.0**	**87.7**	**79.5**	**97.8**	**96.8**	89.0	99.0	89.2	**86.2**	80.7

4.5.2.2 Details of Experimental Results under MTIL benchmark

Table 4.4 presents results of the proposed MM-MoE on the Multi-domain Task Incremental Learning (MTIL) benchmark under task order setting I, showing the performance across 11 datasets after continuous training on each task.

TABLE 4.4. Task-wise accuracy (%) across all datasets in the MTIL benchmark, based on task order setting I, following the completion of training on each task.

	Aircraft	Caltech101	CIFAR100	DTD	EuroSAT	Flowers	Food	MNIST	OxfordPet	Cars	SUN397
Aircraft	56.3	88.4	68.2	44.7	55.3	71.0	88.5	59.5	89.0	64.7	65.6
Caltech101	56.3	95.0	68.2	42.8	55.3	70.7	88.5	59.5	89.0	64.7	65.6
CIFAR100	56.3	94.1	87.7	44.3	55.3	70.7	88.5	59.5	89.0	64.7	65.6
DTD	56.3	95.0	87.7	79.5	55.3	70.7	88.5	59.5	89.0	64.7	65.6
EuroSAT	56.3	95.0	87.7	79.5	97.8	70.7	88.5	59.5	89.0	64.7	65.6
Flowers	56.3	95.0	87.7	79.5	97.8	96.8	88.5	59.5	89.0	64.7	65.6
Food	56.3	95.0	87.7	79.5	97.8	96.8	89.0	59.5	89.0	64.7	65.6
MNIST	56.3	95.0	87.7	79.5	97.8	96.8	89.0	99.0	89.0	64.7	65.6
OxfordPet	56.3	95.0	87.7	79.5	97.8	96.8	89.0	99.0	89.2	64.7	65.6
Cars	56.3	95.0	87.7	79.5	97.8	96.8	89.0	98.9	89.2	86.2	65.6
SUN397	56.3	95.0	87.7	79.5	97.8	96.8	89.0	99.0	89.2	86.2	80.7
Transfer		88.4	68.2	43.9	55.3	70.7	88.5	59.5	89.0	64.7	65.6
Avg.	56.3	94.3	84.2	69.8	82.3	84.9	88.7	73.9	89.1	68.6	67.0
Last	56.3	95.0	87.7	79.5	97.8	96.8	89.0	99.0	89.2	86.2	80.7

Additionally, we provide further experimental details regarding task order setting II. The task sequence for this setting is as follows: StanfordCars, Food, MNIST, OxfordPet, Flowers, SUN397, Aircraft, Caltech101, DTD, EuroSAT, and CIFAR100. The detailed experimental results of MM-MoE after each task are presented in Table 4.5. Furthermore, Table 4.6 provides a comparison of MM-MoE against baselines. Despite the change in task order, the proposed MM-MoE continues to effectively mitigate catastrophic forgetting while preserving the zero-shot capability of CLIP.

4.5.3 Results of Class Incremental Learning

4.5.3.1 Main Results

We also conduct experiments on class incremental learning (CIL) using CIFAR100 and TinyImageNet to validate the effectiveness of the proposed method. Specifically, in a 10-step setting, the classes in CIFAR100 are evenly divided into 10 groups, with each group

TABLE 4.5. Task-wise accuracy (%) across all datasets in the MTIL benchmark, based on task order setting II, following the completion of training on each task.

	Cars	Food	MNIST	OxfordPet	Flowers	SUN397	Aircraft	Caltech101	DTD	EuroSAT	CIFAR100
Cars	86.8	88.5	60.7	89.0	71.0	65.4	24.3	88.4	44.7	55.3	68.2
Food	86.8	89.1	55.5	89.0	71.0	65.4	24.3	88.4	44.7	54.6	68.2
MNIST	86.8	89.1	98.8	89.0	71.0	65.4	24.3	88.4	44.7	54.6	68.2
OxfordPet	86.8	89.1	98.8	89.2	71.0	65.4	24.3	88.4	44.7	54.6	68.2
Flowers	86.8	89.1	98.8	89.2	71.0	65.4	18.1	88.4	44.7	54.6	68.2
SUN397	86.8	89.1	98.8	89.2	96.8	65.4	24.3	88.4	44.4	54.6	68.2
Aircraft	86.8	89.1	98.8	89.2	96.8	80.3	54.9	88.4	44.4	54.6	68.2
Caltech101	86.8	89.1	98.8	89.2	96.8	80.3	54.9	94.2	42.1	54.6	68.2
DTD	86.8	89.1	98.8	89.2	96.8	79.2	54.9	92.8	79.0	54.6	68.2
EuroSAT	86.8	89.1	98.8	89.2	96.8	80.3	54.9	94.2	79.0	97.4	68.2
CIFAR100	86.8	89.1	98.8	89.2	96.8	80.4	54.9	94.1	79.0	97.4	87.5
Transfer		88.5	58.1	89.0	71.0	65.4	23.3	88.4	44.3	54.6	68.2
Avg.	86.8	88.6	91.4	89.1	87.4	73.4	37.6	90.4	53.8	62.4	70.0
Last	86.8	89.0	98.8	89.2	96.9	80.4	54.8	94.1	79.0	97.4	87.5

representing a task. For TinyImageNet, 100 classes are learned in the base step, while the remaining 100 classes are divided into 10 groups.

Since the task ID of images is not provided in CIL benchmark, we use only a single router for the image encoder of CLIP without employing DDAS, which is not effective for single-domain tasks. The average accuracy after each step on both datasets is presented in Table 4.7 and Table 4.8, respectively. As shown in the results of Table 4.7, full-parameter training methods, including ZSCL, LwF-VR, and iCaRL, achieve higher accuracy on the first task. In contrast, MM-MoE shows lower performance on the first task due to the limited number of trainable parameters. However, for all subsequent tasks, the proposed MM-MoE achieves the highest accuracy, surpassing the state-of-the-art full-parameter methods by 2.1% at the last step. The results in Table 4.8 indicate a significant improvement of our method over all baselines on TinyImageNet, showing a consistent trend. The proposed method consistently demonstrates the best performance among all approaches throughout the learning process.

TABLE 4.6. Performance of the proposed method and baselines on each task in the MTIL benchmark for order I. The best result is highlighted in **bold**, and the second-best is underlined.

Method	Cars	Food	MNIST	OxfordPet	Flowers	SUN397	Aircraft	Caltech101	DTD	EuroSAT	CIFAR100
Zero-shot	64.7	88.5	59.4	89.0	71.0	65.2	24.3	88.4	44.6	54.9	68.2
Fine-tuning	89.6	92.7	99.6	94.7	97.5	81.8	62.0	95.1	79.5	98.9	89.6
						Transfer					
Continual FT		85.9	**59.6**	57.9	40.0	46.7	11.1	70.0	30.5	26.6	37.7
LwF (Li and Hoiem 2017)		87.8	58.5	71.9	46.6	57.3	12.8	81.4	34.5	33.7	46.8
iCaRL (Rebuffi et al. 2017)		86.1	51.8	67.6	50.4	57.9	11.0	72.3	31.2	32.7	48.1
LwF-VR (Ding et al. 2022)		88.2	57.0	71.4	50.0	58.0	13.0	82.0	34.4	29.3	47.6
WiSE-FT (Wortsman et al. 2022)		87.2	57.6	67.0	45.0	54.0	12.9	78.6	35.5	28.4	44.3
ZSCL (Zheng et al. 2023b)		88.3	57.5	84.7	68.1	64.8	21.1	88.2	**45.3**	**55.2**	**68.2**
MoE-Adapters (Zheng et al. 2023b)		**88.8**	59.5	**89.1**	69.4	65.3	15.0	87.9	43.9	54.6	**68.2**
Ours		88.5	58.1	89.0	**71.0**	**65.4**	**23.3**	**88.4**	44.3	54.6	**68.2**
						Avg.					
Continual FT	42.1	70.5	**92.2**	80.1	54.5	59.1	19.8	78.3	41.0	38.1	42.3
LwF (Li and Hoiem 2017)	49.0	77.0	92.1	85.9	66.5	67.2	20.9	84.7	44.6	45.5	50.5
iCaRL (Rebuffi et al. 2017)	52.0	75.9	77.4	74.6	58.4	59.3	11.7	79.6	42.1	43.2	51.7
LwF-VR (Ding et al. 2022)	44.9	75.8	91.8	85.3	63.5	67.6	16.9	84.9	44.0	40.6	51.3
WiSE-FT (Wortsman et al. 2022)	52.6	79.3	91.9	83.9	63.4	65.2	23.3	83.7	45.4	40.0	48.2
ZSCL (Zheng et al. 2023b)	81.7	**91.3**	91.1	**91.0**	82.9	72.5	33.6	89.7	53.3	**62.8**	69.9
MoE-Adapters (Zheng et al. 2023b)	**86.8**	89.3	**92.2**	89.1	86.0	73.0	30.8	90.0	53.1	62.6	69.9
Ours	**86.8**	88.6	91.4	89.1	**87.4**	**73.4**	**37.6**	**90.4**	**53.8**	62.4	**70.0**
						Last					
Continual FT	24.0	67.3	99.1	87.4	44.3	67.0	29.5	92.3	61.3	81.0	**88.1**
LwF (Li and Hoiem 2017)	34.6	69.6	99.3	88.7	61.1	72.5	32.5	88.1	65.6	90.9	87.9
iCaRL (Rebuffi et al. 2017)	46.0	81.5	91.3	82.8	66.5	72.2	16.3	91.6	68.1	83.2	87.8
LwF-VR (Ding et al. 2022)	27.4	61.2	99.4	86.3	60.6	70.7	23.4	88.0	61.3	84.3	**88.1**
WiSE-FT (Wortsman et al. 2022)	35.6	76.9	**99.5**	89.1	62.1	71.8	27.8	90.8	67.0	85.6	87.6
ZSCL (Zheng et al. 2023b)	78.2	**91.1**	97.6	**92.5**	87.4	78.2	45.0	92.3	72.7	96.2	86.3
MoE-Adapters (Zheng et al. 2023b)	83.7	88.5	99.2	88.7	92.9	75.9	42.6	93.1	76.6	**98.6**	86.4
Ours	**86.8**	89.0	98.8	89.2	**96.9**	**80.4**	**54.8**	**94.1**	**79.0**	97.4	87.5

Additionally, the results in Table 4.7 and Table 4.8 both show that “MM-MoE*", representing the proposed MM-MoE method without Multi-Modality Router for the text modality, has a lower average accuracy at each step compared to “MM-MoE". This comparison indicates that

TABLE 4.7. Performance of the trained model at each step in the CIL benchmark on CIFAR-100. The “Average Accuracy" represents the mean accuracy across all previous tasks.“MM-MoE*" indicates the proposed method without Multi-Modality Router for the two modalities. The best result is highlighted in **bold**.

Method	Step 1	Step 2	Step 3	Step 4	Step 5	Step6	Step 7	Step8	Step 9	Step 10
iCaRL	98.8	92.5	86.9	78.6	72.7	73.7	73.6	71.7	71.4	71.9
LwF-VR	**99.0**	91.6	86.4	78.4	73.6	75.3	73.9	72.0	67.6	70.5
Continual-CLIP	96.7	92.2	86.0	80.4	77.5	75.8	73.0	71.4	69.8	66.7
ZSCL	98.8	94.8	90.1	85.8	82.7	80.8	80.9	80.1	77.6	77.5
MoE-Adapters	98.4	94.7	89.3	85.4	83.6	83.3	81.4	79.8	78.9	77.5
MM-MoE*	98.1	93.6	88.5	85.5	83.8	82.9	80.4	80.4	79.1	77.9
MM-MoE	98.4	**95.6**	**90.8**	**87.6**	**85.4**	**84.7**	**82.6**	**81.5**	**80.3**	**79.6**

TABLE 4.8. Performance of the trained model at each step in the CIL benchmark on TinyImageNet. The “Average Accuracy" represents the mean accuracy across all previous tasks. “MM-MoE*" indicates the proposed method without Multi-Modality Router for the two modalities. The best result is highlighted in **bold**.

Method	Base	Step 1	Step 2	Step 3	Step 4	Step 5	Step6	Step 7	Step8	Step 9	Step 10
iCaRL	**88.0**	82.3	78.2	77.8	73.6	71.2	70.7	69.0	68.3	66.1	65.4
LwF-VR	87.8	80.4	77.4	77.9	74.4	71.3	71.7	71.7	69.7	70.4	68.8
ZSCL	87.0	84.8	83.8	82.6	81.5	80.2	79.1	79.1	78.1	77.2	76.2
MoE-Adapters	85.5	83.9	82.8	82.0	80.7	79.6	79.0	78.2	77.5	76.9	76.4
MM-MoE*	86.1	84.5	83.7	82.4	82.1	80.8	79.7	79.0	78.3	77.2	76.8
MM-MoE	86.9	**85.6**	**84.4**	**83.9**	**83.3**	**82.8**	**81.6**	**80.9**	**79.8**	**79.4**	**78.5**

incorporating information exchange between the two modalities to guide the selection of text experts effectively helps improve the continual learning performance of CLIP.

Overall, the proposed MM-MoE, which trains only a limited number of parameters, effectively mitigates catastrophic forgetting in the CIL benchmark, demonstrating substantial effectiveness across different datasets.

4.5.3.2 More Results under CIL benchmark

We provide additional results on the CIFAR100 dataset under the Class Incremental Learning (CIL) benchmark. Specifically, in the t-step setting, the classes in CIFAR100 dataset are evenly divided into t groups, with each group representing a task. We set t = {10, 20, 50}.

We present the experimental results in Table 4.9. From the results, it is evident that as the number of steps increases, the proposed method consistently outperforms others, both in terms of "Avg." and "Last." Overall, MM-MoE is more effective in mitigating catastrophic forgetting in continual learning, as it fully takes into account the information from both modalities.

TABLE 4.9. Results on the CIFAR100 dataset across different steps under CIL benchmark.

	10 steps		20 steps		50 steps	
Method	**Avg.**	**Last**	**Avg.**	**Last**	**Avg.**	**Last**
UCIR (Hou et al. 2019)	58.66	43.39	58.17	40.63	56.86	37.09
Bic (Wu et al. 2019)	68.80	53.54	66.48	47.02	62.09	41.04
PODNet (Douillard et al. 2020)	58.03	41.05	53.97	35.02	51.19	32.09
DER (Yan et al. 2021)	74.64	64.35	73.98	62.55	72.05	59.76
DyTox+ (Douillard et al. 2022)	74.10	62.34	71.62	57.43	68.90	51.09
Zero-shot (Radford et al. 2021)	74.57	65.98	75.34	65.98	75.88	65.98
Continual FT	65.46	53.23	59.69	43.13	39.23	18.89
LwF (Li and Hoiem 2017)	65.86	48.04	60.64	40.56	47.69	32.90
iCaRL (Rebuffi et al. 2017)	79.18	71.86	73.99	64.46	69.95	58.56
LwF-VR (Ding et al. 2022)	78.81	70.48	74.66	62.13	72.46	61.06
ZSCL (Radford et al. 2021)	85.09	77.18	82.94	74.71	81.91	70.17
MoE-ada. (Yu et al. 2024)	85.21	77.52	83.72	76.20	83.60	75.24
MM-MoE	**86.72**	**79.63**	**85.12**	**78.16**	**84.97**	**76.79**

4.5.4 Analysis of the Compute Costs

The proposed MM-MoE not only achieves satisfactory experimental results but also demonstrates considerable efficiency. To measure runtime, we used 4 NVIDIA V100 GPUs, an Intel 6278C CPU, and PyTorch 2.1.0 with CUDA 11.4. Table 4.10 presents the comparison results of compute costs. As indicated by the results, MoE-based methods, including both MoE-Adapters and MM-MoE, lead to an overall increase in parameter count but significantly

reduce the number of training parameters. Specifically, the proposed MM-MoE requires only 4.12M trainable parameters per task, which is 0.08× that of MoE-Adapters and 0.03× that of LWF and ZSCL. Moreover, MM-MoE imposes the lowest GPU burden and achieves a 21.6% speed improvement over the most computationally efficient baseline.

TABLE 4.10. Comparison of the number of computational cost during the training process of each task between MM-MoE and others.

Method	Params↓	Train Params↓	GPU↓	Runtime↓
LWF (Li and Hoiem 2017)	**149.6M**	149.6M	34836MiB	2.86s/it
ZSCL (Zheng et al. 2023b)	**149.6M**	149.6M	29174MiB	5.47s/it
MoE-Ada. (Yu et al. 2024)	200.7M	51.1M	20375MiB	2.68s/it
MM-MoE	197.3M	**4.12M**	**18631MiB**	**2.03s/it**

TABLE 4.11. Impact of Multi-Modality Router and Dynamic Expert Expansion Mechanism.

Multi-Modality Router	Dynamic Expert Expansion	Transfer	Avg.	Last
✗	✗	67.5	71.1	77.9
✓	✗	69.1	77.3	85.8
✗	✓	68.6	75.8	83.3
✓	✓	**69.4**	**78.2**	**87.1**

4.5.5 Ablation Study

In this section, we focus on analyzing the different aspects of the effectiveness of MM-MoE. All experiments are conducted on the MTIL benchmark under task order setting I.

4.5.5.1 Impact of Multi-Modality Router and Dynamic Expert Expansion Mechanisms.

To enable MoE to assist vision-language models in continual learning tasks, we incorporate a Multi-Modality Router mechanism that facilitates interactions between images and text to select experts for each sample. Additionally, We introduce a Dynamic Expert Expansion Mechanism with task-adaptive loss to facilitate the addition of new experts. We conduct experiments to evaluate the impact of these two components on continual learning performance, with the results presented in Table 4.11.

The results show that the model performs worst when the Multi-Modality Router is removed and the router and experts are not increased as tasks grow. Introducing the Multi-Modality Router improves the model's performance on three metrics. Furthermore, the best results are achieved when both two components are adopted, with maximum performance improvements of 1.6% for "Transfer", 7.1% for "Avg.", and 9.2% for "Last". These experimental results indicate that leveraging multimodal information for expert selection and dynamically adding MoE components as tasks increase can both contribute to improving the performance of vision-language models in continual learning tasks.

TABLE 4.12. Ablation study of Mixture-of-Experts on the number of experts per task and the top-k of router gates.

Experts per task				**Top-k of router gates**			
m	Transfer	Avg.	Last	k	Transfer	Avg.	Last
1	68.9	77.8	86.6	1	68.5	74.2	81.0
2	69.4	78.2	87.1	2	69.2	77.5	86.4
3	69.6	78.3	87.3	3	69.4	78.2	87.1

4.5.5.2 Analysis of incremental MoE.

We conduct experiments to explore two parameters in MoE: the number of experts per task m and the top-k value of the router gates. When changing the number of experts per task m, k is kept one greater than the number of experts to ensure that the knowledge learned by previous experts can be utilized. When k is varied, the number of experts is set to 2.

We present the experimental results in Table 4.12. It can be observed that setting the number of experts per task m to 2 yields acceptable performance. Increasing the number of experts per task m to 3 brings only a slight improvement in performance, but more experts means introducing more trainable parameters. Therefore, our experiments ultimately set the number of experts per task m to 2. On the other hand, when the number of experts per task m is fixed at 2 and only the top-1 expert is selected, the results are the worst across all three metrics, especially for "Last". Moreover, when k is set to 3, the model can select not only the newly added experts but also those trained on previous tasks, thereby leveraging learned knowledge. This achieves the best performance across all three metrics. On the other hand, when k =

2, which matches the number of newly added experts, Due to the balance loss, the model struggles to select the experts trained on previous tasks and leverage the knowledge they carry, leading to a decrease in performance. Moreover, increasing k to 4 does not yield noticeable performance gains.

4.5.5.3 Impact of λ.

We introduce a task-adaptive loss to ensure that newly added experts receive sufficient training in the early training steps, and we set a hyperparameter, λ, to balance the contribution between different losses. We conducted experiments to investigate the effect of λ on the results, and the outcomes are presented in Figure 4.2. When λ is set to 0, indicating no task-adaptive loss, the results across all three metrics are the worst. When λ is greater than 0, all three metrics show significant improvement, especially “Last," and there is little fluctuation in the results across the metrics. This demonstrates both the necessity of adding task-adaptive loss and the model’s insensitivity to the λ parameter. In our experiments, λ is set to 0.05.

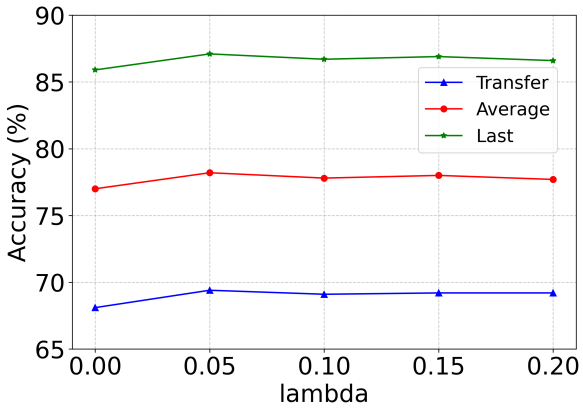


FIGURE 4.2. Impact of λ

4.5.6 Discussion

The results of our experiments clearly demonstrate that the integration of dynamic routing and cross-attention significantly enhances the ability of models to learn incrementally without forgetting previous knowledge. The dynamic router allows for better expert selection tailored

to the specific input features, while the cross-attention mechanism enriches the model's representation by leveraging multimodal information.

Furthermore, the ablation study confirms the effectiveness of each component, underscoring the importance of both dynamic routing and multimodal interaction in achieving superior performance in continual learning tasks.

4.6 Conclusion

In this paper, we proposed a Multimodal MoE architecture and a Dynamic Expert Expansion Mechanism to address the challenges of applying CLIP in continuous learning scenarios. Our approach leverages cross-attention to maintain cross-modal alignment and improve expert utilization, ensuring balanced learning across image and text modalities. Additionally, the dynamic expansion mechanism enables the model to scale its capacity as new tasks are introduced, reducing task interference while maintaining computational efficiency. These innovations provide a scalable and flexible framework for continuous learning, offering significant improvements in task adaptability and performance.

4.6.1 Limitations and Future Work

While our proposed method shows promising results, certain limitations remain. For instance, the performance may vary depending on the specific task complexity and the number of classes. Future work will focus on optimizing the dynamic routing process further and exploring additional modalities to enhance the model's adaptability in more complex learning environments.

CHAPTER 5

Adaptive Input-Side Components for Large Language Models

5.1 Related Work

5.1.1 Subword tokenizer

Subword tokenizers are widely applied in many large language models (LLMs) (Sennrich et al. 2015; Kudo 2018; Wang et al. 2021; Su 2023; Kudo and Richardson 2018), such as GPT-3 (Brown et al. 2020), BERT (Devlin et al. 2018), and T5 (Raffel et al. 2020). This is because subword tokenizers do not face the out-of-vocabulary issues that word-level tokenizers do. Unlike character-level tokenizers (Sutskever et al. 2011; Radford et al. 2017), they do not require processing longer sequences at the character level, which significantly increases the complexity of the models quadratically (Ling et al. 2015). Specifically, BPE (Byte Pair Encoding) (Sennrich et al. 2015) applies a compression algorithm (Gage 1994) to the task of word segmentation. Specifically, BPE (Byte Pair Encoding) (Sennrich et al. 2015) applies a compression algorithm (Gage 1994) to the task of word segmentation. First, the entire sentence is split into individual characters. Subsequently, the most frequent adjacent pairs of characters are consecutively merged until the desired vocabulary size is reached. Unlike BPE, which builds a vocabulary from smaller to larger units, Unigram (Kudo 2018) starts by preparing a large seed vocabulary. The vocabulary is pruned until it reaches the specified size. Unlike BPE, which builds a vocabulary from smaller to larger units, Unigram (Kudo 2018) starts by preparing a large seed vocabulary. Under the assumption that each subword occurs independently, it calculates and ranks how likely the marginal likelihood decreases when each subword is removed from the current vocabulary. Based on this ranking, the

vocabulary is pruned until it reaches the specified size. Similar to the Unigram framework and its assumptions, BytePiece (Su 2023) trims the vocabulary based on token frequency to the required size. In its initial text encoding phase, BytePiece converts the corpus into bytes, enhancing the training speed of the tokenizer and achieving a higher compression rate. Obviously, these tokenizers are data-driven (Boecking et al. 2022), with the generated vocabulary built based on the frequencies of word fragments (Bostrom and Durrett 2020). However, they cannot directly interact with LLMs to enhance model capabilities (Clark et al. 2022).

5.1.2 Learnable tokenizer

Unlike traditional tokenizers that offer a predefined and fixed vocabulary, learnable tokenizers (Islam et al. 2022; Godey et al. 2022; Thawani et al. 2023; Tay et al. 2021; Sreedhar et al. 2022; Bursztein et al. 2023) can be integrated with large language models into an end-to-end learning framework, resulting in task-specific tokenization to enhance the performance of LLMs. MANTa (Godey et al. 2022) introduces a gradient-based tokenization and pooling module that can be jointly learned with an encoder-decoder LLM (Beltagy et al. 2020). eChar (Thawani et al. 2023) also introduced a straightforward and effective word pooling method to achieve end-to-end tokenization. RETVec (Bursztein et al. 2023) embeds words into a high-dimensional vector with a pre-trained model to be robust against adversarial attacks. Neural (Islam et al. 2022) adapts the tokenization behavior to the downstream task after pre-training the tokenization by distilling from a language-specific subword tokenizer. However, such tokenizers have high requirements for the quantity and quality of the training data. If the data distribution is unbalanced or contains too much noisy data, it can lead to poor generalization of the tokenizer and negatively affect the performance of LLMs. For instance, Neural (Islam et al. 2022) requires a pre-training dataset curated with space-separated tokens and two carefully crafted heuristics to improve the ground label of the dataset. Therefore, stringent requirements for the quality of training data limit their widespread application.

In addition, some existing works have applied the concept of adaptive tokenizers in several fields, such as neural machine translation (Park et al. 2021), domain adaptation (Sachidananda

et al. 2021), and text generation (Liu et al. 2023c). These task-adaptive tokenizers integrate tokenizers generated from different data distributions, focusing on how to combine the task-specific tokenizer with the other one. In contrast, our proposed model, ADAT, is designed to learn a general tokenizer. Therefore, the purpose of ADAT is different from that of the aforementioned adaptive tokenizers.

5.2 Motivation

In recent years, large language models (LLMs) have emerged as foundational tools across a spectrum of applications in natural language processing (Biderman et al. 2023; Brown et al. 2020; Nawrot et al. 2022). From generating human-like text to enabling complex question-answering systems (Radford et al. 2018), LLMs have proven to be exceptionally versatile and capable. At the core of these models lies the tokenizer, a critical component that dictates how natural language is transformed into a format amenable to computational processing. The effectiveness of a tokenizer directly influences the model's ability to understand and generate language, thus playing a pivotal role in the overall performance of the LLM. Recognizing this integral relationship, it becomes essential to develop tokenizers that are not only effective but also dynamically adaptable to the evolving architectures of contemporary LLMs.

Current tokenization methods for large language models (LLMs) primarily include Byte Pair Encoding (BPE) (Sennrich et al. 2015), WordPiece (Wu et al. 2016), and Unigram (Kudo 2018), each serving to enhance text preprocessing by splitting it into manageable subwords. BPE focuses on reducing the dataset size through a greedy merging strategy based on character or subword frequency, effectively addressing the issue of infrequent words by splitting them into more common subunits. WordPiece, similar to BPE, starts with a base vocabulary and iteratively refines it by merging the most frequent pairs but incorporates a likelihood maximization step, which makes it slightly more context-aware than BPE. Unigram tokenization operates somewhat inversely, beginning with a large vocabulary and iteratively pruning it down based on token utility calculated through negative log likelihood, aiming to optimize the vocabulary against corpus loss metrics. Despite their efficiency in handling large vocabularies

and improving computational feasibility, these tokenization methods are typically fixed once developed and are not designed to adapt or learn from the model's evolving understanding of language during training.

While traditional tokenization methods have played a key role in improving the efficiency and performance of large language models (LLMs), they are generally designed independently of the model's learning dynamics. As a result, they remain static throughout training and are not adapted based on the model's performance or task-specific requirements. These approaches typically focus on vocabulary compression, which may hinder performance in complex language tasks that demand greater adaptability and contextual sensitivity.

Recent developments in end-to-end learnable tokenization attempt to address these shortcomings by integrating the tokenization process more tightly with model training. However, such methods often incur substantial computational overhead—e.g., gradient-based tokenization and learnable pooling—and lack the portability of traditional tokenizers, which are easily transferable across architectures.

In this work, we propose a model-coupled tokenization strategy that dynamically adapts to the evolving performance of the LLM. Specifically, we introduce an adaptive tokenizer that starts with a broad initial vocabulary and is progressively refined during training based on the model's perplexity signals. This enables the tokenizer to co-evolve with the model, maintaining alignment with its internal learning trajectory. Experimental results demonstrate that our adaptive tokenizer consistently improves training efficiency compared to conventional tokenization approaches, highlighting its potential to enhance the overall optimization process of LLMs.

5.3 Adaptive Tokenizers

For Large Language Models (LLMs), two critical aspects are accuracy and inference speed, both of which are deeply intertwined with the design of the tokenizer. Specifically, an optimal vocabulary V with maximum size N in an objective dataset D can be described by the

following optimization problem:

$$\min_{V} \textbf{Length}(D_o, V) - \lambda \textbf{Acc}(D, M, V), \quad |V| \leq N, \tag{5.1}$$

where M denotes the trained LLM, **Length** and **Acc** denotes the sequence length and accuracy (or the performance) using vocabulary V in dataset D, and the λ is a hyper-parameter to balance the two terms. This problem incorporates two primary objectives: given a fixed vocabulary size, the first is to maximize inference speed given a fixed vocabulary size, and the second is to maximize model accuracy. However, existing tokenizer schemes typically focus on one aspect over the other; for instance, traditional frequency-based schemes emphasize speed, while end-to-end approaches prioritize accuracy. To address this, we propose a method that optimizes both aspects. Therefore, we introduce an improved approach based on traditional frequency statistics, termed "adaptive tokenizers." This method aims to refine the balance between vocabulary efficiency and performance, thereby enhancing both the speed and accuracy of the model.

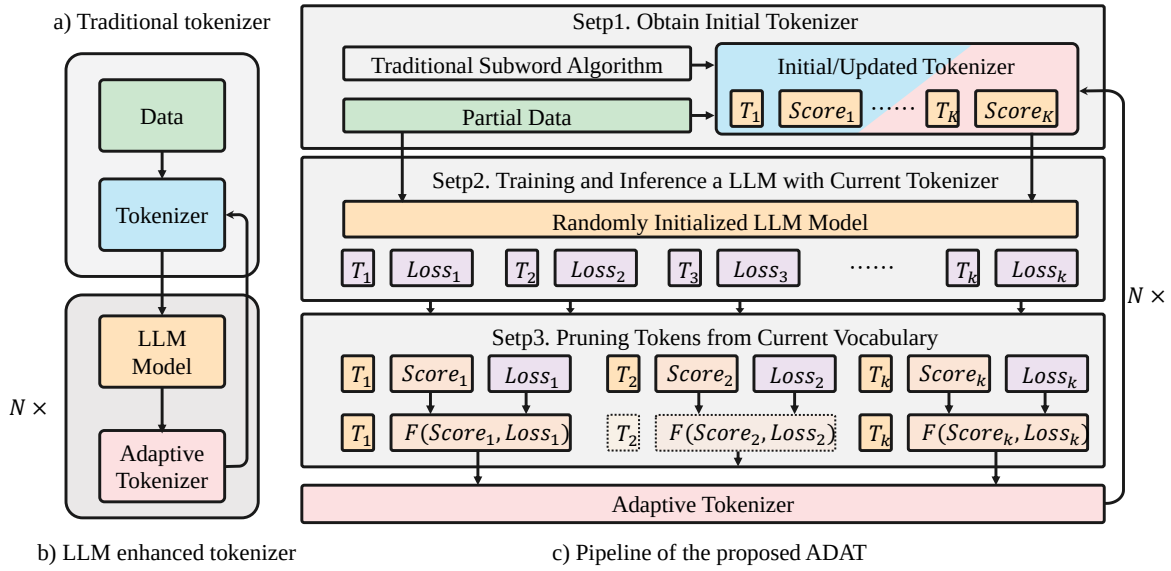


FIGURE 5.1. Illustration of the proposed ADAT pipeline. (a) The traditional tokenizer algorithm that directly extracts vocabulary from data. (b) The framework of the LLM-enhanced tokenizer, iteratively refining vocabulary based on model feedback. (c) Overview of ADAT, encompassing initial tokenizer acquisition, training and inference to derive token losses, token pruning based on scores and losses.

5.3.1 Unigram Model

Traditional tokenization methods generally fall into two categories: one approach, exemplified by Byte Pair Encoding (BPE) and WordPiece, starts with a small set of symbols and incrementally builds a larger vocabulary by merging the most frequent adjacent pairs. The other approach, typified by the Unigram method, begins with a large initial vocabulary which is progressively pruned based on token utility, a method found to generally offer superior performance due to its probabilistic foundation.

The Unigram model operates on the principle that the probability of a sentence is determined by the individual probabilities of its tokens. Here we briefly review the Unigram model. Initially, a large vocabulary V is established. This extensive initial set includes potentially every unique word or subword unit observed in the training corpus, ensuring that the vocabulary can cover all possible textual inputs. The process of refining the vocabulary involves several key steps repeated in cycles: 1. **Probability Estimation**: For each token x_i in the current vocabulary, we calculate its probability $p(x_i)$ based on its frequency of occurrence in the corpus. 2. **Loss Calculation**: We then compute the loss for each token, which is determined by how much the overall loss of the model would decrease if that token were removed. The loss function is calculated as:

$$\mathcal{L}_P(V) = \sum_{s=1}^{|V|} \log(p(X^{(s)}) = \sum_{i=1}^{|V|} \log(\sum_{\mathbf{x} \in S((X^{(s)})}^{P} (\mathbf{x})), \tag{5.2}$$

where $S(X)$ is a set of segmentation candidates built from the input sentence X, and $\mathbf{x} = (x_1, ..., x_K)$ is a subword sequence that $P(\mathbf{x}) = \prod_{i=1}^{K} p(x_i)$. The loss for each token x_i is then formulated as $\mathcal{L}_P(x_i) = \mathcal{L}_P(V) - \mathcal{L}_P(V - x_i)$. 3. **Token Pruning**: Tokens are ranked according to their calculated loss. Finally, a proportion of tokens contributing the most to increasing the overall loss is pruned from the vocabulary.

5.3.2 LLM-Enhanced Tokenization

To enhance the integration of Large Language Models (LLMs) with our tokenization process, we have developed a simple but effective refined method for calculating the loss associated with each token, incorporating insights directly from the LLM's performance metrics. This approach aims to optimize the tokenizer's vocabulary to better align with the LLM's understanding and generation of text. The framework is illustrated in Figure 6.1.

Our revised loss calculation method integrates the traditional Unigram model's frequency-based loss with a performance-driven loss derived from an LLM. Specifically, we first train an LLM M in the vocabulary D using a training dataset T. This model is designed to capture the linguistic nuances relevant to the tasks it is trained for, providing a robust framework for assessing token utility. For each token x_i in the vocabulary, we measure its individual contribution to the model's error using a cross-entropy loss function. The loss for each token is calculated as:

$$\mathcal{L}_M(x_i) = \sum_{x_i \in T} CE(M(x_{i-1}), x_i). \tag{5.3}$$

Here, CE represents the cross-entropy function, $M(x_{i-1})$ is the LLM's output given the previous token x_{i-1}, and x_i is the actual next token. This formula assesses how well the LLM predicts each token following its predecessor, providing a direct measure of each token's impact on model performance. Finally, The cross-entropy loss for each token is then combined with the traditional Unigram frequency-based loss. This combined loss ensures that tokens are evaluated not only on their frequency of occurrence but also on their actual contribution to the LLM's task performance. The final loss for pruning the vocabulary is given by:

$$\mathcal{L}(x_i) = F(\mathcal{L}_P(x_i), \mathcal{L}_M(x_i)), \tag{5.4}$$

where $F(\cdot, \cdot)$ is a function to balance the importance of frequency-based loss and LLM-driven loss, which will be discussed in experiments. Using this enhanced loss metric, we iteratively refine the vocabulary by pruning tokens that contribute the least to the combined loss, thus optimizing the vocabulary for both general language understanding and specific

task performance. This process continues until the vocabulary is compact enough to manage while still being comprehensive enough to support the LLM effectively.

5.3.2.1 Random sampling

In the training of Large Language Models (LLMs), ensuring that each token within the set vocabulary receives equal and substantial training is crucial to prevent loss bias due to uneven training. While iterating over all possible tokenizations of the training corpus would ideally provide the most comprehensive learning experience, this approach is computationally prohibitive due to the immense variety of potential segmentations. To address this, we adopt the classic Viterbi algorithm (Viterbi 1967) to perform randomized tokenization of the training data. This method allows for a diverse and balanced exposure of all tokens within the vocabulary to the learning process. The Viterbi algorithm efficiently determines the most probable tokenization paths through a probabilistic model of token occurrence, which significantly reduces the computational overhead compared to exhaustive methods. By leveraging this approach, our LLM can learn each token in the vocabulary more uniformly, enhancing the overall robustness and performance of the model.

5.3.2.2 Loss momentum

In the iterative process of training Large Language Models, maintaining the stability of the vocabulary is crucial to ensure consistent learning outcomes. To achieve this, we propose a momentum-based improvement for calculating the loss during each iteration. Specifically, the loss for iteration j of token x_i, denoted as $\mathcal{L}^j(x_i)$, is not solely computed based on the current data but is also weighted by the loss from the previous iteration L^{j-1}. This approach allows for a smoother convergence and mitigates fluctuations in training dynamics. The formula for updating the loss at each iteration is given by:

$$\mathcal{L}^j_{\text{momentum}}(x_i) = \beta \mathcal{L}^{j-1}_{\text{momentum}}(x_i) + \mathcal{L}^j(x_i), \tag{5.5}$$

where β is the momentum coefficient that controls the extent to which the previous loss influences the current loss. This methodology not only stabilizes the vocabulary updates

across iterations but also enhances the model's ability to generalize from the training data by reducing the variability in loss across successive training epochs.

5.4 Experiments

In this section, we outline the comprehensive experimental framework designed to assess the effectiveness of our proposed tokenizer, **Ada**ptive **T**okenizer (**ADAT**), in comparison to established methods such as Byte Pair Encoding (BPE) (Sennrich et al. 2015) and the Unigram model (Kudo 2018). These evaluations utilize the Pythia (Biderman et al. 2023) suite of models at various scales, leveraging a substantial corpus to ensure robust and generalizable results.

5.4.1 Experimental Setup

5.4.1.1 Model Framework

We deploy the Pythia framework (Biderman et al. 2023) for its lightweight design and adaptability across different computational setups. Pythia's flexibility facilitates reproducibility and consistent assessment of performance, making it an ideal choice for evaluating the scalability and efficiency of various tokenization strategies across model sizes of 70M, 160M, and 410M parameters.

5.4.1.2 Data Corpus

The study utilizes a substantial corpus extracted from The Pile (Gao et al. 2020), consisting of 56GB of raw data across 91 files. We specifically excluded subsets from DM_Mathematics and Github to ensure the relevance and quality of the data. The remaining data, approximately 16 billion tokens after a random shuffle, was tokenized using a Unigram (Kudo 2018) tokenizer with a vocabulary size of 50,000 tokens.

5.4.1.3 Baseline Methods

Our investigation compares four tokenization methods: Bytepiece (Su 2023), Byte Pair Encoding (BPE) (Sennrich et al. 2015), Unigram (Kudo 2018), and our proposed **ADAT**. These tokenizers were selected based on their established efficacy in handling large corpora and their theoretical implications for processing complex linguistic data.

5.4.1.4 Evaluation Metrics

The effectiveness of each tokenization strategy is rigorously evaluated using several metrics. These include Perplexity (PPL), which measures the model's predictive accuracy, and Compression Rate, assessing how efficiently the tokenization process reduces vocabulary size while preserving linguistic diversity. We calculate PPL for all models on PG19 (Rae et al. 2019) dataset. Specifically, we use its test set and the first 2048 tokens for each book. Furthermore, we use the Language Model Evaluation Harness (Gao et al. 2023a) to run five-shot evaluations on eight common language modeling benchmarks: Lambada (OpenAI) (Paperno et al. 2016), PIQA (Bisk et al. 2020), WinoGrande (Sakaguchi et al. 2019), ARC-Easy (Clark et al. 2018), ARC-Challenge (Clark et al. 2018), SciQ (Johannes Welbl 2017), LogiQA (Liu et al. 2020), and SST-2 (Socher et al. 2013; Wang et al. 2018), to provide a comprehensive insight into each method's capabilities.

By analyzing the impact of tokenization on model scalability and the influence of vocabulary size variations, this study aims to enhance our understanding of how tokenization strategies can optimize language models for efficiency and linguistic performance. The findings are expected to contribute significantly to the development of more robust and adaptable language processing tools, catering to a wide array of NLP applications. The Pythia models are trained using a corpus of 15B tokens, where training the 70M model consumes approximately 48 GPU hours with FlashAttention (Dao et al. 2022). The models used for loss calculation require additional 2 GPU hours.

5.4.2 Tokenization Methods Evaluation

In this section, we examine the effects of different tokenization strategies on the training effectiveness. The core objective is to explore how variations in the vocabulary, induced by different tokenization methods, affect model training and performance.

Baseline tokenization methods (BPE, Unigram, BytePiece) generate a vocabulary with 50,000 tokens directly with initial data, which is 1/10th of the training corpus (around 1.5 billion tokens). For the proposed ADAT, the initial vocabularies is created with Bytepiece or Unigram with 150,000 tokens. Then it is subsequently refined to 50,000 tokens through 5 iterative steps, aligning with the baseline size. In each step, a randomly initialized model is trained on about 0.3 billion tokens from the initial data. The obtained model then performs inference on a 0.1 billion token subset, during which token loss is calculated. This loss data, combined with token frequency using the formula $\frac{a}{\lambda \log(b+1)}$, informs vocabulary pruning. Ablation study on conbination manners will be disscussed in Sec. 5.4.6.4. The Pythia-70M model is selected due to its moderate size and efficiency, which help mitigate the complexities associated with larger model architectures. The model is initialized with random weights and subjected to a single epoch of training using pre-training data processed with vocabularies created through different tokenization methods. The baseline methods—BPE, Unigram, and BytePiece each produced a vocabulary directly from 1/10th of the training corpus (around 1.5 billion tokens), and is consite of 50,000 tokens. The size is same as the setting of Pythia (Biderman et al. 2023). The proposed tokenizer approach initially obtains larger vocabularies of 150,000 tokens using baseline methods, such as Unigram. Then it is methodically reduced to 50,000 tokens through 5 iterative steps, aligning with baseline sizes. In each step, a randomly initialized model is trained using one-fifth of the initial corpus, approximately 0.3 billion tokens. This model is then deployed for inference on a smaller subset of the corpus, consisting of 0.1 billion tokens, during which the loss associated with each token is recorded. This loss data is integrated with the token frequency with the manner of $\frac{a}{\lambda \log(b+1)}$ to serve as the basis for pruning the vocabulary.

TABLE 5.1. Comparison of Different Tokenization Methods in Terms of PPL and Accuracy Across Multiple Datasets.

Metric	**BPE**	**BytePiece**	**+ADAT(Ours)**	**Unigram**	**+ADAT(Ours)**
PPL	22.31	71.5	67.19(-4.31)	16.52	**6.97(-9.55)**
ARC-C	17.32 ± 1.11	18.69 ± 1.14	18.94 ± 1.15	$\mathbf{19.54} \pm 1.16$	18.46 ± 1.12
ARC-E	37.58 ± 0.99	33.80 ± 0.97	33.71 ± 0.97	37.04 ± 0.99	$\mathbf{40.57} \pm 0.99$
Boolq	61.28 ± 0.85	42.12 ± 0.87	$\mathbf{62.20} \pm 0.85$	53.06 ± 0.87	61.19 ± 0.85
Lambda	10.89 ± 0.43	8.80 ± 0.39	13.55 ± 0.48	17.27 ± 0.53	$\mathbf{17.97} \pm 0.52$
LogiQA	23.04 ± 1.65	20.28 ± 1.58	22.27 ± 1.63	23.20 ± 1.66	$\mathbf{24.22} \pm 1.70$
PIQA	59.25 ± 1.15	57.83 ± 1.15	56.96 ± 1.16	$\mathbf{60.50} \pm 1.14$	59.93 ± 1.14
SciQ	66.60 ± 1.49	54.01 ± 1.58	51.90 ± 1.58	68.10 ± 1.47	$\mathbf{72.40} \pm 1.44$
SST-2	51.26 ± 1.69	49.08 ± 1.69	50.23 ± 1.69	49.77 ± 1.69	$\mathbf{54.24} \pm 1.69$
Winogrande	49.96 ± 1.41	50.31 ± 1.41	49.41 ± 1.41	51.46 ± 1.40	$\mathbf{51.62} \pm 1.40$
Avg. (%)	41.91	37.21	$39.91(+2.70)$	42.22	$\mathbf{44.51}(+\mathbf{2.29})$

The models underwent a comprehensive evaluation using multiple benchmark datasets to assess their post-training linguistic processing capabilities. As illustrated in Table 5.1, our method achieves its best performance when initialized with the Unigram vocabulary, recording a score of 44.51. This score represents a considerable improvement of 2.29 points over the standard Unigram model and surpasses the BPE model by 2.6 points. Additionally, our approach shows a notable enhancement of 2.7 points when utilizing BytePiece as the initial vocabulary. Although the BytePiece vocabulary generally exhibits inferior baseline results, our method effectively elevates its performance, indicating robustness across both high-quality (Unigram) and lower-quality (BytePiece) vocabularies. These results not only affirm the efficacy of our method but also demonstrate its adaptability to different initial conditions, thereby validating its potential for broad adeptness on diverse vocab initialization.

5.4.3 Scalability

This subsection explores the scalability of the proposed tokenization methods across different configurations of the Pythia model, specifically examining their performance across models with 70M, 160M, and 410M parameters. Scalability in this context refers to the ability of the tokenization methods to maintain or enhance performance as model size increases, an essential attribute for practical applications in machine learning and natural language processing.

TABLE 5.2. Evaluation on Three Different Model Scales: 70M, 160M, and 410M.

Metric	**70M**		**160M**		**410M**	
	Unigram	**ADAT**	**Unigram**	**ADAT**	**Unigram**	**ADAT**
PPL	16.52	**6.97(-9.55)**	13.97	**6.19(-7.78)**	10.92	**5.78(-5.14)**
ARC-C	**19.54**±1.16	18.46 ± 1.12	18.69 ± 1.14	**18.94**±1.15	**20.82**±1.19	19.29 ± 1.21
ARC-E	37.04±0.99	**40.57**±0.99	39.52 ± 1.00	**42.87**±1.01	44.65 ± 1.02	**46.69**±1.03
Boolq	53.06 ± 0.87	**61.19**±0.85	**58.56**±0.86	57.68 ± 0.86	54.80 ± 0.87	**60.81**±0.87
Lambda	17.27 ± 0.53	**17.97**±0.52	19.06 ± 0.55	**25.02**±0.60	27.81 ± 0.62	**28.94**±0.66
LogiQA	23.20 ± 1.66	**24.22**±1.70	**25.65**±1.71	25.04 ± 1.65	23.20 ± 1.66	**24.32**±1.68
PIQA	**60.50**±1.14	59.93 ± 1.14	60.83 ± 1.14	**61.86**±1.14	63.38 ± 1.12	**64.61**±1.11
SciQ	68.10 ± 1.47	**72.40**±1.44	72.10 ± 1.42	**79.60**±1.28	80.70 ± 1.25	**83.50**±1.14
SST-2	49.77 ± 1.69	**54.24**±1.69	52.06 ± 1.69	**52.78**±1.69	50.69 ± 1.69	**54.71**±1.69
Winogrande	51.46 ± 1.40	**51.62**±1.40	49.88 ± 1.41	**50.69**±1.41	52.41 ± 1.40	**52.93**±1.41
Avg	42.22	**44.51**	44.04	**46.05**	46.50	**48.42**

We examine the scalability of a proposed tokenization method that tailors the vocabulary to model size, unlike the static Unigram method which maintains a consistent vocabulary across various model capacities. We evaluate this scalability across models with 70M, 160M, and 410M parameters to understand its impact on performance. The scalability of the tokenization methods is tested using the Pythia framework configured at three different levels of computational complexity: 70M, 160M, and 410M parameters. For each model size, our method generates an optimized vocabulary specific to that configuration, allowing us to analyze how adjustments in vocabulary affect performance as model size increases. In contrast, the Unigram method employs a uniform 50,000-word vocabulary across all sizes, serving as a baseline. We gauge performance using Perplexity (PPL) and scores from benchmark datasets designed to assess the linguistic capabilities of each model under various conditions, providing insights into the efficiency and adaptability of the tokenization methods at scale. For training larger models, the same volume of data will lead to insufficient warm-up, potentially resulting in a slight decline in the accuracy of loss computations used for determining token priority. As a result, we increase the data volume for training the loss calculation model according to the size of model.

The results of this experimental framework, as presented in Table 5.2, indicate substantial performance variations across different model sizes employing varied tokenization strategies.

Specifically, average performance scores across all evaluated metrics demonstrate consistent improvements with increases in model sizes: ADAT achieves a score of 44.51 in the 70M model, significantly surpassing Unigram's 42.22; 46.05 compared to 44.04 in the 160M model; and 48.32 versus 46.50 in the 410M model. These findings highlight the superior efficacy of ADAT in managing diverse model volumes compared to the more static approach of Unigram, which exhibits limited scalability with increasing model size. Remarkably, the performance of the 70M model using ADAT exceeded that of the Unigram on the 160M model by nearly 5%, illustrating the substantial enhancement and ability of our method to bridge a parameter gap of over double. Furthermore, the performance of our 160M model approaches that of the 410M model, emphasizing the robust adaptability of the ADAT method across varying computational scales.

TABLE 5.3. Cross-Model Adaptability of Vocabularies. Performance of vocabularies obtained from different models when applied to models of other sizes.

Model Size	**Unigram**	**70M-Model Vocabulary**	**410M-Model Vocabulary**
70M	42.22	44.51	42.62
160M	44.04	45.03	45.83
410M	46.50	47.66	48.42

5.4.4 Cross-Model Adaptability

This experiment evaluates the adaptability of vocabularies generated by our proposed tokenization method across various configurations of the Pythia model, particularly assessing whether vocabularies optimized for one model size can effectively scale to others. We initially create vocabularies using the 70M and 410M configurations. These are then used to train models at both scales to evaluate performance in downstream tasks, allowing us to assess how vocabularies designed for a specific size perform when applied to both smaller and larger models, thus examining their cross-model adaptability.

Table 5.3 illustrates the cross-model adaptability of vocabularies across different model sizes. By applying vocabularies derived from different model sizes to various models, we observe that vocabularies generated by the 70M and 410M models surpass the performance of the

standard Unigram model. This indicates the adaptability of the ADAT vocabularies across different model sizes. Furthermore, we note that the vocabulary from the 410M model achieves only a marginal improvement of 0.4 when applied to the 70M model, significantly less than the 2.29 increase afforded by the 70M model's vocabulary. This suggests that the vocabularies selected by ADAT possess a strong capacity for targeted optimization, enabling the selection of tokenization strategies that are specifically tailored to the characteristics of different models.

5.4.5 Model and Vocabulary Size

The experiment aims to assess the impact of different tokenizer strategies on model performance across two vocabulary sizes, comparing a standard 50,000-token set with a reduced 30,000-token set. We utilize two configurations of the Pythia model—70M and 160M—to explore how vocabulary size influences model efficiency. Each model is tested using both the standard Unigram and our proposed tokenization method. This setup allows us to directly observe the effects of reduced vocabulary sizes on the performance dynamics, providing insights into how smaller vocabularies impact the computational efficiency and efficacy of language models.

The experimental results, as presented in Table 5.4, support the hypothesis that changes in vocabulary size can significantly affect model performance, with different impacts observed across varying model sizes. For large language models, it is common for models of vastly different sizes to utilize vocabularies of similar or identical sizes (Biderman et al. 2023; Touvron et al. 2023). This practice can lead to issues of performance or efficiency. Our method offers a more effective solution by tailoring tokenization strategies to the specific sizes of models, thereby mitigating these challenges. For the 70M model, ADAT achieved a notable improvement in accuracy from 42.22 to 44.51 (+2.29) and a substantial reduction in perplexity from 16.52 to 6.97 when using a 50,000-word vocabulary. Even with a reduced vocabulary of 30,000, ADAT enhances accuracy to 43.33 (+2.40) and decreases perplexity to 7.38, suggesting robustness against vocabulary size reduction. In contrast, the 160M model, which has a greater parameter capacity, also shows improvements with ADAT: accuracy

TABLE 5.4. Impact of Vocabulary Size on Model Performance Across Different Model Sizes.

Model Size	Vocabulary Size	Tokenization Method	Accuracy	Perplexity (PPL)
70M	50,000	**Unigram**	42.22	16.52
		ADAT	44.51(+2.29)	6.97
	30,000	**Unigram**	40.93	32.53
		ADAT	43.33(+2.40)	7.38
160M	50,000	**Unigram**	44.04	13.97
		ADAT	46.05(+2.01)	6.19
	30,000	**Unigram**	43.08	15.21
		ADAT	45.26(+2.14)	6.11

increases from 44.04 to 46.05 (+2.01), and perplexity drops sharply from 13.97 to 6.19 for the 50,000 vocabulary size. With a 30,000 vocabulary, accuracy still increases to 45.26 (+2.14), and perplexity remains low at 6.11, underscoring that larger models not only handle vocabulary reductions well but also benefit significantly in terms of computational efficiency and model quality.

5.4.6 Ablation Study

This ablation study is structured into three distinct parts to explore how variations in the inference corpus size used for calculating token loss, initial vocabulary sizes, momentum strategy, and balance function $F(a, b)$ influence the efficacy of our proposed tokenization method on a 70M parameter model. More ablation results can be referred to in the supplementary.

5.4.6.1 Corpus Size used in Loss Calculation

We conducted an experimental study to investigate the effects of varying corpus sizes on the accuracy of token loss calculations. The experiment assessed the performance of models trained on different sizes of inference data, specifically 1 million (1M), 10 million (10M), and 100 million (100M) tokens. The results are summarized in the table 5.5.

These results indicate a direct correlation between the volume of the corpus used during the loss calculation phase and the overall accuracy of token loss estimates. When smaller

TABLE 5.5. Ablation Study Results on Inference Data Tokens, Initial Vocabulary Size, Momentum, and different choice of Balance $F(a, b)$.

Infer Data Volume		**Initial Vocabulary Size**		**Momentum**		**Balance** $F(a, b)$	
Tokens	**Acc.**	**Init Size**	**Acc.**	**Methods**	**Acc.**	**Methods**	**Acc.**
1M	43.13	75k	43.42	Unigram	42.20	$a - \lambda b$	42.70
10M	43.74	100k	43.78	ADAT+By	44.51	$log(a) - \lambda b$	43.23
100M	44.51	150k	44.51	-w/o Mnt.	43.16	$a/\lambda \log(b + 1)$	44.51

corpora are used, a significant number of tokens are absent, resulting in numerous instances where loss values cannot be computed. Furthermore, the precision of token loss estimations tends to decrease with smaller data sets. Even with just 1M tokens, there was a noticeable improvement over the baseline unigram vocabulary accuracy of 42.22. This enhancement became more pronounced with larger data volumes, reaching an increase of 44.51 in accuracy with 100M tokens.

5.4.6.2 Initial Vocabulary Size

This segment of our study assesses the effect of different initial vocabulary sizes on model performance. Adjusting the vocabulary from a baseline of 150,000 tokens to either 100,000 or 75,000 tokens, we explore the influence of vocabulary scale on training outcomes. The results, detailed in Table 5.5, illustrate the trade-offs associated with varying vocabulary sizes.

From the experiment, it is evident that models equipped with a larger initial vocabulary of 150,000 tokens tend to achieve lower Perplexity and higher Accuracy, indicating a robust ability to capture diverse linguistic nuances that significantly enhance performance. In contrast, reducing the vocabulary size to 75,000 tokens results in increased perplexity and decreased accuracy, highlighting a potential compromise in linguistic detail that adversely affects model functionality, especially in complex linguistic scenarios.

5.4.6.3 Momentum Strategy

This experiment evaluates the impact of incorporating a momentum strategy into our tokenization algorithm's vocabulary pruning process. The performance of vocabularies pruned under both the momentum and non-momentum conditions is directly compared in Table 5.5.

The results in Table 5.5 demonstrate that the momentum approach significantly enhances model accuracy, with a notable improvement from 43.16% to 44.51% in the ADAT method with Unigram initialization vocabulary when momentum is applied. Similarly, the Unigram method shows a baseline performance of 42.20% accuracy. These results confirm that integrating momentum allows for a more refined pruning process by effectively utilizing historical performance data to make more informed decisions, thereby preserving valuable linguistic features.

5.4.6.4 Balance Strategy

In this ablation study, we investigate various functions to balance token frequency and loss value in our tokenization algorithm. The primary objective is to adhere to the principle that tokens with higher frequency and lower loss should be assigned higher priority. Given the significant difference in their magnitudes, we explored subtraction and division methods. We evaluated three functions, detailed in Table 5.5. Here, λ is a scaling factor introduced to adjust the balance between frequency and loss, and we set it as 1 in practice.

The results demonstrate that the subtraction methods $a-\lambda b$ and $a-\lambda \log(b)$ yielded accuracies of 42.70 and 43.23 respectively. These results indicate relatively poor performance, even with the logarithmic transformation applied to the score $\log(a) - \lambda b$. This underperformance is likely attributable to the significant disparity in the magnitudes of frequency and loss values, which the subtraction methods struggle to reconcile effectively. In contrast, the division method $\frac{a}{\lambda \log(b+1)}$ significantly outperformed the subtraction approaches with an accuracy of 44.51. This superior performance suggests that the division method more naturally balances the influence of frequency and loss by scaling the loss logarithmically before the division, thereby mitigating the impact of numerical range discrepancies. This method's ability to

integrate frequency and loss without requiring additional adjustments for scale disparities results in more stable and effective prioritization of tokens.

5.4.7 Additional analysis experiments

5.4.7.1 Analysis of the Compute Costs

To prove that the proposed method is feasible in practice. We analyzed the empirical runtime introduced by ADAT. To measure runtime, we used 8 NVIDIA A100 GPUs, an Intel 8378A CPU, and PyTorch 2.1.2 with CUDA 12.1. The tokenizer optimization involves 5 epochs, where each epoch consists of training the LLM on a 0.3B corpus, followed by inference on a 0.1B corpus, and concludes with a vocabulary pruning step (90 seconds for a 100K tokens vocabulary). Therefore, the total computational cost of the tokenizer optimization process is calculated as:

$$5\times(0.3B \text{ training}+0.1B \text{ inference}+ \text{ pruning time}) = 1.5B \text{ training}+0.5B \text{ inference}+450s.$$

ADAT introduces an additional training cost of 1.5B tokens and an inference cost of 0.5B tokens, along with minimal vocabulary pruning time. Compared to the hundreds of billions or even trillions of tokens required for LLM training, these computational costs are negligible. As shown in Table B.1, the full-scale training of the LLM incurs significantly higher computational costs. For instance, when training models with a 16B and 60B corpus, the tokenizer optimization accounts for only 4.17% and 1.04% of the total training time, respectively. The Pythia-70M model takes 510 GPU hours to train with the full Pile dataset (Biderman et al. 2023), and exceeds the tokenizer optimization's computational cost by over 255 times. Therefore, the additional computational cost introduced by our method is minimal, making it feasible in practice.

TABLE 5.6. Comparison of ADAT Optimization Runtime and Training Time Across Different Model Sizes.

	Tokenizer Opt.	**Training on 16B**	**Training on 60B**	**Pythia Report**
Runtime	2 GPU hours	48 GPU hours	192 GPU hours	510 GPU hours

5.4.7.2 Compression Rate results.

The compression rate of the proposed model and baselines across Pythia model[1]-70M models are shown in Table 5.7.

TABLE 5.7. Performance comparison of different tokenization methods.

Metric	BPE	BytePiece	ADAT+By	Unigram	ADAT+U
Compression Rate	4.38	4.81	4.98	3.96	2.91

5.4.7.3 Analysis of Differences Between Vocabularies

To illustrate the differences between the vocabulary results obtained by ADAT and those from unigram, we calculated the overlap ratio of the two token sets as follows:

$$Ratio = |A_{vocab} \cap B_{vocab}|/|A_{vocab} \cup B_{vocab}|. \tag{5.6}$$

Where $\cap$ and $\cup$ denote the intersection and union of two sets, respectively, we present the overlap ratios between the tokenizers obtained using ADAT and Unigram on different models under the same vocabulary size setting, as well as the overlap ratios between these tokenizers themselves in Table 5.8. As shown in Table 1, there are significant differences between the vocabulary generated by ADAT and that generated by unigram. This disparity arises because ADAT is not entirely data-driven in its vocabulary generation process. By incorporating the loss from the LLM, ADAT can simultaneously focus on enhancing the performance of the LLM. Additionally,the overlap ratio between ADAT-160M and ADAT-410M is higher than that between ADAT-160M and ADAT-70M. This also indirectly explains why, as shown in Table 5.3, the tokenizer generated by ADAT-410M is more suitable for Pythia-160M compared to the tokenizer generated by ADAT-70M.

To compare the vocabulary obtained by ADAT with the initial vocabulary, we sorted the 100k initial vocabulary by score in descending order and calculated the percentage of tokens,

[1]with Apache-2.0 license

TABLE 5.8. Comparison between Vocabulary of ADAT and Unigram.

Unigram vs.		ADAT	
Model size	**Ratio**	**Model size**	**Ratio**
ADAT-70M	0.18	70M | 160M	0.71
ADAT-160M	0.19	70M | 410M	0.69
ADAT-410M	0.21	160M | 410M	0.84

pruned by ADAT, that fall into each score interval. As shown in Table 5.9, ADAT-70M-50K tokens are most densely distributed not in the 0-25% interval, indicating that ADAT relies not only on token frequency but also on the prediction difficulty of tokens in the LLM during training.

TABLE 5.9. Comparison with the Initial Vocabulary (100k size): Percentage of Tokens Falling into Each Score Interval.

Vocabulary	**0-25%**	**25-50%**	**50-75%**	**75-100%**
Unigram-50k	54.75%	23.95%	8.11%	13.19%
ADAT-70M-50k	21.43%	31.60%	32.72%	14.24%

5.4.7.4 Impact of Training Epochs

We investigate the effects of varying the number of training epochs for developing the vocabulary. Specifically, the model is trained using vocabularies that have been developed over 3, 5, and 7 epochs. The model is initialized with random weights for each training session to evaluate the immediate impact of the epoch variation.

As shown in Table 5.10, the results from varying the number of training epochs suggest a clear relationship between training duration and vocabulary efficacy. Before the epoch reaches a certain value (5 epochs), increasing the number of epochs benefits accuracy, indicating a more refined and effective vocabulary. However, accuracy does not significantly improve with further increases in the number of epochs. This suggests that, given the specified vocabulary size, ADAT can quickly and efficiently learn an appropriate vocabulary without the need for prolonged training over many epochs. Therefore, in our experiments, the default number of training epochs is set to 5.

TABLE 5.10. Impact of Training Epochs on Model Performance.

Training Epochs	Accuracy
3	43.67
5	44.51
7	44.58

5.4.7.5 Expanded Analysis on Infer Data Volume Tokens and Initial Vocabulary Size

we have expanded the analysis of 'infer data volume tokens' and 'initial vocabulary size'—both variables are explored within and beyond settings in Table 5.5. The expanded results are displayed in the table 5.11. We observe that an inference data volume of 100M tokens is sufficient, with larger volumes yielding only marginal improvements. Regarding the initial vocabulary size, increasing it to 150K is important to enhance performance. However, when it increases to 200K, the score shows almost no improvement, indicating that the 150K vocabulary likely already includes most of the potential final candidate tokens. Therefore, further increasing the initial vocabulary size will not bring additional benefits.

TABLE 5.11. Results for different Infer Data Volume Tokens and Initial Vocabulary Sizes.

Infer Data Volume Tokens	75K	100K	150K	200K
1M	42.89	43.07	43.13	43.20
10M	43.19	43.39	43.74	43.77
100M	43.42	43.78	44.51	44.56
1000M	43.45	43.83	44.53	44.57

5.5 Conclusion

In this paper, we have presented a novel approach to tokenizer design that integrates key aspects of both accuracy and inference speed, addressing the inherent limitations found in existing tokenizer schemes. By innovating beyond the traditional frequency-based and end-to-end methodologies, our adaptive tokenizer framework strategically optimizes vocabulary construction, ensuring both rapid processing and high precision in language modeling tasks. Our results demonstrate that the adaptive tokenizer significantly enhances the performance of

Large Language Models (LLMs) across various benchmarks, providing a balanced solution that does not sacrifice speed for accuracy or vice versa. Future work will focus on refining these adaptive tokenization techniques, exploring further integration with neural network architectures, and expanding their applicability to a broader range of languages and complex linguistic tasks.

CHAPTER 6

Adaptive Output-Side Components for Vision Transformers

6.1 Related Work

6.1.1 Natural language prompting and visual prompting

Prompt-based learning in natural language processing is a new paradigm that adapting frozen, pre-trained language models to diverse downstream scenarios (Liu et al. 2023a; Ma et al. 2023). Inspired by the purpose of prompting that adapting a frozen pre-trained model by modifying the data space, Visual Prompting (Bahng et al. 2022) designs a specific prompt for each task and applies it to the input. In addition, Visual-Prompting Tuning (Jia et al. 2022) are specially proposed for Transformer pre-trained model, adding learnable parameters into Transformer's first or every input layer. The new paradigm "prompting-based learning" in natural language processing and vision enables pre-trained models to be easily applied to various downstream tasks while consuming minimal memory space and computational resources (Schick and Schütze 2020; Sohn et al. 2023).

6.1.2 Label mapping for visual prompting

Label mapping (LM) are applied in various machine learning tasks such as transfer learning (Sang et al. 2022; Wu and He 2020), domain adaptation (Kurmi and Namboodiri 2019; Zhao et al. 2021; Le et al. 2021), and instance segmentation (Hu et al. 2023). Unlike the aforementioned tasks where LM is integrated into the model, trainable parameters are only present in the input, and the entire pre-trained model is inaccessible in visual prompting (Bahng et al. 2022). Consequently, LM has to independently assign source labels to target labels. The

existing label mapping methods in visual prompting are not yet mature. Firstly, random-based label mapping strategies are employed in visual prompt methods (Bahng et al. 2022; Li et al. 2023b; Huang et al. 2023). In addition, frequency-based mapping (Tsai et al. 2020; Chen et al. 2021c; Chen et al. 2023) maps target labels to source labels by the prediction frequencies of pre-trained model on the target dataset. These methods overlook the potential to alleviate the training burden on input prompts during label mapping. In contrast, our proposed method minimizes the cost when mapping labels.

6.1.3 Optimal transport in machine learning

Optimal Transport (OT) is a mathematical framework of seeking the most cost-efficient way to transport mass from one distribution to another (Villani et al. 2008). After the emergence of new methods addressing high computational cost issues (Bonneel et al. 2015; Bai et al. 2023), OT has found extensive application across various domains in machine learning, such like Generative Adversarial Networks (Arjovsky et al. 2017; Salimans et al. 2018; Korotin et al. 2022), diffusion models (Gu et al. 2023; Lipman et al. 2022), and domain adaptation (Courty et al. 2017; Damodaran et al. 2018; Gu et al. 2022; Courty et al. 2016). WGAN (Arjovsky et al. 2017) replaces the KL divergence in traditional GANs with the Wasserstein distance. To the best of our knowledge, Optimal Transport has not yet been employed to tackle the Visual Prompting challenge.

6.2 Motivation

As the amount of data and computational resources increases, large models trained on extensive datasets are able to effectively capture general features (Dosovitskiy et al. 2020b; Chen et al. 2021b), which present impressive generalization ability on adapting to other datasets. Following this, the ‘pretraining and finetuning’ paradigm—finetuning a pre-trained model on unseen datasets— has achieved notable success and been with widespread application (Bao et al. 2021). However, this approach encounters certain limitations in practical applications. As pre-trained models are often typically large (e.g. 632 million parameters in ViT-Huge),

there is nearly no chance to deploy several models fine-tuned on different datasets for a variety of scenarios, which would involve unacceptable memory requirement for storing these models with large volume. Moreover, fully finetuning them for each task incurs extensive computational costs, expanding to a broader range of downstream tasks (Pfeiffer et al. 2020).

Drawing inspiration from the outstanding performance of prompting in the NLP domain, Visual Prompting (VP) is proposed to solve this problem (Bahng et al. 2022). VP reinterprets the problem of adapting a frozen, pre-trained model to an unseen dataset as a data-space adaptation by transforming both the input and the output. Specifically, a small amount of learnable perturbations is introduced into the input. On the output side, label mapping provides a function to map source labels to target labels. This strategy introduces two significant advantages. Firstly, VP considerably reduces the number of parameters requiring tuning. More importantly, it maintains the pre-trained model unchanged, enabling various scenarios to share the same pre-trained model identically. Consequently, this leads to a substantial reduction in both memory requirements and computational expenses.

In visual prompting, only a limited number of learnable parameters are introduced, in the form of prompts that are incorporated with images before the first layer. Therefore, how to efficiently utilize the adaptation capability provided by prompts is a research topic worth exploring. Some approaches aim to enhance the adaptation capability of prompts through input transformation. For instance, DAM-VP (Huang et al. 2023) offers unique prompts for each subset rather than a universal prompt for all images, aiming to address the challenges posed by highly diverse downstream datasets. Nonetheless, this method necessitates additional training data for clustering and leads to an increase in the number of parameters.

While significant attention has been dedicated to innovations at the input, the transformation applied to the output plays an equally pivotal role within the VP framework. Often referred to as label mapping, this transformation is crucial for aligning the pre-trained model's source domain labels with the target domain's labels. Label mapping serves to correlate the model's source domain classification outcomes with meaningful labels of the target domain. Several methodologies for label mapping have been explored. For example, random-based label mapping is employed in Prom-PATE (Li et al. 2023b). Hard-coded label mapping is utilized

in the output transformation of VP (Bahng et al. 2022), simply hard-coding the first n outputs of the model layer directly to the task data categories. Apparently, random-based methods do not consider the rationality of the mapping at all. Such random methods completely discard the information of labels. Another kind of label mapping method is frequency-based label mapping (Chen et al. 2023; Tsai et al. 2020; Chen et al. 2021c), which relies on the source model's prediction frequencies for target data points. Furthermore, ILM (Chen et al. 2023) incorporates the relationships of label mapping as trainable parameters into its training process, aiming to discover interpretable label mapping results. These methods just heuristically rely on the voting of samples to produce mapping results.

The choice of label mapping strategy plays a critical role in determining the effectiveness of visual prompting. We argue that an ideal strategy should minimize the extent to which the model must adjust its output distribution to align with target labels, thereby improving the efficiency and overall performance of prompt optimization. A well-designed mapping can reduce unnecessary changes in the output space, enabling more efficient use of prompt learning capabilities and yielding better results. However, existing label mapping approaches have largely overlooked this perspective and fail to improve prompting effectiveness accordingly.

In this paper, we introduce the concept of minimal transfer load for prompt design and define label mapping as a type of partial optimal transport problem, where the transfer load is delineated through distribution migration. To solve this optimal transport problem, we propose a novel cost matrix that defines the mapping cost between labels from the perspective of transfer load. Then, we derive a label mapping strategy **OTLM** (**O**ptimal **T**ransport-based **L**abel **M**apping) that minimizes the transfer load of output distribution (shown in Figure 6.1). Moreover, we analyze the existing frequency-based label mapping method from the optimal transport perspective, providing insights into the advantages of our approach. The results of the experiment, conducted across multiple datasets and model architectures, showcase the substantial performance improvements brought about by the OTLM. Additionally, we conduct interpretative experiments to understand the effectiveness of the proposed method.

6.3 Preliminaries

6.3.1 Visual Prompting (VP) Framework.

We formally introduce the concepts of Visual Prompting (VP) and Label Mapping (LM), which lay the foundational framework for our proposed methodology.

Visual prompting employs a pre-trained model $\mathcal{M}$, with parameters θ_s, initially trained on a source dataset $\mathcal{D}_S$ with labels $\mathcal{Y}_S \subseteq \mathbb{R}^{n_s}$, where n_s is the number of class of source dataset. VP aims to adapt $\mathcal{M}$ to a target task defined by a new dataset $\mathcal{D}_T$ with labels $\mathcal{Y}_T$, without modifying θ_s. This is achieved through the introduction of the learnable prompt θ_p, which is optimized to manipulate the input data that satisfies the target task performance. Given a target data point $\mathbf{x}_t \in \mathcal{D}_T$, the prompted input $\mathbf{x}'((\theta_p))$ is generated as:

$$\mathbf{x}'(\theta_p) = h(\mathbf{x}_t, \theta_p), \tag{6.1}$$

where $h(\cdot, \theta_p)$ represents the transformation function incorporating the prompt θ_p into the target input $\mathbf{x}_t$. The objective is to optimize θ_p such that $\mathcal{M}(\mathbf{x}')$ aligns with the target labels $y_t \in \mathcal{Y}_T$, effectively reprogramming $\mathcal{M}$ for the target task.

Label mapping is a critical component of VP, facilitating the translation of model predictions from the source label space $\mathcal{Y}_S$ to the target label space $\mathcal{Y}_T$. The LM progress is to find a mapping function $\mathcal{F}_{LM} : \mathcal{Y}_S \to \mathcal{Y}_T$ based on specific principles. With the help of label mapping function $\mathcal{F}_{LM}$, the optimization paradigm can be established. The optimization objective can be expressed as:

$$\min_{\theta_p} \mathbb{E}_{\{\mathbf{x}_t, y_t \in \mathcal{D}_T\}} [\mathcal{L}\left(\mathcal{F}_{LM}(\mathcal{M}(h(\mathbf{x}_t, \theta_p))), y_t\right)], \tag{6.2}$$

where $\mathcal{L}(\cdot)$ denotes a loss function measuring the discrepancy between the mapped predictions $\mathcal{F}_{LM}(\mathcal{M}(\mathbf{x}'(\theta_p)))$ and the true target labels $y_t \in \mathcal{Y}_T$. This optimization is performed only on the prompt parameters θ_p, leaving the original model parameters $\theta_s \in \mathcal{M}$ untouched.

6.4 Methodology

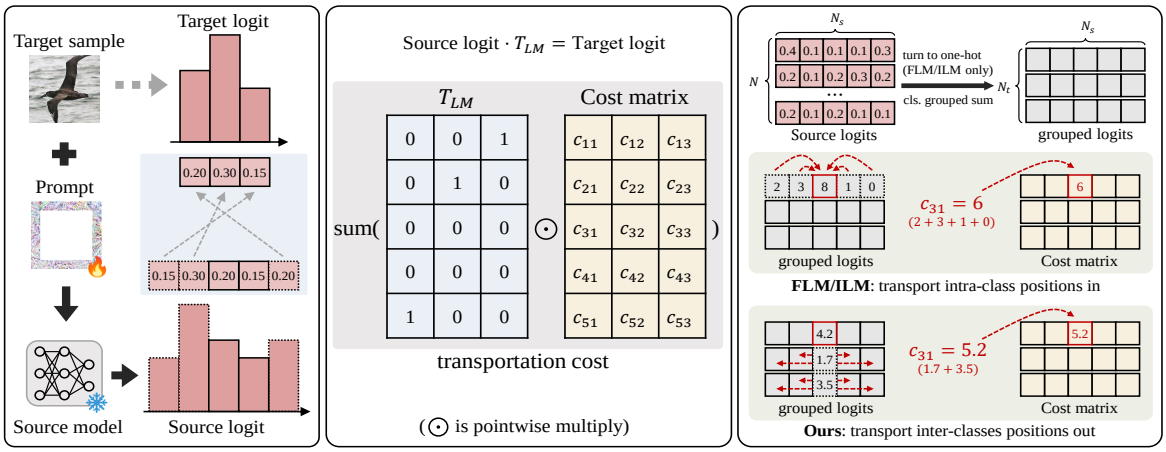
The optimization burden hinges critically on the disparity between the initial and final states of the feature distributions influenced by visual prompts. It is imperative to recognize that the initial state of these features is solely determined by the intrinsic characteristics of the data and the model architecture. Consequently, for a given target data distribution, the magnitude of the optimization burden is directly influenced by the final state to which the system aspires. As delineated in the preceding sections, this final state is fundamentally contingent upon the label mapping function $\mathcal{F}_{LM}$, which governs the transition of prompted output distributions $\hat{f}_i$ towards the desired targets y_i.

Given the pivotal role of $\mathcal{F}_{LM}$ in determining the optimization burden, it is logical to explore methodologies for selecting or designing an appropriate mapping function that can significantly alleviate this burden. To this end, we propose an approach grounded in the principles of optimal transport to conceptualize label mapping.

6.4.1 Label Mapping from an Optimal Transport Perspective

Our objective is to devise a label mapping strategy, denoted as $\mathcal{F}_{LM}$, aimed at substantially minimizing the necessity for feature transfer. In this framework, a prompt θ_p is meticulously optimized to adjust $f_i = \mathcal{M}(\mathbf{x}'_i(\theta_p)) \in \mathbb{R}^{n_s}$, ensuring f_i closely mirrors the target feature $\hat{f}_i \in \mathbb{R}^{n_t}$, i.e., the one-hot representation of label $y_i \in \mathcal{Y}_T$. The selection of a label mapping strategy introduces variable levels of operational challenge for the prompt, emphasizing that the optimal label mapping is one that requires the least deviation in feature distribution between the source and target. To achieve this, we encapsulate the concept of label mapping within an optimal transport, setting the stage for a strategic transformation that aligns with our goal of efficiency and precision in feature adaptation.

Considering that label mapping facilitates the transition from the source label distribution $\mathbf{p}$ with n_s classes to the target label distribution $\mathbf{q}$ with n_t classes. $n_s > n_t$ is introduced as a hypothesis. We leverage empirical sampling from input examples $\mathbf{x}'$ to define these



(a) Visual prompting (b) POT framework (c) Cost matrix design

FIGURE 6.1. Overview of OTLM: (a) VP pipeline entails transforming input by incorporating prompts with images and transforming output by mapping source labels to target labels; (b) Label mapping within the Partial Optimal Transportation (POT) framework involves computing the transportation cost as the sum of the pointwise multiplication between the mapping matrix and the cost matrix; (C) The design of the cost matrix involves obtaining predictions of the entire target dataset and then aggregating them by class through grouped summation. In FLM/ILM, the cost at each position is defined by transporting the intra-class distribution into the selected position, while OTLM defines it by transporting the inter-class distribution outside the selected position.

distributions as $(\mathbf{p}, \mathbf{q}) \in \Sigma_{n_s} \times \Sigma_{n_t}$:

$$\mathbf{p} = \sum_{j=1}^{n_s} p_j \delta_{f_{,j}}, \quad \mathbf{q} = \sum_{j=1}^{n_t} q_j \delta_{\hat{f}_{,j}}, \tag{6.3}$$

where $\delta_{f_{,j}}$ is the Dirac function at location $f_{,j} \in \mathbb{R}^N$, with N being the total number of samples in the target dataset $\mathcal{D}_T$ and $f_{i,j}$ signifying the j-th component of feature $f_i \in \mathbb{R}^{n_s}$. Σ_{n_s} and Σ_{n_t} correspond to histograms with n_s and n_t bins, respectively. p_j and q_j are the probability mass associated to the j-th element of feature f_i. In the VP-LM scenario, each target label correlates with a specific source label, disregarding the non-selected source labels. For sample-balance tasks, we adopt $p_j = q_j = \frac{1}{n_s}$, which leads that $\sum_j p_j = 1$ and $\sum_j q_j = \frac{n_t}{n_s}$. As a result, label mapping can be defined as a Partial Optimal Transport (POT), which extends the classical Optimal Transport (OT) concept to accommodate for the transfer of mass between distributions of differing total masses (Chizat et al. 2018), aptly suiting the label mapping context.

The set of partial transportation plans $\Pi(\mathbf{p}, \mathbf{q})$ between two discrete probability measures $\mathbf{p}$ and $\mathbf{q}$ is,

$$\Pi(\mathbf{p}, \mathbf{q}) = \{\mathbf{T} \in \mathbb{R}_{+}^{n_s \times n_t} | \mathbf{T}\mathbb{1}_{n_t} \leq \mathbf{p}, \mathbf{T}^{\top}\mathbb{1}_{n_s} = \mathbf{q}\}, \tag{6.4}$$

where $\mathbb{1}_d$ is a d-dimensional vector of ones, and $\mathbf{T} = (T_{ij})_{i,j}$ is the transportation plan matrix with an entry T_{ij} represents the fraction of mass transported from class $i \in \mathcal{Y}_S$ to class $j \in \mathcal{Y}_T$, which is the label mapping strategy.

The total transportation cost and the optimal transportation plan (Chapel et al. 2020) (Figure 6.1(b)) can be defined as,

$$\mathrm{POT}(\mathbf{p}, \mathbf{q}) = \min_{\mathbf{T} \in \Pi(\mathbf{p}, \mathbf{q})} \langle \mathbf{C}, \mathbf{T} \rangle_F = \min_{\mathbf{T} \in \Pi(\mathbf{p}, \mathbf{q})} \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} C_{ij} T_{ij}, \tag{6.5}$$

where C_{ij} is the cost of transporting unit mass from class $i \in \mathbf{p}$ to class $j \in \mathbf{q}$, $\mathbf{C} \in \mathbb{R}^{n_s \times n_t}$ is the cost matrix, and $\langle \cdot, \cdot \rangle_F$ is the Frobenius dot product.

6.4.2 Cost Matrix Definition

As defined by Eq.(6.5), solving for the optimal transport plan $\mathbf{T}$ necessitates a clear definition of the cost matrix $\mathbf{C}$. As mentioned above, the matrix element C_{ij} quantifies the necessary adjustments to transition a label i to a label j in the source and target domain respectively. An effective mapping strategy aims to minimize the overall feature disparity throughout the data distribution, significantly reducing the prompt's workload. This diminished feature variation facilitates the prompt's task of tailoring the output distribution to match the target labels more efficiently. Therefore, a suitable transportation cost matrix is of paramount importance. Next, we start with an intuitive approach to define the cost of transport within samples. By analyzing the limitations of this method, we further develop the inter-sample transport cost matrix proposed in this work.

6.4.2.1 Inter-sample Cost Definition.

In traditional classification tasks, aligning a feature vector $f_i \in \mathbb{R}_{+}^{n_s}$ with its label typically uses Kullback-Leibler (KL) divergence as the loss function to measure disparities. In this

study, we instead apply Earth Mover's Distance (EMD) to assess the discrepancy between f_i and the one-hot label vector h_t, denoted by $\mathrm{EMD}(f_i, h_t)$, where $h_t \in \mathbb{R}^{n_s}$represents the n_s-dimensional one-hot vector for label t.

To articulate the dimensions of the feature, we adopt the Earth Mover's Distance, derived from optimal transport theory, to quantify the "transport" needed to adjust f_i to h_t. Therefore, for a normalized feature f_i, we have:

$$\mathrm{EMD}(f_i, h_t) = \sum_{j \neq t} f_{i,j} = 1 - f_{i,t}, \tag{6.6}$$

where $f_{i,j}$ denotes the j-th component of feature vector f_i, implying that the transport required for feature-to-label alignment equates to the sum necessary to transition all feature components, barring the one associated with label t, to that specific label's position. In our ensuing discussion, we will illustrate that methodologies based on frequency (such as FLM, ILM (Chen et al. 2023)) potentially offer an approximative resolution to this theoretical framework.

6.4.2.2 Intra-sample Cost Definition.

As mentioned, label mapping essentially constitutes a partial optimal transport problem. This characterizes a scenario where the target domain necessitates fewer features than those generated by the source domain. As a result, the alignment of selected features with their corresponding labels does not require the repositioning of all unassociated feature values to the designated label's position. Instead, it specifically entails the reallocation of chosen feature values to align with the label. Accordingly, we can reformulate the transport cost as:

$$\hat{D}_{\mathbf{T}}(f_i, h_t) = \mathrm{EMD}(f_i\mathbf{T}, h_t\mathbf{T}). \tag{6.7}$$

This revised definition of transport cost selectively incorporates the n_t features chosen from the source domain labels, while disregarding the rest set of n_s features. An issue emerges from this definition because the transformation $\mathbf{T}$ isolates n_t features out of n_s from the source domain, resulting in the aggregated sum of $f_i\mathbf{T}$ diverging from unity. In practical applications, the alignment of the transformed feature vector $f_i\mathbf{T}$ with the target label t

involves reallocating the feature values from non-target positions to the target label's position. This procedure entails setting the feature values at non-target positions to zero while ensuring that the feature value at the target position remains positive, reflecting the activation state of the target label. Consequently, this allows us to reformulate the loss function as,

$$\hat{D}_{\mathbf{T}}(f_i, h_t) = \sum_{j \neq t} (f_i \mathbf{T})_j. \tag{6.8}$$

The preceding Eq.(6.8) redefines the cost for the alignment of the feature vector f_i with its corresponding label, in the context of label mapping. It quantifies the cost involved in adjusting the feature values at positions not specified by the label to zero, within the subset of n_t selected features from the source domain. With this framework in mind, we proceed to define C_{ij}, the cost associated with mapping a feature at index i from the source to a corresponding label at index j in the target domain. According to our refined definition, the loss incurred by choosing the source feature at index i for mapping to target label j is indirectly determined by the selection of the other $n_t - 1$ indices, which make a direct calculation of C_{ij} complicated. C_{ij} essentially represents the mapping of feature $f_{,i}$ to class j. For $f_{k,i}$, where $k \in \mathcal{C}_j$ and $\mathcal{C}_j = \{k | y_k = j\}$ denotes the set of sample indices labeled j, the transport cost for $f_{k,i}$ according to our definition is null, while the transport cost for $f_{k',i}$, with $k' \notin \mathcal{C}_j$, accumulates to $\sum_{k'} f_{k',i}$. (see Figure 6.1(c)) Therefore, C_{ij} can be expressed as,

$$C_{ij} = \sum_{k'} f_{k',i}, \quad \forall k' \notin \mathcal{C}_j. \tag{6.9}$$

For computational simplicity, we adopt a normalized formulation:

$$C_{ij} = \frac{\sum_{k'} f_{k',i}}{\sum_{n=1}^{N} f_{n,i}}, \quad \forall k' \notin \mathcal{C}_j \rightarrow C_{ij} = 1 - \frac{\sum_{k} f_{k,i}}{\sum_{n=1}^{N} f_{n,i}}, \quad \forall k \in \mathcal{C}_j, \tag{6.10}$$

where N represents the total number of samples, offering a clear and computationally efficient approach to calculating the mapping cost between source and target label indices.

6.4.3 Solution of the Optimal Transport Plan

With a well-defined cost matrix $\mathbf{C}$, we can establish the label mapping strategy for our method by resolving the partial optimal transport plan. The objective defined in Eq.(6.5) is to minimize this cost, leading us to the formulation:

$$\mathbf{T}_0 \leftarrow \underset{\mathbf{T}\in\Pi(\mathbf{p},\mathbf{q})}{\arg\min}\langle \mathbf{T}, \mathbf{C}\rangle_F. \tag{6.11}$$

Given the task constraint that $\forall i, j, p_i = q_j$, each target label is uniquely mapped to one source label. In such cases, the optimal transport plan $\mathbf{T}_0$ materializes as a permutation matrix, i.e., a binary matrix with solely 0 and 1, where each row and column contains no more than one entry of 1. The partial optimal transport problem is equivalent to an assignment problem. Concurrently, this specification is also perfectly suited for label mapping in visual prompting contexts.

As a result, the label mapping function can be formulated as $\mathcal{F}_{LM}(f_i) = f_i\mathbf{T}$. In practice, this calculation can be bypassed. We maintain an indices vector $\mathbf{I} \in \mathbb{R}^{n_t}$, where $\mathbf{I}_i$ represents the index of the source feature corresponding to the ith label, determined by $\mathbf{I}_i = \text{Find}(\mathbf{T}_{,i}, 1)$. This approach directly maps each target label to its source index efficiently, leveraging the optimal transport plan without the computational overhead.

This assignment problem can be solved via linear programming algorithms, with a computational complexity of $O((n_s + n_t)n_s n_t \log(n_s + n_t))$ (Ahuja et al. 1993). Considering practical scenarios where n_s and n_t are small enough (less than 10^5), this algorithm's computational demand is trivial compared to the extensive calculations required for deep model's forward and backward progress. Consequently, its impact on the overall training speed is marginal, ensuring the method's feasibility even in computationally intensive scenarios.

6.4.4 Analysis

Within the Optimal Transport (OT) framework, we examine Iterative Label Mapping (ILM) and Frequency-based Label Mapping (FLM), considering ILM as an extension of FLM.

This approach hinges on a frequency-based voting mechanism that selects for each category the most frequently occurring maximum value positions across samples, prioritizing these positions in a descending order of frequency.

We introduce a frequency matrix $\mathbf{A} \in \mathbb{R}_{+}^{n_t \times n_s}$, where $\mathbf{A}_{ij}$ reflects the number of times the maximum value index for samples of target category i is j in the source domain. The transport costs are inversely related to these frequencies, simplified as $\max(\mathbf{A}) - \mathbf{A}$ to derive costs, and finalizing the cost matrix as $\mathbf{C}_f = (\max(\mathbf{A}) - \mathbf{A})^T$. Traditional FLM and ILM methods then use a greedy algorithm to compute the transport plan $\mathbf{T}$.

However, these methods diverge from optimal transport principles, primarily by ignoring the detailed information in soft logits and relying on a greedy solution, limiting their ability to reduce prompts' learning effort and enhance efficiency. Our experiments support this critique, revealing the limitations of frequency-based mapping in prompt learning efficiency.

6.5 Experiment

6.5.1 Setup

6.5.1.1 Frameworks.

To perform vision prompt tuning, researchers have proposed various approaches, with two frameworks commonly used: visual prompting (**VP**) (Bahng et al. 2022) and visual prompt tuning (**VPT**) (Jia et al. 2022).

In our paper, we primarily adhere to the VP framework for conducting our experiments. Additionally, we modify the VPT-deep framework to be compatible with label mapping approaches by excluding the learnable task-specific classification layer. This enables us to compare our proposed method with the baselines, further demonstrating its generalizability.

TABLE 6.1. Specifications of six pre-trained backbones employed in this paper. All of these backbones are pre-trained on ImageNet (Deng et al. 2009).

Backbone	**Pre-trained Dataset**	**# params (M)**	**Feature dim** d
vit-B/16 (Dosovitskiy et al. 2020b)	ImageNet-1k	85	768
vit-B/16 (Dosovitskiy et al. 2020b)	ImageNet-22k	85	768
swin-B (Liu et al. 2021)	ImageNet-22k	88	1024
ResNet50 (He et al. 2016)	ImageNet-22k	25	2048
vit-L/16 (Dosovitskiy et al. 2020b)	ImageNet-1k	307	1024
ConvNeXt-B/16 (Liu et al. 2022)	ImageNet-2k	88	1024

6.5.1.2 Baselines.

Although VP can adapt pretrained models without changing any model parameters, the hard-coded mapping results in a degradation of performance. To achieve better label mapping, **FLM** and **ILM** (Chen et al. 2023) adopt label mapping based on prediction frequencies on target data points of the model without employing a vision prompt. We compare our proposed method with these competitive label mapping approaches, as well as full fine-tuning (**FF**) and linear probing (**LP**), serving as baselines.

6.5.1.3 Backbones Specifications

In our experiments, we use a total of six backbones, with their specifics outlined in Table 6.1. Specifically, under the VP framework, all these six backbones are utilized. The results for Vit-B pretrained on ImageNet-22K and ResNet50 are presented in the manuscript while the results for Vit-B/Vit-L pretrained on ImageNet-1K, ConvNeXt-B and Swin-B are presented in this supplementary. Meanwhile, Swin-B is also utilized under VPT framework.

6.5.1.4 Downstream Datasets.

Detailed information about the downstream datasets used under two framework groups is displayed in the table. Specifically, within the VP framework, we utilized ten datasets. including DTD (Cimpoi et al. 2014), CUB200 (Wah et al. 2011), NABirds (Van Horn et al. 2015), StanfordDogs (Khosla et al. 2011), Flowers102 (Nilsback and Zisserman 2008), Food101 (Bossard

TABLE 6.2. Description of datasets evaluated under the Visual Prompting and Visual Prompt Tuning frameworks.

Dataset	# Classes	# Train	# Val	# Test	Description
Evaluated under VP framework					
DTD (Cimpoi et al. 2014)	47	1880	1880	1880	texture database
CUB-200-2011 (Wah et al. 2011)	200	5394	600	5794	images of birds
NABirds (Van Horn et al. 2015)	555	21536	2393	24633	images of birds
Stanford Dogs (Khosla et al. 2011)	120	10800	1200	8580	images of dogs
Oxford Flowers (Nilsback and Zisserman 2008)	102	1020	1020	6149	images of flowers
Food101 (Bossard et al. 2014)	101	60600	15150	25250	images of food
CIFAR100 (Krizhevsky, Hinton et al. 2009)	100	40000	10000	10000	real-life object
CIFAR10 (Krizhevsky, Hinton et al. 2009)	10	40000	10000	10000	real-life object
GTSRB (Houben et al. 2013)	43	21312	2526	12630	traffic signs
SVHN (Netzer et al. 2011)	10	58605	14652	26032	numbers in real-life
Evaluated under VPT framework					
CUB-200-2011 (Wah et al. 2011)	200	5394	600	5794	images of birds
NABirds (Van Horn et al. 2015)	555	21536	2393	24633	images of birds
Oxford Flowers (Nilsback and Zisserman 2008)	102	1020	1020	6149	images of flowers
Stanford Dogs (Khosla et al. 2011)	120	10800	1200	8580	images of dogs
Stanford Cars (Gebru et al. 2017)	196	7329	815	8041	images of cars

et al. 2014), CIFAR10/100 (Krizhevsky, Hinton et al. 2009), GTSRB (Houben et al. 2013), and SVHN (Netzer et al. 2011). Under the VPT framework, we leverage Swin-B (Liu et al. 2021) models pretrained on ImageNet-22K (Deng et al. 2009) and assess their performance on CUB200 (Wah et al. 2011), NABirds (Van Horn et al. 2015), Flowers102 (Nilsback and Zisserman 2008), StanfordDogs (Khosla et al. 2011), and StanfordCars (Gebru et al. 2017). For datasets without publicly available splits, we followed (Huang et al. 2023) for the allocation of training and validation sets. Additionally, the five datasets we used within the VPT framework are from Fine-Grained Visual Classification tasks. For datasets without publicly available splits, we followed (Jia et al. 2022).

6.5.2 Performance Under the VP Framework

6.5.2.1 Main results

Our evaluation begins with experiments conducted under the VP framework (Bahng et al. 2022). We utilize ten distinct downstream tasks to evaluate the transfer performance. We train the source model for 100 epochs with SGD optimizer.

TABLE 6.3. Comparison of performance on 10 downstream datasets under the VP framework. The utilized source model is ViT-B pretrained on ImageNet-22K. The highest performance achieved among prompt-based methods is highlighted in **bold**.

	Fine-tuning		**Prompt-based**			
Dataset	FF	LP	VP	FLM-VP	ILM-VP	OTLM
DTD	64.3	63.2	50.0	52.0	48.3	**56.2**
CUB200	87.3	85.3	32.6	41.4	45.2	**56.6**
NAbirds	82.7	75.9	8.4	19.0	19.6	**33.8**
StanfordDogs	89.4	86.2	57.0	58.5	60.3	**67.9**
Flowers102	98.8	97.9	64.1	64.3	62.6	**86.6**
Food101	84.9	84.4	55.6	55.0	56.1	**61.9**
CIFAR100	68.9	63.4	49.4	54.5	54.0	**62.1**
CIFAR10	97.4	96.3	94.6	94.6	94.7	**95.2**
GTSRB	97.1	68.0	87.7	84.3	85.2	**88.6**
SVHN	87.4	36.6	87.0	88.5	**88.8**	88.7
Average	85.82	75.72	58.60	61.20	61.47	**69.77**

We first adopt a ViT-B model pretrained on ImageNet-22K as the source model and train it with an initial learning rate of 1e4. The results are summarized in Table 6.3, with results of fine-tuning methods reported in (Huang et al. 2023). Based on the results, it is evident that fine-tuning methods outperform prompt-based approaches in the majority of cases. Among prompt-based methods, the original VP method achieves the worst performance as it only employs a hard-coded mapping of labels. In contrast, the other three approaches achieve better performance. Specifically, our method achieves an average performance of 69.77% across all ten datasets, surpassing FLM and ILM by 8.57% and 8.30%, respectively. Following previous work (Chen et al. 2023), we employ a convolutional model, i.e., ResNet50 pretrained on ImageNet-1K, to further evaluate the proposed method. For this model, we initialize the learning rate to 1e2 and present the results in Table 6.4. With this source model, similar relevant accuracy is observed across each dataset for each method, where fine-tuning methods consistently outperform prompt-based approaches, and our proposed OTLM method outperforms other prompt-based approaches. Nevertheless, the absolute accuracy values are significantly lower than those achieved by the pretrained ViT-B model, primarily due to the limited model capacity of the ResNet50 model.

TABLE 6.4. Comparison of performance on 10 downstream datasets under the VP framework. The utilized source model is ResNet50 pretrained on ImageNet-1K. The highest performance achieved among prompt-based methods is highlighted in **bold**.

	Fine-tuning		**Prompt-based**			
Dataset	FF	LP	VP	FLM-VP	ILM-VP	OTLM
DTD	62.1	64.8	14.2	36.8	38.6	**42.7**
CUB200	76.5	68.1	1.1	9.7	11.0	**12.2**
NAbirds	73.7	58.7	5.9	5.9	7.6	**8.1**
StanfordDogs	75.8	88.5	4.3	51.8	51.9	**56.1**
Flowers102	88.1	81.0	6.9	11.3	17.1	**20.5**
Food101	84.0	71.8	7.8	20.6	**22.2**	21.8
CIFAR100	81.2	71.4	12.2	25.8	30.4	**34.8**
CIFAR10	95.8	89.9	55.9	61.3	64.7	**67.9**
GTSRB	95.2	79.4	33.7	39.4	43.2	**48.2**
SVHN	96.5	45.3	55.2	56.9	61.0	**62.7**
Average	82.89	71.89	19.71	31.94	34.77	**37.48**

6.5.2.2 More Results on Other Backbones

We assess our OTLM on Swin-B pretrained on ImageNet-22K and Vit-B pretrained on ImageNet-1K, presented in Table 6.5 and Table 6.6, respectively. We train the source models for 100 epochs. And we also assess OTLM on ViT-L pretrained on ImageNet-1K and ConvNeXt-B pretrained on ImageNet-22K, presented in Tabe 6.7, we train these two source models for 50 epochs. As shown in Table 6.5, the performance of the original VP with Swin-B as the backbone is not ideal. All LM methods, especially OTLM, significantly enhance VP's performance on all ten downstream tasks. Moreover, OTLM notably outperforms both FLM and ILM. Additionally, as mentioned in experiment section, pre-trained datasets with smaller output dimensions make label mapping simpler. Therefore, compared to backbones pre-trained on ImageNet-22K, the performance of frequency-based label mapping methods improves, as shown in Table 6.6. Nevertheless, performance of OTLM remains significantly superior to other label mapping methods across all datasets.

TABLE 6.5. Comparison of performance on 10 downstream datasets under the VP framework. The utilized source model is Swin-B pretrained on ImageNet-22K. The highest performance achieved among prompt-based methods is highlighted in **bold**.

	Fine-tuning		Prompt-based			
Dataset	FF	LP	VP	FLM-VP	ILM-VP	OTLM
DTD	72.4	73.6	28.0	56.1	51.1	**62.2**
CUB200	89.7	88.6	5.4	54.5	57.9	**60.9**
NAbirds	86.8	85.2	3.2	46.8	47.7	**52.9**
StanfordDogs	86.2	86.8	12.5	81.9	82.4	**86.6**
Flowers102	98.3	99.4	13.7	89.5	90.8	**95.5**
Food101	91.7	88.2	19.8	62.9	64.9	**66.1**
CIFAR100	73.3	61.6	24.6	68.5	71.4	**77.9**
CIFAR10	98.3	96.3	76.9	93.4	92.9	**95.1**
GTSRB	97.1	93.8	64.3	68.8	72.3	**76.8**
SVHN	91.2	43.5	74.7	77.6	78.5	**81.9**
Average	88.50	81.70	32.29	69.97	71.00	**75.58**

TABLE 6.6. Comparison of performance on 10 downstream datasets under the VP framework. The utilized source model is ViT-B pretrained on ImageNet-1K. The highest performance achieved among prompt-based methods is highlighted in **bold**.

	Fine-tuning		Prompt-based			
Dataset	FF	LP	VP	FLM-VP	ILM-VP	OTLM
DTD	70.6	68.7	47.8	53.0	54.2	**57.4**
CUB200	84.7	83.5	40.6	53.5	56.8	**63.2**
NAbirds	72.3	69.3	13.8	30.5	34.6	**38.8**
StanfordDogs	84.6	84.4	61.9	71.1	70.3	**73.7**
Flowers102	98.3	97.7	56.5	76.7	74.7	**82.0**
Food101	83.0	78.5	55.7	62.3	63.4	**64.1**
CIFAR100	87.5	77.6	54.4	63.7	64.0	**64.3**
CIFAR10	97.4	92.9	92.9	93.8	93.7	**94.9**
GTSRB	96.8	65.6	86.0	86.1	86.4	**89.0**
SVHN	96.9	61.1	87.8	88.0	88.2	**88.7**
Average	87.21	77.93	59.74	67.87	68.67	**71.61**

6.5.3 Performance Under the VPT Framework

We further assess our method under the VPT-deep framework (Jia et al. 2022), employing five different datasets. We adapt VPT by removing its learnable head to accommodate label mapping methods. The number of added prompt tokens for each dataset adheres to the original

TABLE 6.7. Comparison of performance on 3 downstream datasets under the VP framework. The utilized source models are ViT-L pretrained on ImageNet-1K and ConvNeXt-B pretrained on ImageNet-22K. The highest performance achieved among prompt-based methods is highlighted in **bold**.

Model	**ViT-L-1k**			**ConvNeXt-B-22k**		
Method	FLM-VP	ILM-VP	OTLM	FLM-VP	ILM-VP	OTLM
CIFAR10	92.9	91.7	**94.9**	79.1	81.3	**83.2**
CIFAR100	50.6	51.4	**72.4**	18.6	23.7	**29.9**
StanfordDogs	62.3	64.5	**87.9**	14.8	32.4	**62.5**

TABLE 6.8. Comparison of performance on five datasets under the VPT framework using model Swin-B pretrained on ImageNet-22K. The highest performance achieved among label mapping methods without additional heads is highlighted in **bold**.

	Fine-tuning			**Label mapping**		
Dataset	FF	LP	VPT	FLM	ILM	OTLM
Extra head	✓	✓	✓	✗	✗	✗
CUB-200-2011	88.8	87.7	85.4	80.7	81.8	**84.7**
NABirds	86.3	82.6	81.1	71.3	74.4	**77.2**
Flowers102	98.8	98.0	98.7	95.8	96.1	**98.5**
StanfordDogs	84.3	82.7	84.6	79.1	81.1	**83.1**
StanfordCars	90.4	68.2	80.3	73.4	74.8	**78.0**
Average	89.72	83.8	86.0	80.1	81.6	**84.30**

implementation of VPT. Source model is trained for 100 epochs on each dataset, utilizing a learning rate of 0.1 and a weight decay of 1e−4 with the SGD optimizer. Considering that the original VPT method necessitates training an extra classification layer for each downstream dataset, we exclude it from the comparison of label mapping methods.

We utilize Swin-B models pretrained on ImageNet-22K as source model and train them on downstream datasets. The corresponding results are presented in Table 6.8. Once again, fully fine-tuning (FF) achieves the best performance. Original VPT enhances performance over LP due to introducing not only trainable tokens at the input layer but also an extra head for each dataset at the output layer. On the contrary, applying label mapping to VPT enables it to map source labels to target labels without an extra classification head, which means less parameters to optimize. Moreover, our method significantly outperforms the other two LM

methods and achieves the best results on all five datasets. Notably, average metric of our approach even surpasses that of LP and is comparable to the results of methods with and additional head, except for FF.

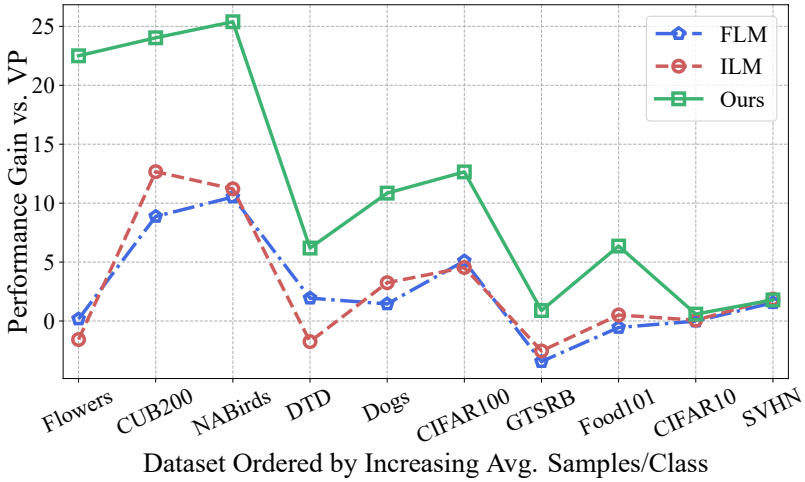
6.5.4 Data Efficiency in Downstream Task Transfer

In many cases, downstream tasks lack sufficient labeled data, making data efficiency — the ability to achieve proficient performance using limited data — crucial. To assess the data efficiency of various label mapping methods, we analyze the effectiveness of different approaches for integrating label mapping into the VP framework using the prompt-tuned ViT-B model. Specifically, we introduce a metric to measure data scale, defined as the average number of samples in each class. A higher value indicates a larger dataset scale, while a lower value suggests a smaller one. We explore the relationship between data scale and the efficacy of label mapping.

In Figure 6.2, we present the performance improvement of label mapping methods relative to the original VP method, arranged in ascending order of data scale. The figure illustrates that all three approaches exhibit substantial performance gains with smaller data scales. However, the performance gap compared to VP diminishes as the scale increases. Notably, our OTLM method consistently outperforms both FLM and ILM, particularly on small-scale data. This finding underscores the superior data efficiency of our method, which is significant in practical applications.

6.5.5 Scalability

A larger pretrained dataset typically yields a source model with enhanced representation extraction capabilities. However, the commonly concomitant increase in output dimensionality may also complicate the transportation process. To further investigate the performance of label mapping methods across source datasets of varying scales, we utilize Vit-B source models pretrained on both ImageNet-1K and ImageNet-22K, transferring them to the CUB200 downstream dataset. Alongside evaluating performance enhancements over the VP method,



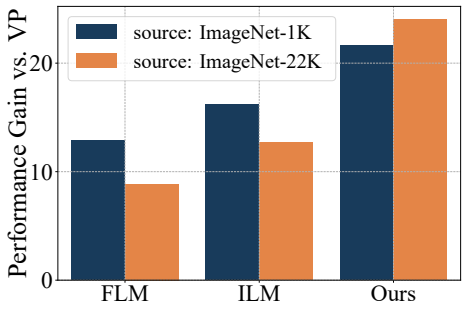
Dataset	S/C
Flowers	10
CUB200	27
NABirds	39
DTD	40
Dogs	90
CIFAR100	400
GTSRB	496
Food101	600
CIFAR10	4000
SVHN	5860

FIGURE 6.2. Data efficiency of variant label mapping approaches. Compared to FLM and ILM, our method demonstrates significantly higher accuracy gains over VP, particularly evident when the downstream dataset has fewer averaged samples per class. This highlights exceptional data efficiency of our method.

we employ the average cost as a metric for comparison. The average cost is computed by averaging the transportation cost of predictions from the source model into the target label space over the whole downstream dataset.

The results are depicted in Figure 6.3. In the left figure, it is evident that as the dataset scale increases, FLM and ILM show diminished improvement in downstream performance, while our method demonstrates enhanced performance improvement. Upon examining the values of the average cost, we observe that for each method, the cost value increases with the scale of the source dataset. This phenomenon arises because the source model pretrained on ImageNet-22K possesses a larger output dimension, making label mapping to a downstream task more challenging due to the increased number of possible mapping solutions. However, our partial optimal transportation-based method only considers a subset of the output dimension, which matches the output dimension of the downstream dataset.

This approach mitigates the negative impact of the increasing source dataset, as evidenced by the lower cost and higher performance improvement of OTLM compared to FLM and ILM. Furthermore, across each dataset, our OTLM method consistently outperforms FLM



Method	Average cost IN-1K	IN-22K
FLM	17.96	43.85
ILM	16.28	45.92
Ours	13.53	34.18

FIGURE 6.3. Comparison of different source datasets by employing Vit-B source models pretrained on either ImageNet-1K or ImageNet-22K. The performance improvement over VP and average cost under the CUB200 downstream dataset is reported.

and ILM in terms of performance gain over VP and average cost, underscoring its superior performance and scalability.

6.5.6 Ablation Study

We conducted an ablation study to examine the effectiveness of each module within our proposed OTLM method. The primary enhancements of our method lie in the cost matrix obtained under the optimal transportation framework and the utilization of linear programming for solving the label mapping result. We employ the ImageNet-22K pretrained ViT-B as the source model and adopt the Flowers102 dataset to investigate the impact of these two modules.

TABLE 6.9. Ablation study investigating the effects of various solving algorithms and cost matrix designs on model performance.

Algorithm	Cost matrix	Accuracy
Greedy	Frequency	62.56
Linear programming	Frequency	64.29
Greedy	OT	79.92
Linear programming	OT	**86.63**

The ablation results are presented in Table 6.9. As shown, the combination of the greedy solver and the frequency-based cost matrix achieves the worst result, indicating the limitations

of the existing ILM method. Building upon this, integrating either the optimal transportation-based cost matrix or linear programming helps improve performance. Adopting both of these modules, as OTLM does, yields the best performance, achieving a top-1 accuracy of 86.63%. These results confirm the effectiveness of each module within our proposed method.

6.5.7 Data Efficiency

To assess the data efficiency of various label mapping methods, we present the detailed results when reducing training samples per clasee from 4k to 4 on CIFAR10 dataset using Vit-B pretrained on ImageNet-22K in Tabel 6.10.

TABLE 6.10. Comparison of the efficiency between the proposed model and other models under different proportions of training samples.

Method	100%(4000)	10%(400)	1%(40)	0.1%(4)
FLM	94.6	88.5	70.6	12.7
ILM	94.7	88.5	69.8	10.9
OTLM	**95.2**(+0.5)	**93.6**(+5.1)	**80.3** (+9.7)	**18.4**(+5.7)

6.5.8 Discussion on Tunable Parameters and Running Speed

6.5.8.1 Tunable parameters.

We present the amount of trainable parameters of various methods on Vit-B pretrained on ImageNet-22K in Table 6.11. VP incorporates a minimum of trainable parameters with the input images without access to the model instead of tuning a great number of parameters of pretrained model. As shown in Table 6.11. Compared to FF and LP, the amount of parameters that VP needs to train is negligible. Furthermore, compared to VP, existing label mapping methods, especially our OTLM, do not introduce additional parameters yet significantly enhance model performance. Therefore, it is evident that a VP method equipped with an appropriate label mapping strategy can significantly improve training efficiency and greatly reduce the memory for model storage.

TABLE 6.11. Total trainable parameters in the input prompt or model finetuning for all 10 downstream datasets under VP framework on Vit-B pretrained on ImageNet-22K. '×' denotes the multiple of the tunable parameter amount relative to the total amount of pre-trained Vit-B pretrained on ImageNet-22K encoder parameters (85.8M).

	FF	**LP**	**VP**	**FLM**	**ILM**	**OTLM**
Total params	10.01×	0.43×	0.01×	0.01×	0.01×	0.01×

6.5.9 Visualization

6.5.9.1 Visualize mapping results.

we present the comparison mapping results on CIFAR10 dataset under the VP framework using model Vit-B pretrained on ImageNet-1K in Figure 6.4. Different label mapping strategies will yield different mapping results, thereby directly affecting the model's accuracy on downstream datasets.

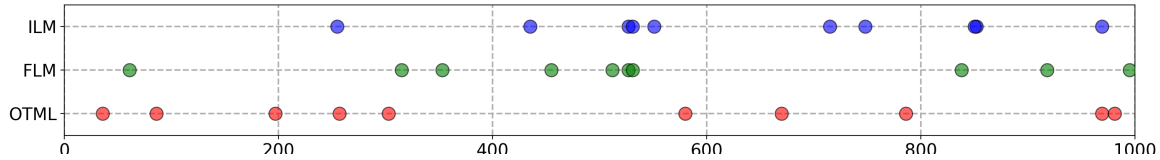


FIGURE 6.4. Visualize mapping results of three label mapping methods on CIFAR10 dataset under the VP framework.

6.5.9.2 Explanation of Different Label Mapping Strategies.

We report the LM strategies of frequency-based LM and OTLM on NABirds dataset using a Vit-B model pretrained on ImageNet-1K at the first epoch in Figure 6.5. Specifically, the results of the average logit for the classes 'gray jay' and 'gray catbird' at the model's output are shown in subfigures (a) and (b), respectively. The average logits for class 'gray jay', based on labels selected by frequency-based LM and OTLM, are presented in subfigures (c) and (d). As Figure 6.5 illustrates, the frequency-based LM, following a greedy strategy, chooses the source label with the highest logits as the mapping for 'gray jay' (blue line). However, this causes a significant transport cost since other target labels also have high logits (orange

line). In contrast, OTLM, employing a linear programming strategy, selects the source label with the minimal transport cost (green line). Although it may not have the highest logit in the output, as shown by subfigure (d), it has the highest logit among the selected labels. Therefore, frequency-based LM focuses solely on local selection results, while OTLM not only identifies the source label with the highest association but also ensures the transport cost is minimized.

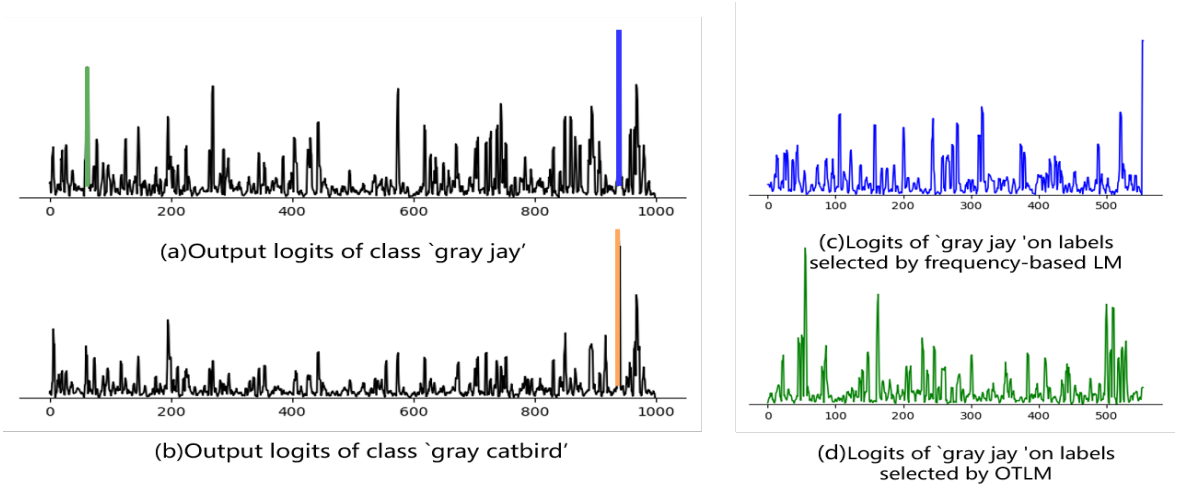


FIGURE 6.5. Explanation of the difference between frequency-based LM and OTLM. (a) The blue line represents the mapping of the 'gray jay' selected from the source labels by FLM. The green line represents the mapping of the 'gray' jay selected from the source labels by OTLM. (b) The orange line indicates the logit for the class 'gray catbird' on the selected label of FLM for 'gray jay'. (c) and (d) report the logits of 'gray jay' on labels selection results by frequency-based LM and OTLM, respectively.

6.6 Conclusion

This paper focuses on how appropriate label mapping can help alleviate the burden on input prompts. Specifically, we define a mapping cost matrix and propose OTLM, a novel label mapping strategy based on optimal transport that can minimize the necessary adjustments of the model when training prompts. Additionally, we theoretically analyze why existing frequency-based methods struggle to achieve this goal. Extensive experimental results prove that the proposed method improves accuracy significantly over all other LM methods on VP framework on 10 datasets. Furthermore, OTLM can also be easily applied to the VPT framework and achieves results comparable to VPT with an extra head.

CHAPTER 7

Conclusion

Transformer-based large-scale models have become the mainstream choice across various modalities. However, their substantial training costs pose a major barrier to both research and real-world deployment. Improving training efficiency and reducing computational overhead have therefore become urgent challenges. Existing approaches—whether through architecture-level interventions or interface-side component optimizations—often overlook the importance of adaptivity to the model itself.

In this thesis, we explore Model-Adaptive Efficiency Strategies for model training from both of these perspectives. Specifically, we first propose model-adaptive knowledge distillation and a model-adaptive Mixture-of-Experts (MoE) architecture for vision-language models (VLMs), addressing the limitations of existing methods in capturing rich cross-modal interactions. Furthermore, on the input side of large language models (LLMs), we propose an adaptive tokenizer that enables tokenization to dynamically align with the model's evolving training behavior. On the output side, we introduce an adaptive label mapping strategy for Vision Transformer (ViTs) that effectively reduces the training burden on model parameters. This work provides a promising direction for advancing future research and development in the field, contributing to more efficient training and fine-tuning of large-scale Transformer models.

7.1 Future Work

Throughout this thesis, we have explored various perspectives on designing model-adaptive components to enable more cost-efficient training of neural networks. Our investigations span across multiple modalities, including computer vision (CV), natural language processing

(NLP), and multimodal learning. The proposed methods collectively demonstrate that designing components specifically adapted to the characteristics of a given model can significantly improve training efficiency. However, these studies only scratch the surface—many important directions remain unexplored.

One promising direction for future research lies in the development of input- or output-level components for multimodal models that can preserve and exploit cross-modal interactions. Since architecture-level modifications often involve substantial changes to the backbone, they can be intrusive and incompatible with existing pretrained models. In contrast, components decoupled from the backbone offer greater flexibility and are more practical for closed-source models. That said, in multimodal architectures, cross-modal interactions are often distributed across different layers. How to capture such interaction information externally—without accessing the model's internals—remains an important and open research challenge.

Another potential research direction is to design unified model-adaptive components that can generalize across modalities. This is particularly challenging, as CV, NLP, and multimodal models exhibit fundamental differences: images are composed of pixels, NLP inputs are sequences of tokens, and multimodal systems must jointly process and align both. Understanding the shared training dynamics across these diverse architectures—and using that understanding to develop a modality-agnostic, adaptive component—is an open and intriguing question for future work.

Furthermore, while the two methods presented in this thesis focus on multimodal models such as VLMs, the proposed approaches can potentially be extended to other widely used models beyond CLIP, such as BLIP and LLaVA. It is worth noting that different VLMs adopt different strategies for modality alignment and cross-modal information interaction. Therefore, designing model-specific improvements based on the unique characteristics of each architecture represents a promising direction for broadening the applicability of the proposed approach.

In addition, research and experiments in the academic domain should ultimately be validated through real-world deployment. For example, incorporating MoE components may introduce

practical challenges during deployment, such as complex scheduling systems and limited compatibility with hardware like GPUs and TPUs. Hence, while MoE provides an effective way to reduce the number of trainable parameters, future work should also focus on addressing the deployment issues that arise in practical applications.

All in all, I believe the methods and results in this thesis can meaningfully contribute to future research on improving training efficiency across different modalities through model-adaptive strategies, especially in relation to the unresolved challenges discussed earlier.

APPENDIX A

Appendix for Chapter 5

A1 Results of ADAT in the 1B model.

To further demonstrate the scalability of our proposed ADAT method, we expanded our experimental results on larger models and larger corpus. Specifically, we add results(shown in Table A.1) from training a 1B model on 60B corpus. The results demonstrate that on larger models and with more data, ADAT continues to show substantial improvements over the baseline, indicating that ADAT has strong scalability.

TABLE A.1. Results in the 1B model.

Metric	**Unigram**	**ADAT**
Avg	49.11	51.20

A2 Details of Model Parameters

The detailed parameters of the 3 different model sizes applied in our experiment are shown in Table A.2.

TABLE A.2. Specifications of different LLMs used in the paper.

Model Size	**Layers**	**Model Dim**	**Heads**	**Learning Rate**	**Batch Size**
70M	6	512	8	10.0×10^{-4}	32
160M	12	768	12	2.5×10^{-4}	16
410M	24	1024	16	2.5×10^{-4}	16

A3 List of Training Dataset

The specific corpus used for training tokenizers is from The Pile[1], and the detailed list of files is shown below.

- pile_ArXiv_025.json
- pile_ArXiv_069.json
- pile_ArXiv_070.json
- pile_ArXiv_092.json
- pile_ArXiv_098.json
- pile_ArXiv_123.json
- pile_ArXiv_124.json
- pile_ArXiv_133.json
- pile_ArXiv_134.json
- pile_ArXiv_157.json
- pile_Books3_015.json
- pile_Books3_016.json
- pile_Books3_052.json
- pile_Books3_057.json
- pile_Books3_071.json
- pile_Books3_083.json
- pile_Books3_084.json
- pile_Books3_093.json
- pile_Books3_115.json
- pile_Books3_134.json
- pile_Books3_173.json
- pile_Books3_197.json
- pile_Books3_203.json
- pile_Books3_235.json
- pile_Books3_242.json

[1]URL:https://pile.eleuther.ai/

- pile_Books3_247.json
- pile_Enron_Emails_004.json
- pile_FreeLaw_031.json
- pile_FreeLaw_083.json
- pile_FreeLaw_104.json
- pile_Gutenberg_PG-19_044.json
- pile_Gutenberg_PG-19_049.json
- pile_OpenSubtitles_008.json
- pile_OpenSubtitles_031.json
- pile_OpenSubtitles_037.json
- pile_OpenWebText2_011.json
- pile_OpenWebText2_050.json
- pile_OpenWebText2_063.json
- pile_OpenWebText2_108.json
- pile_OpenWebText2_118.json
- pile_OpenWebText2_132.json
- pile_OpenWebText2_157.json
- pile_OpenWebText2_162.json
- pile_OpenWebText2_212.json
- pile_OpenWebText2_216.json
- pile_OpenWebText2_242.json
- pile_OpenWebText2_245.json
- pile_OpenWebText2_256.json
- pile_Pile-CC_001.json
- pile_Pile-CC_024.json
- pile_Pile-CC_069.json
- pile_Pile-CC_076.json
- pile_Pile-CC_106.json
- pile_Pile-CC_120.json
- pile_Pile-CC_133.json

- pile_Pile-CC_181.json
- pile_Pile-CC_209.json
- pile_Pile-CC_211.json
- pile_Pile-CC_237.json
- pile_Pile-CC_254.json
- pile_Pile-CC_259.json
- pile_PubMed_Abstracts_037.json
- pile_PubMed_Abstracts_049.json
- pile_PubMed_Abstracts_054.json
- pile_PubMed_Central_028.json
- pile_PubMed_Central_053.json
- pile_PubMed_Central_067.json
- pile_PubMed_Central_069.json
- pile_PubMed_Central_085.json
- pile_PubMed_Central_123.json
- pile_PubMed_Central_125.json
- pile_PubMed_Central_132.json
- pile_PubMed_Central_149.json
- pile_PubMed_Central_165.json
- pile_PubMed_Central_173.json
- pile_PubMed_Central_215.json
- pile_PubMed_Central_220.json
- pile_Stack_Exchange_055.json
- pile_USPTO_Backgrounds_012.json
- pile_USPTO_Backgrounds_027.json
- pile_USPTO_Backgrounds_031.json
- pile_USPTO_Backgrounds_051.json
- pile_Ubuntu_IRC_001.json
- pile_Ubuntu_IRC_017.json
- pile_Ubuntu_IRC_021.json

- pile_Wikipedia_en_006.json
- pile_Wikipedia_en_009.json
- pile_Wikipedia_en_043.json
- pile_Wikipedia_en_053.json
- pile_Wikipedia_en_070.json
- pile_YoutubeSubtitles_008.json

APPENDIX B

Appendix for Chapter 6

B1 Running speed.

Additionally, we report the average execution time for 1 epoch training under the VP framework in Table B.1. OTLM shares the similar run time with ILM which is slightly behind FLM.

TABLE B.1. Running time(s/epoch) under VP framework on DTD dataset with source model Vit-B pretrained on ImageNet-22K.

	FLM-VP	**ILM-VP**	**OTLM**
Running time	20	24	24

Bibliography

Ahuja, Ravindra K, Thomas L Magnanti, James B Orlin et al. (1993). *Network flows: theory, algorithms, and applications*. Vol. 1. Prentice hall Englewood Cliffs, NJ.

Aljundi, Rahaf et al. (2018). 'Memory aware synapses: Learning what (not) to forget'. In: *Proceedings of the European conference on computer vision*, pp. 139–154.

Arjovsky, Martin, Soumith Chintala and Léon Bottou (2017). 'Wasserstein generative adversarial networks'. In: *ICML*, pp. 214–223.

Bahng, Hyojin et al. (2022). 'Exploring visual prompts for adapting large-scale models'. In: *arXiv preprint arXiv:2203.17274*.

Bai, Yikun et al. (2023). 'Sliced optimal partial transport'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13681–13690.

Bao, Hangbo et al. (2021). 'Beit: Bert pre-training of image transformers'. In: *arXiv preprint arXiv:2106.08254*.

Beltagy, Iz, Matthew E Peters and Arman Cohan (2020). 'Longformer: The long-document transformer'. In: *arXiv preprint arXiv:2004.05150*.

Biderman, Stella et al. (2023). 'Pythia: A suite for analyzing large language models across training and scaling'. In: *International Conference on Machine Learning*. PMLR, pp. 2397–2430.

Bisk, Yonatan et al. (2020). 'PIQA: Reasoning about Physical Commonsense in Natural Language'. In: *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Boecking, Benedikt et al. (2022). 'Making the most of text semantics to improve biomedical vision–language processing'. In: *European conference on computer vision*. Springer, pp. 1–21.

Bonneel, Nicolas et al. (2015). 'Sliced and radon wasserstein barycenters of measures'. In: *Journal of Mathematical Imaging and Vision* 51, pp. 22–45.

Bossard, Lukas, Matthieu Guillaumin and Luc Van Gool (2014). 'Food-101–mining discriminative components with random forests'. In: *Proceedings of the European conference on computer vision*. Springer, pp. 446–461.

Bostrom, Kaj and Greg Durrett (2020). 'Byte pair encoding is suboptimal for language model pretraining'. In: *arXiv preprint arXiv:2004.03720*.

Brown, Tom et al. (2020). 'Language models are few-shot learners'. In: *Advances in neural information processing systems* 33, pp. 1877–1901.

Bursztein, Elie et al. (2023). 'Retvec: Resilient and efficient text vectorizer'. In: *Advances in Neural Information Processing Systems* 36.

Buzzega, Pietro et al. (2020). 'Dark experience for general continual learning: a strong, simple baseline'. In: *Advances in neural information processing systems* 33, pp. 15920–15930.

Chapel, Laetitia, Mokhtar Z Alaya and Gilles Gasso (2020). 'Partial optimal tranport with applications on positive-unlabeled learning'. In: *Advances in Neural Information Processing Systems* 33, pp. 2903–2913.

Chen, Aochuan et al. (2023). 'Understanding and improving visual prompting: A label-mapping perspective'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19133–19143.

Chen, Hanting et al. (2020a). 'Learning student networks via feature embedding'. In: *IEEE Transactions on Neural Networks and Learning Systems* 32.1, pp. 25–35.

Chen, Hanting et al. (2021a). 'Learning student networks in the wild'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6428–6437.

Chen, Hanting et al. (2021b). 'Pre-trained image processing transformer'. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12299–12310.

Chen, Lingwei, Yujie Fan and Yanfang Ye (2021c). 'Adversarial reprogramming of pretrained neural networks for fraud detection'. In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 2935–2939.

Chen, Ting et al. (2020b). 'A simple framework for contrastive learning of visual representations'. In: *International conference on machine learning*. PMLR, pp. 1597–1607.

Chizat, Lenaic et al. (2018). 'Unbalanced optimal transport: Dynamic and Kantorovich formulations'. In: *Journal of Functional Analysis* 274.11, pp. 3090–3123.

Cimpoi, Mircea et al. (2014). 'Describing textures in the wild'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3606–3613.

Clark, Jonathan H et al. (2022). 'Canine: Pre-training an efficient tokenization-free encoder for language representation'. In: *Transactions of the Association for Computational Linguistics* 10, pp. 73–91.

Clark, Peter et al. (2018). 'Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge'. In: *arXiv:1803.05457v1*.

Courty, Nicolas et al. (2016). 'Optimal transport for domain adaptation'. In: *IEEE transactions on pattern analysis and machine intelligence* 39.9, pp. 1853–1865.

Courty, Nicolas et al. (2017). 'Joint distribution optimal transportation for domain adaptation'. In: *Advances in neural information processing systems* 30.

Dai, Damai et al. (2024). 'Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models'. In: *arXiv preprint arXiv:2401.06066*.

Damodaran, Bharath Bhushan et al. (2018). 'Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation'. In: *Proceedings of the European conference on computer vision*, pp. 447–463.

Dao, Tri et al. (2022). 'Flashattention: Fast and memory-efficient exact attention with io-awareness'. In: *Advances in Neural Information Processing Systems* 35, pp. 16344–16359.

Deng, Jia et al. (2009). 'Imagenet: A large-scale hierarchical image database'. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255.

Deng, Li (2012). 'The mnist database of handwritten digit images for machine learning research [best of the web]'. In: *IEEE signal processing magazine* 29.6, pp. 141–142.

Devlin, Jacob et al. (2018). 'Bert: Pre-training of deep bidirectional transformers for language understanding'. In: *arXiv preprint arXiv:1810.04805*.

Ding, Yuxuan et al. (2022). 'Don't Stop Learning: Towards Continual Learning for the CLIP Model'. In: *arXiv preprint arXiv:2207.09248*.

Dong, Xiaoyi et al. (2022). 'Clip itself is a strong fine-tuner: Achieving 85.7% and 88.0% top-1 accuracy with vit-b and vit-l on imagenet'. In: *arXiv preprint arXiv:2212.06138*.

Dosovitskiy, Alexey et al. (2020a). 'An image is worth 16x16 words: Transformers for image recognition at scale'. In: *arXiv preprint arXiv:2010.11929*.

— (2020b). ‘An image is worth 16x16 words: Transformers for image recognition at scale’. In: *arXiv preprint arXiv:2010.11929*.

Douillard, Arthur et al. (2020). ‘Podnet: Pooled outputs distillation for small-tasks incremental learning’. In: *European Conference on Computer Vision*. Springer, pp. 86–102.

Douillard, Arthur et al. (2022). ‘Dytox: Transformers for continual learning with dynamic token expansion’. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9285–9295.

Fedus, William, Barret Zoph and Noam Shazeer (2022). ‘Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity’. In: *Journal of Machine Learning Research* 23.120, pp. 1–39.

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

Fernando, Chrisantha et al. (2017). ‘Pathnet: Evolution channels gradient descent in super neural networks’. In: *arXiv preprint arXiv:1701.08734*.

Gage, Philip (1994). ‘A new algorithm for data compression’. In: *The C Users Journal* 12.2, pp. 23–38.

Gao, Leo et al. (2020). ‘The Pile: An 800GB Dataset of Diverse Text for Language Modeling’. In: *arXiv preprint arXiv:2101.00027*.

Gao, Leo et al. (2023a). *A framework for few-shot language model evaluation*. Version v0.4.0. URL: https://zenodo.org/records/10256836.

Gao, Peng et al. (2023b). ‘Clip-adapter: Better vision-language models with feature adapters’. In: *International Journal of Computer Vision*, pp. 1–15.

Gebru, Timnit et al. (2017). ‘Fine-grained car detection for visual census estimation’. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 31. 1.

Geng, Shijie et al. (2023). ‘HiCLIP: Contrastive language-image pretraining with hierarchy-aware attention’. In: *arXiv preprint arXiv:2303.02995*.

Godey, Nathan et al. (2022). ‘Manta: Efficient gradient-based tokenization for robust end-to-end language modeling’. In: *arXiv preprint arXiv:2212.07284*.

Gu, Xiang et al. (2022). 'Keypoint-guided optimal transport with applications in heterogeneous domain adaptation'. In: *Advances in Neural Information Processing Systems* 35, pp. 14972–14985.

Gu, Xiang et al. (2023). 'Optimal transport-guided conditional score-based diffusion model'. In: *Advances in Neural Information Processing Systems* 36, pp. 36540–36552.

Hao, Zhiwei et al. (2024). 'ADEM-VL: Adaptive and Embedded Fusion for Efficient Vision-Language Tuning'. In: *arXiv preprint arXiv:2410.17779*.

He, Jiaao et al. (2021). 'Fastmoe: A fast mixture-of-expert training system'. In: *arXiv preprint arXiv:2103.13262*.

He, Jiaao et al. (2022). 'Fastermoe: modeling and optimizing training of large-scale dynamic pre-trained models'. In: *Proceedings of the 27th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pp. 120–134.

He, Kaiming et al. (2016). 'Deep residual learning for image recognition'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Helber, Patrick et al. (2019). 'Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification'. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 12.7, pp. 2217–2226.

Hinton, Geoffrey, Oriol Vinyals and Jeff Dean (2015). 'Distilling the knowledge in a neural network'. In: *arXiv preprint arXiv:1503.02531*.

Hou, Saihui et al. (2019). 'Learning a unified classifier incrementally via rebalancing'. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 831–839.

Houben, Sebastian et al. (2013). 'Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark'. In: *The 2013 international joint conference on neural networks (IJCNN)*, pp. 1–8.

Hu, Edward J et al. (2021). 'Lora: Low-rank adaptation of large language models'. In: *arXiv preprint arXiv:2106.09685*.

Hu, Jie et al. (2023). 'Pseudo-label alignment for semi-supervised instance segmentation'. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16337–16347.

Huang, Qidong et al. (2023). 'Diversity-aware meta visual prompting'. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10878–10887.

Islam, Md Mofijul et al. (2022). 'A vocabulary-free multilingual neural tokenizer for end-to-end task learning'. In: *arXiv preprint arXiv:2204.10815*.

Jia, Chao et al. (2021). 'Scaling up visual and vision-language representation learning with noisy text supervision'. In: *International conference on machine learning*. PMLR, pp. 4904–4916.

Jia, Menglin et al. (2022). 'Visual prompt tuning'. In: *European conference on computer vision*. Springer, pp. 709–727.

Johannes Welbl Nelson F. Liu, Matt Gardner (2017). 'Crowdsourcing Multiple Choice Science Questions'. In: *arXiv preprint arXiv:1707.06209*.

Kamra, Nitin, Umang Gupta and Yan Liu (2017). 'Deep generative dual memory network for continual learning'. In: *arXiv preprint arXiv:1710.10368*.

Khosla, Aditya et al. (2011). 'Novel dataset for fine-grained image categorization: Stanford dogs'. In: *Proceedings of the IEEE international conference on computer vision workshops*. Vol. 2. 1.

Kirkpatrick, James et al. (2017). 'Overcoming catastrophic forgetting in neural networks'. In: *Proceedings of the national academy of sciences* 114.13, pp. 3521–3526.

Korotin, Alexander, Daniil Selikhanovych and Evgeny Burnaev (2022). 'Neural optimal transport'. In: *arXiv preprint arXiv:2201.12220*.

Krause, Jonathan et al. (2013). '3d object representations for fine-grained categorization'. In: *Proceedings of the IEEE international conference on computer vision workshops*, pp. 554–561.

Krizhevsky, Alex, Geoffrey Hinton et al. (2009). 'Learning multiple layers of features from tiny images'. In.

Kudo, Taku (2018). 'Subword regularization: Improving neural network translation models with multiple subword candidates'. In: *arXiv preprint arXiv:1804.10959*.

Kudo, Taku and John Richardson (2018). 'Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing'. In: *arXiv preprint arXiv:1808.06226*.

Kurmi, Vinod Kumar and Vinay P Namboodiri (2019). 'Looking back at labels: A class based domain adaptation technique'. In: *2019 international joint conference on neural networks (IJCNN)*. IEEE, pp. 1–8.

Lan, Zhenzhong et al. (2019). 'Albert: A lite bert for self-supervised learning of language representations'. In: *arXiv preprint arXiv:1909.11942*.

Le, Trung et al. (2021). 'Lamda: Label matching deep domain adaptation'. In: *International Conference on Machine Learning*. PMLR, pp. 6043–6054.

Le, Ya and Xuan Yang (2015). 'Tiny imagenet visual recognition challenge'. In: *CS 231N* 7.7, p. 3.

Lewis, Mike et al. (2021). 'Base layers: Simplifying training of large, sparse models'. In: *International Conference on Machine Learning*. PMLR, pp. 6265–6274.

Li, Bo et al. (2024). 'Llava-onevision: Easy visual task transfer'. In: *arXiv preprint arXiv:2408.03326*.

Li, Junnan et al. (2022a). 'Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation'. In: *International conference on machine learning*. PMLR, pp. 12888–12900.

Li, Junnan et al. (2023a). 'Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models'. In: *International conference on machine learning*. PMLR, pp. 19730–19742.

Li, Liunian Harold et al. (2022b). 'Grounded language-image pre-training'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10965–10975.

Li, Xilai et al. (2019). 'Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting'. In: *International conference on machine learning*. PMLR, pp. 3925–3934.

Li, Yizhe et al. (2023b). 'Exploring the benefits of visual prompting in differential privacy'. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5158–5167.

Li, Zhizhong and Derek Hoiem (2017). 'Learning without forgetting'. In: *IEEE transactions on pattern analysis and machine intelligence* 40.12, pp. 2935–2947.

Ling, Wang et al. (2015). 'Character-based neural machine translation'. In: *arXiv preprint arXiv:1511.04586*.

Lipman, Yaron et al. (2022). 'Flow matching for generative modeling'. In: *arXiv preprint arXiv:2210.02747*.

Liu, Haotian et al. (2024). 'Visual instruction tuning'. In: *Advances in neural information processing systems* 36.

Liu, Jian et al. (2020). 'Logiqa: A challenge dataset for machine reading comprehension with logical reasoning'. In: *arXiv preprint arXiv:2007.08124*.

Liu, Pengfei et al. (2023a). 'Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing'. In: *ACM computing surveys* 55.9, pp. 1–35.

Liu, Qidong et al. (2023b). 'Moelora: An moe-based parameter efficient fine-tuning method for multi-task medical applications'. In: *arXiv preprint arXiv:2310.18339*.

Liu, Siyang et al. (2023c). 'Task-Adaptive Tokenization: Enhancing Long-Form Text Generation Efficacy in Mental Health and Beyond'. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 15264–15281.

Liu, Xialei et al. (2023d). 'Class incremental learning with pre-trained vision-language models'. In: *arXiv preprint arXiv:2310.20348*.

Liu, Yufan et al. (2019). 'Knowledge distillation via instance relationship graph'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7096–7104.

Liu, Ze et al. (2021). 'Swin transformer: Hierarchical vision transformer using shifted windows'. In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022.

Liu, Zhuang et al. (2022). 'A convnet for the 2020s'. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986.

Lopes, Raphael Gontijo, Stefano Fenu and Thad Starner (2017). 'Data-free knowledge distillation for deep neural networks'. In: *arXiv preprint arXiv:1710.07535*.

Lopez-Paz, David and Marc'Aurelio Ranzato (2017). 'Gradient episodic memory for continual learning'. In: *Advances in neural information processing systems* 30.

Loshchilov, I (2017). 'Decoupled weight decay regularization'. In: *arXiv preprint arXiv:1711.05101*.

Ma, Xinhong et al. (2023). 'When visual prompt tuning meets source-free domain adaptive semantic segmentation'. In: *Advances in Neural Information Processing Systems* 36, pp. 6690–6702.

Maji, Subhransu et al. (2013). 'Fine-grained visual classification of aircraft'. In: *arXiv preprint arXiv:1306.5151*.

Mallya, Arun and Svetlana Lazebnik (2018). '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.

Masana, Marc et al. (2022). 'Class-incremental learning: survey and performance evaluation on image classification'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.5, pp. 5513–5533.

Mustafa, Basil et al. (2022). 'Multimodal contrastive learning with limoe: the language-image mixture of experts'. In: *Advances in Neural Information Processing Systems* 35, pp. 9564–9576.

Nawrot, Piotr et al. (2022). 'Efficient transformers with dynamic token pooling'. In: *arXiv preprint arXiv:2211.09761*.

Netzer, Yuval et al. (2011). 'Reading digits in natural images with unsupervised feature learning'. In: *NIPS workshop on deep learning and unsupervised feature learning*. Vol. 2011. 2. Granada, p. 4.

Nie, Ying et al. (2023). 'Lightclip: learning multi-level interaction for lightweight vision-language models'. In: *arXiv preprint arXiv:2312.00674*.

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

Paperno, Denis et al. (Aug. 2016). *The LAMBADA dataset*. DOI: 10.5281/zenodo.2630551.

Park, Chanjun et al. (2021). 'Should we find another model?: Improving neural machine translation performance with ONE-piece tokenization method without model modification'. In: *Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics: human language technologies: Industry papers*, pp. 97–104.

Parkhi, Omkar M et al. (2012). 'Cats and dogs'. In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE, pp. 3498–3505.

Pfeiffer, Jonas et al. (2020). 'Adapterhub: A framework for adapting transformers'. In: *arXiv preprint arXiv:2007.07779*.

Radford, Alec, Rafal Jozefowicz and Ilya Sutskever (2017). 'Learning to generate reviews and discovering sentiment'. In: *arXiv preprint arXiv:1704.01444*.

Radford, Alec et al. (2018). 'Improving language understanding by generative pre-training'. In.

Radford, Alec et al. (2021). 'Learning transferable visual models from natural language supervision'. In: *International conference on machine learning*. PMLR, pp. 8748–8763.

Rae, Jack W. et al. (2019). *Compressive Transformers for Long-Range Sequence Modelling*. arXiv: 1911.05507 [cs.LG].

Raffel, Colin et al. (2020). 'Exploring the limits of transfer learning with a unified text-to-text transformer'. In: *Journal of machine learning research* 21.140, pp. 1–67.

Rajbhandari, Samyam et al. (2022). 'Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale'. In: *International conference on machine learning*. PMLR, pp. 18332–18346.

Rannen, Amal et al. (2017). 'Encoder based lifelong learning'. In: *Proceedings of the IEEE international conference on computer vision*, pp. 1320–1328.

Rebuffi, Sylvestre-Alvise et al. (2017). 'icarl: Incremental classifier and representation learning'. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010.

Romero, Adriana et al. (2014). 'Fitnets: Hints for thin deep nets'. In: *arXiv preprint arXiv:1412.6550*.

Sachidananda, Vin, Jason S Kessler and Yi-An Lai (2021). 'Efficient domain adaptation of language models via adaptive tokenization'. In: *arXiv preprint arXiv:2109.07460*.

Sakaguchi, Keisuke et al. (2019). 'An adversarial winograd schema challenge at scale'. In: vol. 6.

Salimans, Tim et al. (2018). 'Improving GANs using optimal transport'. In: *arXiv preprint arXiv:1803.05573*.

Sang, Jianghui et al. (2022). 'Multi-label transfer learning via latent graph alignment'. In: *World Wide Web* 25.2, pp. 879–898.

Sau, Bharat Bhusan and Vineeth N Balasubramanian (2016). 'Deep model compression: Distilling knowledge from noisy teachers'. In: *arXiv preprint arXiv:1610.09650*.

Schick, Timo and Hinrich Schütze (2020). 'It's not just size that matters: Small language models are also few-shot learners'. In: *arXiv preprint arXiv:2009.07118*.

Schuhmann, Christoph et al. (2022). 'Laion-5b: An open large-scale dataset for training next generation image-text models'. In: *Advances in Neural Information Processing Systems* 35, pp. 25278–25294.

Sennrich, Rico, Barry Haddow and Alexandra Birch (2015). 'Neural machine translation of rare words with subword units'. In: *arXiv preprint arXiv:1508.07909*.

Shazeer, Noam et al. (2017). 'Outrageously large neural networks: The sparsely-gated mixture-of-experts layer'. In: *arXiv preprint arXiv:1701.06538*.

Shen, Li et al. (2023a). 'On efficient training of large-scale deep learning models: A literature review'. In: *arXiv preprint arXiv:2304.03589*.

Shen, Sheng et al. (2023b). 'Scaling vision-language models with sparse mixture of experts'. In: *arXiv preprint arXiv:2303.07226*.

Shmelkov, Konstantin, Cordelia Schmid and Karteek Alahari (2017). 'Incremental learning of object detectors without catastrophic forgetting'. In: *Proceedings of the IEEE international conference on computer vision*, pp. 3400–3409.

Socher, Richard et al. (2013). 'Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank'. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642.

Sohn, Kihyuk et al. (2023). 'Visual prompt tuning for generative transfer learning'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19840–19851.

Sreedhar, Makesh Narsimhan et al. (2022). 'Local byte fusion for neural machine translation'. In: *arXiv preprint arXiv:2205.11490*.

Su, Jianlin (2023). *BytePiece: A more pure and effective tokenizer*. https://github.com/bojone/bytepiece.

Sun, Quan et al. (2024). 'Eva-clip-18b: Scaling clip to 18 billion parameters'. In: *arXiv preprint arXiv:2402.04252*.

Sun, Zhiqing et al. (2020). 'Mobilebert: a compact task-agnostic bert for resource-limited devices'. In: *arXiv preprint arXiv:2004.02984*.

Sutskever, Ilya, James Martens and Geoffrey E Hinton (2011). 'Generating text with recurrent neural networks'. In: *Proceedings of the international conference on machine learning*, pp. 1017–1024.

Tang, Jian et al. (2015). 'Line: Large-scale information network embedding'. In: *Proceedings of the international conference on world wide web*, pp. 1067–1077.

Tay, Yi et al. (2021). 'Charformer: Fast character transformers via gradient-based subword tokenization'. In: *arXiv preprint arXiv:2106.12672*.

Thawani, Avijit et al. (2023). 'Learn Your Tokens: Word-Pooled Tokenization for Language Modeling'. In: *arXiv preprint arXiv:2310.11628*.

Thengane, Vishal et al. (2022). 'CLIP model is an Efficient Continual Learner'. In: *arXiv:2210.03114*.

Touvron, Hugo et al. (2023). 'Llama: Open and efficient foundation language models'. In: *arXiv preprint arXiv:2302.13971*.

Tsai, Yun-Yun, Pin-Yu Chen and Tsung-Yi Ho (2020). 'Transfer learning without knowing: Reprogramming black-box machine learning models with scarce data and limited resources'. In: *International Conference on Machine Learning*. PMLR, pp. 9614–9624.

Tschannen, Michael et al. (2025). 'Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features'. In: *arXiv preprint arXiv:2502.14786*.

Van Horn, Grant et al. (2015). 'Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 595–604.

Villani, Cédric et al. (2008). *Optimal transport: old and new*. Vol. 338. Springer.

Viterbi, Andrew (1967). 'Error bounds for convolutional codes and an asymptotically optimum decoding algorithm'. In: *IEEE transactions on Information Theory* 13.2, pp. 260–269.

Wah, Catherine et al. (2011). 'The caltech-ucsd birds-200-2011 dataset'. In.

Wang, Alex et al. (2018). 'GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding'. In: *Proceedings of the EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355.

Wang, Liyuan et al. (2024). 'A comprehensive survey of continual learning: theory, method and application'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wang, Xinyi, Sebastian Ruder and Graham Neubig (2021). 'Multi-view subword regularization'. In: *arXiv preprint arXiv:2103.08490*.

Wang, Zifeng et al. (2020). 'Learn-prune-share for lifelong learning'. In: *IEEE International Conference on Data Mining*. IEEE, pp. 641–650.

Wortsman, Mitchell et al. (2022). 'Robust fine-tuning of zero-shot models'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7959–7971.

Wu, Jun and Jingrui He (2020). 'Continuous transfer learning with label-informed distribution alignment'. In: *arXiv preprint arXiv:2006.03230*.

Wu, Yonghui et al. (2016). 'Google's neural machine translation system: Bridging the gap between human and machine translation'. In: *arXiv preprint arXiv:1609.08144*.

Wu, Yue et al. (2019). 'Large scale incremental learning'. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 374–382.

Xiao, Jianxiong et al. (2010). 'Sun database: Large-scale scene recognition from abbey to zoo'. In: *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, pp. 3485–3492.

Xing, Yinghui et al. (2022). 'Class-aware visual prompt tuning for vision-language pre-trained model'. In: *arXiv preprint arXiv:2208.08340*.

Yan, Shipeng, Jiangwei Xie and Xuming He (2021). 'Der: Dynamically expandable representation for class incremental learning'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3014–3023.

Yao, Lewei et al. (2021). 'Filip: Fine-grained interactive language-image pre-training'. In: *arXiv preprint arXiv:2111.07783*.

Yao, Lewei et al. (2022). 'Detclip: Dictionary-enriched visual-concept paralleled pre-training for open-world detection'. In: *Advances in Neural Information Processing Systems*, pp. 9125–9138.

Yim, Junho et al. (2017). ‘A gift from knowledge distillation: Fast optimization, network minimization and transfer learning’. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4133–4141.

Yoon, Jaehong et al. (2017). ‘Lifelong learning with dynamically expandable networks’. In: *arXiv preprint arXiv:1708.01547*.

Yu, Jiahui et al. (2022). ‘Coca: Contrastive captioners are image-text foundation models’. In: *arXiv preprint arXiv:2205.01917*.

Yu, Jiazuo et al. (2024). ‘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.

Zagoruyko, Sergey and Nikos Komodakis (2016). ‘Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer’. In: *arXiv preprint arXiv:1612.03928*.

Zang, Yuhang et al. (2022). ‘Unified vision and language prompt learning’. In: *arXiv preprint arXiv:2210.07225*.

Zhang, Jihai et al. (2024a). ‘Clip-moe: Towards building mixture of experts for clip with diversified multiplet upcycling’. In: *arXiv preprint arXiv:2409.19291*.

Zhang, Jingyi et al. (2024b). ‘Vision-language models for vision tasks: A survey’. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhang, Renrui et al. (2022). ‘Tip-adapter: Training-free adaption of clip for few-shot classification’. In: *European conference on computer vision*. Springer, pp. 493–510.

Zhao, Yue et al. (2021). ‘Label-based alignment multi-source domain adaptation for cross-subject EEG fatigue mental state evaluation’. In: *Frontiers in Human Neuroscience* 15, p. 706270.

Zheng, Mengyu et al. (2024a). ‘Adapt without forgetting: Distill proximity from dual teachers in vision-language models’. In: *European Conference on Computer Vision*. Springer, pp. 109–125.

Zheng, Mengyu et al. (2024b). ‘Enhancing Large Language Models through Adaptive Tokenizers’. In: *Advances in Neural Information Processing Systems* 37, pp. 113545–113568.

Zheng, Mengyu et al. (2024c). 'Visual Prompting via Partial Optimal Transport'. In: *European Conference on Computer Vision*. Springer, pp. 1–18.

Zheng, Mengyu et al. (2025a). 'Adapt Without Forgetting: Distill Proximity from Dual Teachers in Vision-Language Models'. In: *European Conference on Computer Vision*. Springer, pp. 109–125.

Zheng, Mengyu et al. (2025b). *Multimodal Mixture of Experts for Continual Learning in Vision-Language Models*. ICCV submission ID 05780.

Zheng, Zangwei et al. (2023a). 'Preventing Zero-Shot Transfer Degradation in Continual Learning of Vision-Language Models'. In: *arXiv preprint arXiv:2303.06628*.

— (2023b). 'Preventing Zero-Shot Transfer Degradation in Continual Learning of Vision-Language Models'. In: *arXiv preprint arXiv:2303.06628*.

Zhou, Kaiyang et al. (2022). 'Learning to prompt for vision-language models'. In: *International Journal of Computer Vision* 130.9, pp. 2337–2348.