

Rochester Institute of Technology

RIT Digital Institutional Repository

Theses

7-2-2024

Towards Human-Embodied Visual Intelligence

Cheng Han
ch7858@rit.edu

Follow this and additional works at: <https://repository.rit.edu/theses>

Recommended Citation

Han, Cheng, "Towards Human-Embodied Visual Intelligence" (2024). Thesis. Rochester Institute of Technology. Accessed from

This Dissertation is brought to you for free and open access by the RIT Libraries. For more information, please contact repository@rit.edu.

Towards Human-Embodied Visual Intelligence

by

Cheng Han

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the Chester F. Carlson Center for Imaging Science

College of Science

Rochester Institute of Technology

July 2, 2024

Towards Human-Embodied Visual Intelligence

by

Cheng Han

Submitted to the

Chester F. Carlson Center for Imaging Science
in partial fulfillment of the requirements
for the Doctor of Philosophy Degree
at the Rochester Institute of Technology

Abstract

The rapid development of artificial intelligence (AI) in computer vision has garnered considerable attention across diverse research fields (*e.g.*, image classification [1, 2], image segmentation [3, 4], object detection [5, 6], optical flow [7–12], depth estimation [13, 14]). However, this progress is met with an array of challenges that hinder them from real-world deployments. In this paper, three major challenges are discussed: ❶ *Low Task Generalizability*. Current visual models are often specifically designed for targeted tasks, constraining their generalizability with varying adaptation scenarios; ❷ *Weak Model Interpretability*. Within the paradigm of connectionism [15], most of these models are frequently regarded as “black-box” systems, making them challenging for humans to understand and control; ❸ *High Computational Consumption*. As the quest for superior performance persists, there is a prevailing trend towards scaling up visual models, which consequentially incurs significant computational costs. Human visual intelligence, on the other hand, provides natural solutions to these challenges. I thus seek to embody and mimic capabilities with human visual intelligence. Consequently, three primary contributions are covered in this paper in response to these challenges.

Universal Visual Learner. While current computer vision techniques provide specialized solutions for different vision tasks (*e.g.*, optical flow, depth estimation), human understands and explores the world by complex visual stimuli, unbound by task-specific constraints. To bridge this gap, I propose the Prototypical Transformer (ProtoFormer), a general and unified framework that addresses various motion tasks from a prototype-based perspective. ProtoFormer seamlessly integrates prototype learning with the Transformer architecture by thoughtfully incorporating motion dynamics through two innovative designs. First, *Cross-Attention Prototyping* identifies prototypes based on distinct motion patterns, enhancing transparency in the interpretation of motion scenes. Second, *Latent Synchronization* steers feature representation learning via prototypes, effectively reducing motion uncertainty.

Interpretable Visual Intelligence. Due to the connectionism nature of the current deep neural networks, how to explain the network behaviors becomes a critical topic. In light of this view, I first introduce DNC (Deep Nearest Centroids)—a rejuvenation of the classic Nearest Centroids classifier, envisioned for large-scale visual recognition. In contrast to conventional deep models, which often overlook latent data structures, DNC employs a non-parametric, case-based reasoning approach. By utilizing sub-centroids of training samples to represent class distributions, DNC classifies by measuring the proximity of test data to these sub-centroids within the feature space. This distance-based approach provides unparalleled flexibility, allowing complete knowledge transfer across diverse recognition tasks. Moreover, DNC’s inherent simplicity, combined with its intuitive decision-making process, ensures explainability when sub-centroids are actual training images. Another promising direction for interpretable visual intelligence is to provide explicit symbols at each programming stage, enabling users to intuitively interpret and modify results. Recognizing that current approaches in Image-to-Image translation [16–18] are generally unexplainable, I propose a novel Diffusion Visual Programmer (DVP), a neuro-symbolic image translation framework. The proposed DVP seamlessly integrates a condition-flexible diffusion model within the GPT architecture, orchestrating a coherent sequence of visual programs (*i.e.*, computer vision models) for various pro-symbolic tasks, such as RoI identification, style transfer, and position manipulation. This integration facilitates transparent and controllable image translation processes. Several key features contribute to DVP’s success: First, DVP achieves condition-flexible translation through instance normalization, which eliminates sensitivity caused by manual guidance and allows the model to optimally focus on textual descriptions for high-quality content generation. Second, the framework enhances in-context reasoning by transforming complex high-dimensional concepts in feature spaces into more accessible low-dimensional symbols (*e.g.*, [Prompt], [RoI object]), enabling localized, context-free editing while maintaining overall coherence. Lastly, DVP improves system controllability and explainability by providing explicit symbolic representations at each programming stage, empowering users to intuitively interpret and modify results. This research marks a significant advancement towards harmonizing artificial image translation processes with cognitive intelligence, promising broader applications.

Carbon-Efficient Visual Intelligence System. During my study, the heavy training burden appears with high frequency. When considering human visual intelligence system, they can efficiently and effectively realize vision tasks with low energy cost, this phenomenon inspires me to make investigation on parameter-efficient training to networks. Transformer-based models are nowadays trend in visual related tasks, however, their size continue to grow, and fine-tuning these large-scale pretrained vision models for new tasks has become increasingly parameter-intensive. While parameter-efficient learning emerges as a solution, it often lags behind full fine-tuning in performance. To address this challenge, I study and introduce an Effective and Efficient Visual Prompt Tuning (E²VPT) mechanism. E²VPT incorporates learnable key-value prompts, enhancing the

model’s fine-tuning efficiency. Furthermore, a strategic prompt pruning approach maintains performance while significantly reducing parameters. Another promising avenue toward carbon-efficient visual intelligence system is knowledge distillation. I focus on studying the transformer-based architectures since they are the de-facto standard models for diverse vision tasks owing to their superior performance. As the size of the models, especially transformer-based models, continue to scale up, model distillation becomes extremely important in various real-world applications, particularly on devices limited by computational resources (*i.e.*, edge devices). However, prevailing knowledge distillation methods exhibit diminished efficacy when confronted with a large capacity gap between the teacher and the student, *e.g.*, $10\times$ compression rate. I thus present Automatic Multi-step Distillation (AMD) for large-scale vision model compression. In particular, the distillation process unfolds across multiple steps. Initially, the teacher undergoes distillation to form an intermediate teacher-assistant model, which is subsequently distilled further to the student. An efficient and effective optimization framework is introduced to automatically identify the optimal teacher-assistant that leads to the maximal student performance.

To sum up, by drawing inspiration from the innate capabilities of human visual intelligence, this research underscores the necessity of fostering models that are not just proficient but also versatile, interpretable, and parameter-efficient. My endeavors, ranging from the development of universal visual learners to carving paths for carbon-efficient AI systems, manifest my commitment to driving AI research that resonates with real-world intricacies. It is my fervent hope that the foundations laid in this paper serve as a prompting avenue to the AI community on continually striving for models that seamlessly bridge the gap between machine efficiency and human intuitiveness.

Acknowledgements

I would like to first express my deepest gratitude to my advisor, Prof. Dongfang Liu and Prof. Ying Nian Wu, for their unwavering support, insightful feedback, and invaluable mentorship throughout my research journey. My sincere thanks to the members of my committee, Prof. Qi Yu, and Prof. Bartosz Krawczyk, for their constructive suggestions and essential revisions. On a personal note, I owe my deepest thanks to my family for their unfailing love, patience, and encouragement throughout my academic journey. My partner Qing Shen has been an invaluable support and encouragement. She has been a cornerstone in my life, aiding me in navigating the complexities of work and life. One of the most cherished experiences she has introduced me to is the world of Broadway musicals (*e.g.*, “Alexander Hamilton”, “Hadestown”, “Chicago”). These outings to the theater have become a sanctuary for me, offering a much-needed respite from the rigors of research. My friends Chengxuan Zhang, Naixian Zheng, James C. Liang, Yuxing Wang have been a source of strength and humor, reminding me of life outside of research, and help me survive my hardest time. I also owe my thanks to my lovely cat, Sherri. I’ll always remember the days and nights she spent by my side at the table, even though she once accidentally deleted my code, and I had to spend extra 5 days to fix them ;) Lastly, a big thank you to all who have indirectly contributed to this work and whose names I might have unintentionally left out. Thank you very much for all your support.

Contents

1	Introduction	17
1.1	Universal Visual Learner	17
1.2	Interpretable Visual Intelligence	19
1.2.1	Towards Network Explainability	19
1.2.2	Neural Symbolic Approach for Network Explainability and Control- lability	21
1.3	Carbon-Efficient Visual Intelligence System	23
1.3.1	Visual Prompt tuning	23
1.3.2	Knowledge Distillation	26
2	Background	28
2.1	Universal Visual Learner	28
2.1.1	Motion Task	28
2.1.2	Prototype Learning	28
2.1.3	Transformer Architecture	29
2.2	Interpretable Visual Intelligence	29
2.2.1	Distance-/Prototype-based Classifiers	29
2.2.2	Neural Network Interpretability	30
2.2.3	Metric Learning	31
2.2.4	Clustering-based Self-supervised Representation Learning	31
2.2.5	Image-to-Image Translation	32
2.2.6	Text-guided Diffusion Models	33
2.2.7	Visual Programming	33
2.3	Carbon-efficient Visual Intelligence System	34
2.3.1	Vision Transformers	34
2.3.2	Parameter-efficient Fine-tuning	34
2.3.3	Large-scale Vision Models	35
2.3.4	Model Pruning	35
2.3.5	Knowledge Distillation	36















3	Method	37
3.1	Universal Visual Learner	37
3.1.1	Prototypical TransFormer (ProtoFormer)	37
3.2	Interpretable Visual Intelligence	41
3.2.1	Deep Nearest Centroids (DNC)	41
3.2.2	Image Translation as Diffusion Visual Programmer (DVP)	46
3.3	Carbon-efficient Visual Intelligence System	50
3.3.1	An Efficient and Effective Approach for Visual Prompt Tuning (E ² VPT)	50
3.3.2	Automatic Multi-step Distillation of Large-scale Vision Models (AMD)	54
4	Results	59
4.1	Experiment for ProtoFormer	59
4.1.1	Experiments on Optical Flow	59
4.1.2	Experiments on Scene Depth	60
4.1.3	Diagnostic Experiments	62
4.2	Experiment for DNC	63
4.2.1	Experiments on Image Classification	63
4.2.2	Experiments on Semantic Segmentation	66
4.2.3	Experiments on Point Cloud Segmentation	69
4.2.4	Experiments on Sub-Categories Discovery	71
4.2.5	Study of Ad-hoc Explainability	72
4.2.6	Diagnostic Experiment	74
4.3	Experiment for DVP	76
4.3.1	Comparisons with Current Methods	76
4.3.2	Systemic Diagnosis	78
4.4	Experiment for E ² VPT	82
4.4.1	Comparison with State-of-the-Arts	82
4.4.2	Diagnostic Experiments	85
4.4.3	Visualization	87
4.5	Experiment for AMD	88
4.5.1	Main Results	89
4.5.2	Diagnostic Experiments	89
5	Research Plan	95
5.1	Datasets	95
5.1.1	Universal Visual Learner	95
5.1.2	Interpretable Visual Intelligence	95
5.1.3	Carbon-efficient Visual Intelligence System	96
5.2	Evaluation Metrics	96
5.2.1	Universal Visual Learner	96
5.2.2	Interpretable Visual Intelligence	97



CONTENTS

9



5.2.3 Carbon-efficient Visual Intelligence System	97
5.3 Research Timeline	98

List of Figures

1.1	ProtoFormer as a unified framework considers motion as different levels of dynamics granularity (<i>e.g.</i> , instance-driven flow, pixel-anchored depth, <i>etc.</i>).    are prototypes.	18
1.2	(a) Visual recognition setup. (b) Prevalent visual recognition models  , built upon parametric softmax classifiers, suffering from few limitations, such as their non-transparent decision-making process. (c) Humans  can use past cases as models when solving new problems [19,20] (<i>e.g.</i> , comparing  with a few familiar/exemplar animals). (d) DNC  makes classification based on the similarity of  to class sub-centroids (representative training examples) in the feature space. The class sub-centroids are vital for capturing underlying data structure, enhancing interpretability, and boosting recognition.	20
1.3	Working pipeline showcase. DVP represents a solution rooted in visual programming, demonstrating pronounced capabilities <i>in-context reasoning</i> and <i>explainable control</i> , in addition to its remarkable efficacy in style transfer.	22
1.4	E²VPT (ours) vs concurrent arts (<i>i.e.</i> ,  partial tuning [21],  extra module [22], and  prompt tuning [23] methods) under <i>pretrain-then-finetune</i> paradigm. Our method yields solid performance gains over state-of-the-art fine-tuning methods and competitive to full fine-tuning on a wide range of classification tasks adapting the pretrained ViT-Base/16 [24] as backbone with considerable lower parameter usage.    colors represent results on VTAB-1k [25] <i>Specialized</i> , <i>Natural</i> and <i>Structure</i> , respectively.	24

1.5	A preliminary study on the impact of teacher-assistants with different scales and performance <i>w.r.t.</i> the performance of the student. In (a) and (b), ViT-Tiny and ViT-Base are used as the teacher models, and distilled to a 10% student via teacher-assistants [26] at different scales on CIFAR-10 and CIFAR-100 [27] respectively. There are several key observations: ❶ The performance of the teacher assistant degrades when its scale decreases (see green curve); ❷ The performance of the student varies with different teacher-assistants in scales (see yellow curve); ❸ We ascertain that the Negative Performance-Scale Derivative (NPSD) metric (see §3.3.2) exhibits a positive correlation with the performance of student models (see red curve).	26
3.1	(a) Overall pipeline of ProtoFormer. Movement of a small part of an object within an image is being considered as a rigid motion. In our approach, we use prototypes to understand or predict this kind of motion pattern. (b) In each layer of the <i>Cross-Attention Prototyping</i> (see §3.1.1), there are N sequential iterations encompassing the assignment of feature-prototypes (<i>i.e.</i> , E -step) and the subsequent updating of these prototypes (<i>i.e.</i> , M -step) via Eq. 3.5. (c) Concurrently, the <i>Latent Synchronization</i> process (see §3.1.1) associates the feature representations via the freshly updated motion prototypes, (see Eq. 3.6). For (b) and (c), we apply optical flow for illustration, which demonstrates straightforward systemic explainability.	38
3.2	DNC uses a distance-/case-based criterion to combine unsupervised sub-pattern discovery and supervised representation learning in a synergistic fashion.	42
3.3	Diffusion Visual Programmer (DVP) overview. Our proposed framework contains two core modules:  is the condition-flexible diffusion model (see §3.2.2), augmented by the integration of instance normalization (see Fig. 3.4), aimed to achieve a more generalized approach to translation;  stands for visual programming (see §3.2.2), fulfilled by a series of off-the-shelf operations (<i>e.g.</i> , Segment operation for precise RoI segmentation). The overall neuro-symbolic design enables in-context reasoning for context-free editing. We also enjoy enhanced controllability and explainability by intuitively explicit symbols (<i>e.g.</i> , [Prompt], [RoI object], [Scenario], [Translated object]) at each intermediate stage, facilitating human interpretation, comprehension and modification.	46
3.4	Instance Normalization Guidance.	48

3.5	Overview of our E²VPT framework. Under the <i>pretrain-then-finetune</i> paradigm, only the prompts in the transformer’s input and backbone (§3.3.1), are updated during the fine-tuning process, while all other components remain frozen. We further introduce pruning (§3.3.1) at two levels of granularity (<i>i.e.</i> , token-wise and segment-wise) in (d) to eliminate unfavorable input prompts during rewinding.	50
3.6	Overview of different approaches on knowledge distillation. (a) Traditional distillation methods directly distill the teacher to the student (<i>i.e.</i> , without considering additional teacher-assistant models). (b) Multi-step distillation methods first distill the teacher to a teacher-assistant (requires a large search), which is then further distilled to the student. (c) Manual Multi-step Distillation (MMD) effectively identifies a set of teacher-assistants with different scales and performs multi-step distillation. However, MMD still requires separate distillations with different-scale teacher assistants. (d) Automatic Multi-step Distillation (AMD) efficiently and effectively selects the optimal teacher-assistant through one single optimization, which include three stages: <i>Structural Pruning</i> , <i>Joint Optimization</i> and <i>Optimal Selection</i>	54
4.1	Qualitative results on the Sintel. The red boxes highlight the regions compared. Matchflow [28] appears blurry and ambiguous on textureless and occluded objects, while Flowformer [10] fails to recover complete and detailed information. Ours can estimate clear and complete flow motion, which is closer to ground truth.	61
4.2	Qualitative depth comparison results on the KITTI. The red boxes indicate the highlighted regions. P3Depth [14] and AdaBins [13] have limited receptive fields and do not consider conceptual object-level groupings, thus producing discontinuous and ambiguous predictions. While ours can estimate consistent and sharp depths, which is closer to ground truth.	62
4.3	Visualization of proto-feature mapping , which demonstrates distinct prototypes with similar representations.	64
4.4	<i>Top</i> : sub-centroid images. <i>Bottom</i> : rule created for “goose”.	73
4.5	DNC can provide (dis)similarity-based interpretation. For the two test samples, we only plot the normalized similarities for their corresponding closest sub-centroids from top-4 scoring classes.	74
4.6	Qualitative results with the state-of-the-art baselines. DVP exhibits rich capability in style transfer, achieving realistic quality while retaining high fidelity. Owing to the context-free manipulation (see §3.2.2), the DVP framework is capable of flawlessly preserving the background scenes while specifically targeting the translation of the RoI. Note that while VISPROG also enables context-free editing, it exhibits considerable limitations in rational manipulation (see Fig. 4.8).	77

4.7	Visualization results of instance normalization compared with various guidance scales w	78
4.8	Visualization results of in-context reasoning against attention-based Prompt2Prompt [29] and programming-based VISPROG [30].	79
4.9	Explainable controllability during program execution, which is easy for error assessment and correction.	80
4.10	Human-annotated and <i>Prompter</i>-generated descriptions. Descriptions generated via the <i>Prompter</i> yield finer reconstructions, as well as subsequent translated outputs, suggesting a relaxation of label dependency.	81
4.11	Hyperbolic visualization results from VPT [23] and ours on 3 FGVC tasks (<i>i.e.</i> , FGVC CUB-200-2011 [31], Oxford Flowers [32] and Stanford Dogs [33]). Our method shows consistently better clustering pushed to the border of the Poincaré disk.	86
4.12	Sensitivity of input prompt and key-value prompt lengths. We vary the number of prompts for different combinations, and show their results on VTAB-1k <i>Natural</i> SVHN [34].	86
4.13	The sufficiency of using one teacher-assistant.  colors represent the performance of AMD_{Min} using zero, one, two, and three teacher-assistants, respectively. The results from the training on the ViT-Base CIFAR-100 is posited at (a), CIFAR-10 at (b).	91
4.14	Efficient training schedule. Our proposed AMD enjoys superior performance among the overall training scene.	91
4.15	Weight values from different loss objectives. We present the performance on different values of α and β from Eq. 3.29. The highest performance is marked in red (<i>i.e.</i> , $\alpha = 0.2, \beta = 100$).  colors represent performance with respect to different $\beta \in \{1, 10, 50, 100\}$	92

List of Tables

4.1	Quantitative results on standard Sintel and KITTI flow benchmarks. ‘A’ denotes the Autoflow dataset; ‘C + T’ denotes training on the FlyingChairs and FlyingThings datasets only; ‘C + T + S + K + H’ fine-tunes on a combination of Sintel, KITTI, and HD1K training sets. Error metrics are lower is better with “↓”, and accuracy metrics are higher is better with “↑”. Same for Table 4.2.	60
4.2	Quantitative results on Sintel and KITTI depth datasets. With both test data unseen by the model, we can achieve leading performance over state-of-the-art methods [13, 14, 35–38].	61
4.3	A set of ablative studies on optical flow (see §4.1.3). The best performances are marked in bold	62
4.4	Classification top-1 accuracy on CIFAR-10 [39] test . #Params: the number of learnable parameters (same for other tables).	64
4.5	Classification top-1 accuracy on CIFAR-100 [39] test	65
4.6	Classification top-1 and top-5 accuracy on ImageNet [40] val	65
4.7	Classification top-1 and top-5 accuracy on CUB-200-2011 test [31].	66
4.8	Classification top-1 and top-5 accuracy on ImageNetv2 test sets [41].	67
4.9	Segmentation mIoU score on ADE20K [42] val , Cityscapes [43] val and COCO-Stuff [44] val , respectively (top-1 acc. on ImageNet [40] val of backbones are also reported for reference).	68
4.10	Semantic segmentation results on S3DIS [45] Area 5 and ScanNet v2 [46] (Ranked by S3DIS’s results).	69
4.11	Instance segmentation results on S3DIS [45] Area 5 and ScanNet v2 [46].	69
4.12	Panoptic segmentation results on SemanticKITTI [47] test	69
4.13	Top-1 induction accuracy on CIFAR-100 [39] test using CIFAR-20 pre-trained models. Numbers reported with k -nearest neighbor classifiers. See §4.2.4.	72
4.14	Top-1 induction accuracy on ImageNet [40] val using ImageNet-127 pre-trained models. Numbers reported with k -nearest neighbor classifiers. See §4.2.4.	72

4.15	Classification top-1 and top-5 accuracy on ImageNet [40] val, using cluster center <i>vs</i> resembling real observation as class sub-centroids, based on DNC-ResNet50 architecture.	73
4.16	A set of ablative experiments on ImageNet [40] val, ADE20K [42] val and S3DIS [45] Area 5.	75
4.17	User study, CLIP-Score, and DINO-Score on the comparison between our proposed framework and state-of-the-art baselines. The metrics are detailed in §5.2.2.	78
4.18	Ablative results on instance normalization guidance and label efficiency.	82
4.19	Image classification accuracy for ViT-Base/16 [24] pretrained on supervised ImageNet-21k. Following [23], we report the average test accuracy (three runs) on FGVC [23] and VTAB-1k [25] benchmarks, and “Number of Wins” in [·] compared to full fine-tuning (Full) [48]. “Tuned/Total” is the average percentage of tuned parameters required by 24 tasks. “Scope” indicates the tuning scope of each method. “Additional parameters” is the existence of parameters in addition to the pretrained backbone and linear head. The highest accuracy among all approaches except FULL are shown in bold . E ² VPT outperforms the full fine-tuning in 19 of 24 instances with far fewer trainable parameters. More impressively, we further report “Number of Wins to VPT” in {·}. Our method beats VPT in 21 of 24 cases with considerably lower parameters. Same for Table 4.20 and 4.21.	83
4.20	Image classification accuracy for Swin-Base [49] pretrained on supervised ImageNet-21k.	83
4.21	Image Classification accuracy for different pretrained objectives — MAE [50] and MoCo v3 [51] with ViT-Base [24] as backbone. Our method enjoys significant performance gains to VPT [23] while having lower parameter usage.	84
4.22	Impact of different components in E ² VPT on two instances: VTAB-1k <i>Natural</i> SVHN [34] and FGVC NABirds [52].	84
4.23	Prompt location and Initialization results on VTAB-1k [25] in three runs.	85

4.24	The results of knowledge distillation methods on CIFAR-10 [27] and CIFAR-100 [27] datasets. All results are reported using the same teacher and student (10% scale). The best results are highlighted in bold , and the second best are shown in <u>underline</u> . Follow [53], as our student design via <i>Structural Pruning is orthogonal</i> to current approaches (see §3.3.2), we rerun and reproduce the results of all methods based on author-provided or author-verified code, and report the average performance over five runs. The GPU hours are estimated with respect to their conventional counterparts. ↓ denotes the performance lost with respect to their teacher models, a smaller gap represent a superior performance.	88
4.25	Comparison results on ImageNet [40] dataset.	89
4.26	Impact of student model scalability ranging from 5% to 20%. In our study, we deliberately exclude models of a higher scale, taking into full account the constraints imposed by computational capacities.	90
4.27	Impact of candidate scaling m	92
4.28	Impact of different loss components , including three variants from original training objectives (see Eq. 3.29)	93
4.29	Discussion on NPSD metric . We further introduce a general form of NPSD metric as λ -NPSD.	93
5.1	User study on controllability and user-friendliness.	97
5.2	Overall timeline for the Ph.D. research dissertation.	98

Chapter 1

Introduction

1.1 Universal Visual Learner

“All is flux, nothing is stationary.”

– Heraclitus [54]

The aphorism attributed to Heraclitus underscores the foundation of physics in the natural world. Understanding motion has the potential to unveil the intricate secrets of intelligence, shedding light on the systematic construction of artificial entities [55]. However, the proliferation of excessively granular motion tasks has shifted the focus towards *specialized* models within deep learning. This shift contrasts sharply with the longstanding scientific tradition of seeking *general* solutions to elegantly describe physical phenomena in the universe. This raises the following question: ① Can we discover a *unified* model that serves as a comprehensive motion learner?

Motion learning tasks fundamentally involve pixel-level dynamics and correspondences, such as optical flow and depth scene estimation. A prevalent challenge is the presence of photometric and geometric inconsistencies, such as shadows and occlusions, which introduce significant uncertainty during the matching process [56, 57]. Consequently, the accuracy of pixel-wise feature matching is compromised, detrimentally impacting the learning of the underlying motion representation. A promising solution to this challenge is prototype learning [58, 59], which categorizes motion measurements into discrete exemplars. Within each exemplar, a prototype serves as a central archetype, encapsulating the essential attributes of its associated motion patterns observed in the data. Clustering similar patterns around prototypes can effectively reduce the impact of noise and outlier pixels in feature matching, thereby significantly mitigating uncertainty. In this context, we can address question ① by asking: How can we design a model that incorporates the principles of prototype learning in motion tasks?

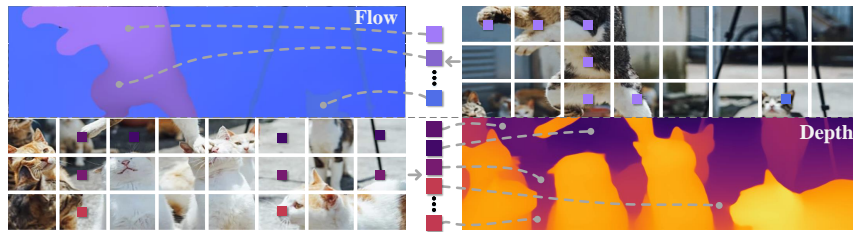


Figure 1.1: **ProtoFormer as a unified framework** considers motion as different levels of dynamics granularity (e.g., instance-driven flow, pixel-anchored depth, etc.). ■■■ are prototypes.

Recently, the Transformer architecture has attained ubiquitous adoption, enjoying unambiguous acclaim in both the domains of vision and language [60–62]. Its accomplishments are underpinned by the attention mechanism, endowing models with the capability to selectively attend to salient entities within input data. The capacity to generate context-aware feature representations represents a substantial enhancement of the model’s effectiveness, enabling it to apply as a general solution to diverse vision tasks. Inspired by its encouraging success, our inquiry naturally delves into a more specific dimension: ③ How to incorporate the prototype learning capacity into the architecture of Transformer?

To this demand, we employ Prototypical TransFormer (ProtoFormer), a unified motion solution. Specifically, ProtoFormer incorporates prototype learning with Transformer. The method first tokenizes images features into patches, where the features are initialized into distinct prototypes. These prototypes are recursively updated via *Cross-Attention Prototyping* (§3.1.1) to capture representative motion characteristics through clustering. After assignments and updates, *Latent Synchronization* (§3.1.1) builds up prototype-feature association, which helps denoise and mitigate motion ambiguity. The refined features are finally fed into the decoder for task-specific predictions.

ProtoFormer exhibits several compelling attributes. ❶ **Architectural elegance:** ProtoFormer leverages a prototype-guided Transformer architecture, allowing it to handle heterogeneous motion tasks with different levels of dynamic granularity in a unified manner (see Fig. 1.1). ❷ **Predictive robustness:** Prototype learning inherently diminishes noisy outliers through its density criterion (§2.1.2). By anchoring on recursively refined prototypes, feature learning is guided towards more robust representations (§3.1.1), thus offering a viable solution to the challenge of motion ambiguity (see Fig. 4.1 and Fig. 4.2). ❸ **Systemic explainability:** The density-based nature of recursive prototyping provides intuitive visual demonstrations of motion prototypes (see Fig. 4.3), enabling direct interpretation of various dynamic patterns captured by the system.

1.2 Interpretable Visual Intelligence

In §1.1, I investigate a unified approach for various visual comprehension tasks, under the spirit of clustering. Its transparency is built upon the *post-hoc* explainability. While various studies [63–66] have shown that such explanations might not be sufficient enough for higher-level transparency, I thus study the possibility of *ad-hoc* explainability, integrating case based reasoning to network design in §1.2.1. Furthermore, I investigate another promising path to network controllability and explainability in §1.2.2.

1.2.1 Towards Network Explainability

Deep learning models, ranging from convolutional networks (*e.g.*, VGG [67], ResNet [68]) to Transformer-based architectures (*e.g.*, Swin [49]), have significantly advanced the field of visual recognition. With these advancements, parametric softmax classifiers, which learn a set of class-specific parameters, *i.e.*, weight vector, and bias term, have become the *de facto* regime in the area (Fig. 1.2(b)). However, due to their parametric nature, these classifiers encounter several limitations: **First**, they are non-tangible because their parameters are abstract and not inherently linked to the physical nature of the modeled problem [69]. Thus, these classifiers pose challenges in providing explanations that can be easily interpretable by humans [70]. **Second**, these linear classifiers primarily focus on optimizing classification accuracy, with less emphasis on capturing the latent data structure. For each class, only one single weight vector is learned in a fully parametric manner, assuming *unimodality* for each class [71, 72] and exhibiting less tolerance for intra-class variation. **Third**, deep parametric classifiers, where each class has its own set of parameters, require the output space to have a fixed dimensionality equal to the number of classes [73]. This fixed output space restricts their transferability when trying to utilize ImageNet-trained classifiers to initialize segmentation networks (*i.e.*, pixel classifiers). Consequently, the knowledge retained in the form of trainable parameters from the image classification task must be discarded.

To address these limitations of parametric softmax classifiers, we introduce deep nearest centroids (DNC), a powerful and nonparametric classification network (Fig. 1.2(d)). Nearest Centroids, which has historical roots dating back to the dawn of artificial intelligence [74–79], is arguably the simplest classifier. Nearest Centroids operates on an intuitive principle: A test sample is directly classified to the class of training examples whose mean (centroid) is closest to it. Apart from its internal transparency, Nearest Centroids is a classical form of exemplar-based reasoning [70, 76], which is fundamental to our most effective strategies for tactical decision-making [80] (Fig. 1.2(c)). Numerous past studies [19, 20, 81] have shown that humans learn to solve new problems by utilizing past solutions to similar problems. Despite its conceptual simplicity and empirical support in cognitive sciences [82–84], the utility of Nearest Centroids in large datasets with high-dimensional input spaces remains largely unexplored or overlooked by the current community. Building upon the intuitive

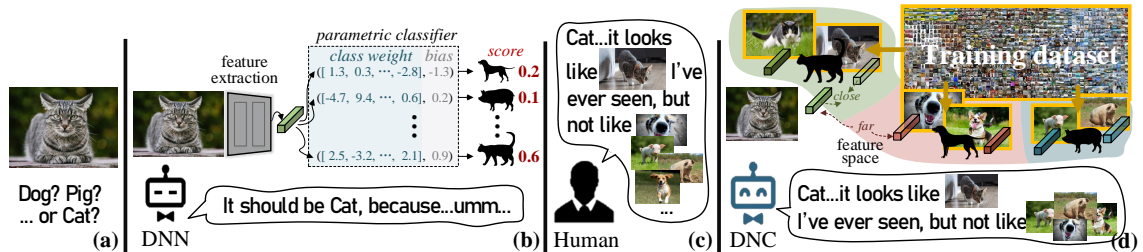


Figure 1.2: (a) Visual recognition setup. (b) Prevalent visual recognition models DNN , built upon parametric softmax classifiers, suffering from few limitations, such as their non-transparent decision-making process. (c) Humans Human can use past cases as models when solving new problems [19, 20] (e.g., comparing Image with a few familiar/exemplar animals). (d) DNC DNC makes classification based on the similarity of Image to class sub-centroids (representative training examples) in the feature space. The class sub-centroids are vital for capturing underlying data structure, enhancing interpretability, and boosting recognition.

strength of Nearest Centroids, the proposed DNC serves as a *strong yet interpretable backbone* for large-scale visual recognition. It fully addresses the aforementioned limitations of parametric classifiers while demonstrating superior performance.

Specifically, DNC condenses each class into a collection of sub-centroids (sub-cluster centers) by clustering the training data within the same class. Each test sample is then assigned to the class with the nearest sub-centroid. DNC operates as an *experience/distance*-based classifier by relying solely on the proximity of test query to the local means of training data (“quintessential” past observations) in the deep feature space. As such, DNC learns visual recognition by directly optimizing the representation, eliminating the need for deep parametric models that require an additional softmax classification layer after feature extraction. During training, DNC alternates between two steps: **i)** *class-wise clustering* for automatic discovery of class sub-centroids, and **ii)** *classification prediction* for supervised representation learning by retrieving the nearest sub-centroids. However, as the feature space evolves continually during training, computing the sub-centroids becomes computationally expensive – it requires a pass over the full training dataset after each batch update, which limits the scalability of DNC. To overcome this problem, we employ a Sinkhorn Iteration [85] based clustering algorithm [86] for fast cluster assignment. We further adopt momentum update with an external memory for estimating online the sub-centroids (whose amount is more than 1K on ImageNet [40]) with small-batch size (e.g., 256). Consequently, DNC can be trained efficiently by simultaneously conducting clustering and stochastic optimization on large datasets with small batches, with only a minor reduction in training speed (e.g., $\sim 5\%$ on ImageNet).

Overall, DNC enjoys several appealing qualities: **First**, improved *simplicity* and *transparency*. The intuitive working mechanism and statistical interpretation of class sub-centroids make DNC elegant and intuitive to understand. **Second**, automated discovery

of the *underlying data structure*. Through within-class deterministic clustering, the latent distribution of each class is automatically discovered and captured as a set of representative local means. In contrast, parametric classifiers learn a single weight vector per class and are oblivious to rich intra-class variations. **Third**, direct supervision of *representation learning*. DNC make classification decisions by comparing data samples and class sub-centroids in the feature space. Its distance-based nature allows DNC to blend unsupervised sub-pattern mining (class-wise clustering) and supervised representation learning (nonparametric classification) synergistically. Local significant patterns are automatically mined to facilitate classification decision-making, and the supervisory signal from classification directly optimizes the representation, which in turn boosts meaningful clustering. **Fourth**, strong *modularity* and *transferability*. DNC learns by optimizing *only* the feature representation, eliminating the need for the output dimensionality to match the number of classes. With this algorithmic merit, all useful knowledge (parameters) learned from a source task (*e.g.*, ImageNet classification) is stored in the representation space and can thus be completely transferred to target tasks (*e.g.*, Cityscapes segmentation). Demonstrating versatility across diverse tasks and datasets, DNC exhibits robust performance via a plug-and-play approach, underscoring its inherent modularity. **Fifth**, *ad-hoc explainability*. By restricting the class sub-centroids to be samples (images) of the training set, DNC gains the ability to explain its predictions based on *IF...Then* rules. This allows users to intuitively view the class representatives and appreciate the similarity of the test data to the representative images. Such *ad-hoc* explainability [87] is valuable in safety-sensitive scenarios and differentiates DNC from most existing network interpretation techniques [88–90] that only investigate *post-hoc* explanations and thus fail to elucidate precisely how a model works [64, 91, 92].

1.2.2 Neural Symbolic Approach for Network Explainability and Controllability

Current state-of-the-art image translation methods predominantly adhere to the connectionism paradigm [93], focusing on answering the question of “*what*” — specifically, translating an image from the source domain to the target domain with high fidelity. These methods can be broadly categorized into *Generative Adversarial Network (GAN)-based* and *Diffusion-based* approaches (see §2.2.5). Diffusion-based methods often exhibit superior performance compared to their GAN-based counterparts, particularly in terms of image quality and coherence, training stability, and fine-grained control [94]. Consequently, diffusion-based approaches are perceived to have greater potential in the field of image translation [29].

Despite their significant success [94, 95], these diffusion-based methods exhibit several limitations: ① *Condition-rigid learning*. Existing methods based on the principles of classifier-free guidance [29, 96] face a significant challenge in achieving a harmonious balance between unconditional and conditional predictions. Typically, these approaches rely

on manually crafted guidance scale parameters to control the translation process for each individual image. This inherent limitation restricts algorithmic scalability, thus hindering their potential for fully automated applications in real-world scenarios; ② *Context-free incompetence*. Current methods predominantly focus on the global manipulation of various attributes (*e.g.*, stylistic and content-related elements) within images, prioritizing the preservation of contextual integrity over local modifications. However, the lack of context-free reasoning impedes the precision needed for specific RoI modifications while maintaining overall coherence. Achieving such contextual understanding requires a high degree of semantic acuity and a robust comprehension of image structure, which contemporary image translation solutions [97–99] largely lack; ③ *System opacity*. Due to their black box nature, diffusion-based methods — rooted in connectionism [1] — often exhibit a level of abstraction that detaches them from the intrinsic physical characteristics of the problems they aim to model [69]. Consequently, users have limited control over the model’s behaviors prior to obtaining the final outputs, making it challenging to establish trustworthiness in decision-making or pursue systematic improvements.

In this work, we revolutionize the focus of “*what*” into a more flexible and controllable image translation paradigm of “*where-then-what*.” Under this paradigm, we first answer the question of “*where*” by finding the instructed target region accurately. After getting the target region (*i.e.*, Region of Interest (RoI)), we answers the question of “*what*” by translating it into the targeted domain with high-fidelity. Considering the aforementioned discussions, we are approaching the task of image translation from a fresh neuro-symbolic perspective, presenting a novel method named the Diffusion Visual Programmer (DVP). More concretely, we architect an condition-flexible diffusion

model [100], harmoniously integrated with the foundation of GPT [101], which orchestrates a concatenation of off-the-shelf computer vision models to yield the coveted outcomes. In Fig. 1.3, we present a possible procedure of DVP execution process. Given an input instruction that specifies the translation from the “left dog” to a “sheep,” DVP first utilizes GPT as an AI agent to plan a sequence of programs with operations, subsequently invoked in the following procedure. It first addresses the fundamental question of “*where*” by identifying and segmenting the RoI of the “dog.” After getting the segmentation result, the background undergoes an inpainting to restore the regions obscured by the “dog,” and

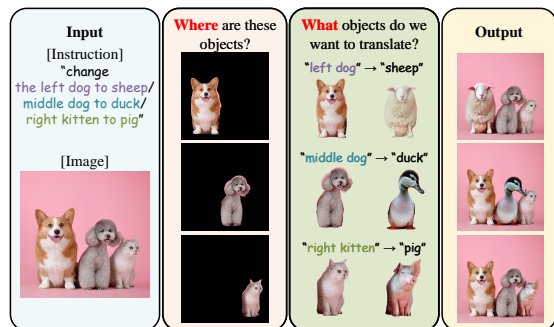




Figure 1.3: **Working pipeline showcase.** DVP represents a solution rooted in visual programming, demonstrating pronounced capabilities *in-context reasoning* and *explainable control*, in addition to its remarkable efficacy in style transfer.

our condition-flexible diffusion model translates  \rightarrow , addressing the query of “*what*.” DVP, leveraging spatial data, positions the “sheep” in the instructed spatial location, and further enables various context-free manipulations (see §3.2.2).

DVP is an intuitive and potent image translation framework, showing superior performance qualitatively and quantitatively (§4.3.1) over state-of-the-art approaches [16–18, 29, 102–106]. It enjoys several attractive qualities: ❶ **Condition-flexible translation.** Our proposed condition-flexible diffusion model creatively utilizes instance normalization guidance (see §3.2.2) to mitigate the global noises in the unconditional embeddings by adaptive distribution shifting, and ensure that the model remains optimally conditioned based on the textual descriptions without any parameter engineering. It offers a streamlined solution tackling the challenge of hand-crafted guidance sensitivity on condition-rigid learning. This innovation addresses the “*what*” inquiry and enables a generalized learning paradigm for diffusion-based solutions. ❷ **Effective in-context reasoning.** By decoupling the high-dimensional, intricate concepts in feature spaces into low-dimensional, simple symbols (*e.g.*, [Prompt], [RoI object]), we enable context-free manipulation of imagery contents via visual programming (see §3.2.2). It essentially includes a sequence of operations (*e.g.*, segmentation, inpainting, translation) to establish in-context reasoning skills. Such a neuro-symbolic design fortifies our method with the capability to discern the concept of “*where*” with commendable precision. ❸ **Enhanced controllability and explainability.** Our modulating scheme uses explicit symbolic representations at each intermediate stage, permitting humans to intuitively interpret, comprehend, and modify. By leveraging a step-by-step pipeline, we introduce not only a novel strong baseline but also a controllable and explainable framework to the community (see §4.3.2). Instead of requiring the redesign of networks for additional functions or performance enhancement, our approach is distinguished by its user-centric design, enabling the seamless integration of future advanced modules.

1.3 Carbon-Efficient Visual Intelligence System

1.3.1 Visual Prompt tuning

The development of artificial intelligence (AI) should prioritize not only performance advancements but also sustainable deployment [107–110]. Despite the compelling pursuit of performance improvements in visual-related tasks, the size of current models has been rapidly increasing, leading to energy-intensive and computationally expensive training processes [111–113]. Currently, Transformer-based architectures dominate visual-related models (*e.g.*, ViT-Huge [24] (632M) and Swin-Large [49] (197M)) with significantly more parameters than the Convolutional Neural Networks (CNN) like ResNet [68] (25M). Training such large models from scratch poses challenges, including limited data [101, 114, 115] and slow convergence at low accuracy [116, 117]. A prevalent approach to overcoming

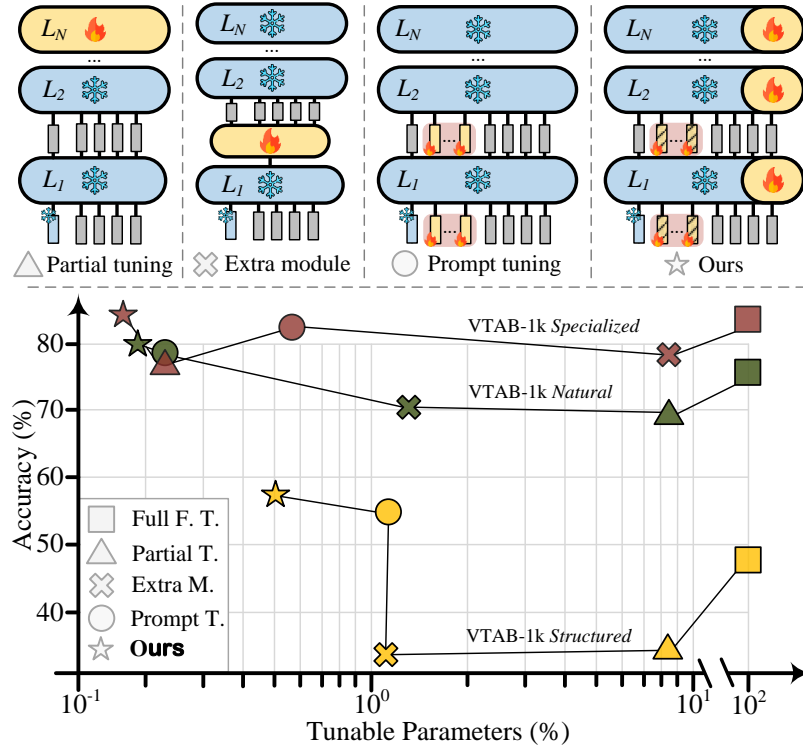


Figure 1.4: **E²VPT (ours)** vs concurrent arts (*i.e.*, \triangle partial tuning [21], \otimes extra module [22], and \circ prompt tuning [23] methods) under *pretrain-then-finetune* paradigm. Our method yields solid performance gains over state-of-the-art fine-tuning methods and competitive to full fine-tuning on a wide range of classification tasks adapting the pre-trained ViT-Base/16 [24] as backbone with considerable lower parameter usage. \blacksquare \blacksquare \blacksquare colors represent results on VTAB-1k [25] *Specialized*, *Natural* and *Structure*, respectively.

these challenges is *pretrain-then-finetune*, which diminishes the need for extensive training data and accelerates the processing of various visual tasks. However, the traditional *full fine-tuning* involves storing and deploying a complete copy of the backbone parameters for every single task [23], which remains computationally expensive and unsuitable for fast model deployment.

To address this issue, various approaches have been developed, which can be divided into three main categories (see Fig. 1.4): partial tuning, extra module, and prompt tuning methods. *Partial tuning* methods [51, 118–121] only fine-tune part of the backbone, such as the classifier head or last few layers, while freezing the others. *Extra module* methods insert learnable bias term [22] or additional adapters [122, 123] to the network for adaptation. *Prompt tuning* methods add prompt tokens [23, 124–127] to the input layer of the transformer without altering or fine-tuning the backbone itself. All of these methods

operate within the *pretrain-then-finetune* paradigm, which reduces the number of learnable parameters compared to full fine-tuning [51, 118, 119, 122, 123]. However, despite their promising results, there are two main limitations in existing parameter-efficient methods. **First**, they do not examine the core architecture of the transformer’s self-attention mechanism, resulting in a significant performance gap compared to full fine-tuning. **Second**, they typically require fine-tuning a relatively large number of parameters to achieve reasonable performance, failing to explore the extremes of parameter efficiency.

The perspective outlined above raises two fundamental questions: ❶ *How can we establish the effectiveness of prompt tuning for large-scale Transformer-based vision models?* ❷ *How can we explore the extremes of parameter efficiency to minimize the number of tunable parameters?* These questions form the cornerstone of our research. Our approach is guided by the intuition that, rather than merely modifying inputs as in traditional prompt tuning methods, we should explicitly investigate enhancing the self-attention mechanism during fine-tuning and push the boundaries of parameter efficiency.

To address question ❶, we discuss and analyze the self-attention mechanism of the transformer, which is vital for capturing long-range token dependencies within a global context [128–130]. Beyond traditional input visual prompts, we introduce learnable key-value prompts that are integrated into the Key and Value matrices of the self-attention layers. These key-value prompts are jointly learned with the input visual prompts during fine-tuning. This innovative approach capitalizes on the sophisticated prompt architecture of transformers, leading to substantial performance enhancements. Furthermore, it offers a versatile plug-and-play prompt module for existing transformer architectures, presenting a fine-tuning solution that is conceptually distinct from all previously mentioned methods in the vision domain. Motivated by ❷, We propose a pruning strategy to further reduce the number of parameters while maintaining model performance. Our approach is inspired by the lottery ticket hypothesis (LTH) [131, 132], which suggests that for any given task, there exists a sub-network capable of matching the test accuracy of the original over-parameterized network without the extraneous weights [133–137]. Building on this concept, we re-evaluate the core design of prompt tuning methods to reduce the number of learnable parameters. Specifically, we aim to retain only those prompt tokens that significantly contribute to performance, pruning those that are redundant or unnecessary during fine-tuning. This pruning process enhances the efficiency of prompt tuning while preserving the model’s overall performance.

To this end, we propose **E²VPT**, namely **E**ffective and **E**fficient **V**isual **P**rompt **T**uning. E²VPT is a novel prompt tuning framework that is both architecture-aware and pruning-anchored (see Fig. 1.4). In §2.3, we conduct a literature review and discuss relevant works. Our proposed approach is presented in §3.3, where we describe in detail how we design visual and key-value prompts to achieve superior performance with fewer parameters.

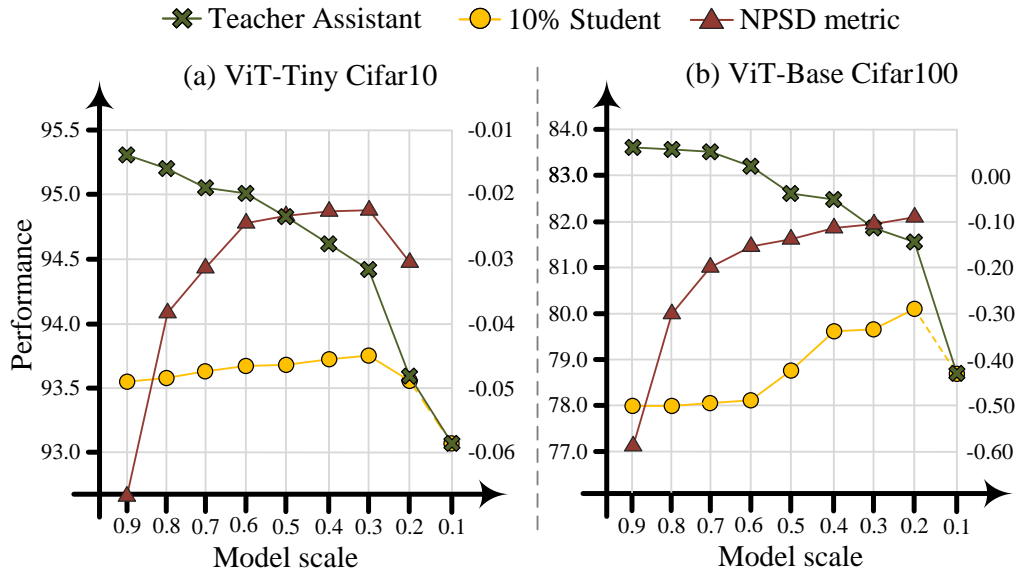


Figure 1.5: **A preliminary study on the impact of teacher-assistants with different scales and performance *w.r.t.* the performance of the student.** In (a) and (b), ViT-Tiny and ViT-Base are used as the teacher models, and distilled to a 10% student via teacher-assistants [26] at different scales on CIFAR-10 and CIFAR-100 [27] respectively. There are several key observations: ❶ The performance of the teacher assistant degrades when its scale decreases (see green curve); ❷ The performance of the student varies with different teacher-assistants in scales (see yellow curve); ❸ We ascertain that the Negative Performance-Scale Derivative (NPSD) metric (see §3.3.2) exhibits a positive correlation with the performance of student models (see red curve).

1.3.2 Knowledge Distillation

Recent advancements in foundation models for vision, such as BiT [138], ViT [24], Swin [49], and Florence [139], have garnered considerable attention due to their groundbreaking performance across a range of tasks. Transformer-based architectures like ViT-Large [24] (61.6 GFlops) and Swin-Large [49] (103.9 GFlops) exemplify this new class of visual foundation models [140], showcasing significantly more complex operations compared to traditional Convolutional Neural Networks (CNNs) such as ResNet18 [68] (1.8 GFlops). However, as these models scale up to achieve higher performance, their inherent complexity becomes a bottleneck, posing significant challenges for deployment on low-power processors and mobile devices, which often have constrained computational resources [2, 53, 141, 142].

A conventional method for leveraging the high performance of large-scale models while

managing limited computational resources is knowledge distillation [143]. This technique involves transferring the knowledge acquired by large teacher models to smaller, more deployable student models. Although effective in numerous applications, recent studies [26, 144] have shown that traditional distillation methods experience significant performance drops when there is a substantial capacity gap between teacher and student models. To address this, multi-step distillation approaches [145, 146] have been proposed, where the teacher model is first distilled into an intermediate teacher-assistant model, which then aids in transferring knowledge to the student model. While multi-step distillation generally enhances student model performance, our preliminary study (Figure 1.5) indicates that the effectiveness of the teacher-assistant model greatly influences the final student performance. However, selecting the appropriate scale/size for the teacher-assistant model often lacks clear guidance, necessitating trial training. This process, which involves evaluating all possible scales to identify the optimal teacher-assistant, incurs significant computational costs.

To address this challenge, we introduce an Automatic Multi-step Distillation (AMD) approach for large vision model compression. AMD employs a cascade strategy that includes three phases. First, *Structural Pruning* utilizes a grating and pruning algorithm to create a range of teacher-assistant architectures at different scales. Second, *Joint Optimization* involves a framework that efficiently identifies the best teacher-assistants across all scales in a single run. Finally, *Optimal Selection* chooses the most suitable teacher-assistant from the candidates using the Negative Performance-Scale Derivative (NPSD) metric, which assesses the performance-scale optimality of each candidate. In summary, the results we present hold particular significance given the limited research on compressing vision models with substantial capacity differences. We believe that this study advances the foundational understanding of this domain and opens new avenues for future research.

Chapter 2

Background

This chapter describes important preliminaries on the universal visual learner (§1.1), interpretable visual intelligence (§1.2) and carbon-efficient visual intelligence system (§2.3).

2.1 Universal Visual Learner

2.1.1 Motion Task

Motion tasks encompass the identification, modeling, and prediction of movement patterns in objects and scenes, serving as the cornerstone for a wide array of computer vision applications. These include vehicle and pedestrian motion detection [147–150], abnormal activity detection [151–153], and video compression [154–156]. Within this domain, optical flow [7–9] and depth estimation [10, 11, 157] are particularly influential, significantly impacting motion-related downstream tasks such as object tracking and video stabilization. Current research efforts largely focus on task-specific solutions, leading to duplicated work and inefficient hardware usage. In contrast, ProtoFormer represents a novel approach by striving to integrate motion tasks under a unified paradigm. This conceptual innovation sets ProtoFormer apart from existing methods in the field, offering a more cohesive and efficient framework for tackling motion-related challenges in computer vision.

2.1.2 Prototype Learning

Traditionally, prototype learning in machine learning involves establishing a metric space where features are differentiated by calculating their distances or densities relative to prototypical representations [158]. Early techniques included classical methods such as support vector machines [159], random forests [160], and logistic regression [161]. With the rise of deep neural networks (DNNs), prototype-based models have found extensive applications in areas such as few-shot learning [162–166], zero-shot learning [167, 168], and explainable classifiers [1, 4, 72]. In this context, we propose that movements within the same

object or nearby regions exhibit significant similarities, forming clusters of prototypes. By integrating prototype learning into model design, we can naturally encapsulate diverse dynamic characteristics, thereby enhancing the model’s capacity to understand motion across various contexts. This integration facilitates a more nuanced comprehension of motion patterns, improving the overall performance and interpretability of motion-related tasks.

2.1.3 Transformer Architecture

The groundbreaking success of Transformers in natural language processing (NLP) [101, 169–172] has led to their widespread adoption in vision-related tasks, such as image classification [1, 24, 49, 173] and image segmentation [174–177]. Transformers have proven to excel in these visual applications, often surpassing convolutional neural networks (CNNs) [140]. This advantage stems from their capability to capture extensive token dependencies in a global context, addressing a key limitation of CNN-based methods that primarily focus on local interactions within convolutional layers [128–130, 140, 178]. The distinctive attention mechanism in Transformers facilitates the comprehension of global spatial relationships, making them particularly suitable for motion-related tasks where extensive spatial interconnections are crucial. By integrating the attention mechanism with prototype learning, we aim to leverage the representational strength of Transformers to unravel complex patterns in motion tasks, providing a unified, Transformer-based solution.

2.2 Interpretable Visual Intelligence

2.2.1 Distance-/Prototype-based Classifiers

Among the various classification algorithms, such as logistic regression [179], Naive Bayes [180], random forest [160], support vector machines [159], and deep neural networks (DNNs) [181], distance-based methods are distinguished by their intuitive mechanism. Distance-based classifiers are nonparametric and exemplar-driven, relying on similarities between samples and stored exemplars or prototypes. They employ case-based reasoning that mimics the natural problem-solving approach of humans, making them both appealing and interpretable [19, 66]. One prominent example is the k -Nearest Neighbors (k -NN) algorithm [74, 75], which utilizes all training data as exemplars [182, 183]. Significant progress has been made in implementing k -NN within neural networks [184–186], notably by Wu *et al.* [187], whose k -NN network outperforms the parametric softmax-based ResNet [68] and excels in few-shot learning scenarios. However, k -NN classifiers, including their deep learning counterparts, require substantial storage and computational resources, as they must retain the entire training dataset and perform full-dataset retrieval for each query [188, 189]. Additionally, the nearest neighbors may not always serve as good class representatives [66]. Nearest Centroids classifier [76–79] can mitigate some of k -NN’s

deficiencies. Nearest Centroids uses representative class centers as exemplars instead of all training data [66, 83]. Guerriero *et al.* [190] explored incorporating Nearest Centroids into DNNs. However, their approach abstracts each class into one single mean, failing to capture complex class-wise distributions and producing suboptimal results even on small datasets like CIFAR [39].

The concept of distance-based classification has also inspired the development of *prototypical networks*, which are particularly prominent in few-shot [162, 191] and zero-shot [167, 168] learning. These networks typically associate a single representation (prototype) with each class [192], and these prototypes are often either flexible parameters [168, 189, 191] or predefined before training [73, 167]. In DNC, a prototype (sub-centroid) serves as a generalization of several observations or a representative training example. By leveraging clustering-based sub-class mining, DNC addresses two key properties of prototypical exemplars: *sparsity* and *expressivity* [193, 194]. This approach enables the representation to capture the intrinsic structure of each class, facilitating large-scale visual recognition while maintaining transparency and interpretability.

2.2.2 Neural Network Interpretability.

Given the constraints of the *black-box* nature of deep neural networks (DNNs) in decision-critical tasks, there has been a growing interest in enhancing the interpretability of DNNs. Most existing interpretability techniques, however, offer post-hoc explanations for already-trained models. These techniques typically involve analyzing reverse-engineered importance values [88–90, 195–200] and input sensitivities [201–204]. As highlighted in the literature, these post-hoc explanations can be problematic and misleading [63–66], as they often fail to elucidate the actual reasoning process behind a DNN’s decisions [205]. In pursuit for *ad-hoc* explainability, recent efforts have focused on developing inherently interpretable DNNs by incorporating more transparent mechanisms into the models [206–208]. These approaches include regularizing representations with specific properties, such as sparsity [209], decomposability [210], and monotonicity [211], to enhance the transparency and comprehensibility of the decision-making process.

DNC fundamentally relies on the retrieval of class sub-centroids, offering theoretical simplicity and clarity. Anchoring these sub-centroids to available observations provides a natural criterion for measuring the similarity between a test sample and representative data. This dual function of representation learning and case-based reasoning enables DNC to be inherently self-explainable, eliminating the need for *post-hoc* analysis [87]. DNC shares similarities with *concept*-based explainable networks [69, 70, 205–207, 212–214] that integrate human-friendly concepts or prototypes into the decision-making process. However, these methods typically require significant architectural modifications and often rely on pre-trained models, which serve as backbone networks. In stark contrast, DNC introduces minimal architectural changes to parametric classifier-based DNNs and demonstrates exceptional performance on ImageNet [40] through training from scratch, all while

providing *ad-hoc* explainability. To our knowledge, this is the first instance of solid empirical evidence showcasing the effectiveness of case-based reasoning in large-scale visual recognition.

2.2.3 Metric Learning

The objective of metric learning, also known as distance learning, is to develop a distance metric or embedding that clusters similar samples while separating dissimilar ones. This concept has a long history, with its roots extending back several decades [215, 216]. Over the years, various objective functions for metric learning have been introduced, including contrastive loss [217–219], triplet loss [220], quadruplet loss [221], and n -pair loss [222]. These functions have been instrumental in measuring similarity in the feature space for representation learning and have demonstrated significant benefits in numerous applications such as image retrieval [223], face recognition [220, 224–227], and person re-identification [228]. Recently, metric learning has achieved notable success in deriving transferable deep representations from large-scale unlabeled data [229]. A category of *instance-based* methods employs contrastive loss [230, 231] to directly compare pairs of image representations [231–235]. Another approach utilizes a *clustering-based* strategy, learning unsupervised representations by distinguishing between groups of images without the need for costly pairwise comparisons [86, 236–243]. More recently, there has been a renewed focus on applying metric learning within supervised learning settings [219, 244–246].

Distance- and similarity-based classifiers utilize the similarity between samples and class representatives for classification, naturally relating them to the fields of metric learning and distance-based classification. The choice of an appropriate distance measure is crucial to the success of these classifiers [247]. Historically, metric learning and class center discovery have been fundamental research topics in distance-based classification. DNC, a nonparametric, distance-based classifier, can be viewed as a learnable metric function that compares data samples based on the corresponding semantic labels. While existing distance learning algorithms optimize the feature space by comparing data samples, they often rely on parametric softmax for classification. Consequently, these models function as black-box parametric classifiers, lacking interpretability. In contrast, DNC assigns observations to the class of the nearest centroids directly, without the need for parametric softmax. This distance-based decision-making approach allows DNC to seamlessly incorporate existing metric learning techniques. In fact, its training process can already be considered a form of metric learning.

2.2.4 Clustering-based Self-supervised Representation Learning

There is a growing trend to integrate self-supervised representation learning with clustering. Clustering-based self-supervised representation learning offers significant advantages in efficiency for large-scale training data and resilience to the similarity in semantic struc-

tures among data samples, compared to instance-level approaches. More specifically, early methods [86, 236, 238, 239, 242, 248, 249] focused on learning image representations and cluster assignments in an *alternating* manner. This involved grouping features into clusters to generate pseudo supervisory signals, which were then used to guide the representation learning process. More recently, efforts have shifted towards *simultaneous* clustering and representation learning. These newer approaches leverage techniques such as data reconstruction [237, 250], mutual information maximization [248, 251, 252], and contrastive instance discrimination [240, 241, 243, 253–255].

Our work is closely related to clustering-based unsupervised representation learning methods, particularly those that utilize the fast Sinkhorn-Knopp algorithm [85] for robust clustering [86, 240]. These methods aim to *learn transferable representations* from large amounts of *unlabeled* data. Although DNC also employs a clustering procedure for automatic sub-pattern mining, it is designed to establish a robust *similarity-based classification network* within a *supervised learning* setting. In DNC, the automatically discovered class sub-centroids serve as informative class representatives, explicitly capturing the latent data structure of each class and providing clear physical meaning as classification evidence. The entire training process in DNC is a hybrid approach that integrates two key components: class-wise online clustering for unsupervised sub-class discovery and sub-centroid-based classification for supervised representation learning.

2.2.5 Image-to-Image Translation

Image-to-Image (I2I) translation focuses on mapping images from a source domain to a target domain while retaining the domain-invariant context of the input image [256, 257]. Current data-driven methods for I2I tasks can be classified into two main groups: *GAN-based* and *Diffusion-based* approaches.

GAN-based methods, though achieving high fidelity in translation performance [258–261], pose significant challenges in training [262, 263] and often suffer from mode collapse in the output distribution [264, 265]. In addition, these models frequently struggle to produce diverse translation outcomes [266].

Diffusion-based methods, on the other hand, have recently demonstrated competitive performance in generating high-fidelity images. Conditional diffusion models [16–18] show promising results by integrating the encoded features of a reference image during the generation process [267]. However, while successful, these methods typically offer only coarse guidance in embedding spaces, leading to ambiguity in more complex scenarios [102]. In contrast, our approach leverages in-context reasoning to decompose complex scenes into low-dimensional concepts, effectively addressing these challenges.

2.2.6 Text-guided Diffusion Models

Several concurrent works contribute to text-guided diffusion models. DreamBooth [104] and Textual Inversion [105] develop pre-trained text-to-image diffusion models that require several provided images. Prompt-to-Prompt [29, 106] manipulates local or global details by adjusting the text prompt. By injecting internal cross-attention maps, these methods preserve the spatial configuration and geometry of images, enabling regeneration and modification through prompt editing. However, [106] does not use an inversion technique, restricting it to synthesized images [29]. Both methods introduce an extra hyperparameter, w (i.e., guidance scale parameter), which significantly impacts the translated performance (see Fig. 4.7). In light of these considerations, our approach rethinks the design of classifier-free guidance [96] prediction and eliminates such an oscillating parameter for more robust, fully-automatic predictions. Further details of our findings are included in §4.3.2.

2.2.7 Visual Programming

Visual programming serves as an intuitive method for specifying programmatic operations and data flows to tackle complex visual tasks [30], grounded in the neuro-symbolic paradigm. The integration of Large Language Models (LLMs) into visual programming has demonstrated exceptional performance across various vision tasks, such as visual relationship understanding [268, 269], visual question answering [99, 270, 271], commonsense reasoning [272], and image translation [30]. This approach holds significant potential to revolutionize the field of image translation by enhancing *transparency and explainability*, as well as *in-context reasoning*.

We recognize several insightful approaches, notably AnyDoor [102], Inst-Inpaint [273] and VISPROG [30] are particularly relevant to our work. However, [102] involves manually selecting an object’s location in a scene, classifying objects with an additional ID extractor, and extracting detail maps for hierarchical resolutions. Similarly, [273] requires additional training on the GQA-Inpaint dataset [274]. These processes remain opaque and necessitate additional training, making them unsuitable for manual modifications or training-free paradigms. [30] introduces visual programming for compositional visual tasks but primarily focuses on building a modular neuro-symbolic system, overlooking strong editing abilities for compositional generalization. Moreover, it replaces original objects with text-to-image generation, disregarding the preservation of content from the original image.

Our DVP, on the other hand, provides a nuanced framework that delineates each intermediate step in a comprehensible and modifiable manner for human users (see §4.3.2). By translating complex feature spaces into lower-dimensional symbols, DVP enables effective context-free manipulations (see §3.2.2).

2.3 Carbon-efficient Visual Intelligence System

2.3.1 Vision Transformers

Inspired by the remarkable success of transformers in natural language processing (NLP) [1, 101, 169–172], researchers have extended the transformer architecture for various supervised vision tasks, including image classification [24, 49, 173, 275], image segmentation [174–177, 276, 277, 277–281], object detection [282–287], and pose estimation [288–291]. Additionally, self-supervised pretraining paradigms [50, 51, 292] have been explored, achieving state-of-the-art results. Transformers have become dominant in visual-related fields due to their superior performance and scalability compared to convolutional neural networks (CNNs) [23, 293]. However, the substantial computational and parameter overhead required to adapt transformers to various vision tasks remains a significant challenge [294–296]. Recent transformer-based models such as MViTv2-Large [297] (218M parameters), ViT-G [298] (1.8B parameters), SwinV2-G [173] (3.0B parameters), and V-MoE [299] (14.7B parameters) incur considerable computational costs.

To address these challenges, we propose E²VPT, a solution designed to reduce the computational burden of transformer-based architectures while maintaining high performance within the *pretrain-then-finetune* paradigm.

2.3.2 Parameter-efficient Fine-tuning

Efficient model training has become a significant focus within the vision community, particularly with the rise of Vision Transformers [24, 49, 300–302]. Despite their efficacy and widespread adoption, these models are often too large for practical deployment and adaptation. Consequently, the *pretrain-then-finetune* paradigm is frequently employed. While full fine-tuning guarantees strong performance, it is a costly approach that requires updating all network parameters [115, 293]. To overcome this challenge, researchers are exploring alternatives that strike a balance between parameter efficiency and robust performance. These alternatives can be broadly classified into three categories: *partial tuning*, *extra module*, and *prompt tuning* methods.

Partial tuning techniques are commonly employed for parameter-efficient fine-tuning. These approaches involve keeping the majority of the backbone network unchanged and fine-tuning only a small subset of the parameters. This subset typically includes elements such as linear layers [48], MLP heads [233], or specific blocks or layers within the backbone [21, 50, 303, 304]. Although these methods are simple to implement and relatively straightforward [51, 118, 119], they frequently exhibit a significant performance gap when compared to full fine-tuning. *Extra module* methods involve designing additional learnable architectures that can be plugged into existing networks for fine-tuning. For instance, [123] introduces an alternative side structure while keeping the original network frozen. Similarly, [22] and [122] add extra residual units into the backbone architecture. However, a

significant drawback of these approaches is that the inserted modules are often tailored to specific architectures, limiting their generalizability to other models. Furthermore, these extra modules typically require more parameters than partial tuning methods.

Prompt tuning or prompting [127, 305–307] was initially introduced for fast model adaptation in the language domain. These approaches involve prepending a set of learnable vectors to the input of the backbone, with only these task-specific prompts being updated during fine-tuning. Recently, visual-related prompting [23, 308, 309] has been applied to the vision domain, incorporating visual prompts into the input sequence and demonstrating competitive performance with full fine-tuning. However, existing methods typically overlook the internal design of transformer-based architectures, leading to less effective prompting solutions. In contrast, our approach takes into account the architectural intricacies of transformers and is anchored in pruning strategies, fundamentally differentiating it from the previously discussed methods.

2.3.3 Large-scale Vision Models

Building on the significant advancements of transformers in natural language processing (NLP) [101, 169–172], researchers have successfully adapted the transformer architecture for various vision tasks, including image classification [1, 24, 49, 173], image segmentation [174–177], and object detection [282, 283, 285–287]. Transformers have emerged as dominant models in vision-related disciplines due to their superior performance and scalability compared to convolutional neural networks (CNNs) [23, 293]. However, the significant computational complexity of transformers [140, 294–296] poses challenges for their deployment in real-world scenarios with limited computational resources. For instance, the well-known transformer-based architecture ViT-Base [24] requires 2.23 times more computational overhead than ResNet101 [68] (*i.e.*, 17.6 GFlops *v.s.* 7.9 GFlops). While there is a strong desire to leverage the capabilities of large-scale vision models, much of the current research has focused on knowledge distillation within traditional CNNs, neglecting the growing need for scale reduction in transformer-based architectures. To address this gap, we propose AMD, a knowledge distillation approach designed to reduce the computational overhead of large-scale vision models while maintaining strong performance.

2.3.4 Model Pruning

Model pruning can be generally classified into unstructured [140, 310–312] and structured pruning [137, 313–317]. The primary distinction between these approaches lies in their impact on the neural network architecture. Structured pruning modifies the network by physically removing groups of parameters, while unstructured pruning zeros out specific weights without changing the network’s inherent structure [315]. Although unstructured pruning allows for more fine-grained parameter reduction, it typically requires specialized hardware or software to be effective. In contrast, structured pruning reduces memory

usage and computational overhead without the need for specialized accelerators or frameworks [318, 319], making it suitable for a wider range of practical applications. Within transformer-based architectures, various strategies have been proposed to achieve efficient designs at the desired scale, including dynamic search [320], layer dropping [321], and pruning [322]. In our research, we adopt structured pruning to derive candidate structures due to its advantages in facilitating distillation.

2.3.5 Knowledge Distillation

Knowledge distillation [143] has garnered significant attention in the vision community, particularly in the area of model compression [146, 323, 324]. The core idea involves transferring the knowledge from a large-scale model (*i.e.*, the teacher) to a smaller, more computationally efficient model (*i.e.*, the student). Recent research in this field can be broadly categorized into single-step and multi-step distillation methods. Single-step distillation methods, initially inspired by [143], involve using the teacher model’s predictions as “soft” labels [325, 326], which the student model mimics to reproduce the teacher’s final predictions [327]. These methods include logit-based approaches [328–330] and feature-mimicking techniques [331–336], which utilize spatial-wise knowledge to transfer intermediate features directly or indirectly [337–339]. Additionally, relation-based studies have explored the relationships between different layers or data samples [340, 341], leveraging various forms of relational knowledge such as FSP matrices [342], instance relations [343–345], similarity matrices [346], and mutual information flows [347]. Multi-step distillation methods gained prominence when researchers observed that a significant capacity discrepancy between large-scale teacher models and smaller student models could hinder effective knowledge transfer [26, 348]. These methods address the challenge by using intermediate teacher-assistant models to bridge the gap. For instance, [146] employs multiple teacher-assistants and introduces “random drop” to enhance efficiency, while [349] constructs a gradual mimicking sequence. Despite mitigating performance degradation, these methods are often computationally expensive due to the overhead associated with multiple teacher-assistants, especially as teacher model sizes increase. Moreover, current methods [350–352] seldom consider the needs of large-scale vision models, which require higher compression ratios due to their substantial size compared to traditional convolutional architectures. Although recent research, such as CSKD [353], DeiT [354], CviT [355], and DearKD [351], has achieved impressive performance, their primary focus is on enhancing large-scale models through knowledge distillation rather than on model compression and deployment on resource-constrained devices. In contrast, our approach prioritizes both computational efficiency and achieving a high compression ratio, while preserving the superior performance of the teacher models.

Chapter 3

Method

3.1 Universal Visual Learner

3.1.1 Prototypical TransFormer (ProtoFormer)

After reviewing the existing literature (§2.1), we find that integrating prototype learning with the Transformer architecture offers a promising approach for addressing various motion tasks. In this section, we first revisit the Transformer architecture and reinterpret its attention mechanism through the lens of prototype learning (§2.1.2). Building on this foundation, we introduce ProtoFormer, which includes two key innovations: *Cross-Attention Prototyping* (§3.1.1) and *Latent Synchronization* (§3.1.1). These contributions directly address question ③. We elaborate our approach below.

Preliminary. In our study, we re-conceptualize the Transformer’s attention mechanism through the framework of classical clustering. Traditionally, attention maps are generated by computing the similarity between all query-key pairs [356, 357]. Our approach, on the other hand, introduces a density-based cross-attention estimation, tailored to accommodate motion characteristics by aggregating local rigid motion patterns into distinct prototype clusters.

Classic clustering, a widely recognized paradigm, involves partitioning m observations into k distinct groups. Each observation is assigned to the cluster it most closely associates with, based on the highest likelihood or minimal distance (*e.g.*, proximity to the cluster mean). Formally, the clustering process can be optimized iteratively through two phases:

- *Assignment Phase* assigns each observation to the cluster for which it has the highest probability of membership or the shortest spatial distance.
- *Centroid Recalculation Phase* updates the centroids of the clusters to accurately represent the current distribution of observations within each cluster.

These two phases continue until convergence is achieved, which is indicated either by the stabilization of assignments or when the changes in assignments fall below a predefined threshold, implying that the clusters have stabilized.

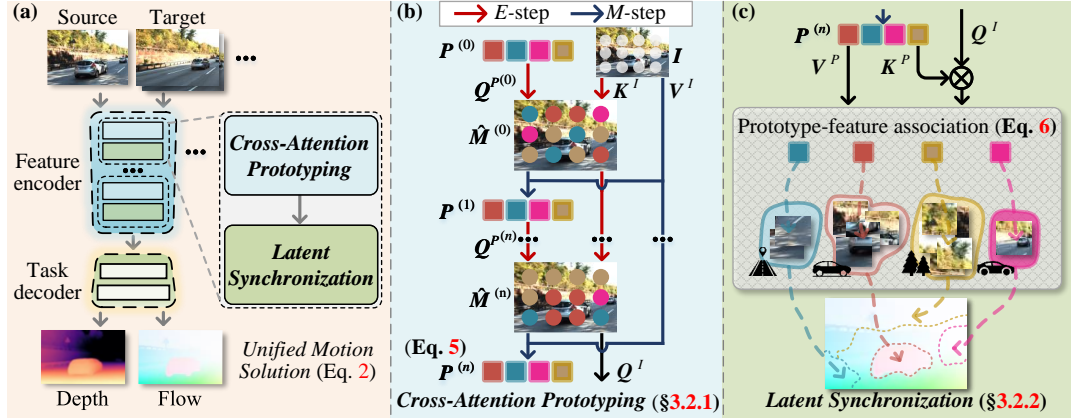


Figure 3.1: (a) **Overall pipeline of ProtoFormer**. Movement of a small part of an object within an image is being considered as a rigid motion. In our approach, we use prototypes to understand or predict this kind of motion pattern. (b) In each layer of the **Cross-Attention Prototyping** (see §3.1.1), there are N sequential iterations encompassing the assignment of feature-prototypes (*i.e.*, E -step) and the subsequent updating of these prototypes (*i.e.*, M -step) via Eq. 3.5. (c) Concurrently, the **Latent Synchronization** process (see §3.1.1) associates the feature representations via the freshly updated motion prototypes, (see Eq. 3.6). For (b) and (c), we apply optical flow for illustration, which demonstrates straightforward systemic explainability.

From a mathematical perspective, let θ denote the centroids [1] of the clusters, and consider $x \in \mathcal{X}$ as an individual observation:

$$\theta^{(n+1)} = \arg \max_{\theta} \mathbb{E}(p_{x \sim \mathcal{X}}(x|\theta^{(n)})). \quad (3.1)$$

Here, $\theta^{(n)}$ refers to the centroid calculated during the n -th iteration, while $p(\cdot)$ denotes the posterior probability associated with the data assignments.

Methodology. The main objective of ProtoFormer is to optimize the expected likelihood function within the framework of clustering as part of a *unified motion solution* as:

$$\hat{\theta}_k = \sum_{j=1}^K p(\mathcal{X}|\theta_j) \cdot P(\theta_k, \theta_j). \quad (3.2)$$

Here θ denotes the centroid representations, which we identify as *Prototypes*, our optimization targets. The total number of clusters is represented by K . The probabilities $p(\mathcal{X}|\theta_k) \in (0, 1)$ are the mixing coefficients for each cluster $k \in \mathcal{K}$, adhering to the constraint $\sum_k p(x|\theta_k) = 1$. The projected prototype representation $P(\cdot)$ refers to the learnable dense vector derived from a shared parametric family associated with the k -th prototype.

This function aggregates information across all clusters \mathcal{K} , with each projected prototype $P(\theta_k, \theta_j)$ representing the new projected representation for its respective cluster. $\hat{\theta}_k$ indicates the updated prototype, taking into account the prototype representation θ and the posterior probability $p(\mathcal{X}|\theta_k)$, which is the conditional likelihood of assigning the data \mathcal{X} to the prototype parameterized by θ_k .

Cross-Attention Prototyping via EM clustering

To develop a unified motion solution utilizing Transformers, we redefine the traditional Transformer’s self-attention mechanism [172] into an innovative prototypical cross-attention mechanism. This new mechanism is optimized using *Expectation-Maximization (EM)* clustering. The optimization process employs density-based estimation to compute the maximum likelihood for p_k and θ_k , leveraging posterior probabilities.

E-Step: For each observation $x_i \in \mathcal{X}$, *E-Step* computes the n -th iteration posterior probabilities $p_k(x_i)$. These probabilities represent the likelihood of x_i being associated with center θ_k with the logit vector $s_{x_i,k}$ as:

$$p_k^{(n)}(x_i) = \frac{s_{x_i,k}^{(n)} \cdot P(x_i, \theta_k^{(n)})}{\sum_{j=1}^K s_{x_i,j}^{(n)} \cdot P(x_i, \theta_j^{(n)})}. \quad (3.3)$$

$s_{x_i,k}^{(n)}$, $\theta_k^{(n)}$ are the parameters estimated at the n -th iteration.

M-Step: For each cluster θ_k , it obtains the maximum likelihood estimations $p_k^{(n)}$ and $\theta_k^{(n)}$ from projected sub-sample representations P' , updated as:

$$\theta_k^{(n+1)} = \frac{1}{N} \sum_{i=1}^N \sum_{j=1}^K p_k^{(n)}(x_i) \cdot P'(\theta_k^{(n)}, \theta_j^{(n)}). \quad (3.4)$$

In practice, given feature embeddings $\mathbf{I} \in \mathbb{R}^{HW \times D}$ and initializing $\mathbf{P}^{(0)}$ as K prototype cluster centers, we incorporate the *EM* clustering process within a *Cross-Attention Prototyping* layer (see Fig. 3.1(b)) that iterates N times as:

$$\begin{aligned} E\text{-step:} \quad & \hat{\mathbf{M}}^{(n)} = \text{softmax}_K(\mathbf{Q}^{P^{(n)}}(\mathbf{K}^I)^\top), \\ M\text{-step:} \quad & \mathbf{P}^{(n+1)} = \hat{\mathbf{M}}^{(n)} \mathbf{V}^I \in \mathbb{R}^{K \times D}, \end{aligned} \quad (3.5)$$

where $n \in \{1, \dots, N\}$. $\hat{\mathbf{M}} \in [0, 1]^{K \times HW}$ represents the “soft” pixel-prototype assignment matrix, serving as the probability maps of prototypes. $\mathbf{Q}^P \in \mathbb{R}^{K \times D}$ is the query vector projected from the prototype representation \mathbf{P} , and $\mathbf{V}^I, \mathbf{K}^I \in \mathbb{R}^{HW \times D}$ are the value and key vectors projected from the image features \mathbf{I} , respectively. Overall, our proposed layer iteratively updates the prototype membership $\hat{\mathbf{M}}$ (*i.e.*, *E-step*) and the prototypes \mathbf{P} (*i.e.*, *M-step*).

The key characteristic of this approach is its assurance of incremental convergence in the likelihood function with each iteration (see Eq. 3.4). Essentially, the *E-step* assesses the current membership of data representations based on existing prototypes, while the *M-step* refines these prototypes to better match the pixels, ensuring steady progression towards optimal clustering. By applying cross-attention prototyping to the source and target images separately, our method effectively handles the complexities of motion uncertainty

and photometric inconsistency. Additionally, we modify the default *softmax* operator from HW to K , emulating the *EM* clustering process.

The proposed layer enjoys various compelling characteristics:

- *Convergence*: *EM* clustering consistently enhances the marginal likelihood with each iteration and has been empirically shown to converge to a local optimum, given a sufficient number of iterations [358–360]. In this context, our proposed *Cross-Attention Prototyping* leverages the power of recursive clustering, iterated over N steps, to enhance the probability of achieving an optimal configuration for motion partitioning (see §4.1.3).
- *Transparency*: Prototyping serves as a crucial mechanism for contextual understanding in motion scenes, recognizing and grouping similar patterns and movements. It clusters pixels into prototypes that exhibit homogeneity in characteristics such as flow or depth. By aggregating entities with shared attributes, these prototypes effectively describe the intrinsic dynamics of the scene. Moreover, prototyping offers a foundational framework for motion comprehension, with each prototype representing a microcosm of the objects within the scene, encapsulating their unique elements and interrelations.
- *Efficiency*: *Cross-Attention Prototyping* operates with $\mathcal{O}(NKHWD)$ time complexity, offering a substantial improvement over the self-attention $\mathcal{O}(H^2W^2D)$ complexity (see §4.1.3). The efficiency gain stems from the relationship $NK \ll HW$ (for example, 60 *v.s.* 25,920 in the first stage with an image resolution of 960×432). This difference becomes even more pronounced in pyramid architectures [49,276,302,361], where the cumulative NK value is significantly smaller than HW , especially in the early stages of the network. In each iteration, only the query matrix \mathbf{Q} needs to be updated, while the key \mathbf{K} and value \mathbf{V} matrices are computed once. This selective updating process greatly reduces the computational burden, making it particularly advantageous for handling large-scale data in high-dimensional feature spaces.

Prototype-Feature Corresponding by Latent Synchronization

We further enhance feature representations by synchronizing the projection of K prototypes into an $H \times W$ feature space. This approach aligns the prototype representations with the motion features (see Fig. 3.1(c)).

The core method involves a Feed-Forward Network (FFN) integrated with a masked cross-attention mechanism as:

$$\hat{I} = \text{FFN}(\text{Cross-Attention}(Q^I, K^P, V^P, \mathcal{M}_P)), \quad (3.6)$$

where \mathcal{M}_P denotes the feature assignment mask maps based on the similarity to the corresponding prototypes P . The term \hat{I} represents the refined feature, Q^I indicates the query projection derived from the input image feature, and K^P and V^P indicates the

key and value projections obtained from the learning prototypes, respectively. *Latent Synchronization* is designed to enhance feature learning for prototype-feature association, thereby reducing motion ambiguity. This process enables the extraction and encapsulation of the latent distribution of each feature representation within its respective prototype.

Latent Synchronization also enjoys several appealing characteristics:

- *Blended Paradigm*: *Latent Synchronization* elegantly combines unsupervised prototype mining (§3.1.1) and supervised feature representation learning (§3.1.1) in a synergy manner. It autonomously identifies significant local motion patterns to facilitate density-based prototyping. Concurrently, task-specific supervisory signals directly optimize the feature representations, enhancing the effectiveness of prototyping.
- *Prototype-Anchored Learning*: Density-based prototype learning recursively calculates dependable probabilities for prototype assignment (Eq. 3.5). Anchored by the updated prototypes, features are further refined through prototype-feature associations (Eq. 3.6). This process ensures that motion patterns tend to cluster in regions of high data density, which in turn enhances robustness against motion ambiguities.

3.2 Interpretable Visual Intelligence

3.2.1 Deep Nearest Centroids (DNC)

Problem Statement. Under the standard visual recognition setting, let \mathcal{X} represent the visual space (*e.g.*, image space for recognition, pixel space for segmentation), and let $\mathcal{Y} = \{1, \dots, C\}$ be the set of semantic classes. Given a training dataset $\{(x_n, y_n) \in \mathcal{X} \times \mathcal{Y}\}_{n=1}^N$, the objective is to use the N training examples to fit a *model* (or hypothesis) $h : \mathcal{X} \mapsto \mathcal{Y}$ that accurately predicts the semantic classes for new visual samples.

Parametric Softmax Classifier. Current common practices implement h as DNNs and decompose it into $h = l \circ f$. Here, $f : \mathcal{X} \mapsto \mathcal{F}$ serves as a *feature extractor* (*e.g.*, convolution-based or Transformer-based networks) that maps an input sample $x \in \mathcal{X}$ into a d -dimensional representation space $\mathcal{F} \in \mathbb{R}^d$, *i.e.*, $\mathbf{x} = f(x) \in \mathcal{F}$. The function $l : \mathcal{F} \mapsto \mathcal{Y}$ is a *parametric classifier* (*e.g.*, the final fully-connected layer in visual recognition or the final 1×1 convolution layer in segmentation tasks) that takes \mathbf{x} as input and produces a class prediction $\hat{y} = l(\mathbf{x}) \in \mathcal{Y}$. Specifically, l assigns a query $x \in \mathcal{X}$ to the class $\hat{y} \in \mathcal{Y}$ by:

$$\hat{y} = \arg \max_{c \in \mathcal{Y}} s^c, \quad s^c = (\mathbf{w}^c)^\top \mathbf{x} + b^c, \quad (3.7)$$

where $s^c \in \mathbb{R}$ represents the unnormalized prediction score (*i.e.*, the *logit*) for class c , and $\mathbf{w}^c \in \mathbb{R}^d$ and $b^c \in \mathbb{R}$ are the learnable parameters | specifically, the class weight and bias term for class c . The parameters of both l and f are optimized by minimizing the softmax

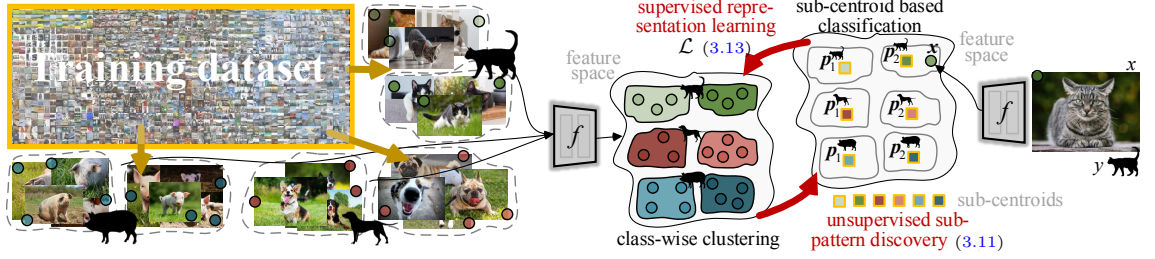


Figure 3.2: DNC uses a distance-/case-based criterion to combine unsupervised sub-pattern discovery and supervised representation learning in a synergistic fashion.

cross-entropy loss:

$$\mathcal{L} = \frac{1}{N} \sum_{n=1}^N -\log p(y_n|x_n), \quad (3.8)$$

$$p(y|x) = \text{softmax}_y(l \circ f(x)) = \frac{\exp(s^y)}{\sum_{c \in \mathcal{Y}} \exp(s^c)}.$$

Though highly successful, the parametric classifier l has several limitations: **i)** The weight matrix $\mathbf{W} = (\mathbf{w}^1, \dots, \mathbf{w}^C) \in \mathbb{R}^{C \times d}$ and bias vector $\mathbf{b} = (b^1, \dots, b^C) \in \mathbb{R}^d$ in l are learnable parameters that do not offer any transparency into the decision-making process of the model h . **ii)** The loss function \mathcal{L} depends solely on the relative relationships among the logits, *i.e.*, $\{s^c\}_c$, and does not directly supervise the representation \mathbf{x} [362, 363]. **iii)** The weight matrix \mathbf{W} and bias vector \mathbf{b} are learned as flexible parameters, which do not explicitly model the underlying data structure. **iv)** The dimensionality of the final output is limited to the number of classes, *i.e.*, C . During transfer learning, as different visual recognition tasks typically involve distinct semantic label spaces (*i.e.*, with varying numbers of classes), the classifier from a pretrained model often needs to be discarded (*i.e.*, the valuable knowledge encapsulated in the learned parameters \mathbf{W} and \mathbf{b} from the source task). In the following part, we demonstrate that not only can these limitations of the parametric classifier be effectively addressed, but they can also be resolved while achieving superior performance via our proposed DNC.

DNC Classifier. DNC (Fig. 3.2) is founded on the intuitive concept of Nearest Centroids, *i.e.*, assign a sample x to the class $\hat{y} \in \mathcal{Y}$ with its closest class center as:

$$\hat{y} = \arg \min_{c \in \mathcal{Y}} \langle \mathbf{x}, \bar{\mathbf{x}}^c \rangle, \quad \bar{\mathbf{x}}^c = \frac{1}{N^c} \sum_{x_n^c: y_n^c = c} x_n^c, \quad (3.9)$$

where $\langle \cdot, \cdot \rangle$ is a distance measurement (*i.e.*, we use cosine-similarity in our settings), given as: $\langle \mathbf{u}, \mathbf{v} \rangle = -\mathbf{u}^\top \mathbf{v} / \|\mathbf{u}\| \|\mathbf{v}\|$. For simplicity, all the features are defaulted to ℓ_2 -normalized from this point forward. Furthermore, $\bar{\mathbf{x}}^c$ represents the mean vector of class c , x_n^c denotes a training sample of c , *i.e.*, $y_n^c = c$, and N^c is the number of training samples in c . Consequently, the feature-to-class mapping $\mathcal{F} \mapsto \mathcal{Y}$ is achieved in a *nonparametric* and more *interpretable* manner from user's view, contrasting with the parametric classifier

l that learns “non-transparent” parameters for each class. In many challenging visual recognition tasks, complex intra-class variations cannot be fully captured with a simple unimodal distribution for the samples in each class. These variations, however, can be successfully addressed by introducing multiple sub-centroids (local means) for each class. When representing each class c with K sub-centroids, denoted as $\{\mathbf{p}_k^c \in \mathbb{R}^d\}_{k=1}^K$, the C -way classification for a sample x follows a *winner-takes-all* rule as:

$$\hat{y} = c^*, \quad (c^*, k^*) = \arg \min_{c \in \mathcal{Y}, k \in \{1, \dots, K\}} \langle \mathbf{x}, \mathbf{p}_k^c \rangle. \quad (3.10)$$

To intuitively estimate class sub-centroids, we need to cluster training samples within each class. Since class sub-centroids are sub-cluster centers in the latent feature space \mathcal{F} , they encapsulate locally significant visual patterns and comprehensively represent class-level characteristics. DNC can be interpreted as the process of selecting and storing prototypical exemplars for each class and identifying classification evidences for a previously unseen sample by retrieving the most similar exemplar. This approach aligns with the *prototype theory* in psychology [81, 364, 365], which posits that prototypes constitute a typical form of cognitive organization for real-world objects. For instance, ornithologists classify a given bird by comparing it to relevant exemplars from known bird species [87]. The distance-based criterion of DNC mirrors this process [66].

Sub-centroid Estimation. Given the representation space \mathcal{F} , we conduct deterministic clustering for each class to identify informative sub-centroids that optimally represent each class. More specifically, for each class c , we cluster all the representations $\{\mathbf{x}_n^c \in \mathbb{R}^d\}_{n=1}^{N^c}$ into K clusters with the cluster centers serving as the sub-centroids for c , *i.e.*, $\{\mathbf{p}_k^c \in \mathbb{R}^d\}_{k=1}^K$. Let $\mathbf{X}^c = [\mathbf{x}_1^c, \dots, \mathbf{x}_{N^c}^c] \in \mathbb{R}^{d \times N^c}$ and $\mathbf{P}^c = [\mathbf{p}_1^c, \dots, \mathbf{p}_K^c] \in \mathbb{R}^{d \times K}$ denote the feature and sub-centroid matrix, respectively. The deterministic clustering, *i.e.*, the mapping from \mathbf{X}^c to \mathbf{P}^c , can be represented by $\mathbf{Q}^c = [\mathbf{q}_1^c, \dots, \mathbf{q}_{N^c}^c] \in \{0, 1\}^{K \times N^c}$, where n -th column $\mathbf{q}_n^c \in \{0, 1\}^K$ is a one-hot assignment vector of n -th sample x_n^c *w.r.t.* the K clusters. \mathbf{Q}^c is designed to maximize the similarity between \mathbf{X}^c and \mathbf{P}^c , leading to the following binary integer programming (BIP) as:

$$\begin{aligned} \max_{\mathbf{Q}^c \in \mathcal{Q}^c} & \text{Tr}((\mathbf{Q}^c)^\top (\mathbf{P}^c)^\top \mathbf{X}^c), \\ \mathcal{Q}^c = & \{\mathbf{Q}^c \in \{0, 1\}^{K \times N^c} \mid (\mathbf{Q}^c)^\top \mathbf{1}_K = \mathbf{1}_{N^c}\}, \end{aligned} \quad (3.11)$$

where $\mathbf{1}_K$ is a K -dimensional all-ones vector. Following [86], we relax \mathcal{Q}^c to a *transportation polytope* [85]: $\mathcal{Q}^c = \{\mathbf{Q}^c \in \mathbb{R}_+^{K \times N^c} \mid (\mathbf{Q}^c)^\top \mathbf{1}_K = \mathbf{1}_{N^c}, \mathbf{Q}^c \mathbf{1}_{N^c} = \frac{N^c}{K} \mathbf{1}_K\}$, casting BIP (3.11) into an *optimal transport* problem. In \mathcal{Q}^c , we incorporate not only the *one-hot assignment* constraint (*i.e.*, $(\mathbf{Q}^c)^\top \mathbf{1}_K = \mathbf{1}_{N^c}$), but also an *equipartition* constraint (*i.e.*, $\mathbf{Q}^c \mathbf{1}_{N^c} = \frac{N^c}{K} \mathbf{1}_K$). This ensures that the N^c samples can be evenly distributed among the K clusters, efficiently avoiding degeneracy (*i.e.*, mapping all the data to the same cluster). The solution can then be obtained using a fast version of the Sinkhorn-Knopp algorithm [366], as a normalized exponential matrix:

$$\mathbf{Q}^{c*} = \text{diag}(\boldsymbol{\alpha}) \exp\left(\frac{(\mathbf{P}^c)^\top \mathbf{X}^c}{\varepsilon}\right) \text{diag}(\boldsymbol{\beta}), \quad (3.12)$$

where the exponentiation is performed element-wise, $\alpha \in \mathbb{R}^K$ and $\beta \in \mathbb{R}^{N^c}$ are two renormalization vectors, which can be computed with a small number of matrix multiplications via Sinkhorn-Knopp Iteration [85], and $\varepsilon = 0.05$ balances the trade-off between convergence speed and the accuracy of the approximation to the original transport problem. In summary, by mapping data samples into a limited numbers of clusters under the constraints \mathcal{Q}^c , we aim to achieve both *sparsity* and *expressivity* [193, 194], while assigning the class sub-centroids as the representatives of the dataset.

Training of DNC = Supervised Representation Learning + Automatic Sub-class Pattern Mining. Ideally, based on class-wise cluster assignments $\{\mathcal{Q}^c\}_{c=1}^C$, a total of CK sub-centroids $\{\mathbf{p}_k^c\}_{c,k=1}^{C,K}$ can be calculated, *i.e.*, mean feature vectors of the training data within the CK clusters. The training objective then becomes:

$$\begin{aligned} \mathcal{L} &= \frac{1}{N} \sum_{n=1}^N -\log p(y_n|x_n), \\ p(y|x) &= \frac{\exp(-\min(\{\langle \mathbf{x}, \mathbf{p}_k^y \rangle\}_{k=1}^K))}{\sum_{c \in \mathcal{Y}} \exp(-\min(\{\langle \mathbf{x}, \mathbf{p}_k^c \rangle\}_{k=1}^K))}. \end{aligned} \quad (3.13)$$

Comparing (3.8) and (3.13), we can see that since the class sub-centroids $\{\mathbf{p}_k^c\}_{c,k}$ are derived solely from data representations, DNC learns visual recognition by directly optimizing the representation f , rather than the parametric classifier l . Moreover, with this non-parametric, distance-based scheme, DNC establishes a closer link to metric learning [218, 219, 222, 229, 244, 245, 367]. Consequently, DNC can be viewed as learning a metric function f to compare data samples $\{x_n\}_n$, guided by their corresponding semantic labels $\{y_n\}_n$.

During training, DNC alternates between two steps iteratively: **i**) class-wise clustering (3.11) to automatically discover sub-centroids, and **ii**) sub-centroid based classification to supervise the learning of representations (3.13). Through clustering, DNC explores the underlying data distribution of each class, and generates informative sub-centroids by aggregating statistics from data clusters. This automatic sub-class discovery process also shares a similar philosophy with recent clustering-based unsupervised representation learning methods [86, 236–243]. However, DNC operates in a class-wise manner, utilizing class labels. In this way, DNC optimizes the representation by adjusting the alignment between sub-centroids and data samples. The enhanced representation, in turn, helps identify more informative sub-centroids, benefiting classification performance. As such, DNC conducts *unsupervised sub-class pattern discovery* during *supervised representation learning*, setting it apart from most current visual recognition models.

As the latent representation f continually evolves during training, it is essential to keep class sub-centroids synchronized. This requires performing class-wise clustering on all training data after each batch update. However, this step can be computational expensive, especially on large-scale datasets, even considering the high efficiency of the Sinkhorn-Knopp iteration [85] based clustering presented in (3.12). To avoid the costly offline sub-centroid estimation, we implement *momentum update* and *online clustering*. Specifically,

at each training iteration, we perform class-wise clustering on the current batch and update each sub-centroid as:

$$\mathbf{p}_k^c \leftarrow \mu \mathbf{p}_k^c + (1 - \mu) \bar{\mathbf{x}}_k^c. \quad (3.14)$$

Here $\mu \in [0, 1]$ is a momentum coefficient, and $\bar{\mathbf{x}}_k^c \in \mathbb{R}^d$ is the mean feature vector of data assigned to (c, k) -cluster in current batch. This ensures that the sub-centroids remain updated with parameter changes. Although this batch-wise clustering is effective in most cases, it may not scale well to a large number of classes. For example, when training on ImageNet [40] with 1,000 classes using a batch size of 256, not all classes/clusters are visited in a single batch (at most 256 classes are visited). To mitigate this, we store features from several prior batches in a memory, and perform clustering on both the memory and the current batch. Overall, DNC can be trained by gradient backpropagation in a small-batch setting with minimal delay ($\sim 5\%$ training overhead on ImageNet).

In the context of point cloud segmentation, we encounter a prevalent class imbalance issue, a phenomenon not typically shown in image classification and semantic segmentation. This often leads to sub-optimal sub-centroids, especially for rare classes. To counter this, we propose a calibration strategy with a learnable factor, capable of assessing the representational capacity of each class sub-centroid. Specifically, the calibration factor for class c , denoted as z^c , is defined by a rectified sigmoid function that adjusts the distance between the sub-centroids matrices \mathbf{P}^c and the point embedding feature matrices \mathbf{X}^c :

$$z^c = \frac{1}{N_c K} \sum_{i=1}^{N_c} \sum_{k=1}^K \frac{1}{1 + \exp((\mathbf{x}_i^c)^\top \mathbf{p}_k^c)}, \quad z^c \in [0, 1]. \quad (3.15)$$

Thus, the modified Eq. 3.14 with calibration turns into:

$$\mathbf{p}_k^c \leftarrow \mu \mathbf{p}_k^c + z^c (1 - \mu) \bar{\mathbf{x}}_k^c, \quad (3.16)$$

in point cloud segmentation settings.

Versatility. DNC is a general framework that can be effortlessly integrated into any parametric classifier based DNN with minimal architectural change by simply replacing the parametric softmax layer. However, DNC changes the classification decision-making mode, reforms the training regime, and makes the reasoning process more transparent, without slowing down the inference speed. DNC can be seamlessly applied to various visual recognition tasks.

Transferability. As a nonparametric scheme, DNC can handle an arbitrary number of classes with fixed output dimensionality (d); all the knowledge learnt on a source task (*e.g.*, ImageNet classification with 1K classes) are stored as a fixed number of parameters in f , and thus can be *completely* transferred to a new task (*e.g.*, Cityscapes [43] segmentation with 19 classes), under the ‘‘pre-training and fine-tuning’’ paradigm. Under a similar setting, its parametric counterpart has to discard around 2M parameters during transfer learning ($d=2,048$ when using ResNet101 [68]).

Ad-hoc Explainability. DNC is a transparent classifier containing a built-in case-based reasoning process, as the sub-centroids are summarized from real observations and are

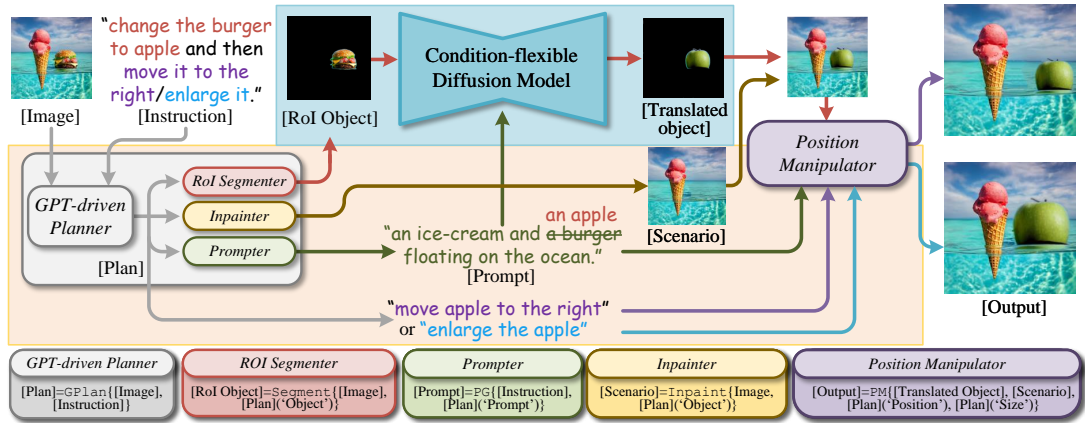


Figure 3.3: **Diffusion Visual Programmer (DVP) overview.** Our proposed framework contains two core modules: ■ is the condition-flexible diffusion model (see §3.2.2), augmented by the integration of instance normalization (see Fig. 3.4), aimed to achieve a more generalized approach to translation; ■ stands for visual programming (see §3.2.2), fulfilled by a series of off-the-shelf operations (e.g., Segment operation for precise ROI segmentation). The overall neuro-symbolic design enables in-context reasoning for context-free editing. We also enjoy enhanced controllability and explainability by intuitively explicit symbols (e.g., [Prompt], [RoI object], [Scenario], [Translated object]) at each intermediate stage, facilitating human interpretation, comprehension and modification.

referenced during classification. So far we have investigated the scenario where the sub-centroids are the mean feature vectors of several training samples with similar patterns. When we further constrain the sub-centroids to be elements of the training set (i.e., representative training images), DNC naturally portrays human-tangible explanations for each prediction. The explanations stay true to the internal decision mode and do not create a post-hoc justification.

3.2.2 Image Translation as Diffusion Visual Programmer (DVP)

In this part, we present Diffusion Visual Programmer (DVP), a visual programming pipeline for image translation (see Fig. 3.3). Our framework decomposes image translation into two distinct sub-objectives: ① style transfer, which involves translating RoIs within images while maintaining contextual coherence; and ② context-free editing, which allows for unrestricted yet judicious modifications. To address ①, we introduce the Condition-flexible diffusion model, designed for autonomous, non-human-intervened translation (§3.2.2). For ②, we propose In-context Visual Programming, which breaks down high-level concepts into human-understandable symbols, enabling adaptable manipulation (§3.2.2). We elaborate on our techniques below.

Condition-flexible Diffusion Model

Preliminaries. Text-guided diffusion models convert a stochastic noise vector z_t and a textual prompt embedding \mathcal{P} into an output image z_0 that aligns with the specified conditioning prompt. To accomplish step-by-step noise reduction, the model ϵ_θ is calibrated to estimate synthetic noise as:

$$\min_{\theta} E_{z_0, \epsilon \sim N(0, I), t \sim \text{Uniform}(1, T)} \|\epsilon - \epsilon_\theta(z_t, t, \mathcal{P})\|^2, \quad (3.17)$$

where \mathcal{P} represents the conditioning prompt embedding and z_t is a hidden layer perturbed by noise, which is introduced to the original sample data z_0 based on the timestamp t . During inference, the model progressively reduces the noise over T steps starting from a noise vector z_T . To accurately reconstruct a given real image, the deterministic diffusion model sampling [100] is defined as:

$$z_{t-1} = \sqrt{\frac{\alpha_{t-1}}{\alpha_t}} z_t + \left(\sqrt{\frac{1}{\alpha_{t-1}} - 1} - \sqrt{\frac{1}{\alpha_t} - 1} \right) \cdot \epsilon_\theta(z_t, t, \mathcal{P}), \quad (3.18)$$

where $\alpha_t := \prod_{i=1}^t (1 - \beta_i)$, and $\beta_i \in (0, 1)$ is a hyper-parameter for the noise schedule. Inspired by [106], we generate spatial attention maps corresponding to each textual token. Specifically, a cross-attention mechanism is incorporated for controllability during translation, facilitating interaction between the image and prompt during noise prediction:

$$\epsilon_\theta(z_t, t, \mathcal{P}) = \text{Softmax}\left(\frac{\mathbf{1}_Q(\phi(z_t))\mathbf{1}_K(\psi(\mathcal{P}))}{\sqrt{d}}\right)\mathbf{1}_V(\psi(\mathcal{P})), \quad (3.19)$$

where $\mathbf{1}_Q$, $\mathbf{1}_K$, $\mathbf{1}_V$ are learned linear projections. $\phi(z_t)$ is the spatial features of the noisy image, and $\psi(\mathcal{P})$ stands for the textual embedding.

Instance Normalization Guidance. Text-guided generation often encounters the challenge of magnifying the influence of the conditioning text during the generation process [100]. To address this, [96] proposed a classifier-free guidance approach. In this method, an initial prediction is generated without any specific conditioning. This unconditioned output, modulated by the guidance scale parameter w , is then linearly combined with predictions influenced by the conditioning text scaled by $(1 - w)$. Formally, given $\emptyset = \psi(\cdot)$ as the feature representation from the null text, we have:

$$\tilde{\epsilon}_\theta(z_t, t, \mathcal{P}, \emptyset) = w \cdot \epsilon_\theta(z_t, t, \mathcal{P}) + (1 - w) \cdot \epsilon_\theta(z_t, t, \emptyset). \quad (3.20)$$

In practice, we observe that the scaling factor w is extremely sensitive. Even slight variations in its value can significantly impact the final images. This necessity for meticulous fine-tuning on a per-image basis renders it impractical for widespread adoption in real-world applications. In the light of this view, we introduce the concept of adaptive distribution shift for the condition-flexible translation. Specifically, we examine the roles

of two types of embeddings: the conditional prediction, $\epsilon_\theta(z_t, t, \mathcal{P})$, and the unconditional noise prediction, $\epsilon_\theta(z_t, t, \emptyset)$. We propose that these embeddings require distinct treatment, moving beyond a rudimentary linear combination from Eq. 3.20.

In the works of [368, 369], it is suggested that the success of instance normalization in style transfer is due to its resilience to variations in the content image. Similarly, classifier-free guidance [96] employs a similar form by merging stable predictions from a conditional embedding with those from a concurrently trained unconditional embedding.

We, therefore, employ instance normalization, which aligns effectively with the intended translation direction by mitigating the influence of unconditional embeddings (see Fig. 3.4). This ensures that the diffusion model remains strictly conditioned on the prompt, thereby eliminating any unwanted variations. The implementation of instance normalization not only improves translation performance but also strengthens the model’s ability to manage variations from the input distribution, regardless of potential discrepancies in unconditional distributions (see Fig. 4.7). Formally, the instance normalization guidance is:

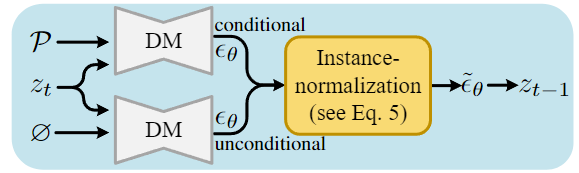


Figure 3.4: **Instance Normalization Guidance.**

$$\tilde{\epsilon}_\theta(z_t, t, \mathcal{P}, \emptyset) = \sigma(\epsilon_\theta(z_t, t, \emptyset)) \text{conv} \left(\frac{\epsilon_\theta(z_t, t, \emptyset) - \mu(\epsilon_\theta(z_t, t, \mathcal{P}))}{\sigma(\epsilon_\theta(z_t, t, \mathcal{P}))} \right) + \mu(\epsilon_\theta(z_t, t, \emptyset)), \quad (3.21)$$

Intuitively, the network $\tilde{\epsilon}_\theta$ is rescaled by σ and shifted by μ , where μ and σ represent the mean of the unconditional embedding and the standard deviation of the conditional embedding, respectively. Since both μ and σ are known values derived from predictions, this approach allows us to avoid operations influenced by human intervention for further tuning of unconditional textual embeddings.

In-context Visual Programming

Condition-flexible diffusion model in §3.2.2 offers a generalized solution for image translation, addressing the “*what*” aspect as the neural embodiment in our proposed framework. Additionally, we introduce in-context visual programming, which employs a symbolic reasoning process to bridge the understanding of visual concepts and text instructions, effectively tackling the concept of “*where*” (see Fig. 4.8). By decomposing the rich complexity of high-dimensional concepts into simpler, low-dimensional symbolic forms, we enable a cascade of logical operations. This approach fosters the development of nuanced, in-context reasoning capabilities. The core components are articulated below.

Symbols. Our framework generates intermediate steps, including [Prompt], [RoI object], and [Scenario]. These intermediary outcomes enhance transparency and explainability, allowing for human interpretation, understanding, and modifications. Additionally, these

steps can be viewed as symbols [370], which serve as low-dimensional data that enable structured operations. This approach bridges the gap between raw data-driven methods and symbolic reasoning.

Operations. We carefully grounded five operations, {GPlan, PG, Segment, Inpaint, PM}, in in-context visual programming (see Fig. 3.3). We first leverage the capabilities of the advanced language model GPT-4 [371] as an AI agent for perceiving, planning, and generating program directives [372,373]. This component, named as the *GPT-driven Planner* module, manages the creation of programs based on a few examples of similar instructions and initiates the necessary operations through the GPlan function during the reasoning process. All programs are constructed following first-order logic principles, enabling the articulation of more complex statements than those possible with propositional logic alone [99]. Specifically, these programs feature a hierarchical structure of symbolic, functional operations (*e.g.*, PG, Segment, Inpaint, PM), directing collections of modules such as the *Prompter*, *RoI Segmenter*, *Inpainter*, and *Position Manipulator* at each phase. These operations can be executed in parallel, allowing for flexible combinations and systemic controllability via program execution (see Fig. 4.9). The *Prompter* module uses GPT-4 to generate detailed descriptions of any given input image through the PG operation. This capability extends beyond human-annotated images, allowing for random unlabeled images as inputs and thus broadening application scenarios and enhancing data efficiency. The *RoI Segmenter* module, leveraging a pre-trained Mask2former [374], performs flexible RoI segmentation, aligning with the Segment operation. The *Inpainter* module operates based on the logic of \neg RoI, facilitating the completion of foreground or background elements via stable diffusion v1.5 [17, 375], and is associated with the Inpaint operation. Additionally, the *Position Manipulator* module categorizes rational concepts (*e.g.*, position, scale) and translates programmed human language instructions into a domain-specific language (DSL) designed for positioning tasks. The DSL includes fundamental operations such as **Enlarge**, **Shrink**, **Left**, **Right**, **Up**, and **Down**. These intuitive commands share a common input format and output interface, ensuring programming flexibility (see Fig. 4.9).

Program Execution. The programs are managed by the *Compiler* [30], which establishes a mapping between variables and values. It processes the program step-by-step, executing the appropriate operations line-by-line. At each stage of execution, the program activates the specified operation, producing intermediate outputs (*e.g.*, [prompts], [RoI object]) in human-interpretable symbols. This approach enhances the system’s explainability, allowing for direct visual evaluations of the outputs. Users can choose to either repeat the current step or proceed to the next, thereby improving the system’s controllability and usability (see §4.3.2).

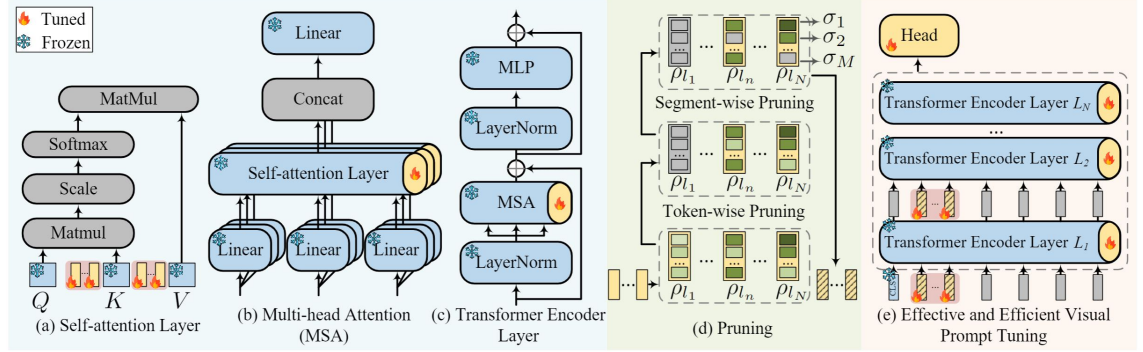


Figure 3.5: **Overview of our E^2VPT framework.** Under the *pretrain-then-finetune* paradigm, only the prompts in the transformer’s input and backbone (§3.3.1), are updated during the fine-tuning process, while all other components remain frozen. We further introduce pruning (§3.3.1) at two levels of granularity (*i.e.*, token-wise and segment-wise) in (d) to eliminate unfavorable input prompts during rewinding.

3.3 Carbon-efficient Visual Intelligence System

3.3.1 An Efficient and Effective Approach for Visual Prompt Tuning (E^2VPT)

We present E^2VPT , an innovative visual prompt tuning approach for the effective and efficient fine-tuning of large-scale transformer-based models. The problem definition and relevant notations are introduced in §3.3.1. In §3.3.1, we discuss effective prompt tuning, focusing on the design of visual and key-value prompts. This is followed by a detailed explanation of efficient prompt pruning in §3.3.1. The overall framework is illustrated in Fig. 3.5.

Problem Definition

In this section, we define the problem of E^2VPT and introduce the relevant notations. We start with a backbone vision transformer model \mathbf{T} , pretrained on a large set of data and tasks. The input to the vision transformer is a sequence of image patches $I = \{I_1, I_2, \dots, I_m\}$, where m is the total number of image patches. Each patch is projected into a d -dimensional embedding with positional encoding, denoted as $E = \{E_j \mid 1 \leq j \leq m\}$, where $E_j = \text{Emb}(I_j)$. The vision transformer \mathbf{T} comprises N identical transformer layers, represented as:

$$\begin{aligned} Z^1 &= L_1(E) \\ Z^i &= L_i(Z^{i-1}) \quad i = 2, 3, \dots, N \end{aligned} \quad (3.22)$$

Each transformer layer consists of a stack comprising a multi-head self-attention (MSA) mechanism and a feed-forward network (FFN):

$$L(\cdot) = \text{FFN} (\text{MSA} (\cdot)) \quad (3.23)$$

For a new vision task, the objective is to fine-tune a model $\hat{\mathbf{T}}$ that performs well on the task while minimizing the number of parameters that need to be tuned. In the context of visual prompt tuning, $\hat{\mathbf{T}} = \{\mathbf{T}, \mathbf{P}\}$ comprises a **frozen** backbone \mathbf{T} and **trainable** prompts \mathbf{P} , which include only a small number of tunable parameters.

Effective Prompting

Most existing prompt tuning methods focus on tuning a set of visual prompts by prepending them to the input sequence in transformer layers, largely ignoring the internal architecture of the transformers. To enhance the effectiveness of prompt tuning and achieve optimal fine-tuning performance, we propose a novel approach that integrates key-value prompts (\mathbf{P}_K and \mathbf{P}_V) along with the traditional input visual prompts (\mathbf{P}_I) within our visual prompt tuning framework. In our approach, input visual prompts are added to the input sequence of each encoder layer to learn representations for the new task. Concurrently, key-value prompts are concatenated with the key and value parameter matrices in the self-attention module, enabling the model to capture new attention patterns from the data.

Visual Prompts. Visual prompts are a set of d -dimensional embedding vectors that match the dimensionality of the input visual tokens. These prompts are prepended to the input sequence at each transformer encoder layer, allowing them to interact with all input tokens. Similar to prompt tokens in prompt tuning methods [23, 306], visual prompts learn task-specific embeddings to guide the model in performing new tasks. Formally, these visual prompts are defined as $P_I = \{P_I^1, P_I^2, \dots, P_I^N\}$, where P_I^i denotes the learnable visual prompts in the i_{th} encoder layer, and N is the total number of layers. The encoder layers can then be represented as:

$$\begin{aligned} Z^1 &= L_1(P_I^1, E) \\ Z^i &= L_i(P_I^i, Z^{i-1}) \quad i = 2, 3, \dots, N \end{aligned} \quad (3.24)$$

where Z^i denotes the contextual embeddings computed by the i_{th} encoder layer. Different colors are used to distinguish between **trainable** and **frozen** parameters. The embeddings for the input image patches E are initialized using a frozen Emb projection derived from the backbone.

Key-Value Prompts.

Visual prompts are effective for learning knowledge about new tasks, but they fall short in guiding information interaction within transformer encoder layers. This limitation arises because, during fine-tuning, the image distribution can differ significantly from the

examples used for pretraining the backbone model. Consequently, it is essential to enhance the model’s ability to capture new information from the fine-tuning data and to facilitate more effective attention among input tokens to learn new patterns.

To this end, we introduce a novel set of key-value prompts, P_K and P_V , incorporated into the attention module within each encoder layer (see Fig. 3.5(a)). These key-value prompts are small matrices with only a few columns, but they share the same number of rows as the key and value matrices in the original attention module. The key and value matrices are concatenated with their corresponding P_K and P_V prompts, respectively, to perform new attention computations. This process is defined as:

$$L(\cdot) = \text{FFN}(\text{MSA}(\cdot))$$

$$\text{MSA}(\cdot) = \text{concat}(\text{softmax}(\frac{Q_h K_h'^T}{\sqrt{d}}) V_h')$$
(3.25)

where FFN is the feed-forward network and MSA is the multi-head attention inside the encoder layer. h represents the h_{th} head. K' and V' are the new key and value embedding matrices defined as:

$$K' = \text{concat}(K, P_K), \quad V' = \text{concat}(V, P_V)$$
(3.26)

where K and V represent the original key and value matrices in the backbone. The key-value prompts facilitate the model’s adaptation to new data by providing additional guidance. In our implementation, we extend this approach by enabling parameter sharing of the P_K and P_V prompts within each transformer layer, rather than tuning separate learnable vectors for each instance. Our motivation is twofold: First, our experimental results indicate that using shared prompts consistently enhances fine-tuning performance across various instances. Second, shared prompt vectors significantly reduce the number of parameters in the learnable transformer component by half, increasing parameter efficiency. We will further discuss the exploration of prompt locations, specifically whether they should be placed before or after K and V .

It is important to note that the query matrix Q is another crucial component in the self-attention mechanism. However, additional prompting on Q is not desirable for two reasons: First, prompting on Q is akin to prepending on K for computing attention scores between each pair of Q and K . Therefore, prompting on both Q and K is redundant. Second, alterations in Q impact the output shape of the attention map, necessitating an additional linear projection to match dimensions in the subsequent layer, which is not feasible under a parameter-efficient design.

Efficient Prompting

Our approach to effective prompting seeks to enhance the performance of fine-tuned models. However, a pertinent question arises: Can we reduce the number of tunable prompts

without compromising model performance? The lottery ticket hypothesis (LTH) [131, 132] suggests that for a given task, there exists a sub-network capable of achieving the same test performance as the original over-parameterized network, without the need for redundant weights. Inspired by this hypothesis, we conducted an experiment where we masked different visual prompts. Our findings revealed that various prompts have distinct effects on model performance, with some even negatively impacting it. This observation aligns with prior research [136, 307].

Based on our findings, we propose a prompt pruning method for visual prompts. The primary objective of this method is to retain the most influential prompts while eliminating redundant or unnecessary ones. By removing less important prompts, we can significantly enhance the efficiency of prompt tuning without compromising performance. To achieve this goal, we design a cascade pruning strategy that operates at two levels of granularity: token-wise pruning and segment-wise pruning (see Fig. 3.5(d)). Token-wise pruning first identifies and removes the least important visual prompts. Following this, segment-wise pruning divides each remaining prompt into multiple segments and filters out the negative segments. By jointly reducing the parameter usage in learnable visual prompts, our two-level pruning approach creates soft-filtered prompts that can be re-trained during the rewinding stage.

Token-wise Pruning. We introduce a learnable mask variable $\rho = \{\rho_1, \rho_2, \dots, \rho_M\}$ (*i.e.*, M is the length of visual prompts) and associate it with the input visual prompts in each transformer layer. Here $\rho_k \in \{0, 1\}$, where 0 means the corresponding learnable input prompt is pruned (*i.e.*, “masked” with zero value). Then the masked version of the visual prompts updates into $\tilde{P}_k = \rho_k \cdot P_k$. In order to determine the pruning position, we calculate the importance score [131, 307] of each prompt token and eliminate those positions with lowest scores. The importance score is defined as the anticipated sensitivity of the model to the mask variables ρ_k [322]:

$$S_{P_k} = \mathbb{E}_{x \sim \mathcal{D}_x} \left| \frac{\partial \mathcal{L}(x)}{\partial \rho_k} \right| \quad (3.27)$$

where \mathcal{L} is the loss function, \mathcal{D}_x is the training data distribution [322]. The importance score assigned to each visual prompt indicates its contribution to the fine-tuning performance. A low score signifies that the prompt has minimal or potentially negative impact on the fine-tuning process. In contrast, a high importance score denotes that the prompt is valuable and significantly enhances the fine-tuning process.

Segment-wise Pruning. We extend our investigation to segment-wise pruning to eliminate ineffective segments within each prompt. Initially, the embedding of each prompt token is divided equally into R parts, with each part treated as a distinct unit subject to joint optimization. Similar to token-wise pruning, a mask variable is assigned to each segment within the prompt token. Segments with low importance scores are then filtered out.

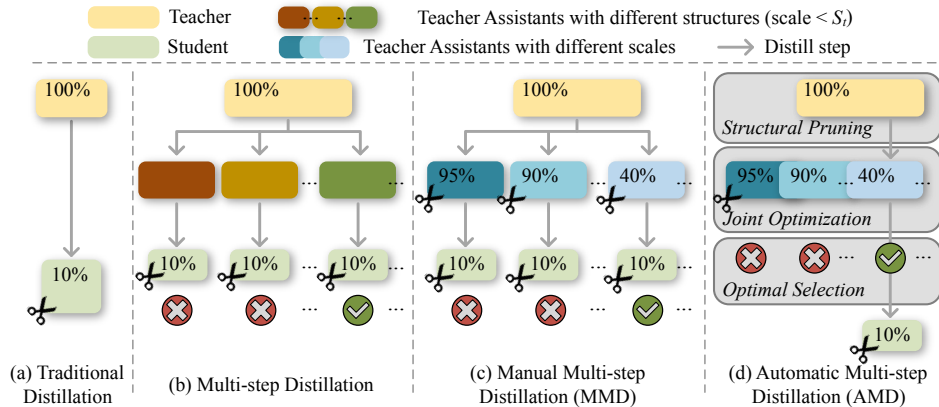


Figure 3.6: Overview of different approaches on knowledge distillation. (a) Traditional distillation methods directly distill the teacher to the student (*i.e.*, without considering additional teacher-assistant models). (b) Multi-step distillation methods first distill the teacher to a teacher-assistant (requires a large search), which is then further distilled to the student. (c) Manual Multi-step Distillation (MMD) effectively identifies a set of teacher-assistants with different scales and performs multi-step distillation. However, MMD still requires separate distillations with different-scale teacher assistants. (d) Automatic Multi-step Distillation (AMD) efficiently and effectively selects the optimal teacher-assistant through one single optimization, which include three stages: *Structural Pruning*, *Joint Optimization* and *Optimal Selection*.

Rewinding. Following the execution of the two-level cascade pruning, the weight rewinding phase centers on re-training the softly filtered prompt tokens. This process entails ranking the importance scores for each layer obtained during pruning and assigning a value of 0 to the corresponding mask variables for tokens with relatively low importance scores. Subsequently, the softly filtered input prompts are re-trained alongside other learnable parameters, utilizing the original learning rate and weight decay settings during the fine-tuning.

3.3.2 Automatic Multi-step Distillation of Large-scale Vision Models (AMD)

In this section, we begin by formally defining the multi-step distillation problems and establishing the necessary notations. We then introduce the Negative Performance-Scale Derivative (NPSD) metric, which serves as a criterion for evaluating the optimality of intermediate teacher-assistant models. Our preliminary study illustrates the efficacy of NPSD in assessing teacher-assistant performance. Following this, we present our proposed automatic multi-step knowledge distillation approach — AMD.

Problem Formulation

Within the multi-step distillation paradigm (see Fig. 3.6), the primary objective is to identify an optimal teaching-assistant, denoted as $TA(S_{ta}, P_{ta})$, to facilitate the distillation of knowledge from a fully-scaled teacher model, $T(S_t, P_t)$ where $S_t = 100\%$, to a student model, $S(S_s, P_s)$. Here P and S represent the performance and scale of the models, respectively. This process aims to enhance the student model’s performance, P_s , by leveraging the appropriate teaching assistant (*i.e.*, $T \rightarrow TA \rightarrow S$). In practice, the scale S_{ta} and performance P_{ta} of the teacher assistant must fall within two boundaries: for scalability, $S_s < S_{ta} < S_t$, and for performance, $P_s < P_{ta} < P_t$.

Negative Performance-Scale Derivative

In this work, we propose a method to identify an optimal teacher-assistant for multi-step distillation by balancing performance and scale. Finding an appropriate objective function to simultaneously optimize these factors is challenging. Ideally, we seek a teacher-assistant with maximum performance and minimal scale. However, this scenario is often impractical. As the scale of the teacher-assistant approaches that of the student model, its performance typically decreases, making complete knowledge transfer difficult. Conversely, a larger-scale teacher-assistant may still have a significant scale gap with the student model, resulting in a notable performance drop during distillation. Therefore, it is essential to find a teacher-assistant that achieves an optimal balance between performance and scale.

To address this challenge, we introduce the Negative Performance-Scale Derivative (NPSD) as our optimization objective. NPSD evaluates the benefits, assessing how closely the teacher-assistant’s scale matches that of the student, against the costs, considering the associated performance drop. A higher NPSD indicates a more favorable benefit-cost ratio. Our preliminary studies confirm that NPSD positively correlates with downstream student performance across various tasks and backbones (refer to Figure 1.5). Based on this finding, our goal is to maximize NPSD while maintaining manageable time complexity.

Definition of NPSD. Formally, NPSD stands for the negative derivation of performance to scale, which is:

$$NPSD_{ta} = - \frac{P_t - P_{ta}}{S_t - S_{ta}} \quad (3.28)$$

where t and ta refer to the teacher and teacher-assistant, respectively. P_t and P_{ta} indicate the performances of these models, while S_t and S_{ta} denote their corresponding model scales. Importantly, for a given teacher model, P_t and S_t are constants. Intuitively, a teacher-assistant that achieves high performance with a small scale results in a high NPSD value.

Preliminary Results. To evaluate the effectiveness of NPSD as an optimality measure, we conducted multi-step distillation experiments on CIFAR-10 and CIFAR-100 [27] using various teacher models. These teacher models are distilled into a 10% student through teacher-assistants [26] at different scales ranging from 10% to 90% in 10% increments. The architecture of each teacher-assistant model is derived through structural pruning [319, 322], which involved pruning the least important parameters based on their importance scores from the teacher model.

The results for ViT-Tiny on CIFAR-10 and ViT-Base on CIFAR-100 are presented in Fig. 1.5. Notably, as the size of the teacher-assistant approaches that of the student, its performance steadily declines. Conversely, the performance of students distilled from different teacher-assistants fluctuates based on the scales of the teacher-assistants. Additionally, we observe a positive correlation between the NPSD measure and the performance of the student models. The highest performance is achieved when students are distilled with teacher-assistants having the maximum NPSD values, demonstrating the effectiveness of the NPSD metric.

Automatic Multi-step Distillation

The objective mentioned above is thus transformed into an optimization problem, where the goal is to identify the optimal teacher-assistant with the highest NPSD_{ta} value. However, identifying an optimal teacher-assistant presents several significant challenges. First, the number of potential teacher-assistants is theoretically infinite due to the continuous nature of the search space. Even within a fixed-scale model, there can be substantial architectural variations. Second, the computational expense of identifying the highest-performing teacher-assistants at every scale becomes prohibitive when performing multi-step distillation for each one. Lastly, selecting the best teacher-assistant from among all candidates adds another layer of complexity to the process.

To tackle these challenges, we break the problem down into three distinct stages. In the *Structural Pruning* stage, we generate a series of candidate teacher-assistants at various scales using gridding and pruning techniques. Next, we introduce a *Joint Optimization* framework to identify the highest-performing teacher-assistants across all scales within a single optimization process. Finally, in the *Optimal Selection* stage, we select the best teacher-assistant based on the highest NPSD score.

Structural Pruning. We utilize *gridding* and *pruning* to construct teacher-assistants at various scales. Initially, *gridding* is introduced to limit the number of candidates by converting a continuous search space into a discrete one. Ideally, generating candidates at every possible scale would be necessary to continuously find the “perfect” teacher-assistant. However, this approach is impractical (due to infinite subdivisions) and inefficient (as it would require optimizing numerous teacher assistants simultaneously). Therefore, we evenly divide the scales into m parts, with the gap between each candidate defined as

$\delta = \frac{S_t - S_s}{m}$, resulting in m candidates between the teacher and student. Next, to determine the architecture of each candidate at different scales, various *pruning* techniques can be applied. In this study, we adopt the structural pruning method [319, 322] due to its well-documented advantages in knowledge distillation [376]. This method gradually prunes the network by removing the least important parameters based on their importance scores, which are obtained by introducing learnable mask variables during the teacher’s inference stage. The resulting teacher-assistant candidates are represented as $\{\mathcal{M}_i \mid i = 1 \text{ to } m\}$.

Joint Optimization. The straightforward approach to identifying the optimal teacher-assistant involves maximizing the performance of each candidate and then using NPSD to evaluate their optimality. However, this process remains computationally intensive due to the large number of candidates. To further reduce computational complexity, we investigate parameter sharing among teacher-assistant candidates across different scales by leveraging the *incremental property* inherent in candidates derived from structural pruning. The incremental property asserts that for two candidates, \mathcal{M}_i and \mathcal{M}_j , if $S_i < S_j$, the parameters of \mathcal{M}_i are a subset of those in \mathcal{M}_j . This allows a larger candidate to be transformed into a smaller one through continuous pruning of less significant parameters, thus enabling efficient parameter sharing across different candidates.

By utilizing parameter sharing, the total number of parameters to be optimized is reduced to that of the largest candidate model. We therefore develop a joint optimization framework that simultaneously handles the distillations from the teacher to all candidate models in a single run. This approach effectively consolidates the memory requirements of all candidates to match that of the largest model. Additionally, the computational costs are significantly lowered due to the parameter-sharing mechanism. The overall objective of this optimization includes the cross-entropy loss, logit-based loss, and feature-based loss:

$$\mathcal{L} = \sum_{i=1}^m (\mathcal{L}_{ce}(\mathcal{T}, \mathcal{M}_i) + \alpha \mathcal{L}_{logit}(\mathcal{T}, \mathcal{M}_i) + \beta \mathcal{L}_{feat}(\mathcal{T}, \mathcal{M}_i)) \quad (3.29)$$

where $\mathcal{L}_{ce} = CE(y^T, y^M)$ is the cross-entropy loss, $\mathcal{L}_{logit} = \text{KL}[\text{softmax}(\frac{l^t}{\gamma}) \parallel \text{Softmax}(\frac{l^s}{\gamma})]$ is the kullback-leibler divergence loss [377, 378] measured between the softened output logits, and $\mathcal{L}_{feat} = \text{MSE}(H^T, H^M)$ is the mean squared error measured between the last layer of hidden states. y , l , γ , H are the labels, output logits, temperature value, and last layer of hidden states, respectively.

Optimal Selection. The optimal teacher-assistant is then chosen based on the highest NPSD value among all candidates. Subsequently, an additional distillation is performed between the selected teacher-assistant and the student, adhering to the training objective outlined in Eq. 3.29. It is worth noting that, in the domain of NLP, there is substantial research [319, 379] that defines the training objective through empirical analysis of training curves and hyper-parameter adjustments. In contrast, our method emphasizes

the relationship between the teacher and teacher-assistant, allowing for the examination of robustness across a wide range of datasets and models. This approach enhances our understanding of knowledge distillation dynamics, offering nuanced insights into the process.

Chapter 4

Results

4.1 Experiment for ProtoFormer

We comprehensively evaluate the performance and coherence of our proposed ProtoFormer on two key motion tasks: optical flow (see §4.1.1) and scene depth estimation (see §4.1.2). By striving for a unified solution for motion-related tasks, we highlight the advantages of this integration while demonstrating superior performance.

4.1.1 Experiments on Optical Flow

Quantitative Results. Table 4.1 presents the evaluation results of our model on the Sintel and KITTI datasets. Under the ‘C+T’ setting, ProtoFormer demonstrates its generalization capability, achieving scores of 1.04 and 2.43 on the clean and final passes of Sintel, respectively, surpassing the recently popular CRAFT [157] by 0.23 and 0.36. Following training in the mixed ‘C+T+S+K+H’ setting, our prototype-based model attains scores of 1.06 and 2.07 on the clean and final passes of Sintel, and 4.35 F1-epe on KITTI.

Qualitative Results. Figure 4.1 presents qualitative results on the Sintel flow dataset. ProtoFormer demonstrates superior performance by capturing more global and finer details in both object and motion boundaries, without being affected by shadows and textureless surfaces. In the first and fourth examples, our model excels at recovering full shapes and intricate details, *e.g.*, bamboo and weapons, contrasting sharply with other methods that struggle with clear predictions due to occlusions and variations in illumination. In the second and third examples, ProtoFormer consistently estimates occluded and textureless regions, such as the backpack and birds in the sky, with significant accuracy. The red-highlighted regions prove ProtoFormer’s effectiveness in object clustering and mitigating motion ambiguity.

Table 4.1: **Quantitative results on standard Sintel and KITTI flow benchmarks.** ‘A’ denotes the Autoflow dataset; ‘C + T’ denotes training on the FlyingChairs and FlyingThings datasets only; ‘C + T + S + K + H’ fine-tunes on a combination of Sintel, KITTI, and HD1K training sets. Error metrics are lower is better with “↓”, and accuracy metrics are higher is better with “↑”. Same for Table 4.2.

Training	Method	Sintel (train)		KITTI-15 (train)		Sintel (test)		KITTI-15 (test)
		Clean ↓	Final ↓	F1-epe ↓	F1-all ↓	Clean ↓	Final ↓	F1-all ↓
A	Perceiver IO [380]	1.81	2.42	4.98	-	-	-	-
	RAFT-A [381]	1.95	2.57	4.23	-	-	-	-
C+T	RAFT [9]	1.43	2.71	5.04	17.4	-	-	-
	Separable Flow [382]	1.30	2.59	4.60	15.9	-	-	-
	GMA [383]	1.30	2.74	4.69	17.1	-	-	-
	AGFlow [384]	1.31	2.69	4.82	17.0	-	-	-
	KPA-Flow [385]	1.28	2.68	4.46	15.9	-	-	-
	DIP [386]	1.30	2.82	4.29	13.7	-	-	-
	GMFlowNet [387]	1.14	2.71	4.24	15.4	-	-	-
	GMFlow [388]	1.08	2.48	7.77	23.4	-	-	-
	CRAFT [157]	1.27	2.79	4.88	17.5	-	-	-
	FlowFormer [10]	1.01	2.40	4.09	14.7	-	-	-
	SKFlow [389]	1.22	2.46	4.27	15.5	-	-	-
	MatchFlow [28]	1.14	2.71	4.19	13.6	-	-	-
	ProtoFormer (Ours)	1.04	2.43	4.08	14.6	-	-	-
	C+T+S+K+H	RAFT [9]	0.76	1.22	0.63	1.5	1.61	2.86
RAFT-A [381]		-	-	-	-	2.01	3.14	4.78
Separable Flow [382]		0.69	1.10	0.69	1.60	1.50	2.67	4.64
GMA [383]		0.62	1.06	0.57	1.2	1.39	2.47	5.15
AGFlow [384]		0.65	1.07	0.58	1.2	1.43	2.47	4.89
KPA-Flow [385]		0.60	1.02	0.52	1.1	1.35	2.36	4.60
DIP [386]		-	-	-	-	1.44	2.83	4.21
GMFlowNet [387]		0.59	0.91	0.64	1.51	1.39	2.65	4.79
GMFlow [388]		-	-	-	-	1.74	2.90	9.32
CRAFT [157]		0.60	1.06	0.58	1.34	1.45	2.42	4.79
Flowformer [10]		0.48	0.74	0.53	1.11	1.20	2.12	4.68
SKFlow [389]		0.52	0.78	0.51	0.94	1.28	2.23	4.87
MatchFlow [28]		0.51	0.81	0.59	1.3	1.33	2.64	4.72
ProtoFormer (Ours)		0.48	0.69	0.50	1.09	1.06	2.07	4.35

4.1.2 Experiments on Scene Depth

Quantitative Results. Table 4.2 presents the test results on the Sintel and KITTI datasets. ProtoFormer outperforms others on most error and accuracy metrics, demonstrating its robust feature representation and generalization capabilities across different scenarios. Compared to recent directly supervised depth estimation methods [13, 35, 37], ProtoFormer shows significantly superior performance, thanks to the integration of prototypical learning and cross-attention architecture. Remarkably, even *without* using additional constraints and priors, such as surface normal and piecewise planarity [14, 38], our model surpasses concurrent P3Depth in error reduction on KITTI by a substantial margin. Moreover, our approach also outperforms methods utilizing self-supervised consistency and strategies [36, 390]. This highlights the effectiveness of our model, which exhibits exceptional adaptability and learning capacity, making it highly suitable for fine-tuning



Figure 4.1: **Qualitative results on the Sintel.** The red boxes highlight the regions compared. Matchflow [28] appears blurry and ambiguous on textureless and occluded objects, while Flowformer [10] fails to recover complete and detailed information. Ours can estimate clear and complete flow motion, which is closer to ground truth.

across a broad range of motion-related tasks.

Table 4.2: **Quantitative results on Sintel and KITTI depth datasets.** With both test data unseen by the model, we can achieve leading performance over state-of-the-art methods [13, 14, 35–38].

Method	Sintel			KITTI		
	Abs Rel ↓	RMSE ↓	Sq Rel ↓	Abs Rel ↓	RMSE ↓	δ_1 ↑
Eigen et al.	0.797	0.834	0.703	0.203	6.307	0.702
Godard et al.	-	-	-	0.114	4.935	0.861
Fu et al.	-	-	-	0.072	2.727	0.932
Yin et al.	0.746	0.611	0.652	0.072	3.258	0.938
AdaBins	0.730	0.572	0.647	0.067	2.960	0.949
P3Depth	0.653	0.396	0.571	0.071	2.842	0.953
Ours	0.594	0.486	0.538	0.062	2.716	0.949

Qualitative Results. Figure 4.2 presents a qualitative comparison of depth estimation on the KITTI Eigen depth datasets. ProtoFormer exhibits superior capability in delineating object surface contours, especially in dynamic scenarios involving pedestrians and vehicles, as well as capturing fine details of objects like traffic signs and light poles. For instance, in Sample 1 and Sample 3, ProtoFormer provides more consistent and complete depth estimations on moving pedestrians and vehicles, offering clearer boundaries compared to P3Depth [14] and AdaBins [13]. In Samples 2, 3, and 4, our method effectively

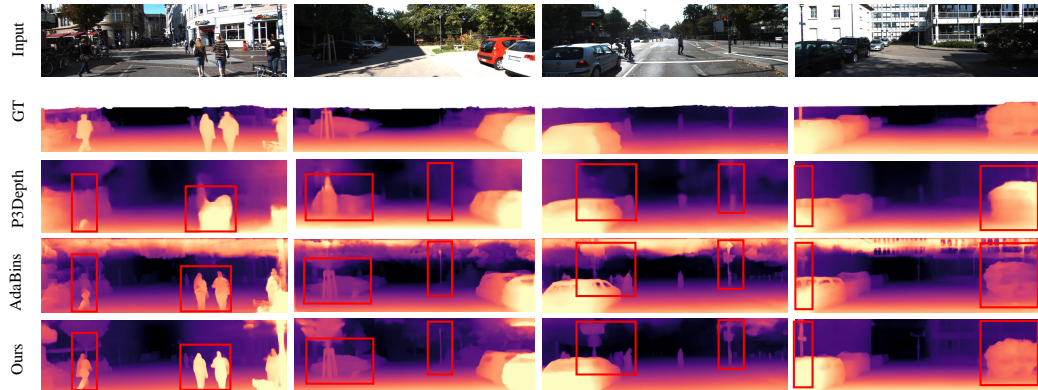


Figure 4.2: **Qualitative depth comparison results on the KITTI.** The red boxes indicate the highlighted regions. P3Depth [14] and AdaBins [13] have limited receptive fields and do not consider conceptual object-level groupings, thus producing discontinuous and ambiguous predictions. While ours can estimate consistent and sharp depths, which is closer to ground truth.

Table 4.3: A set of **ablative studies** on optical flow (see §4.1.3). The best performances are marked in **bold**.

Algorithm Component	#Params	Sintel clean	Sintel final
Base	9.63M	0.55	0.81
+ Cross-Attention Prototyping	11.57M	0.51	0.74
+ Latent Synchronization	10.26M	0.53	0.77
ProtoFormer (All included)	11.90M	0.48	0.69

(a) Key Component Analysis

Variant Prototype Updating Strategy	#Params	Sintel clean	Sintel final
Cosine Similarity	10.28M	0.51	0.75
Vanilla Cross-Attention [172]	14.88M	0.50	0.73
Criss Cross-Attention [391]	14.56M	0.50	0.72
<i>K</i> -Means [392]	11.81M	0.49	0.71
Cross-Attention Prototyping (Eq. 3.5)	11.90M	0.48	0.69

(b) *Cross-Attention Prototyping*

#Iterations (<i>N</i>)	#Params	Sintel clean	Sintel final
1	11.90M	0.52	0.75
2		0.49	0.71
3		0.48	0.69
4		0.48	0.68

(c) Number of Iterations

#Prototypes (<i>K</i>)	#Params	Sintel clean	Sintel final
10	8.95M	0.53	0.78
50	9.78M	0.51	0.73
100	11.90M	0.48	0.69
200	14.21M	0.49	0.71

(d) Number of Prototypes

Latent Synchronization	#Params	Sintel clean	Sintel final
None	11.27M	0.51	0.74
Vanilla FC Layer	11.64M	0.50	0.73
FC w/ Similarity [393]	11.76M	0.49	0.71
Ours (Eq. 3.6)	11.90M	0.48	0.69

(e) *Latent Synchronization*

distinguishes the plant stand and traffic poles from the noisy and complex backgrounds, highlighting its proficiency in such environments. These results demonstrate the advantages of incorporating prototype learning into depth estimation, allowing for geometric consistency and improved handling of motion ambiguity.

4.1.3 Diagnostic Experiments

This section evaluates the key components and configurations of ProtoFormer.

Key Components Analysis. We study the key components of ProtoFormer: *Cross-Attention Prototyping* (§3.1.1) and *Latent Synchronization* (§3.1.1). We designed a **Base** model that does not incorporate prototype updating or prototype-feature assignment. As

shown in Table 4.3a, the **Base** model achieves an average EPE of 0.55 and 0.81. When *Cross-Attention Prototyping* is added, significant improvements are observed (0.55 \rightarrow 0.51 in the clean pass), indicating the effectiveness of prototype updating even without explicit prototype-feature assignment. Adding *Latent Synchronization* to the **Base** model results in a noticeable performance gain (0.81 \rightarrow 0.77 in the final pass). Ultimately, integrating both techniques leads to optimal performance.

Cross-Attention Prototyping. We then examine the effectiveness of *Cross-Attention Prototyping* by comparing it with various updating methods, including cosine similarity, conventional cross-attention [172], Criss cross-attention [391], and *K*-Means [392]. From both efficiency and effectiveness perspectives, *Cross-Attention Prototyping* surpasses these competitive methods (see Table 4.3b). Additionally, we analyze the iteration step N in Table 4.3c, finding that the error decreases progressively from 0.52 to 0.48 as N increases from 1 to 4, stabilizing at 4. Considering computation time, we set $N = 3$ to balance performance and computational cost. The number of prototypes K is crucial in defining the central grouping points for motion features. Therefore, we investigate the impact of varying K in Table 4.3d.

Latent Synchronization. Next, we study our *Latent Synchronization* as presented in Table 4.3e. In the standard setting, without any prototype-feature correspondence (*i.e.*, None), the model achieves a final accuracy of 0.74. Introducing a basic fully-connected layer to update the features reduces the error to 0.73. Though pretty inspiring, our proposed *Latent Synchronization*, which incorporates carefully anchored prototypes, demonstrates superior performance, achieving an accuracy of 0.69 across all ablative methods.

Systemic Explainability. Finally, we examine the prototype-feature correspondence map for optical flow in Fig. 4.3. The systemic explainability relies on the **Prototypes**, which are generated through the integration of probability density estimation within our cross-attention prototyping layer. These recursively optimized prototypes capture the most representative features of motion patterns at their respective density centers. By visualizing the feature correspondence derived from these updated prototypes, we improve the interpretability of the network and the transparency of the model’s decision-making process.

4.2 Experiment for DNC

4.2.1 Experiments on Image Classification

Dataset. The evaluation for image classification is conducted on CIFAR-10 [39], CIFAR-100 [39] and ImageNet [40] datasets, respectively. Specifically, CIFAR-10 contains 60K (50K/10K for **train/test**) 32×32 colored images of 10 classes; CIFAR-100 dataset contains 100 classes with 500 training and 100 testing images per class; and ImageNet contains 1.2M images for **train** and 50K images for **validation** of 1K classes.

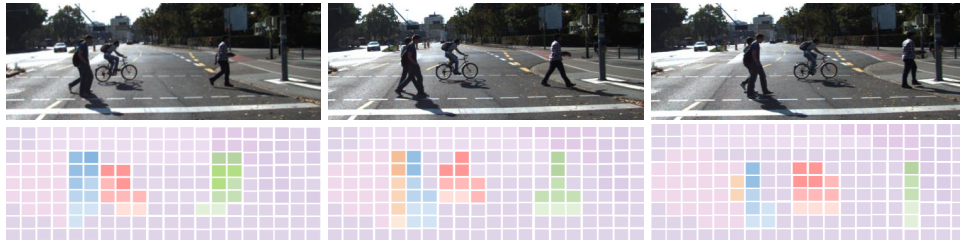


Figure 4.3: **Visualization of proto-feature mapping**, which demonstrates distinct prototypes with similar representations.

Table 4.4: **Classification top-1 accuracy on CIFAR-10 [39] test.** #Params: the number of learnable parameters (same for other tables).

Method	Backbone	#Params	top-1
ResNet [68]	ResNet50	23.52M	95.55%
DNC-ResNet		23.50M	95.78%
ResNet [68]	ResNet101	42.51M	95.58%
DNC-ResNet		42.49M	95.82%

Network Architecture. In order to evaluate the robustness of DNC, we integrate it into both CNN-based and Transformer-based architectures. Specifically, we replace the final linear classification layer of ResNet50/101 [68], which has a dimensionality of 2,048, and the final layers of Swin-Small/Base [49], with dimensionalities of 768 and 1,024, respectively, with DNC.

Training. We utilize the `mmclassification`¹ as our codebase and adhere to the *default* training settings used by its discriminative counterparts. For the CIFAR-10 and CIFAR-100 datasets, ResNet models were trained for 200 epochs with a batch size of 128. The memory size for DNC models was set to 100 batches. For the ImageNet dataset, we trained ResNet for 100 epochs and Swin for 300 epochs, with a batch size of 16. The initial learning rates for ResNet and Swin were set to 0.1 and 0.0005, respectively, with a step policy and polynomial annealing policy for learning rate scheduling. Due to GPU capacity limitations, the memory sizes were set to 1,000 batches for the DNC version of ResNet and 500 batches for Swin. Other hyperparameters were empirically set to $K = 4$ and $\mu = 0.999$ (see §4.2.6). All models were trained *from scratch* on eight V100 GPUs.

Results on CIFAR datasets. Table 4.4 and Table 4.5 compare the performance of DNC with its parametric counterparts using the ResNet architecture on CIFAR-10 and CIFAR-100 datasets, respectively. As seen, DNC outperforms the parametric models on both datasets. For instance, on CIFAR-10, DNC achieves a **0.23%** higher accuracy on ResNet50 and a **0.24%** higher accuracy on ResNet101, with fewer learnable parameters. Similar

¹<https://github.com/open-mmlab/mclassification>

Table 4.5: **Classification top-1 accuracy on CIFAR-100 [39] test.**

Method	Backbone	#Params	top-1
ResNet [68]	ResNet50	23.71M	79.81%
DNC-ResNet		23.50M	79.91%
ResNet [68]	ResNet101	42.70M	79.83%
DNC-ResNet		42.49M	79.99%

Table 4.6: **Classification top-1 and top-5 accuracy on ImageNet [40] val.**

Method	Backbone	#Params	top-1	top-5
ResNet [68]	ResNet50	25.56M	76.20%	93.01%
DNC-ResNet		23.51M	76.49%	93.08%
ResNet [68]	ResNet101	44.55M	77.52%	93.06%
DNC-ResNet		42.50M	77.80%	93.85%
Swin [49]	Swin-S	49.61M	83.02%	96.29%
DNC-Swin		48.84M	83.26%	96.40%
Swin [49]	Swin-B	87.77M	83.36%	96.44%
DNC-Swin		86.75M	83.68%	97.02%

trends are seen on the CIFAR-100 dataset. These results indicate that with identical backbone architectures and training schemes, the performance improvement can be safely attributed to DNC, specifically due to its robust modularity.

Results on ImageNet dataset. Table 4.6 illustrates our outstanding results across various vision network architectures on ImageNet. In terms of top-1 accuracy, DNC surpasses the parametric classifier by **0.29%** and **0.28%** on ResNet50 and ResNet101, respectively. Additionally, DNC achieves remarkable results with Transformer architecture: **83.26%** vs 83.02% on Swin-S, and **83.68%** vs 83.36% on Swin-B. These results are particularly notable given DNC’s transparent, case-based reasoning nature. It is also worth mentioning that another nonparametric classifier, k -NN [187], reports a top-1 accuracy of 76.57% based on ResNet50. However, [187] was trained with 130 epochs, under which DNC achieves 76.64%. As discussed in §2.2.1, [187] demands substantial storage, retaining the entire ImageNet training set (*i.e.*, 1.2M images) to perform the k -NN decision rule, and suffers from low efficiency due to the extensive comparisons required between each test image and all training images. These limitations hinder the practical application of [187] in real, complex scenarios. In contrast, DNC relies only on a small set of class representatives (*i.e.*, 4 sub-centroids per class) for decision-making in classification tasks and does not require any additional computational resources during network deployment.

Analysis on Transferability. We further evaluate transfer learning performance by applying ImageNet-trained weights to the Caltech-UCSD Birds-200-2011 (CUB-200-2011) dataset [31], following [394–396]. The CUB-200-2011 dataset consists of 11,788 bird photographs divided into 200 categories, with 5,994 images for training and 5,794 images for testing. All models utilize the ResNet50 architecture [68] and are trained for 100 epochs.

We employ an SGD optimizer with an initial learning rate of 0.01, following a polynomial annealing policy. Standard data augmentation techniques such as flipping, cropping, and normalizing are applied. The experimental results are presented in Table 4.7. As observed, DNC achieves a Top-1 acc. improvement of **0.73%** and a Top-5 acc. improvement of **0.39%**. For a parametric softmax classifier, both the feature network and the softmax layer are fully learnable parameters. This implies that for a new task, it is necessary to finetune both the feature network and train a new softmax layer. In contrast, DNC requires *only* the feature network to be learnable. The class centers are not freely learnable parameters but are instead directly computed from the training data in the feature space. For a new task, DNC needs to finetune only the feature network, while the class centers are recalculated from the training data based on clustering assignments, without requiring end-to-end training. This characteristic grants DNC superior transferability during fine-tuning.

Table 4.7: **Classification top-1 and top-5 accuracy on CUB-200-2011 test** [31].

Model (ImageNet-trained) Backbone	top-1	top-5
ResNet [68] ResNet50	84.48%	96.31%
DNC-ResNet	85.21%	96.70%

Analysis on adaptive overfitting. We further evaluate DNC-ResNet50 on ImageNetv2 [41] test sets (*i.e.*, “Matched Frequency”, “Threshold0.7” and “Top Images”) to investigate the presence of adaptive overfitting to ImageNet [40] during training. Specifically, each test set contains 10 images for each ImageNet class, collected from MTurk. Each worker is asked to select images belonging to their target class, from several candidate images sampled from a large image pool as well as the ImageNet validation set. The output is a *selection frequency* for each image, *i.e.*, the fraction of MTurk workers selected the image in a task for its target class. Then three test sets are developed according to different principles defined on the selection frequency. For “Matched Frequency”, [41] first approximated the selection frequency distribution for each class using those “re-annotated” ImageNet validation images. According to these class-specific distributions, 10 test images are sampled from the candidate pool for each class. For “Threshold0.7”, [41] sampled 10 images from each class with selection frequency at least 0.7. For “Top Images”, [41] selected the 10 images with the highest selection frequency for each class. The results on these three test sets are shown in Table 4.8. As seen, our DNC exceeds the parametric softmax based ResNet50 by **+0.52-0.89%** top-1 and **+0.26-0.47%** top-5 acc., showing its generality and remaining free from adaptive overfitting during the training process.

4.2.2 Experiments on Semantic Segmentation

Dataset. The evaluation for semantic segmentation is conducted on three datasets: ADE20K [42], Cityscapes [43], and COCO-Stuff [44]. ADE20K comprises 20K/2K/3K

Table 4.8: **Classification top-1 and top-5 accuracy** on ImageNetv2 test sets [41].

Dataset	Method	Backbone	top-1	top-5
MatchedFrequency	ResNet [68]	ResNet50	63.30%	84.70%
	DNC-ResNet		63.96%	85.17%
Threshold0.7	ResNet [68]	ResNet50	72.70%	92.00%
	DNC-ResNet		73.59%	92.26%
TopImages	ResNet [68]	ResNet50	78.10%	94.70%
	DNC-ResNet		78.62%	94.96%

general scene images for **train/val/test**, respectively, covering 150 semantic categories. Cityscapes includes 2,975/500/1,524 urban scene images for training, validation, and testing, respectively, encompassing 19 classes. COCO-Stuff consists of 9K/1K images for **train/test**, respectively, featuring 80 object classes and 91 stuff classes.

Segmentation Network Architecture. For a comprehensive evaluation, we integrate DNC into three famous segmentation models: FCN [397], DeepLab_{v3} [398], and UperNet [399], using two backbone architectures: ResNet101 [68] and Swin-B [49]. The only architectural modification to the segmentation models is the removal of the “segmentation head,” specifically the 1×1 convolution-based, and the pixel-wise classification layer. For the backbone networks, we adopt both the traditional parametric classifier-based versions and our nonparametric, DNC-based versions, which are both trained on ImageNet [40] from Table 4.6, for initialization. Thus, for each segmentation model, we derive four variants from the different combinations of parametric and DNC versions of the backbone and segmentation network architectures.

Training. We utilize the `mmsegmentation`² codebase and adhere to the *default* training configurations. For training, we use FCN and DeepLab_{v3} with ResNet101, employing the SGD optimizer with an initial learning rate of 0.1. UperNet with Swin-B is trained using AdamW with an initial learning rate of 6e-5. For all models, the learning rate follows a polynomial annealing policy. Following common practices [3, 400], we train the models on ADE20K **train** with a crop size of 512×512 and a batch size of 16; on Cityscapes **train** with a crop size of 769×769 and a batch size of 8; and on COCO-Stuff **val** with a crop size of 512×512 and a batch size of 16. Models are trained for 160K iterations on ADE20K and Cityscapes datasets, and for 40K iterations on the COCO-Stuff dataset. Standard data augmentation techniques are employed, including scale and color jittering, flipping, and cropping. The hyper-parameters of DNC are set to the default values as: $K = 10$ and $\mu = 0.999$.

Performance on Segmentation. Shown Table 4.9, our DNC-based segmentation models, regardless of the specific DNC-based backbone used, consistently outperform their

²<https://github.com/open-mmlab/msegmentation>

Table 4.9: **Segmentation mIoU score** on ADE20K [42] val, Cityscapes [43] val and COCO-Stuff [44] val, respectively (top-1 acc. on ImageNet [40] val of backbones are also reported for reference).

Method	Backbone	ImageNet top-1 acc.	#Params	ADE20K mIoU	Cityscapes mIoU	COCO-Stuff mIoU
FCN [397]	ResNet101 [68]	77.52%	68.6M	39.9%	75.6%	32.6%
	DNC-ResNet101	77.80%	68.6M	40.4% \uparrow 0.5	76.3% \uparrow 0.7	32.9% \uparrow 0.3
DNC-FCN	ResNet101 [68]	77.52%	68.5M	41.1% \uparrow 1.2	76.7% \uparrow 1.1	33.0% \uparrow 0.4
	DNC-ResNet101	77.80%	68.5M	42.3% \uparrow 2.4	77.5% \uparrow 1.9	33.5% \uparrow 0.9
DeepLabv3 [398]	ResNet101 [68]	77.52%	62.7M	44.1%	78.1%	36.0%
	DNC-ResNet101	77.80%	62.7M	44.6% \uparrow 0.5	78.7% \uparrow 0.6	36.3% \uparrow 0.3
DNC-DeepLabv3	ResNet101 [68]	77.52%	62.6M	45.0% \uparrow 0.9	79.1% \uparrow 1.0	36.5% \uparrow 0.5
	DNC-ResNet101	77.80%	62.6M	45.7% \uparrow 1.6	79.8% \uparrow 1.7	36.8% \uparrow 0.8
UperNet [399]	Swin-B [49]	83.36%	90.6M	48.0%	79.8%	42.7(7)%
	DNC-Swin-B	83.68%	90.6M	48.4% \uparrow 0.4	80.1% \uparrow 0.3	42.8(4)% \uparrow 0.07
DNC-UperNet	Swin-B [49]	83.36%	90.5M	48.6% \uparrow 0.6	80.5% \uparrow 0.7	43.1% \uparrow 0.3
	DNC-Swin-B	83.68%	90.5M	50.5% \uparrow 2.5	80.9% \uparrow 1.1	43.3% \uparrow 0.5

parametric counterparts, such as FCN + ResNet101, DeepLabv3 + ResNet101, and UperNet + Swin-B, across various segmentation tasks and backbone architectures. For instance, the original FCN model achieves 39.91%, 75.6%, and 32.6% mIoU on ADE20K, Cityscapes, and COCO-Stuff, respectively. In comparison, using the same ResNet101 backbone, DNC-FCN improves these scores to 41.1%, 76.7%, and 33.0%. Further improvements are observed when using the DNC pre-trained backbone, DNC-ResNet101. Our DNC-FCN model continues outperforming its parametric counterpart, achieving 42.3% *vs.* 40.4% on ADE20K, 77.5% *vs.* 76.3% on Cityscapes, and 33.5% *vs.* 32.9% on COCO-Stuff. Similar trends are observed for DeepLabv3 and UperNet, demonstrating the robust performance gains provided by the DNC approach.

Analysis on Transferability. A notable feature of DNC is its robust transferability, as it learns to classify by directly comparing data samples within the feature space. The results presented in Table 4.9 further support this point. For instance, even the parametric classifier-based segmentation model DeepLabv3 can demonstrate significant performance improvements when fine-tuned with DNC-ResNet101: 44.6% *vs.* 44.1% on ADE20K, 78.7% *vs.* 78.1% on Cityscapes, and 36.3% *vs.* 36.7% on COCO-Stuff. Additionally, when using DNC-DeepLabv3, performance further improves after replacing ResNet101 with DNC-ResNet101, achieving 45.7% *vs.* 45.0% on ADE20K, 79.8% *vs.* 79.1% on Cityscapes, and 36.7% *vs.* 36.5% on COCO-Stuff. This improvement can be attributed to the fact that when both the backbone and the segmentation networks are built upon DNC, the model only needs to adapt the original representation space to the target task, eliminating the need to learn additional new parameters.

Table 4.10: **Semantic segmentation** results on S3DIS [45] Area 5 and ScanNet v2 [46] (Ranked by S3DIS’s results).

Method	S3DIS [45]			ScanNet v2 [46]	
	OAcc	mAcc	mIoU	mIoU	Test mIoU Val
JSNet [401] [AAAI’20]	87.7%	61.4%	54.5%	-	-
SegGCN [402] [CVPR’20]	88.2%	70.4%	63.6%	58.9%	-
SCF-Net [403] [CVPR’21]	88.4%	71.6%	82.7%	-	-
PACConv [404] [CVPR’21]	-	-	66.6	-	-
RepSurf-U [405] [CVPR’22]	90.2%	76.0%	68.9%	-	-
CBL [406] [CVPR’22]	90.6%	75.2%	69.4%	-	-
PTv1 [407] [ICCV’21]	90.8%	76.5%	70.4%	-	70.6%
FastPT [408] [CVPR’22]	-	77.9%	70.3%	-	-
PointMixer [409] [ECCV’22]	-	77.4%	71.4%	-	-
PTv2 [410] [NeurIPS’22]	91.1%	77.9%	71.6%	75.2%	75.4%
StratifiedFormer [411] [CVPR’22]	91.5%	78.1%	72.0%	73.7%	74.3%
RandLA-Net [412] [CVPR’20]	-	-	-	64.5%	-
PointASNL [413] [CVPR’20]	-	-	-	66.6%	63.5%
RPNet [414] [ICCV’21]	-	-	-	68.2%	-
FusionNet [415] [ECCV’20]	-	-	-	68.8%	-
JSENet [416] [ECCV’20]	-	-	-	69.9%	-
RepSurf-U [405] [CVPR’22]	-	-	-	70.2%	-
PointNeXt [417] [NeurIPS’22]	-	-	-	71.2%	71.5%
DCR-PointTransformer V2	92.0%	78.5%	72.2%	76.1%	75.9%

Table 4.11: **Instance segmentation** results on S3DIS [45] Area 5 and ScanNet v2 [46].

Method	Evaluation Metric			
	S3DIS [45]			
	mCov	mWCov	mPrec	mRec
JSNet [401] [AAAI’20]	48.7	51.5	62.1	46.9
PointGroup [418] [CVPR’20]	-	-	61.9	62.1
MaskGroup [419] [ICME’22]	-	-	62.9	64.7
SSTNet [420] [ICCV’21]	42.7	59.3	65.5	64.2
DyCo3D [421] [CVPR’21]	63.5	64.6	64.3	64.2
HAIS [422] [ICCV’21]	64.3	66.0	71.1	65.0
DKNet [423] [ECCV’22]	64.7	65.6	70.8	65.3
DCR-PointTransformer V2	66.5	68.1	69.6	66.3
	Test		Val	
	mAP	mAP ₅₀	mAP	mAP ₅₀
3D-MPA [424] [CVPR’20]	35.5	61.1	35.5	59.1
DyCo3D [421] [CVPR’21]	39.5	64.1	35.4	57.6
PointGroup [418] [CVPR’20]	40.7	63.6	34.8	56.7
MaskGroup [419] [ICME’22]	43.4	66.4	42.0	63.3
HAIS [422] [ICCV’21]	45.7	69.9	43.5	64.1
OccuSeg [425] [CVPR’20]	48.6	67.2	44.2	60.7
SoftGroup [426] [CVPR’22]	50.4	76.1	46.0	67.6
SSTNet [420] [ICCV’21]	50.6	69.8	49.4	64.3
DCR-PointTransformer V2	51.1	77.9	47.7	68.3

Table 4.12: **Panoptic segmentation** results on SemanticKITTI [47] test.

Method	PQ	PQ [†]	RQ	SQ	PQ Th	RQ Th	SQ Th	PQ St	RQ St	SQ St
LPSAD [427] [IROS’20]	38.0%	47.0%	48.2%	76.5%	25.6%	31.8%	76.8%	47.1%	60.1%	76.2%
Panoster [428] [RAL’21]	52.7%	59.9%	64.1%	80.7%	49.4%	58.5%	83.3%	55.1%	68.2%	78.8%
Panoptic-PolarNet [429] [CVPR’21]	54.1%	60.7%	65.0%	81.4%	53.3%	60.6%	87.2%	54.8%	68.1%	77.2%
DS-Net [430] [CVPR’21]	55.9%	62.5%	66.7%	82.3%	55.1%	62.8%	87.2%	56.5%	69.5%	78.7%
EfficientLPS [431] [TR’21]	57.4%	63.2%	68.7%	83.0%	53.1%	60.5%	87.8%	60.5%	74.6%	79.5%
GP-S3Net [432] [ICCV’21]	60.0%	69.0%	72.1%	82.0%	65.0%	74.5%	86.6%	56.4%	70.4%	78.7%
SCAN [433] [AAAI’22]	61.5%	67.5%	72.1%	84.5%	61.4%	69.3%	88.1%	61.5%	74.1%	81.8%
Panoptic-PHNet [434] [CVPR’22]	61.5%	67.9%	72.1%	84.8%	63.8%	70.4%	90.7%	59.9%	73.3%	80.5%
DCR-PointTransformer V2	62.1%	68.4%	74.2%	83.5%	65.4%	72.1%	89.0%	61.6%	75.3%	79.2%

4.2.3 Experiments on Point Cloud Segmentation

We further extend DNC to point cloud segmentation tasks, including semantic, instance and panoptic segmentation, to validate the high versatility of DNC. PointTransformer V2 [410] is applied as the encoder for all three tasks for consistency. The original “semantic head” is replaced by the proposed nonparametric, DNC based version.

Dataset. For point cloud semantic and instance segmentation, we utilize two datasets for training and evaluation (*i.e.*, S3DIS [45], ScanNet V2 [46] datasets). S3DIS [45] is a commonly applied large-scale indoor point cloud dataset, including point clouds from 271 rooms across 6 areas. Each room is represented by a medium-sized point cloud, annotating each point with a semantic label. ScanNet V2 [46] covers over 1,500 indoor scenes and 2.5 million annotated RGB-D images with 90% surface coverage. It is evaluated on 20 semantic classes, which includes 18 different object classes. For point cloud panoptic segmentation, we follow common practice [429, 435] and apply SemanticKITTI [47], which is

a developed version of the well-known KITTI Vision [436] benchmark on complex outdoor traffic scenarios. Specifically, it contains 22 data sequences, consisting of 43,552 frames of outdoor scenes. For `train/val/test` split, 23,201 frames with panoptic labels are applied for training and validation, and the remaining frames without labels are used for testing. Point-wise labels in 20 classes are annotated for segmentation tasks, 8 of which are “thing” classes.

Metric. Point cloud segmentation tasks (*i.e.*, semantic, instance and panoptic) require different evaluation metrics to provide meaningful comparisons. Specifically, to evaluate on point cloud semantic segmentation, we apply the mean Intersection-over-Union (mIoU), the overall accuracy (OAcc) taking all points into consideration and the average class accuracy (mAcc). For point cloud instance segmentation, mean coverage (mCov), mean weighed coverage (mWCov), mean precision (mPrec), and mean recall (mRec) are applied as the evaluation metrics for S3DIS [45]. Additionally, ScanNet V2 [46] is evaluated by mean average precisions (mAPs) under different IoU thresholds. For point cloud panoptic segmentation, the panoptic quality (PQ) [437] is utilized as our main metric for evaluation. PQ is defined as the multiplication of segmentation quality (SQ) and recognition quality (RQ). Furthermore, these three metrics can be extended to things and stuff classes, denoted as PQ^{Th} , PQ^{St} , RQ^{Th} , RQ^{St} , SQ^{Th} , SQ^{St} , respectively. PQ^\dagger replaces PQ with IoU for stuff classes.

Performance on Point Cloud Segmentation. As summarized in Table 4.10, DNC achieves superior performance on point cloud semantic segmentation over previous approaches on both S3DIS [45] and ScanNet v2 [46] datasets, respectively. For instance, DNC outperforms PTv2 [410] by **0.9%** and **0.5%** mIoU on the ScanNet v2 [46] `test` and `val`, respectively. It also surpasses other methods across all discussed evaluation metrics on the S3DIS [45] dataset. We further evaluate DNC on point cloud instance segmentation and present the corresponding results in Tables 4.11. DNC surpasses most of the concurrent approaches on the S3DIS [45] Area 5 dataset by achieving scores of **66.5%**, **68.1%**, **69.6%** and **66.3%**, respectively, on mCov, mWCov, mPrec, and mRec metrics. We also observe similar trends on ScanNet V2 [46] `test` and `val`. Table 4.12 presents the results on point cloud panoptic segmentation on the SemanticKITTI [47] `test`. DNC demonstrates large performance gap to best-performing baseline approach, with a PQ score of **62.1%**. The results further validate the superior performance of the proposed DNC on both the “things” and the “stuff” classes, as DNC outperforms the second best method, Panoptic-PHNet [434], in 7 of the 10 relevant metrics. In summary, DNC displays a robust performance improvement across various point cloud segmentation tasks on both indoor and outdoor datasets. These impressive results also underscore the potential of applying DNC to more downstream visual-related tasks, either as a new classification network architecture or as a transferable backbone.

4.2.4 Experiments on Sub-Categories Discovery

Through unsupervised, within-class clustering, DNC represents each class as a set of automatically discovered class sub-centroids (*i.e.*, cluster center). This allows DNC to better describe the underlying, multimodal data structure and robust depict for rich intra-class variance. In other words, the proposed DNC can effectively capture sub-class patterns, which is conducive to algorithmic performance. Such a capacity of sub-pattern mining is also considered crucial for good transferable features – representations learned on coarse classes are capable of fine-grained recognition [438].

In order to quantify the ability of DNC for automatic sub-category discovery, we follow the experimental setup posed by [438] – learning the feature embedding using coarse-grained object labels, and evaluating the learned feature using fine-grained object labels. This evaluation paradigm enables us to assess the degree to which the deep model can discover variations within each category. A conjecture is that performance on this test correlates well with the network’s ability to identify and mine sub-class patterns during training, which the proposed DNC seeks to rigorously establish.

In particular, the network is first trained on coarse-grained labels with the baseline parametric softmax and the non-parametric DNC using the same network architecture. After training on coarse classes, we use the **top-1** nearest neighbor accuracy in the final feature space to measure the accuracy of identifying fine-grained classes. The classification performance evaluated in such setting is referred to [438] as **induction accuracy**. Results on CIFAR100 [39] and ImageNet [40] are presented, respectively.

Performance of Sub-category Discovery on CIFAR100. CIFAR100 includes both fine-grained annotations in 100 classes and coarse-grained annotations in 20 classes. We examine sub-category discovery by transferring representation learned from 20 classes to 100 classes. As shown in Table 4.13, DNC consistently outperforms the parametric counterpart: DNC increases 0.12%, in terms of the standard **top-1** accuracy, on both ResNet50 and ResNet101 architectures. Nevertheless, when transferred to CIFAR100 (*i.e.*, 100 classes) using k -NN, a significant loss occurs on the baseline: 53.22% and 54.31% **top-1** acc. on ResNet50 and ResNet101, respectively. Our features, on the other hand, provide 14.24% and 13.82% improvement over the baseline, achieving 67.46% and 68.13% **top-1** acc. on 100 classes on ResNet50 and ResNet101, respectively. In addition, in comparison the parametric model, our approach results in only a smaller drop in transfer performance, *i.e.*, **-18.87%** *vs* -32.99% on ResNet50, and **-18.47%** *vs* -32.17% on ResNet101.

Performance of Sub-category Discovery on ImageNet. Table 4.14 provides experimental results of sub-category discovery on ImageNet **val**. As in [438], 127 coarse ImageNet categories are obtained by top-down clustering of 1K ImageNet categories on WordNet tree. Training on the 127 coarse classes, DNC improves the performance of baseline by 0.10% and 0.03%, achieving 84.39% and 85.91% on ResNet50 and ResNet101, respectively. When transferring to the 1K ImageNet classes using k -NN, our features

Table 4.13: **Top-1 induction accuracy** on CIFAR-100 [39] **test** using CIFAR-20 pre-trained models. Numbers reported with k -nearest neighbor classifiers. See §4.2.4.

Method	Backbone	20 classes	100 classes
ResNet [68]	ResNet50	86.21%	53.22%
DNC-ResNet		86.33%	67.46%
ResNet [68]	ResNet101	86.48%	54.31%
DNC-ResNet		86.60%	68.13%

Table 4.14: **Top-1 induction accuracy** on ImageNet [40] **val** using ImageNet-127 pre-trained models. Numbers reported with k -nearest neighbor classifiers. See §4.2.4.

Method	Backbone	127 classes	1000 classes
ResNet [68]	ResNet50	84.29%	43.23%
DNC-ResNet		84.39%	52.21%
ResNet [68]	ResNet101	85.88%	45.31%
DNC-ResNet		85.91%	54.60%

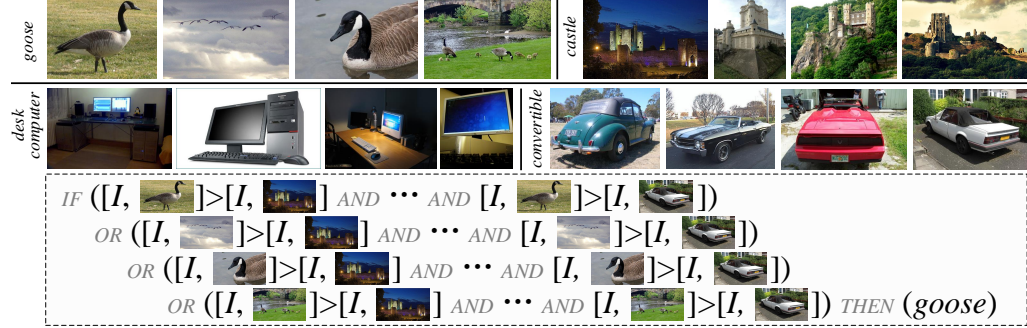
provide huge improvements, *i.e.*, 8.98% and 9.29%, over the baseline.

The promising transfer results on CIFAR100 and ImageNet serve as strong evidence to suggest that the proposed DNC is capable of automatically discovering meaningful sub-class patterns – latent visual structures that are not explicitly presented in the supervisory signal, and hence handle intra-class variance and boost visual recognition.

4.2.5 Study of Ad-hoc Explainability

As demonstrated empirically in §4.2.1-4.2.3, DNC combines the intuitive effectiveness of Nearest Centroids with the robust representation learning capabilities of DNNs, making it a transparent and powerful tool for visual recognition tasks with enhanced transferability. When the class sub-centroids are anchored to real observations (*i.e.*, actual images from the training dataset) rather than selected as cluster centers (*i.e.*, mean features of a set of training images), DNC gains improved *ad-hoc* explainability. Our results indicate that this approach is promising and incurs a negligible performance cost.

Experimental Setup. Following the same experimental settings in §4.2.1, we train the DNC version of ResNet50 on the ImageNet **train** dataset for 100 epochs. During the first 90 epochs, the model is trained in the standard manner, calculating the class sub-centroids as cluster centers. After this, we anchor each sub-centroid to its closest training image based on the cosine similarity of their features. In the final 10 epochs, the sub-centroids are updated solely based on the features of their anchored training images. Apart from this adjustment, all other training settings remained unchanged. This approach results in a more interpretable DNC-ResNet50.

Figure 4.4: *Top*: sub-centroid images. *Bottom*: rule created for “goose”.Table 4.15: **Classification top-1 and top-5 accuracy** on ImageNet [40] val, using clustercenter *vs* resembling real observation as class sub-centroids, based on DNC-ResNet50 architecture.

Sub-centroid	Architecture	top-1	top-5
<i>cluster center</i>	DNC-ResNet50	76.49%	93.08%
<i>real observation</i>		76.37%	93.04%
-	ResNet50 [68]	76.20%	93.01%

Interpretable Class Sub-centroids. The top of Fig. 4.4 displays our identified sub-centroid images for four ImageNet classes. These representative images exhibit diversity in appearance, viewpoint, illumination, scale, and various other attributes. They effectively capture the richness of their respective classes, offering human users a comprehensible perspective of their varied representatives.

Performance with Improved Interpretability. We then present the performance of our DNC-ResNet50, based on interpretable class representatives, on the ImageNet val. As shown in Table 4.15, enforcing class sub-centroids as actual training images results in only a marginal performance decrease (*e.g.*, 76.49% to 76.37% top-1 acc.), while significantly enhancing interpretability. Notably, the explainable DNC-ResNet50 still outperforms the standard black-box ResNet50, achieving 76.37% compared to 76.20%.

Explain Inner Decision-Making Mode based on *IF* \dots *Then* Rules. Using the simple Nearest Centroids mechanism, we can utilize representative images to form a set of *IF* \dots *Then* rules [69], providing an intuitive way to interpret the inner decision-making process of DNC for human users. Specifically, let \hat{I} represent a sub-centroid image for class c , $\check{I}_{1:T}$ denote representative images for all other classes, and I be a query image. One linguistic *IF* \dots *Then* rule that can be generated for \hat{I} :

$$\begin{aligned}
 &IF ([I, \hat{I}] > [I, \check{I}_1] \text{ AND } [I, \check{I}_2] > [I, \check{I}_2] \text{ AND } \dots \\
 &\text{AND } [I, \hat{I}] > [I, \check{I}_T]) \text{ THEN (class } c),
 \end{aligned} \tag{4.1}$$

where $[\cdot, \cdot]$ stands for similarity, given by DNC. The final rule for class c is created by

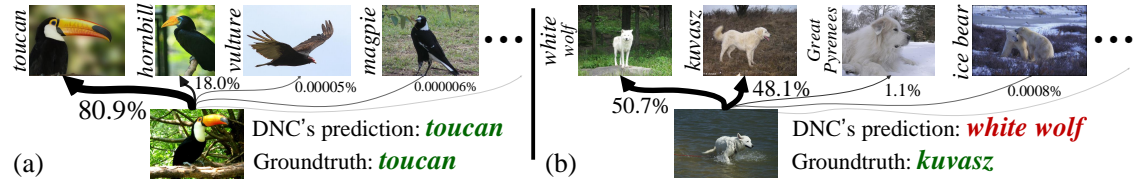


Figure 4.5: DNC can provide (dis)similarity-based interpretation. For the two test samples, we only plot the normalized similarities for their corresponding closest sub-centroids from top-4 scoring classes.

combining all the rules of K sub-centroid images $\hat{I}_{1:K}$ of class c (see Fig. 4.4 bottom):

$$\begin{aligned}
 &IF ([I, \hat{I}_1] > [I, \check{I}_1] \text{ AND} \cdots \text{ AND } [I, \hat{I}_1] > [I, \check{I}_T]) \\
 &OR ([I, \hat{I}_2] > [I, \check{I}_1] \text{ AND} \cdots \text{ AND } [I, \hat{I}_2] > [I, \check{I}_T]) \\
 &OR \cdots OR ([I, \hat{I}_K] > [I, \check{I}_1] \text{ AND} \cdots \text{ AND } [I, \hat{I}_K] > [I, \check{I}_T]) \\
 &THEN (\text{class } c).
 \end{aligned} \tag{4.2}$$

Interpret Prediction Based on (Dis)similarity to Sub-centroid Images. Based on interpretable class representatives, DNC can explain its predictions by allowing users to view and verify the computed (dis)similarity between the query and class sub-centroid images. Shown in Fig. 4.5(a), an observation is correctly classified because DNC determines it closely resembles a specific exemplar of a “toucan.” Conversely, in Fig. 4.5(b), DNC struggles to assign the observation between exemplars from “white wolf” and “kuvasz,” ultimately making an incorrect decision. Although users may not fully understand how DNC maps an image to its feature space, they can easily grasp the decision-making process [66] (*e.g.*, why one class is predicted over another) and verify the calculated (dis)similarity, which serves as the evidence for the classification decision.

4.2.6 Diagnostic Experiment

We further present extensive ablation studies on ResNet101 image classification, DeepLabV3 semantic segmentation and PointTransformer V2 [410] point cloud semantic segmentation models, respectively.

Class Sub-centroids. Table 4.16a examines the impact of varying the number of class sub-centroids (K) for each class. When $K = 1$, each class is represented by its centroid, the average feature vector of all the training samples of the class (Eq. 3.9), without any clustering. The corresponding baseline achieves a **top-1** acc. of 77.31% and a **mIoU** of 43.2% for classification and segmentation, respectively. For classification, increasing K from 1 to 4 results in improved performance (*i.e.*, 77.31% \rightarrow 77.80%), supporting our hypothesis that a single class weight/center is insufficient to capture the underlying data distribution. This validates the effectiveness of our clustering-based sub-class pattern mining approach. We did not use $K > 4$ due to the memory constraints of our hardware.

Table 4.16: A set of ablative experiments on ImageNet [40] val, ADE20K [42] val and S3DIS [45] Area 5.

K	ImageNet		K	ADE20K	K	S3DIS
	top-1	top-5		mIoU		OA _{acc}
1	77.31%	93.01%	1	43.2%	1	91.1%
2	77.54%	93.32%	5	44.0%	5	91.4%
3	77.68%	93.63%	10	44.3%	10	92.2%
4	77.80%	93.85%	20	44.0%	20	91.8%

(a) Number of sub-centroids (K) for each class

Memory (#batch)	ImageNet	
	top-1	top-5
0	77.49%	93.09%
700	77.58%	93.16%
800	77.64%	93.35%
900	77.75%	93.67%
1000	77.80%	93.85%

(b) Memory size

μ	ImageNet		ADE20K	S3DIS
	top-1	top-5	mIoU	OA _{acc}
0	73.82%	93.02%	42.7%	87.5%
0.9	76.41%	93.07%	43.6%	90.8%
0.99	77.33%	93.51%	44.0%	91.2%
0.999	77.80%	93.85%	44.3%	92.2%
0.9999	77.31%	93.48%	44.2%	91.7%

(c) Momentum coefficient (μ)

Output dimensionality	ImageNet	
	top-1	top-5
640	76.23%	92.83%
1024	76.28%	92.90%
1280	76.61%	93.12%
2048	76.49%	93.08%

(d) Output dimensionality

ϵ	ImageNet	
	top-1	top-5
0.01	76.34%	92.97%
0.05	76.49%	93.08%
0.1	76.40%	93.02%

(e) Temperature parameter ϵ in (3.12)

Balancing strategy	S3DIS
	OA _{acc}
w/o	91.4%
z^c	92.2%
Ext_p	91.6%
Ext_n	91.3%

(f) Class balance factor z^c in (3.15)

A similar trend is observed in segmentation; increasing the number of sub-centroids ($K : 1 \rightarrow 10$) yields a noticeable performance improvement (*i.e.*, 43.2% \rightarrow 44.3%). However, increasing K beyond 10 provides marginal or even negative gains. This could be due to over-clustering, which may identify insignificant patterns that are trivial or detrimental to decision-making.

External Memory. We then investigate the impact of external memory, utilized solely in image classification. As illustrated in Table 4.16b, DNC’s performance progressively improves with the expansion of memory size, achieving 77.80% top-1 acc. at a memory size of 1,000. However, the results have not yet reached a saturation point in performance, but are instead constrained by the upper limits of our hardware’s computational capacity.

Momentum Update. We also ablate the impact of the momentum coefficient μ (Eq. 3.14), which governs the speed of sub-centroid online updating. As shown in Table 4.16c, the behavior of μ is consistent across both tasks. Notably, DNC achieves optimal performance with higher coefficients (*i.e.*, $\mu \in [0.999, 0.9999]$), highlighting the importance of slow updates. Performance declines when $\mu \in [0.9, 0.99]$, and there is a significant drop when $\mu = 0$ (*i.e.*, only batch sub-centroids are used as approximations).

Output Dimensionality. As discussed in §4.1, the final output dimensionality of our DNC is set to match that of the last layer of the parametric counterpart to ensure a fair comparison. However, due to its distance- and similarity-based nature, DNC has the flexibility to accommodate any output dimensionality. In Table 4.16d, we further examine the impact of DNC’s output dimensionality. Notably, when the final output dimensionality is set to 1280, we achieve a **76.61%** top-1 acc., which surpasses the initial 2048-dimensional

configuration’s 76.49%. This improvement can be attributed to a better balance between memory capacity and feature dimensionality within the constraints of hardware computational budgets. By reducing the final output dimensionality, the expressiveness of the final feature is slightly diminished, but more image features can be stored in external memory, allowing for more accurate sub-centroid estimation.

Temperature Parameter. Next, parameter ε in (3.12) trades off convergence speed with closeness to the original transport problem [85, 86]. In Table 4.16e, we further study the impact of ε on ImageNet [40] val. We observe that the default $\varepsilon = 0.05$ reaches the best result among all selected values.

Class Balance. Lastly, we validate the effectiveness of our strategy on mitigating category imbalance discussed in §3.2.1. Shown in Table 4.16f, the baseline (*w/o*) fixes $z^c = 1$ in Eq. 3.15, reducing Eq. 3.16 to the form of Eq. 3.14. Ext_p [235] and Ext_n [439] are two alternative strategies employing parametric and non-parametric external memory for class balancing, respectively. The reported results suggest that our proposed calibration strategy outperforms all related class balancing strategies [235, 439], as well as the non-calibrated baseline.

4.3 Experiment for DVP

4.3.1 Comparisons with Current Methods

Qualitative Results. To ensure a fair comparison, we include six state-of-the-art baselines, including SDEdit [440], Prompt-to-Prompt [106], DiffuseIT [441], VQGAN-CLIP [442], Text2LIVE [443], and VISPROG [30], on various prompt-guided image translation tasks. The visualization results with DVP are presented in Fig. 4.6. For style transfer, integrating instance normalization enables our translated images to achieve a balance between closely adhering to the guidance layout and maintaining high fidelity to the target prompt. For instance, DVP vividly transfers the “young boy”, “church”, *etc.* into “robot”, “sandcastle”, *etc.* in Fig. 4.6. In contrast, other methods often produce corrupted translations in these RoIs or introduce excessive variations. DVP, on the other hand, demonstrates robust in-context reasoning by accurately recognizing and transferring objects as specified by the user. For instance, DVP selectively modify the “right fox” to the “arctic wolf,” while leaving other foxes and the background unaffected (see Fig. 4.6 fifth row). Though VISPROG enables local editing, it falls short in cultivating the context reasoning skills, and result in translating all “foxes” into “arctic wolves” (see §2.2.7). The overall results indicate that SDEdit and VQGAN-CLIP compromises between maintaining structural integrity and enabling significant visual modifications. DiffuseIT and Prompt2Prompt preserve the shape of the guidance image but introduce only minimal changes to its appearance. Text2Live demonstrates some correlation between textual descriptions and visual elements (*e.g.*, object shapes); however, transforming objects into



Figure 4.6: **Qualitative results with the state-of-the-art baselines.** DVP exhibits rich capability in style transfer, achieving realistic quality while retaining high fidelity. Owing to the context-free manipulation (see §3.2.2), the DVP framework is capable of flawlessly preserving the background scenes while specifically targeting the translation of the RoI. Note that while VISPROG also enables context-free editing, it exhibits considerable limitations in rational manipulation (see Fig. 4.8).



Figure 4.7: Visualization results of instance normalization compared with various guidance scales w .

Table 4.17: User study, CLIP-Score, and DINO-Score on the comparison between our proposed framework and state-of-the-art baselines. The metrics are detailed in §5.2.2.

Method	User Study			Quantitative Results	
	Quality	Fidelity	Diversity	CLIP-Score	DINO-Score
VQGAN-CLIP [442]	3.25	3.16	3.29	0.749	0.667
Text2Live [443]	3.55	3.45	3.73	0.785	0.659
SDEDIT [440]	3.37	3.46	3.32	0.754	0.642
Prompt2Prompt [106]	3.82	3.92	3.48	0.825	0.657
DiffuseIT [441]	3.88	3.87	3.57	0.804	0.648
VISPROG [30]	3.86	4.04	3.44	0.813	0.651
DVP (ours)	3.95	4.28	3.56	0.839	0.697

completely new ones often does not produce visually satisfactory results. Conversely, DVP effectively manages all examples across diverse scenarios and instructions.

Quantitative Comparisons. Discussed in §5.2.2, we present the user study, CLIP-Score, and DINO-Score results in Table 4.17. These results align with our visual evidence, showing that DVP significantly outperforms other competitors in both fidelity and quality, particularly in fidelity. While existing methods focus solely on semantic consistency, DVP advances further by incorporating visual programming understanding with cognitive reasoning, which preserves instance identity and background scenarios. This approach inherently enhances performance in terms of fidelity, making our model more versatile and applicable across various scenarios.

4.3.2 Systemic Diagnosis

This section analyzes the core design elements of DVP, focusing on *instance normalization guidance* in translation (§3.2.2) and *in-context reasoning* in visual programming (§3.2.2). Additionally, we explore the systemic advantages of our approach, highlighting its *explainable controllability* and *label efficiency*, coined by the neuro-symbolic paradigm.



Figure 4.8: Visualization results of in-context reasoning against attention-based Prompt2Prompt [29] and programming-based VISPROG [30].

Instance Normalization. Our condition-flexible diffusion model diverges from conventional approaches [29, 96] by employing instance normalization guidance to enhance the robustness of translations and capacity to manage variations in the input distribution (see §3.2.2). To verify diffusion model methods with instance normalization and guidance scale [29], we conduct extensive experiments in Fig. 4.7 and Table 4.18(a) qualitatively and quantitatively.

Our condition-flexible diffusion model differentiates itself from conventional approaches [29, 96] by utilizing instance normalization guidance to improve the robustness of translations and better handle variations in input distribution (see §3.2.2). To validate diffusion model methods incorporating instance normalization and guidance scales [29], we conduct extensive qualitative and quantitative experiments in Fig. 4.7 and Table 4.18(a), respectively. To ensure fairness, these comparisons are conducted *without* integrating in-context visual programming into our approach. We select a guidance scale parameter w centered around 7.5 for the linear combination, as recommended by the default settings in [29, 96]. To examine its impact on the translated images, we vary the parameter in increments of 2.5. While the guidance scale baseline shows significant variations in translated images (see Fig. 4.7, right 4 columns) with $w \in \{2.5, 5, 7.5, 10\}$, instance normalization achieves high fidelity and natural, stable translation *without* the need for manually tuned parameters.

In-context Reasoning. DVP employs a set of visual programming operations for image translation, thereby facilitating a powerful in-context reasoning capability during image manipulation. DVP utilizes a series of visual programming operations to achieve image translation, enabling robust in-context reasoning capabilities during image manipulation. To better demonstrate our claim, we present and compare our translated results with the robust cross-attention map baseline method, Prompt2Prompt [29] in Fig. 4.8. Specifically, in the first row of Fig. 4.8, our goal is to translate *only* the right pigeon into an eagle. The cross-attention map on Prompt2Prompt reveals that it recognizes both pigeons, albeit with a notable failure to discern the positional information accurately. As a result, the model incorrectly translates both pigeons as eagles. In contrast, our approach demonstrates a strong in-context understanding of the scene and accurately translates the designated pigeon as instructed. In the second row example in Fig. 4.8, we design more

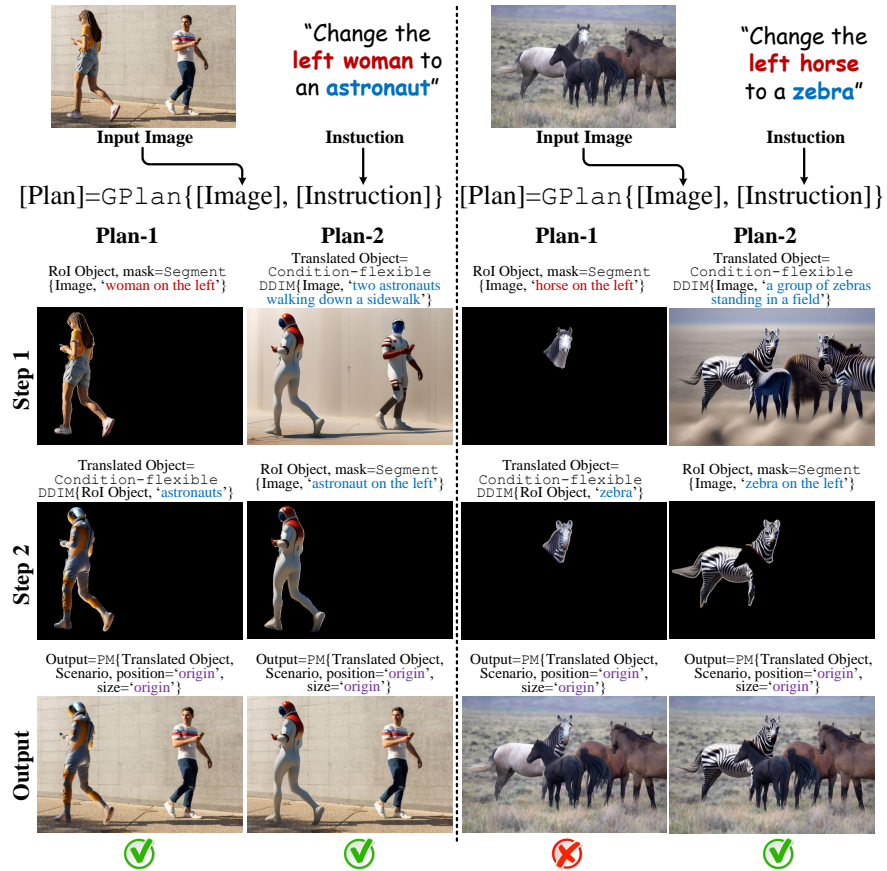


Figure 4.9: **Explainable controllability** during program execution, which is easy for error assessment and correction.

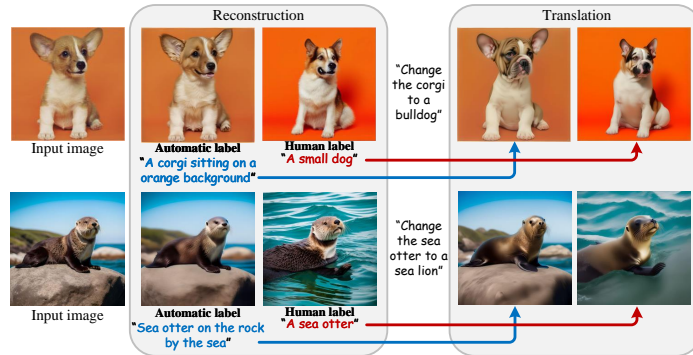


Figure 4.10: **Human-annotated and *Prompter*-generated descriptions.** Descriptions generated via the *Prompter* yield finer reconstructions, as well as subsequent translated outputs, suggesting a relaxation of label dependency.

sophisticated instructions. For Prompt2Prompt, the ambiguous cross-attention on the elephant results in erroneous artifacts across various animals in the translation. This method also shows limitations in performing RoI relation editing (*i.e.*, **Swap**, **Enlarge**). We further compare DVP with VISPROG [30], a visual programming approach that facilitates local image editing (see §2.2.7). Although both DVP and VISPROG support instructive object replacement (Fig. 4.8 right section), VISPROG falls short in translation fidelity (*e.g.*, the translated duck appears too yellow, and the lion assumes a different pose). Moreover, VISPROG fails to support relation manipulations, whereas DVP consistently produces commendable results across all examples, following strictly to human instructions.

Explainable Controllability. Next, we explore the explainable controllability during program execution (see §3.2.2). In this framework, we enable multiple operations to work in parallel, allowing for different program plans that can accommodate diverse sequences of operations. As shown in Fig. 4.9 (left section), the task “change the left woman to an astronaut” can be accomplished using either **Plan-1** or **Plan-2**. During execution, the program runs line-by-line, triggering specified operations and producing human-interpretable intermediate outputs at each step. This approach facilitates systemic explainability for error correction. For instance, as shown in Fig. 4.9 (right section), **Plan-1** mistakenly segments only the head of the “left horse,” resulting in an incomplete transformation to a “zebra.” Thanks to the explainable outputs provided at each step, we can identify specific issues and then employ **Plan-2** as a more suitable strategy for translating the “left horse.”

Label Efficiency. *Prompter* generates detailed image descriptions for arbitrary input images, reducing dependency on human annotations (see §3.2.2). To verify its efficacy, we conducted a user study (Table 4.18(b)) and provided visual evidence (Fig. 4.10), comparing human-generated annotations with those produced by *Prompter*. This ablative study focuses on investigating the impact of labels on translated images, hence visual

Table 4.18: **Ablative results** on instance normalization guidance and label efficiency.

Metrics	$w = 2.5$	$w = 5$	$w = 7.5$	$w = 10$	Inst. norm
CLIP-Score	0.833	0.795	0.742	0.686	0.839
DINO-Score	0.664	0.681	0.712	0.678	0.697

(a) Instance normalization

Methods	CLIP-Score	DINO-Score
Human-annotated	0.817	0.688
<i>Prompter</i>	0.839	0.697

(b) Label efficiency

programming is excluded from the experimental design. In Fig. 4.10, 60% of the optimization steps are bypassed to ensure that the reconstructed images do not closely resemble the input images, even when prompts are entirely erroneous. The results indicate that the GPT-4 generated annotations lead to superior performance in both CLIP-Score and DINO-Score (Table 4.18(b)), suggesting that these annotations enhance the quality of translated images. The visual evidence in Fig. 4.10 supports this claim: with detailed image descriptions, the reconstruction phase yields finer results, improving the quality of the final translated images.

4.4 Experiment for E²VPT

4.4.1 Comparison with State-of-the-Arts

We rigorously evaluate the performance and robustness of E²VPT on ViT [24], Swin [49], as well as on two self-supervised learning objectives: MAE [50] and MoCo v3 [51].

E²VPT on ViT. We present the average accuracy scores on VTAB-1k and FGVC benchmarks across four diverse task groups, based on three runs, in Table 4.19. This comparison includes E²VPT and eight other tuning protocols under the *pretrain-then-finetune* paradigm. Specifically, Full [48] updates both the backbone and classification head, while Linear [48], Partial-1 [21] (top layer), and MLP-3 [233] (3 MLP layers) are partial tuning methods that update only a subset of parameters. Sidetune [123], Bias [122], and Adapter [22] are extra module methods that introduce new trainable parameters to the backbone for adaptation. VPT [23] is the most recent visual prompt tuning method. From these results, several key observations emerge. **First**, E²VPT outperforms the full fine-tuning method in most cases, specifically in 21 out of 24 tasks. For instance, our model achieves a **0.68%** improvement on FGVC and a **9.75%** improvement on VTAB-1k Structured, demonstrating the effectiveness of our approach for rapid large-scale vision model adaptation. Additionally, our model trains only **0.39%** of the backbone parame-

Table 4.19: **Image classification accuracy for ViT-Base/16 [24]** pretrained on supervised ImageNet-21k. Following [23], we report the average test accuracy (three runs) on FGVC [23] and VTAB-1k [25] benchmarks, and “Number of Wins” in $[\cdot]$ compared to full fine-tuning (Full) [48]. “Tuned/Total” is the average percentage of tuned parameters required by 24 tasks. “Scope” indicates the tuning scope of each method. “Additional parameters” is the existence of parameters in addition to the pretrained backbone and linear head. The highest accuracy among all approaches except FULL are shown in **bold**. E²VPT outperforms the full fine-tuning in **19 of 24** instances with far fewer trainable parameters. More impressively, we further report “Number of Wins to VPT” in $\{\cdot\}$. Our method beats VPT in **21 of 24** cases with considerably lower parameters. Same for Table 4.20 and 4.21.

ViT-Base/16 [24] (85.8M)	Tuned/ Total	Scope		Extra params	FGVC [23] [5]	VTAB-1k [25] [19]		
		Input	Backbone			Natural [7]	Specialized [4]	Structured [8]
Full [CVPR22] [48]	100.00%		✓		88.54%	75.88%	83.36%	47.64%
Linear [CVPR22] [48]	0.08%				79.32% [0]	68.93% [1]	77.16% [1]	26.84% [0]
Partial-1 [NeurIPS14] [21]	8.34%				82.63% [0]	69.44% [2]	78.53% [0]	34.17% [0]
MLP-3 [CVPR20] [233]	1.44%			✓	79.80% [0]	67.80% [2]	72.83% [0]	30.62% [0]
Sidetune [ECCV20] [123]	10.08%		✓	✓	78.35% [0]	58.21% [0]	68.12% [0]	23.41% [0]
Bias [NeurIPS17] [122]	0.80%		✓		88.41% [3]	73.30% [3]	78.25% [0]	44.09% [2]
Adapter [NeurIPS20] [22]	1.02%		✓	✓	85.66% [2]	70.39% [4]	77.11% [0]	33.43% [0]
VPT [ECCV22] [23]	0.73%	✓		✓	89.11% [4]	78.48% [6]	82.43% [2]	54.98% [8]
Ours	0.39%	✓	✓	✓	89.22% [4] {4}	80.01% [6] {5}	84.43% [3] {4}	57.39% [8] {7}

Table 4.20: **Image classification accuracy for Swin-Base [49]** pretrained on supervised ImageNet-21k.

Swin-Base [49] (86.7M)	Tuned/ Total	VTAB-1k [25] [19]		
		Natural [7]	Specialized [4]	Structured [8]
Full [ICLR23] [444]	100.00%	79.10%	86.21%	59.65%
Linear [ICLR23] [444]	0.06%	73.52% [5]	80.77% [0]	33.52% [0]
Partial-1 [NeurIPS14] [21]	14.58%	73.11% [4]	81.70% [0]	34.96% [0]
MLP-3 [CVPR20] [233]	2.42%	73.56% [5]	75.21% [0]	35.69% [0]
Bias [NeurIPS17] [122]	0.29%	74.19% [2]	80.14% [0]	42.42% [0]
VPT [ECCV22] [23]	0.25%	76.78% [6]	83.33% [0]	51.85% [0]
Ours	0.21%	83.31% [6] {6}	84.95% [2] {3}	57.35% [3] {7}

ters, significantly enhancing parameter efficiency compared to the full fine-tuned model. **Second**, prompt tuning approaches generally surpass other parameter-efficient methods, such as partial fine-tuning (Partial-1) and extra modules (Adapter), highlighting the superior adaptability of prompt tuning methods for large-scale vision models. Moreover, the number of tunable parameters in prompt tuning methods is smaller compared to other methods. **Third**, our approach consistently outperforms the strong VPT model with fewer tunable prompts, illustrating the effective design of key-value prompting and efficient prompt pruning. This is because VPT focuses solely on designing input visual prompts, which fail to capture accurate interactions between image patches in new data. In contrast, the key-value prompts in E²VPT effectively bridge this gap.

E²VPT on Hierarchical Transformer. To demonstrate the effectiveness and generaliz-

Table 4.21: **Image Classification accuracy for different pretrained objectives** — MAE [50] and MoCo v3 [51] with ViT-Base [24] as backbone. Our method enjoys significant performance gains to VPT [23] while having lower parameter usage.

Pretrained objectives	MAE [50]					MoCo v3 [51]				
	Method	Tuned/ Total	VTAB-1k [25] [19]			Tuned/ Total	VTAB-1k [25] [19]			
<i>Natural</i> [7]			<i>Specialized</i> [4]	<i>Structured</i> [8]	<i>Natural</i> [7]		<i>Specialized</i> [4]	<i>Structured</i> [8]		
Full [CVPR22] [48]	100.00%	59.31%	79.68%	53.82%	100.00%	71.95%	84.72%	51.98%		
Linear [CVPR22] [48]	0.04%	18.87% [0]	53.72% [0]	23.70% [0]	0.04%	67.46% [4]	81.08% [0]	30.33% [0]		
Partial-1 [NeurIPS14] [21]	8.30%	58.44% [5]	78.28% [1]	47.64% [1]	8.30%	72.31% [5]	84.58% [2]	47.89% [1]		
Bias [NeurIPS17] [122]	0.16%	54.55% [1]	75.68% [1]	47.70% [0]	0.16%	72.89% [3]	81.14% [0]	53.43% [4]		
Adapter [NeurIPS20] [22]	0.87%	54.90% [3]	75.19% [1]	38.98% [0]	1.12%	74.19% [4]	82.66% [1]	47.69% [2]		
VPT [ECCV22] [23]	0.10%	36.02% [0]	60.61% [1]	26.57% [0]	0.06%	70.27% [4]	83.04% [0]	42.38% [0]		
Ours	0.07%	59.52% [4] {6}	77.80% [1] {2}	44.65% [3] {8}	0.13%	76.47% [4] {7}	87.28% [2] {4}	54.91% [6] {8}		

Table 4.22: **Impact of different components** in E²VPT on two instances: VTAB-1k *Natural* SVHN [34] and FGVC NABirds [52].

Fine-tuning Techniques			VTAB-1k <i>Natural</i> SVHN [34]			FGVC NABirds [52]		
Visual Prompts	Key-Value Prompts	Pruning & Rewinding	Pruning	Tuned / Total	Accuracy	Pruning	Tuned / Total	Accuracy
✓			0.0%	0.54%	78.1%	0.0%	1.02%	84.2%
✓	✓		0.0%	0.55%	83.8%	0.0%	1.05%	84.5%
✓		✓	56.3%	0.42%	79.0%	34.4%	0.63%	84.2%
✓	✓	✓	62.5%	0.43%	85.3%	40.0%	0.65%	84.6%

ability of our architectural design, we extend E²VPT to a hierarchical transformer model — Swin Transformer [49]. In this model, the MSA layer operates within local shifted windows, and patch embeddings are merged at deeper layers. To maintain generality, we adhere to the same settings as in the ViT [24] architecture by prepending learnable Key-Value pairs and following the approach in [23] for modifying input vectors (*i.e.*, these learnable vectors are attended within the local windows but ignored during patch merging). During pruning, we observed a performance drop when incorporating pruning within the deeper local windows. Consequently, we restricted the pruning stage to the first stage only. Since Swin does not utilize [CLS] tokens and employs global pooling for the classification head [23, 49], we adopted this design when adapting our method. The experiments were exclusively conducted on the ImageNet-21k supervised pretrained Swin-Base [49]. Our E²VPT consistently outperformed all other parameter-efficient methods across all three VTAB-1k problem classes. Notably, for the first time, it surpassed full fine-tuning on VTAB-1k *Specialized* and *Structured* tasks using significantly fewer parameters (*i.e.*, **0.21%**).

Different Pretraining Methods. We conducted experiments using two self-supervised objectives, MAE [50] and MoCo v3 [51], on backbones pretrained without labeled data, following VPT [23]. While VPT yielded inconclusive results with these objectives, our proposed method, E²VPT, outperformed other methods and achieved performance competitive with full fine-tuning. Specifically, E²VPT excelled in **8 out of 19** instances under

Table 4.23: **Prompt location and Initialization** results on VTAB-1k [25] in three runs.

ViT-Base/16 [24] (85.8M)		VTAB-1k [25] [19]		
		<i>Natural</i> [7]	<i>Specialized</i> [4]	<i>Structured</i> [8]
(a)	After	80.67% [6]	84.30% [3]	56.76% [8]
	Before	80.01% [6]	84.43% [3]	57.39% [8]
(b)	<i>Trunc. Norm.</i> [445]	79.77% [6]	84.30% [3]	56.36% [8]
	<i>He</i> [446]	80.01% [6]	84.43% [3]	57.39% [8]

MAE and **12 out of 19** instances under MoCo v3, while using significantly fewer model parameters (**0.07%** for MAE and **0.13%** for MoCo v3). Additionally, our method surpassed VPT by a substantial margin (**59.52%** vs. 36.02% under MAE on VTAB-1k *Natural*). Leveraging the insights from VPT, which suggest that self-supervised ViTs are fundamentally different from supervised ones, we demonstrated the versatility and effectiveness of our method across both pretraining objectives.

4.4.2 Diagnostic Experiments

Impact of Different Components. We first study the impact of different components in E²VPT, including visual prompts, key-value prompts, and pruning and rewinding. As shown in Table 4.22, we propose two example instances in FGVC [23] and VTAB-1k [25] benchmarks respectively for reference. For SVHN [34], the model with visual prompts alone achieves 78.1% in accuracy. Adding key-value prompts or pruning and rewinding technique brings additional gains (*i.e.*, **5.7%/0.9%**), validating the effectiveness of our prompt tuning technique in self-attention module as well as the pruning mechanism. It is not surprising to see that combing all the components together leads to the best performance, yielding 85.3% in accuracy. We find similar trends on FGVC NABirds [52]. **Prompt Location.** A fundamental distinction between E²VPT and other methods is the introduction of learnable key-value prompts to self-attention. In our implementation, we prepend these key-value prompts to the sequence of Key and Value matrices. Thus, further investigation is necessary to determine the optimal placement of the learnable prompts. We provide comprehensive ablation results on VTAB-1k in Table 4.23(a), demonstrating that both prepending the learnable prompts before or after the Key and Value matrices yield competitive results. This validates the robustness of our approach regarding prompt locations. We adopt the “Before” placement as our baseline method in all experiments, as it achieves slightly better average results (73.94% *vs.* 73.91%).

Initialization. Table 4.23(b) presents the performance of our approach using two widely adopted initialization methods: *truncated normal* [445, 447] and *He initialization* [446], evaluated on the VTAB-1k benchmark. The results indicate that *He initialization* generally provides more stable and favorable performance on average. However, in some specific tasks, such as Diabetic Retinopathy Detection in the VTAB-1k *Specialized* category,

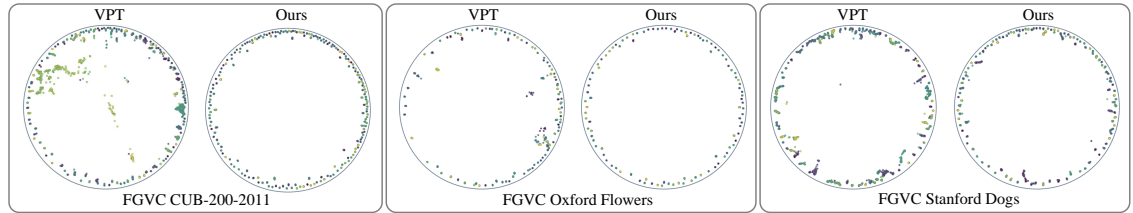


Figure 4.11: **Hyperbolic visualization results** from VPT [23] and ours on 3 FGVC tasks (*i.e.*, FGVC CUB-200-2011 [31], Oxford Flowers [32] and Stanford Dogs [33]). Our method shows consistently better clustering pushed to the border of the Poincaré disk.

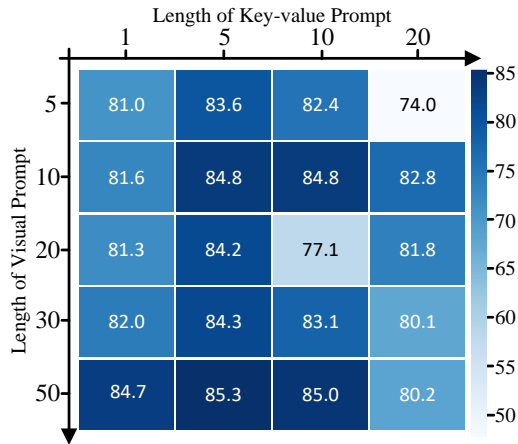


Figure 4.12: **Sensitivity of input prompt and key-value prompt lengths.** We vary the number of prompts for different combinations, and show their results on VTAB-1k *Natural* SVHN [34].

truncated normal outperforms *He* by 1.1% in accuracy. Overall, E²VPT demonstrates robustness across different initialization methods and achieves consistent performance comparable to full fine-tuning.

Prompt Length. The only hyper-parameter that needs tuning in E²VPT is the prompt length. To analyze the impact of different prompt lengths on model performance, we conducted a comprehensive study on the lengths of visual prompts and key-value prompts, specifically focusing on their characteristics on the VTAB-1k *Natural* SVHN dataset [34]. The lengths of visual prompts are tested at [5, 10, 20, 30, 50], while key-value prompts are tested at [1, 5, 10, 50], which is a standard configuration for most datasets. The results of model performance with different combinations of prompt lengths are shown in Fig. 4.12. The results indicate that when using 50 visual prompts, a relatively shorter key-value prompt can significantly improve performance (*e.g.*, 84.7% with one key-value prompt compared to 78.1% without key-value prompts). However, further increasing the

length of the key-value prompt provides only a small performance gain (*e.g.*, 85.3% with five key-value prompts). Additionally, using a large number of key-value prompts results in subpar performance (*e.g.*, 80.2% with 20 key-value prompts). Similar patterns were observed with other visual prompt lengths. We hypothesize that excessive parameter engineering in the self-attention layer might distort the original attention map, adversely affecting adaptation performance.

4.4.3 Visualization

Following [448–452], we present hyperbolic visualization results on the training sets for both VPT and our approach across three FGVC tasks (*i.e.*, CUB-200-2011 [31], Oxford Flowers [32], and Stanford Dogs [33]). Hyperbolic space, a Riemannian manifold with constant negative curvature, is used in our visualizations. Among several isometric models of hyperbolic space, we adhere to the Poincaré ball model, following [449, 450]. Similar to [449], we employ UMAP [453] with the “hyperboloid” distance metric to reduce dimensionality to 2D. ViT-Base serves as the encoder with two types of pretraining (*i.e.*, tuned models under VPT, and our models post-rewinding). During fine-tuning, the models are frozen, and the output embeddings are mapped to hyperbolic space. For all settings, we use the Adam optimizer [454] with a learning rate of 3×10^{-5} , a weight decay of 0.01, and a batch size of 900. All models are trained for 50 steps with gradient clipping by norm 3 to ensure a fair comparison.

Figure 4.11 illustrates the arrangement of learned embeddings on the Poincaré disk. In E²VPT, samples are clustered according to their labels, with each cluster pushed closer to the disk’s border, demonstrating effective class separation by the encoder. In contrast, VPT shows some samples moving towards the center and intermixed [449], suggesting potential confusion during projection.

4.5 Experiment for AMD

Table 4.24: The results of knowledge distillation methods on CIFAR-10 [27] and CIFAR-100 [27] datasets. All results are reported using the same teacher and student (10% scale). The best results are highlighted in **bold**, and the second best are shown in underline. Follow [53], as our student design via *Structural Pruning* is *orthogonal* to current approaches (see §3.3.2), we rerun and reproduce the results of all methods based on author-provided or author-verified code, and report the average performance over five runs. The GPU hours are estimated with respect to their conventional counterparts. ↓ denotes the performance lost with respect to their teacher models, a smaller gap represent a superior performance.

Method	FLOPs	CIFAR-10 [27] top-1	CIFAR-100 [27] top-1	GPU hours
ViT – Tiny _{100%} (teacher)	1.3G	95.98%	76.12%	-
ViT – Tiny _{10%} KD [arXiv15] [143]	0.1G	78.75% _{↑17.23%} ± 0.32	48.85% _{↑27.27%} ± 0.21	1×
ViT – Tiny _{10%} DKD [CVPR22] [330]	0.1G	80.72% _{↑15.26%} ± 0.37	62.21% _{↑13.91%} ± 0.30	1×
ViT – Tiny _{10%} CRD [ICLR20] [455]	0.1G	91.94% _{↑4.04%} ± 0.35	70.42% _{↑5.70%} ± 0.33	1×
ViT – Tiny _{10%} ADKD [ACL23] [319]	0.1G	84.29% _{↑11.69%} ± 0.42	66.43% _{↑9.69%} ± 0.37	1×
ViT – Tiny _{10%} TAKD [AAAI20] [26]	0.1G	81.57% _{↑14.41%} ± 0.67	62.89% _{↑13.23%} ± 0.54	20×
ViT – Tiny _{10%} DGKD [ICCV21] [146]	0.1G	88.13% _{↑7.85%} ± 0.60	69.38% _{↑6.74%} ± 0.52	84×
ViT – Tiny _{10%} MMD	0.1G	<u>93.76%</u> _{↑2.22%} ± 0.53	73.12% _{↑3.00%} ± 0.47	20×
ViT – Tiny _{10%} AMD	0.1G	93.83% _{↑2.15%} ± 0.32	<u>73.10%</u> _{↑3.02%} ± 0.30	2.2×
ViT – Small _{100%} (teacher)	4.6G	97.69%	87.82%	-
ViT – Small _{10%} KD [arXiv15] [143]	0.5G	79.21% _{↑18.48%} ± 0.35	57.38% _{↑30.44%} ± 0.33	1×
ViT – Small _{10%} DKD [CVPR22] [330]	0.5G	85.17% _{↑12.52%} ± 0.37	64.42% _{↑23.40%} ± 0.34	1×
ViT – Small _{10%} CRD [ICLR20] [455]	0.5G	93.12% _{↑4.57%} ± 0.36	77.36% _{↑10.46%} ± 0.34	1×
ViT – Small _{10%} ADKD [ACL23] [319]	0.5G	88.71% _{↑9.56%} ± 0.42	72.84% _{↑14.98%} ± 0.39	1×
ViT – Small _{10%} TAKD [AAAI20] [26]	0.5G	86.10% _{↑11.59%} ± 0.69	64.98% _{↑22.84%} ± 0.63	20×
ViT – Small _{10%} DGKD [ICCV21] [146]	0.5G	91.85% _{↑5.84%} ± 0.56	76.57% _{↑11.25%} ± 0.48	84×
ViT – Small _{10%} MMD	0.5G	95.14% _{↑2.56%} ± 0.57	79.22% _{↑8.60%} ± 0.52	20×
ViT – Small _{10%} AMD	0.5G	<u>95.12%</u> _{↑2.57%} ± 0.34	<u>79.17%</u> _{↑8.65%} ± 0.33	2.2×
ViT – Base _{100%} (teacher)	17.6G	98.01%	89.33%	-
ViT – Base _{10%} KD [arXiv15] [143]	1.8G	79.63% _{↑18.38%} ± 0.38	59.47% _{↑29.86%} ± 0.37	1×
ViT – Base _{10%} DKD [CVPR22] [330]	1.8G	85.71% _{↑12.30%} ± 0.39	69.53% _{↑19.80%} ± 0.36	1×
ViT – Base _{10%} CRD [ICLR20] [455]	1.8G	94.18% _{↑3.83%} ± 0.41	78.29% _{↑11.04%} ± 0.39	1×
ViT – Base _{10%} ADKD [ACL23] [319]	1.8G	88.45% _{↑9.56%} ± 0.35	73.69% _{↑15.64%} ± 0.38	1×
ViT – Base _{10%} TAKD [AAAI20] [26]	1.8G	87.43% _{↑10.58%} ± 0.68	70.89% _{↑18.44%} ± 0.64	20×
ViT – Base _{10%} DGKD [ICCV21] [146]	1.8G	92.78% _{↑5.23%} ± 0.55	78.33% _{↑11.00%} ± 0.53	84×
ViT – Base _{10%} MMD	1.8G	95.54% _{↑2.47%} ± 0.54	<u>80.11%</u> _{↑9.22%} ± 0.53	20×
ViT – Base _{10%} AMD	1.8G	<u>95.52%</u> _{↑2.49%} ± 0.38	80.19% _{↑9.14%} ± 0.37	2.2×
Swin – Base _{100%} (teacher)	15.1G	98.80%	92.68%	-
Swin – Base _{10%} KD [arXiv15] [143]	1.5G	80.29% _{↑18.51%} ± 0.37	61.05% _{↑31.63%} ± 0.37	1×
Swin – Base _{10%} DKD [CVPR22] [330]	1.5G	86.16% _{↑12.64%} ± 0.41	72.13% _{↑20.55%} ± 0.32	1×
Swin – Base _{10%} CRD [ICLR20] [455]	1.5G	94.23% _{↑4.57%} ± 0.39	79.71% _{↑12.97%} ± 0.32	1×
Swin – Base _{10%} ADKD [ACL23] [319]	1.5G	89.58% _{↑9.22%} ± 0.44	76.64% _{↑16.04%} ± 0.37	1×
Swin – Base _{10%} TAKD [AAAI20] [26]	1.5G	86.72% _{↑12.08%} ± 0.59	73.86% _{↑18.82%} ± 0.61	20×
Swin – Base _{10%} DGKD [ICCV21] [146]	1.5G	93.54% _{↑5.26%} ± 0.58	79.15% _{↑13.53%} ± 0.46	84×
Swin – Base _{10%} MMD	1.5G	96.07% _{↑2.73%} ± 0.50	83.61% _{↑9.07%} ± 0.47	20×
Swin – Base _{10%} AMD	1.5G	<u>96.02%</u> _{↑2.78%} ± 0.31	<u>83.55%</u> _{↑9.13%} ± 0.28	2.2×

4.5.1 Main Results

Results on CIFAR-10 and CIFAR-100. Table 4.24 presents the **top-1** accuracy scores on CIFAR-10 and CIFAR-100 datasets across five runs. Several key findings can be observed. **First**, AMD surpasses existing knowledge distillation methods with a significant performance gap on both CIFAR-10 and CIFAR-100 datasets. For instance, our model achieves a **1.34-15.89%** improvement in **top-1** accuracy on ViT-Base knowledge distillation on CIFAR-10. **Second**, the suboptimal performance of logit-based methods (*e.g.*, KD, DKD, and TAKD) across various scales and datasets suggests that these approaches may not be the best solution for knowledge distillation in transformer-based architectures. In contrast, methods utilizing feature-space supervision, such as CRD, ADKD, and our proposed AMD, demonstrate commendable performance. This indicates a need for feature-mimicking studies in knowledge distillation, particularly as transformer architectures become deeper and wider, to maintain the integrity and effectiveness of the distilled models. **Third**, AMD achieves a significantly faster training speed compared to multi-step methods, highlighting the effectiveness of structural pruning and joint optimization. **Finally**, our approach consistently outperforms other knowledge distillation methods across various transformer-based architectural designs, suggesting a promising direction for future model deployment.

Results on ImageNet. To further demonstrate the effectiveness and generalization of our approach, we follow established practices [26, 146, 456] and extend AMD to the ImageNet dataset [40] in Table 4.25. As shown in Table 4.24, logit-based methods result in inferior performance on transformer-based architectures for both CIFAR-10 and CIFAR-100 datasets. Consequently, we only compare our method against the feature-mimicking method CRD in this experiment on ViT and Swin architectures. Our method achieves over **2.61%** higher **top-1** accuracy than CRD on ViT-Base.

Table 4.25: Comparison results on ImageNet [40] dataset.

Method	ImageNet [40] top-1
ViT – Base _{100%} (teacher)	77.90%
ViT – Base _{10%} CRD [ICLR20] [455]	69.66% \uparrow 8.24% \pm 0.31
ViT – Base _{10%} ADKD [ACL23] [319]	68.40% \uparrow 9.50% \pm 0.39
ViT – Base _{10%} DGKD [ICCV21] [146]	69.25% \uparrow 8.65% \pm 0.44
ViT – Base _{10%} MMD	72.43% \uparrow 5.47% \pm 0.27
ViT – Base _{10%} AMD	72.27% \uparrow 5.63% \pm 0.22
Swin – Base _{100%} (teacher)	83.50%
Swin – Base _{10%} CRD [ICLR20] [455]	74.83% \uparrow 8.67% \pm 0.28
Swin – Base _{10%} ADKD [ACL23] [319]	72.38% \uparrow 11.12% \pm 0.32
Swin – Base _{10%} DGKD [ICCV21] [146]	74.39% \uparrow 9.11% \pm 0.37
Swin – Base _{10%} MMD	76.86% \uparrow 6.64% \pm 0.24
Swin – Base _{10%} AMD	76.81% \uparrow 6.69% \pm 0.21

4.5.2 Diagnostic Experiments

We conducted a series of ablation studies to assess the robustness and effectiveness of our proposed AMD approach. The majority of experiments were performed using ViT

Table 4.26: **Impact of student model scalability** ranging from 5% to 20%. In our study, we deliberately exclude models of a higher scale, taking into full account the constraints imposed by computational capacities.

Student scalability	Method	FLOPs	Performance	GPU hours
ViT – Base _{5%}	ADKD	0.88G	68.44%	1×
	AMD		75.88%	2.2×
ViT – Base _{10%}	ADKD	1.76G	73.69%	1×
	AMD		80.19%	2.2×
ViT – Base _{15%}	ADKD	2.64G	75.17%	1×
	AMD		80.23 %	2.2×
ViT – Base _{20%}	ADKD	3.52G	77.09%	1×
	AMD		81.09%	2.2×

trained on CIFAR-100. Additionally, to evaluate the adequacy of employing a single teacher-assistant, we present supplementary results with ViT trained on CIFAR-10 for comprehensive analysis.

Impact of Student Model Scalability. To assess whether AMD can enhance performance across different student model scales, we report the results for AMD at scales ranging from 5% to 20% of ViT-Base in Table 4.26. Specifically, the 20% ViT-Base model, with a computational demand of 3.52G FLOPs, closely matches the performance of a full-sized ResNet34 model, which requires 3.68G FLOPs and achieves a 77.94% **top-1** accuracy on CIFAR-100 [457]. We deliberately avoid expanding the student model to a larger scale, as the main goal of this paper is to maintain a compact student model. Enlarging the student model would significantly increase inference costs, posing challenges for deployment flexibility and requiring more computational resources. The results demonstrate that AMD consistently delivers strong performance across various model scales.

Sufficiency of a Single Teacher-Assistant. To evaluate the efficiency of using a single teacher-assistant, we conducted an ablation study involving multiple teacher-assistants in our method. Figure 4.13(a) illustrates that omitting the teacher-assistant leads to a significant drop in performance (*i.e.*, 79.77%), consistent with the findings in Table 4.24. Additionally, we observed that the performance improvement is minimal when more than one teacher-assistant is used (*i.e.*, 80.19% → 80.24%), despite a substantial increase in GPU hours (*i.e.*, tripled and quadrupled for two- and three-step teacher-assistants, respectively). The results from training on the ViT-Base CIFAR-10 dataset corroborate this observation (see Figure 4.13(b)). Consequently, we opted for a single teacher-assistant in AMD, balancing training efficiency and performance adequacy.

Benefit on Training Schedule. Formally, we follow common practices and set 160 epochs as standard training schedule. However, we notice that AMD enjoys a fast convergence in early epochs. For example, in Figure 4.14, with only 40 epochs, our AMD exhibits competitive performance (*i.e.*, 70.82% in CIFAR-100) in comparison to other

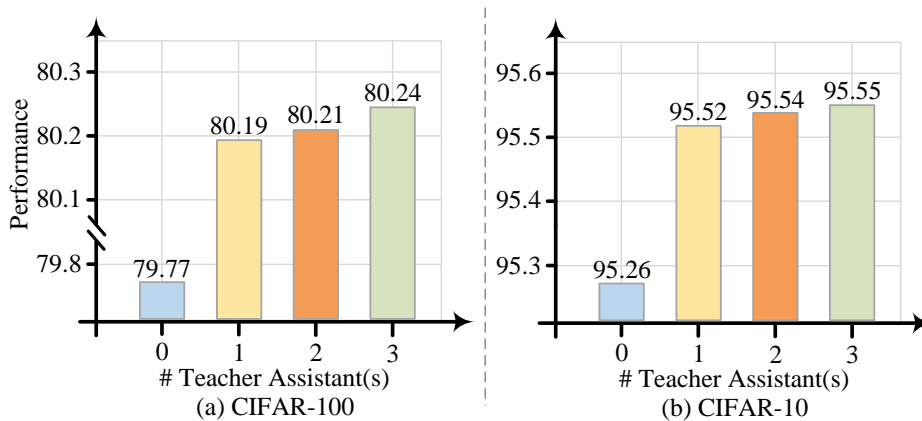


Figure 4.13: **The sufficiency of using one teacher-assistant.** ■ ■ ■ ■ colors represent the performance of AMD_{Min} using zero, one, two, and three teacher-assistants, respectively. The results from the training on the ViT-Base CIFAR-100 is posited at (a), CIFAR-10 at (b).

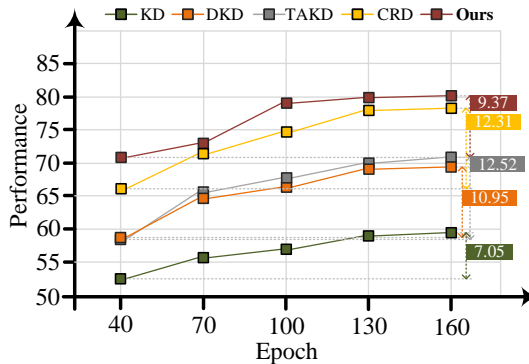


Figure 4.14: **Efficient training schedule.** Our proposed AMD enjoys superior performance among the overall training scene.

prevalent methods (see Table 4.24). Note that though KD [143] exhibits narrow performance gap between epochs (*i.e.*, 7.05%), it can hardly reach a satisfying result under the setting of transformer-based architectures. AMD, on the other hand, performs consistently well even with a reduced number of epochs. This underscores the potential of our approach as an efficacious strategy for training in limited training schedule.

Impact of Candidate Sampling Rate. We further study the variation of candidate sampling rate by changing the number of sampled candidates $m \in \{1, 3, 6, 9, 15\}$. A higher value of m signifies a more refined granularity in the sampling rate. This increased granularity is directly correlated with an extended duration of training time. The GPU hours and their corresponding student performance are reported in Table 4.27. We set

Table 4.27: Impact of candidate scaling m .

Method	Performance	GPU hours
ViT – Base _{100%} (teacher)	89.33%	-
– MMD	80.11%	20×
– AMD ($m = 1$)	78.39%	2×
– AMD ($m = 3$)	79.46%	2×
– AMD ($m = 6$)	79.84%	2.1×
– AMD ($m = 9$)	80.19%	2.2×
– AMD ($m = 15$)	80.22%	2.6×

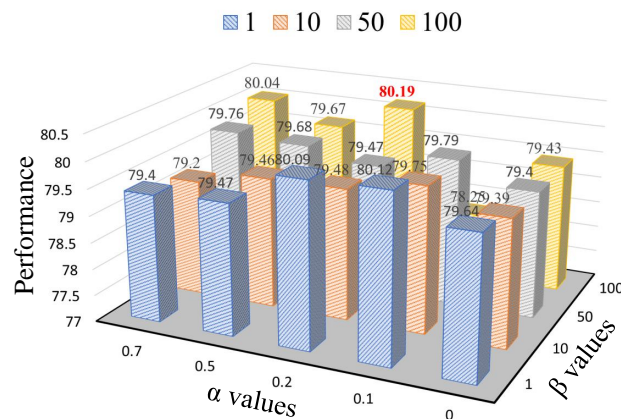


Figure 4.15: **Weight values from different loss objectives.** We present the performance on different values of α and β from Eq. 3.29. The highest performance is marked in red (*i.e.*, $\alpha = 0.2, \beta = 100$). β colors represent performance with respect to different $\beta \in \{1, 10, 50, 100\}$.

$m = 9$ for a satisfying tradeoff between performance and computational overhead. An increased sampling rate invariably leads to a longer training time, which yields marginal enhancements in performance. For example, when having $m = 15$, we observe 0.03% performance gain can be achieved with 18% GPU hour increment. We argue that this is inefficient for training schedule.

Impact of different Hyper-Parameters. In Eq. 3.29, two hyper-parameters, α and β , are introduced to balance the cross-entropy loss, logit-based loss, and feature-mimicking based loss. Figure 4.15 illustrates the impact of different values of α and β on performance. The results demonstrate substantial robustness to variations in these hyper-parameters, as evidenced by a low standard deviation of 0.42. During the experiments, a consistent pattern emerge across various datasets: a larger α correlates with a reduced influence over the teacher-assistant’s logit responses, leading to suboptimal performance. Additionally,

increasing the value of β progressively enhances supervision efficacy. Unlike previous studies in NLP, which suggest varying β values depending on the task [319], β remained relatively stable across several datasets in our experiments. Overall, a hyper-parameter pairing of $\alpha = 0.2$ and $\beta = 100$ is attained for the best performance, which are consistently applied across all experiments.

Impact of Different Loss Components. To examine the impact of different loss components, we conducted ablation studies on three variants of AMD: ❶ AMD without the cross-entropy loss \mathcal{L}_{ce} , ❷ AMD without the logit-based loss \mathcal{L}_{logit} , and ❸ AMD without the feature-mimicking loss \mathcal{L}_{feat} . As shown in Table 4.28, removing the supervision on hidden states (*i.e.*, \mathcal{L}_{feat}) result in a significant performance drop (80.19% \rightarrow 75.24%). This finding aligns with our results in §4.5.1, indicating that performance is suboptimal when relying solely on logit-based methods. Similarly, removing \mathcal{L}_{ce} and \mathcal{L}_{logit} lead to noticeable performance decreases (*i.e.*, 80.19% \rightarrow 78.32% and 80.19% \rightarrow 78.01%, respectively), highlighting the essential roles these losses play in improving model effectiveness. It is worth noting that eliminating \mathcal{L}_{logit} has a more pronounced impact on performance, consistent with our previous ablation study findings (*i.e.*, $\alpha = 1$).

For completeness, we also conduct experiments combining the DKD loss [330], which introduces target class knowledge distillation (TCKD) and non-target class knowledge distillation (NCKD) to further decompose \mathcal{L}_{ce} . Specifically, the combined loss is defined as $\mathcal{L}_{dkd} = \zeta\text{TCKD} + \eta\text{NCKD}$, with balancing parameters ζ and η . Our results indicate that the DKD loss marginally improves model performance (*i.e.*, from 80.19% to 80.22%). However, it is crucial to note that incorporating the DKD loss introduces additional hyper-parameters, leading to increased fluctuations in the pursuit of optimal results. Therefore, to maintain stability in the model’s performance, we adhere to the original design outlined in Eq. 3.29.

Effectiveness of NSPD metric. To demonstrate the effectiveness of the NSPD metric, we also incorporate the combination of the negative derivatives between teacher and teacher-assistant, and teacher-assistant and student, *i.e.*, $-(\frac{P_t - P_{ta}}{S_t - S_{ta}} + \lambda \cdot \frac{P_{ta} - P_s}{S_{ta} - S_s})$, defined as a general form of NSPD metric (*i.e.*, λ -NSPD). We report the results with varying λ values in Table 4.29. The results indicate that λ -NSPD achieves comparable performance to the original NSPD (we have tested other λ values as well). However, incorporating λ introduces additional complexity and necessitates an extra

Table 4.28: **Impact of different loss components**, including three variants from original training objectives (see Eq. 3.29)

Method	Performance
AMD	80.19%
– w/o \mathcal{L}_{CE}	78.32%
– w/o \mathcal{L}_{logit}	78.01%
– w/o \mathcal{L}_{feat}	75.24%
– w/ \mathcal{L}_{dkd}	80.22%

with our previous ablation

Table 4.29: **Discussion on NSPD metric.** We further introduce a general form of NSPD metric as λ -NSPD.

Metric	CIFAR-100 top-1
NSPD ($\lambda = 0$)	80.19%
$\lambda = 0.5$	80.19%
$\lambda = 1$	80.14%

distillation step to determine the student's performance. Therefore, we opt to use the original NSPD as defined in Eq. [3.28](#).

Chapter 5

Research Plan

In this chapter, I will discuss the experimental setup (*i.e.*, datasets (§5.1) and evaluation metrics (§5.2)), and research timeline (§5.3) based on the three tasks I am focusing on. It should be noted that the evaluated datasets and evaluation metrics will undergo replication. I will provide a singular comprehensive overview, and subsequent discussions will reference the established formal introductions.

5.1 Datasets

5.1.1 Universal Visual Learner

We evaluate our methods over two vision tasks *viz.* optical flow and depth estimation.

For **optical flow**, following previous research [10, 28], we first train our approach on FlyingChair [458] and FlyingThings [459], and then fine-tune it on a large combination of datasets (C+T+S+K+H) to allow evaluation on the Sintel and KITTI-2015 benchmarks.

For **depth estimation**, we evaluate ProtoFormer on both real and synthetic datasets, utilizing KITTI [460] and MPI Sintel [461]. As an autonomous driving dataset consisting of 61 outdoor scenes of various modalities, we use the KITTI Eigen depth split, which contains a standard depth estimation split proposed by Eigen et al. [35] consisting of 32 scenes for training and 29 scenes for testing. MPI Sintel is a synthetic dataset featuring long stereo sequences with significant motion and depth variations, containing a total of 35 rendered sequences.

5.1.2 Interpretable Visual Intelligence

Two sub-tasks are included in the discussion of generalizable, interpretable visual intelligence (*i.e.*, see **DNC** and **DVP**).

DNC. The evaluation for image classification is carried out on CIFAR-10 [39], CIFAR-100 [39] and ImageNet [40] datasets, respectively. CIFAR-10 contains 60K (50K/10K

for `train/test`) 32×32 colored images of 10 classes. CIFAR-100 dataset contains 100 classes with 500 training and 100 testing images per class. ImageNet-1K [40] includes high-resolution images spanning distinct categories (*e.g.*, animals, plants, and vehicles). Following conventional procedures, the dataset is split into 1.2M/50K/100K images for `train/ validation/test` splits.

DVP. For quantitative and qualitative results, we conduct a new benchmark, consisting of 100 diverse text-image pairs. Specifically, we manually pick images from web, generated images, ImageNet-R [462], ImageNet [40], MS COCO [463], and other previous work [104, 256]. To ensure a fair comparison and analysis, we construct a list of text templates including 20 different classes for translating, each containing five images of high-resolution and quality. For each originating class (*e.g.*, building), we stipulate corresponding target categories and stylistic attributes, thereby facilitating the automated sampling of various permutations and combinations for translating and evaluation.

5.1.3 Carbon-efficient Visual Intelligence System

E²VPT. Our experiments are carried out on two image classification benchmarks. **VTAB-1k** [25] collects 19 benchmarked Visual Task Adaptation, categorized into three groups: (1) *Natural* contains natural images captured by standard cameras, (2) *Specialized* includes images taken by specialized equipment, and (3) *Structured* covers tasks requiring geometric comprehension (*i.e.*, counting, distance). Each task of VTAB-1k contains 1000 training examples. Following [23, 25], we apply the 800-200 split for training set on hyperparameter tuning. The final evaluation is run on the full training data. **FGVC** contains 5 benchmarked Fine-Grained Visual Classification, including CUB-200-2011 [31], NABirds [464], Oxford Flowers [32], Stanford Dogs [33] and Stanford Cars [465]. Following [23], the training set is randomly split into 90% `train` and 10% `val`. We use `val` for hyperparameter tuning.

AMD. The evaluation is evaluated on CIFAR-10 [27], CIFAR-100 [27] and ImageNet [466], which are specifically discussed in §5.1.2.

5.2 Evaluation Metrics

5.2.1 Universal Visual Learner

As introduced in §1.1, a universal visual learner enjoys handling several tasks (*i.e.*, optical flow, depth estimation), which in turn covers several evaluation metrics.

For **optical flow**, to ensure a fair comparison, we employ standard metrics: the average end-point-error (F1-epe), which measures the average l_2 distance between the predicted values and the ground truth, and the percentage of outliers over all pixels (F1-all). F1-all quantifies the proportion of errors that exceed 3 pixels or 5% relative to the ground truth in optical flow estimation.

Table 5.1: **User study** on controllability and user-friendliness.

Method	Controllability	User-friendliness
VISPROG	25.1%	39.7%
DVP (ours)	74.9%	60.3%

For **depth estimation**, we adhere to the standard metrics of absolute relative error (Abs Rel), root mean square error (RMSE), and the percentage of inlier pixels with $\delta_1 < \tau$ ($\tau = 1.25$).

5.2.2 Interpretable Visual Intelligence

DNC. DNC covers two fundamental image tasks: image classification, and image semantic segmentation. To evaluate its performances, I introduce accuracy (see §5.2.1) for image classification, and mIOU (see §5.2.1) for image semantic segmentation.

DVP. We follow [102, 104], and calculate the CLIP-Score [467] and DINO-Score [468]. These metrics enable the similarity between the generated image and the target prompt. We further conduct user studies involving 50 participants to evaluate 100 sets of results from multiple viewpoints, such as fidelity, quality, and diversity. Each set comprises a source image along with its corresponding translated image. To facilitate the evaluation, we establish comprehensive guidelines and scoring templates, where scores range from 1 (worst) to 5 (best). In Table 4.17, we design user study including “Quality,” “Fidelity” and “Diversity,” respectively. Specifically, we employ user study on Likert scale [469], a well-established rating scale metric utilized for quantifying opinions, attitudes, or behavioral tendencies [470–473]. This scale typically presents respondents with either a statement or a question. In our case, the questions are measuring the overall harmony of the translated image (*i.e.*, “Quality”), the preservation of the identity in the image (*i.e.*, “Fidelity”), and variations between the original and translated images (*i.e.*, “Diversity”) rated by users from 1 (worst) to 5 (best), Respondents then select the choice that most closely aligns with their own perspective or sentiment regarding the given statement or question. We further provide user study in Table 5.1, focusing on the usability of explainable controllability discussed in §4.3.2. The results demonstrate that our approach facilitates user-friendly error correction and permits the intuitive surveillance of intermediate outputs.

5.2.3 Carbon-efficient Visual Intelligence System

Since I am mostly focusing on the conventional image classification tasks, I thus introduce accuracy as the evaluation matrix (see §5.2.1 and §5.2.2) for both **E²VPT** and **AMD**.

Table 5.2: **Overall timeline** for the Ph.D. research dissertation.

Task	Timeline	2023						2024					
		Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun
Universal Visual Learner	ProtoFormer												
Interpretable Visual Intelligence	DNC												
	DVP												
Carbon-efficient Visual Intelligence	E ² VPT												
	AMD												

5.3 Research Timeline

The plans for my future work are shown in Table 5.2. I am planning to graduate in Summer 2024.

Bibliography

- [1] W. Wang, C. Han, T. Zhou, and D. Liu, “Visual recognition with deep nearest centroids,” in *Int. Conf. Learn. Representations*, 2023.
- [2] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [3] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, “Segformer: Simple and efficient design for semantic segmentation with transformers,” in *Advances Neural Inf. Process. Syst*, 2021.
- [4] Z. Qin, C. Han, Q. Wang, X. Nie, Y. Yin, and L. Xiankai, “Unified 3d segmenter as prototypical classifiers,” in *Advances Neural Inf. Process. Syst*, 2023.
- [5] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, “Object detection with deep learning: A review,” *IEEE Trans. on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, 2019.
- [6] C. Szegedy, A. Toshev, and D. Erhan, “Deep neural networks for object detection,” in *Advances Neural Inf. Process. Syst*, 2013.
- [7] A. Ranjan and M. J. Black, “Optical flow estimation using a spatial pyramid network,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017.
- [8] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, “Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [9] Z. Teed and J. Deng, “Raft: Recurrent all-pairs field transforms for optical flow,” in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [10] Z. Huang, X. Shi, C. Zhang, Q. Wang, K. C. Cheung, H. Qin, J. Dai, and H. Li, “Flowformer: A transformer architecture for optical flow,” in *Proc. Eur. Conf. Comput. Vis.*, 2022.

- [11] X. Shi, Z. Huang, D. Li, M. Zhang, K. C. Cheung, S. See, H. Qin, J. Dai, and H. Li, “Flowformer++: Masked cost volume autoencoding for pretraining optical flow estimation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023.
- [12] Y. Lu, D. Liu, Q. Wang, C. Han, Y. Cui, Z. Cao, X. Zhang, Y. V. Chen, and H. Fan, “Promotion: Prototypes as motion learners,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024.
- [13] S. F. Bhat, I. Alhashim, and P. Wonka, “Adabins: Depth estimation using adaptive bins,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [14] V. Patil, C. Sakaridis, A. Liniger, and L. Van Gool, “P3depth: Monocular depth estimation with a piecewise planarity prior,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [15] D. A. Medler, “A brief history of connectionism,” *Neural Computing Surveys*, vol. 1, pp. 18–72, 1998.
- [16] J. Choi, S. Kim, Y. Jeong, Y. Gwon, and S. Yoon, “Ilvr: Conditioning method for denoising diffusion probabilistic models,” *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [17] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [18] C. Saharia, W. Chan, H. Chang, C. Lee, J. Ho, T. Salimans, D. Fleet, and M. Norouzi, “Palette: Image-to-image diffusion models,” in *ACM Special Interest Group on Computer Graphics and Interactive Techniques*, 2022.
- [19] A. Newell, H. A. Simon *et al.*, *Human problem solving*. Prentice-hall Englewood Cliffs, NJ, 1972, vol. 104, no. 9.
- [20] A. Aamodt and E. Plaza, “Case-based reasoning: Foundational issues, methodological variations, and system approaches,” *AI Communications*, vol. 7, no. 1, pp. 39–59, 1994.
- [21] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?” in *Advances Neural Inf. Process. Syst*, 2014.
- [22] H. Cai, C. Gan, L. Zhu, and S. Han, “Tinytl: Reduce memory, not parameters for efficient on-device learning,” in *Advances Neural Inf. Process. Syst*, 2020.
- [23] M. Jia, L. Tang, B.-C. Chen, C. Cardie, S. Belongie, B. Hariharan, and S.-N. Lim, “Visual prompt tuning,” in *Proc. Eur. Conf. Comput. Vis.*, 2022.

- [24] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” in *Int. Conf. Learn. Representations*, 2020.
- [25] X. Zhai, J. Puigcerver, A. Kolesnikov, P. Ruysen, C. Riquelme, M. Lucic, J. Djonlonga, A. S. Pinto, M. Neumann, A. Dosovitskiy *et al.*, “A large-scale study of representation learning with the visual task adaptation benchmark,” *arXiv preprint arXiv:1910.04867*, 2019.
- [26] S. I. Mirzadeh, M. Farajtabar, A. Li, N. Levine, A. Matsukawa, and H. Ghasemzadeh, “Improved knowledge distillation via teacher assistant,” in *AAAI Conference on Artificial Intelligence*, 2020.
- [27] A. Krizhevsky, G. Hinton *et al.*, “Learning multiple layers of features from tiny images,” 2009.
- [28] Q. Dong, C. Cao, and Y. Fu, “Rethinking optical flow from geometric matching consistent perspective,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023.
- [29] R. Mokady, A. Hertz, K. Aberman, Y. Pritch, and D. Cohen-Or, “Null-text inversion for editing real images using guided diffusion models,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023.
- [30] T. Gupta and A. Kembhavi, “Visual programming: Compositional visual reasoning without training,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023.
- [31] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, “The caltech-ucsd birds-200-2011 dataset,” 2011.
- [32] M.-E. Nilsback and A. Zisserman, “Automated flower classification over a large number of classes,” in *Indian Conference on Computer Vision, Graphics & Image Processing*, 2008.
- [33] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-F. Li, “Novel dataset for fine-grained image categorization: Stanford dogs,” in *CVPR Workshop*, 2011.
- [34] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng *et al.*, “Reading digits in natural images with unsupervised feature learning,” in *NIPS Workshop*, vol. 2011, 2011.
- [35] D. Eigen, C. Puhrsch, and R. Fergus, “Depth map prediction from a single image using a multi-scale deep network,” *Advances Neural Inf. Process. Syst.*, 2014.

- [36] C. Godard, O. Mac Aodha, and G. J. Brostow, “Unsupervised monocular depth estimation with left-right consistency,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017.
- [37] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, “Deep ordinal regression network for monocular depth estimation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [38] W. Yin, Y. Liu, C. Shen, and Y. Yan, “Enforcing geometric constraints of virtual normal for depth prediction,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [39] A. Krizhevsky and G. Hinton, *Learning multiple layers of features from tiny images*. Technical report, University of Toronto, 2009.
- [40] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li, “Imagenet large scale visual recognition challenge,” *Int. J. Comput. Vis.*, 2015.
- [41] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, “Do imagenet classifiers generalize to imagenet?” in *Proc. Int. Conf. Mach. Learn.*, 2019.
- [42] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, “Scene parsing through ade20k dataset,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017.
- [43] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016.
- [44] H. Caesar, J. Uijlings, and V. Ferrari, “Coco-stuff: Thing and stuff classes in context,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [45] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, “3d semantic parsing of large-scale indoor spaces,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016.
- [46] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, “ScanNet: Richly-annotated 3d reconstructions of indoor scenes,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017.
- [47] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, “Semantickitti: A dataset for semantic scene understanding of lidar sequences,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [48] E. Iofinova, A. Peste, M. Kurtz, and D. Alistarh, “How well do sparse imagenet models transfer?” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.

- [49] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, “Swin transformer: Hierarchical vision transformer using shifted windows,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [50] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, “Masked autoencoders are scalable vision learners,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [51] X. Chen, S. Xie, and K. He, “An empirical study of training self-supervised vision transformers,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [52] G. Van Horn, S. Branson, R. Farrell, S. Haber, J. Barry, P. Ipeirotis, P. Perona, and S. Belongie, “Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015.
- [53] J. Kim, H. Lee, and S. S. Woo, “Imf: Integrating matched features using attentive logit in knowledge distillation,” in *International Joint Conferences on Artificial Intelligence*, 2023.
- [54] Plato, *Cratylus*. Plato, 402 BC.
- [55] J. Hawkins and S. Blakeslee, *On intelligence*. Macmillan, 2004.
- [56] M. Xiong, Z. Zhang, W. Zhong, J. Ji, J. Liu, and H. Xiong, “Self-supervised monocular depth and visual odometry learning with scale-consistent geometric constraints,” in *International Joint Conferences on Artificial Intelligence*, 2021.
- [57] W. Zhao, S. Liu, Y. Shu, and Y.-J. Liu, “Towards better generalization: Joint depth-pose learning without posenet,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [58] J. D. Smith and J. P. Minda, “Distinguishing prototype-based and exemplar-based processes in dot-pattern category learning.” *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 2002.
- [59] Z. Jiang, Z. Lin, and L. Davis, “Recognizing human actions by learning and matching shape-motion prototype trees,” *IEEE TPAMI*, 2012.
- [60] C. Zhu, W. Ping, C. Xiao, M. Shoeybi, T. Goldstein, A. Anandkumar, and B. Catanzaro, “Long-short transformer: Efficient transformers for language and vision,” in *Advances Neural Inf. Process. Syst.*, 2021.
- [61] J. Yang, J. Liu, N. Xu, and J. Huang, “Tvt: Transferable vision transformer for unsupervised domain adaptation,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023.

- [62] W. Kim, B. Son, and I. Kim, “Vilt: Vision-and-language transformer without convolution or region supervision,” in *Proc. Int. Conf. Mach. Learn.*, 2021.
- [63] T. Laugel, M.-J. Lesot, C. Marsala, X. Renard, and M. Detyniecki, “The dangers of post-hoc interpretability: Unjustified counterfactual explanations,” in *International Joint Conferences on Artificial Intelligence*, 2019.
- [64] C. Rudin, “Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead,” *Nature Machine Intelligence*, vol. 1, no. 5, pp. 206–215, 2019.
- [65] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins *et al.*, “Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai,” *Information Fusion*, vol. 58, pp. 82–115, 2020.
- [66] C. Rudin, C. Chen, Z. Chen, H. Huang, L. Semenova, and C. Zhong, “Interpretable machine learning: Fundamental principles and 10 grand challenges,” *Statistics Surveys*, vol. 16, pp. 1–85, 2022.
- [67] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *Int. Conf. Learn. Representations*, 2015.
- [68] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016.
- [69] P. Angelov and E. Soares, “Towards explainable deep neural networks (xdnn),” *Neural Networks*, vol. 130, pp. 185–194, 2020.
- [70] O. Li, H. Liu, C. Chen, and C. Rudin, “Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions,” in *AAAI Conference on Artificial Intelligence*, 2018.
- [71] H. Hayashi and S. Uchida, “A discriminative gaussian mixture model with sparsity,” in *Int. Conf. Learn. Representations*, 2021.
- [72] T. Zhou, W. Wang, E. Konukoglu, and L. Van Gool, “Rethinking semantic segmentation: A prototype view,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [73] P. Mettes, E. van der Pol, and C. Snoek, “Hyperspherical prototype networks,” in *Advances Neural Inf. Process. Syst.*, 2019.
- [74] E. Fix and J. L. Hodges Jr, “Discriminatory analysis - nonparametric discrimination: Small sample performance,” California Univ Berkeley, Tech. Rep., 1952.

- [75] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE TIT*, vol. 13, no. 1, pp. 21–27, 1967.
- [76] J. L. Kolodner, "An introduction to case-based reasoning," *Artificial Intelligence Review*, vol. 6, no. 1, pp. 3–34, 1992.
- [77] T. Kohonen, "Learning vector quantization," in *Self-Organizing Maps*, 1995, pp. 175–189.
- [78] B. Chaudhuri, "A new definition of neighborhood of a point in multi-dimensional space," *Pattern Recognition Letters*, vol. 17, no. 1, pp. 11–17, 1996.
- [79] A. R. Webb, *Statistical pattern recognition*. John Wiley & Sons, 2003.
- [80] B. Kim, C. Rudin, and J. A. Shah, "The bayesian case model: A generative approach for case-based reasoning and prototype classification," in *Advances Neural Inf. Process. Syst*, 2014.
- [81] E. H. Rosch, "Natural categories," *Cognitive Psychology*, vol. 4, no. 3, pp. 328–350, 1973.
- [82] S. Slade, "Case-based reasoning: A research paradigm," *AI magazine*, vol. 12, no. 1, pp. 42–42, 1991.
- [83] J. S. Sánchez, F. Pla, and F. J. Ferri, "On the use of neighbourhood-based non-parametric classifiers," *Pattern Recognition Letters*, vol. 18, no. 11-13, pp. 1179–1186, 1997.
- [84] J. Gou, Z. Yi, L. Du, and T. Xiong, "A local mean-based k-nearest centroid neighbor classifier," *The Computer Journal*, vol. 55, no. 9, pp. 1058–1071, 2012.
- [85] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Advances Neural Inf. Process. Syst*, 2013.
- [86] Y. M. Asano, C. Rupprecht, and A. Vedaldi, "Self-labelling via simultaneous clustering and representation learning," in *Int. Conf. Learn. Representations*, 2020.
- [87] M. Biehl, B. Hammer, and T. Villmann, "Prototype-based models in machine learning," *Wiley Interdisciplinary Reviews: Cognitive Science*, vol. 7, no. 2, pp. 92–111, 2016.
- [88] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

- [89] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Proc. Eur. Conf. Comput. Vis.*, 2014.
- [90] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016.
- [91] Z. C. Lipton, “The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery.” *Queue*, vol. 16, no. 3, pp. 31–57, 2018.
- [92] Y. Cui, C. Han, and D. Liu, “Ssga-net: Stepwise spatial global-local aggregation networks for autonomous driving,” *arXiv preprint arXiv:2405.18857*, 2024.
- [93] A. d. Garcez, S. Bader, H. Bowman, L. C. Lamb, L. de Penning, B. Illuminoo, H. Poon, and C. G. Zaverucha, “Neural-symbolic learning and reasoning: A survey and interpretation,” *Neuro-Symbolic Artificial Intelligence: The State of the Art*, 2022.
- [94] P. Dhariwal and A. Nichol, “Diffusion models beat gans on image synthesis,” *Advances Neural Inf. Process. Syst.*, 2021.
- [95] D. Epstein, A. Jabri, B. Poole, A. A. Efros, and A. Holynski, “Diffusion self-guidance for controllable image generation,” *arXiv preprint arXiv:2306.00986*, 2023.
- [96] J. Ho and T. Salimans, “Classifier-free diffusion guidance,” *NeurIPS Workshop*, 2021.
- [97] P. Hitzler, A. Eberhart, M. Ebrahimi, M. K. Sarker, and L. Zhou, “Neuro-symbolic approaches in artificial intelligence,” *National Science Review*, 2022.
- [98] A. Oltramari, J. Francis, C. Henson, K. Ma, and R. Wickramarachchi, “Neuro-symbolic architectures for context understanding,” *arXiv preprint arXiv:2003.04707*, 2020.
- [99] J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu, “The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision,” *Int. Conf. Learn. Representations*, 2019.
- [100] J. Song, C. Meng, and S. Ermon, “Denoising diffusion implicit models,” *Int. Conf. Learn. Representations*, 2021.
- [101] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” *NeurIPS*, 2020.

- [102] X. Chen, L. Huang, Y. Liu, Y. Shen, D. Zhao, and H. Zhao, “Anydoor: Zero-shot object-level image customization,” *arXiv preprint arXiv:2307.09481*, 2023.
- [103] G. Batzolis, J. Stanczuk, C.-B. Schönlieb, and C. Etmann, “Conditional image generation with score-based diffusion models,” *arXiv preprint arXiv:2111.13606*, 2021.
- [104] N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman, “Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023.
- [105] R. Gal, Y. Alaluf, Y. Atzmon, O. Patashnik, A. H. Bermano, G. Chechik, and D. Cohen-Or, “An image is worth one word: Personalizing text-to-image generation using textual inversion,” *arXiv preprint arXiv:2208.01618*, 2022.
- [106] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-Or, “Prompt-to-prompt image editing with cross attention control,” *ICLR*, 2023.
- [107] R. Nishant, M. Kennedy, and J. Corbett, “Artificial intelligence for sustainability: Challenges, opportunities, and a research agenda,” *International Journal of Information Management*, 2020.
- [108] A. Van Wynsberghe, “Sustainable ai: Ai for sustainability and the sustainability of ai,” *AI and Ethics*, 2021.
- [109] R. Vinuesa, H. Azizpour, I. Leite, M. Balaam, V. Dignum, S. Domisch, A. Felländer, S. D. Langhans, M. Tegmark, and F. Fuso Nerini, “The role of artificial intelligence in achieving the sustainable development goals,” *Nature Communications*, 2020.
- [110] C.-J. Wu, R. Raghavendra, U. Gupta, B. Acun, N. Ardalani, K. Maeng, G. Chang, F. Aga, J. Huang, C. Bai *et al.*, “Sustainable ai: Environmental implications, challenges and opportunities,” *Proceedings of Machine Learning and Systems*, 2022.
- [111] M. Innes, A. Edelman, K. Fischer, C. Rackauckas, E. Saba, V. B. Shah, and W. Tebbutt, “A differentiable programming system to bridge machine learning and scientific computing,” *arXiv preprint arXiv:1907.07587*, 2019.
- [112] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, “Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter,” *arXiv preprint arXiv:1910.01108*, 2019.
- [113] Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh, “Large batch optimization for deep learning: Training bert in 76 minutes,” in *Int. Conf. Learn. Representations*, 2020.
- [114] D. Guo, A. M. Rush, and Y. Kim, “Parameter-efficient transfer learning with diff pruning,” in *Proc. Int. Conf. Mach. Learn.*, 2021.

- [115] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang, “Convolutional neural networks for medical image analysis: Full training or fine tuning?” *IEEE Transactions on Medical Imaging*, 2016.
- [116] L. Kaiser, A. N. Gomez, N. Shazeer, A. Vaswani, N. Parmar, L. Jones, and J. Uszkoreit, “One model to learn them all,” in *Proc. Int. Conf. Mach. Learn.*, 2017.
- [117] J. Lin, Y. Liu, Q. Zeng, M. Jiang, and J. Cleland-Huang, “Traceability transformed: Generating more accurate links with pre-trained bert models,” in *Int. Conf. on Software Engineering*, 2021.
- [118] M. Jia, Z. Wu, A. Reiter, C. Cardie, S. Belongie, and S.-N. Lim, “Exploring visual engagement signals for representation learning,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [119] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. Van Der Maaten, “Exploring the limits of weakly supervised pretraining,” in *Proc. Eur. Conf. Comput. Vis.*, 2018.
- [120] L. Yan, C. Han, Z. Xu, D. Liu, and Q. Wang, “Prompt learns prompt: Exploring knowledge-aware generative prompt collaboration for video captioning.” in *International Joint Conferences on Artificial Intelligence*, 2023.
- [121] Q. Wang, Y. Mao, J. Wang, H. Yu, S. Nie, S. Wang, F. Feng, L. Huang, X. Quan, Z. Xu *et al.*, “Aprompt: Attention prompt tuning for efficient adaptation of pre-trained language models,” in *Empirical Methods in Natural Language Process.*, 2023.
- [122] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, “Learning multiple visual domains with residual adapters,” in *Advances Neural Inf. Process. Syst.*, 2017.
- [123] J. O. Zhang, A. Sax, A. Zamir, L. Guibas, and J. Malik, “Side-tuning: a baseline for network adaptation via additive side networks,” in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [124] C. Ju, T. Han, K. Zheng, Y. Zhang, and W. Xie, “Prompting visual-language models for efficient video understanding,” in *Proc. Eur. Conf. Comput. Vis.*, 2022.
- [125] Y. Zang, W. Li, K. Zhou, C. Huang, and C. C. Loy, “Unified vision and language prompt learning,” *arXiv preprint arXiv:2210.07225*, 2022.
- [126] C. Han, Q. Wang, Y. Cui, W. Wang, L. Huang, S. Qi, and D. Liu, “Facing the elephant in the room: Visual prompt tuning or full finetuning?” *arXiv preprint arXiv:2401.12902*, 2024.

- [127] L. Yang, Q. Wang, J. Wang, X. Quan, F. Feng, Y. Chen, M. Khabsa, S. Wang, Z. Xu, and D. Liu, "Mixpave: Mix-prompt tuning for few-shot product attribute value extraction," in *Annual Meeting of the Association for Computational Linguistics*, 2023.
- [128] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu *et al.*, "A survey on vision transformer," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2022.
- [129] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *ACM Computing Surveys*, vol. 54, no. 10s, pp. 1–41, 2022.
- [130] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," *AI Open*, 2022.
- [131] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," in *Int. Conf. Learn. Representations*, 2019.
- [132] B. Zhuang, J. Liu, Z. Pan, H. He, Y. Weng, and C. Shen, "A survey on efficient training of transformers," *arXiv preprint arXiv:2302.01107*, 2023.
- [133] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," *Advances Neural Inf. Process. Syst.*, 2015.
- [134] B. Hassibi and D. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," in *Advances Neural Inf. Process. Syst.*, 1992.
- [135] Y. LeCun, J. Denker, and S. Solla, "Optimal brain damage," in *Advances Neural Inf. Process. Syst.*, 1989.
- [136] C. Li, B. Zhuang, G. Wang, X. Liang, X. Chang, and Y. Yang, "Automated progressive learning for efficient training of vision transformers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [137] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," *arXiv preprint arXiv:1608.08710*, 2016.
- [138] A. Kolesnikov, L. Beyer, X. Zhai, J. Puigcerver, J. Yung, S. Gelly, and N. Houlsby, "Big transfer (bit): General visual representation learning," in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [139] L. Yuan, D. Chen, Y.-L. Chen, N. Codella, X. Dai, J. Gao, H. Hu, X. Huang, B. Li, C. Li *et al.*, "Florence: A new foundation model for computer vision," *arXiv preprint arXiv:2111.11432*, 2021.

- [140] C. Han, Q. Wang, Y. Cui, Z. Cao, W. Wang, S. Qi, and D. Liu, “E²vpt: An effective and efficient approach for visual prompt tuning,” *arXiv preprint arXiv:2307.13770*, 2023.
- [141] Y. Li, G. Yuan, Y. Wen, J. Hu, G. Evangelidis, S. Tulyakov, Y. Wang, and J. Ren, “Efficientformer: Vision transformers at mobilenet speed,” *Advances Neural Inf. Process. Syst.*, 2022.
- [142] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *arXiv preprint arXiv:1704.04861*, 2017.
- [143] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” *arXiv preprint arXiv:1503.02531*, 2015.
- [144] Y. Wu, M. Rezagholizadeh, A. Ghaddar, M. A. Haidar, and A. Ghodsi, “Universalkd: Attention-based output-grounded intermediate layer knowledge distillation,” in *EMNLP*, 2021.
- [145] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, “Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers,” in *Advances Neural Inf. Process. Syst.*, 2020.
- [146] W. Son, J. Na, J. Choi, and W. Hwang, “Densely guided knowledge distillation using multiple teacher assistants,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [147] S. Shen, L. Kerofsky, and S. Yogamani, “Optical flow for autonomous driving: Applications, challenges and improvements,” *arXiv preprint arXiv:2301.04422*, 2023.
- [148] A. B. Khalifa, I. Alouani, M. A. Mahjoub, and N. E. B. Amara, “Pedestrian detection using a moving camera: A novel framework for foreground detection,” *Cognitive Systems Research*, 2020.
- [149] A. Marathe, R. Walambe, and K. Kotecha, “Evaluating the performance of ensemble methods and voting strategies for dense 2d pedestrian detection in the wild,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [150] H. Xu, M. Guo, N. Nedjah, J. Zhang, and P. Li, “Vehicle and pedestrian detection algorithm based on lightweight yolov3-promote and semi-precision acceleration,” *IEEE Transactions on Intelligent Transportation Systems*, 2022.
- [151] J. Li, Q. Huang, Y. Du, X. Zhen, S. Chen, and L. Shao, “Variational abnormal behavior detection with motion consistency,” *IEEE Trans. Image Process.*, 2021.

- [152] R. Tudor Ionescu, S. Smeureanu, B. Alexe, and M. Popescu, “Unmasking the abnormal events in video,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017.
- [153] J. T. Zhou, J. Du, H. Zhu, X. Peng, Y. Liu, and R. S. M. Goh, “AnomalyNet: An anomaly detection network for video surveillance,” *IEEE Transactions on Information Forensics and Security*, 2019.
- [154] H. Gao, J. Cui, M. Ye, S. Li, Y. Zhao, and X. Zhu, “Structure-preserving motion estimation for learned video compression,” in *ACM MultiMedia*, 2022.
- [155] Z. Hu, G. Lu, and D. Xu, “Fvc: A new framework towards deep video compression in feature space,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [156] G. Lu, W. Ouyang, D. Xu, X. Zhang, C. Cai, and Z. Gao, “Dvc: An end-to-end deep video compression framework,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019.
- [157] X. Sui, S. Li, X. Geng, Y. Wu, X. Xu, Y. Liu, R. Goh, and H. Zhu, “Craft: Cross-attentional flow transformer for robust optical flow,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [158] M. Lee, S. Cho, S. Lee, C. Park, and S. Lee, “Unsupervised video object segmentation via prototype memory network,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023.
- [159] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.
- [160] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [161] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009, vol. 2.
- [162] N. Dong and E. P. Xing, “Few-shot semantic segmentation with prototype learning,” in *Proceedings of the British Machine Vision Conference*, 2018.
- [163] K. Wang, J. H. Liew, Y. Zou, D. Zhou, and J. Feng, “Panet: Few-shot image semantic segmentation with prototype alignment,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [164] Y. Liu, X. Zhang, S. Zhang, and X. He, “Part-aware prototype network for few-shot semantic segmentation,” in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [165] B. Yang, C. Liu, B. Li, J. Jiao, and Q. Ye, “Prototype mixture models for few-shot semantic segmentation,” in *Proc. Eur. Conf. Comput. Vis.*, 2020.

- [166] G. Li, V. Jampani, L. Sevilla-Lara, D. Sun, J. Kim, and J. Kim, “Adaptive prototype learning and allocation for few-shot segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [167] S. Jetley, B. Romera-Paredes, S. Jayasumana, and P. Torr, “Prototypical priors: From improving classification to zero-shot learning,” *arXiv preprint arXiv:1512.01192*, 2015.
- [168] W. Xu, Y. Xian, J. Wang, B. Schiele, and Z. Akata, “Attribute prototype network for zero-shot learning,” in *Advances Neural Inf. Process. Syst.*, 2020.
- [169] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
- [170] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “Roberta: A robustly optimized bert pretraining approach,” in *Int. Conf. Learn. Representations*, 2020.
- [171] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *The Journal of Machine Learning Research*, 2020.
- [172] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances Neural Inf. Process. Syst.*, 2017.
- [173] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong *et al.*, “Swin transformer v2: Scaling up capacity and resolution,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [174] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, “Segmenter: Transformer for semantic segmentation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [175] H. Wang, Y. Zhu, H. Adam, A. Yuille, and L.-C. Chen, “Max-deeplab: End-to-end panoptic segmentation with mask transformers,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [176] Y. Wang, Z. Xu, X. Wang, C. Shen, B. Cheng, H. Shen, and H. Xia, “End-to-end video instance segmentation with transformers,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [177] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr *et al.*, “Rethinking semantic segmentation from a sequence-to-sequence

- perspective with transformers,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [178] Y. Cai, J. Lin, X. Hu, H. Wang, X. Yuan, Y. Zhang, R. Timofte, and L. Van Gool, “Coarse-to-fine sparse transformer for hyperspectral image reconstruction,” in *Proc. Eur. Conf. Comput. Vis.*, 2022.
- [179] J. H. Friedman, *The elements of statistical learning: Data mining, inference, and prediction*. Springer, 2009.
- [180] D. J. Hand and K. Yu, “Idiot’s bayes—not so stupid after all?” *International Statistical Review*, vol. 69, no. 3, pp. 385–398, 2001.
- [181] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [182] O. Boiman, E. Shechtman, and M. Irani, “In defense of nearest-neighbor based image classification,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008.
- [183] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid, “Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009.
- [184] R. P. Lippmann, “Pattern classification using neural networks,” *IEEE Communications Magazine*, vol. 27, no. 11, pp. 47–50, 1989.
- [185] R. Salakhutdinov and G. Hinton, “Learning a nonlinear embedding by preserving class neighbourhood structure,” in *Artificial Intelligence and Statistics*, 2007, pp. 412–419.
- [186] N. Papernot and P. McDaniel, “Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning,” *arXiv preprint arXiv:1803.04765*, 2018.
- [187] Z. Wu, A. A. Efros, and S. X. Yu, “Improving generalization via scalable neighborhood component analysis,” in *Proc. Eur. Conf. Comput. Vis.*, 2018.
- [188] S. Garcia, J. Derrac, J. Cano, and F. Herrera, “Prototype selection for nearest neighbor classification: Taxonomy and empirical study,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 3, pp. 417–435, 2012.
- [189] H.-M. Yang, X.-Y. Zhang, F. Yin, and C.-L. Liu, “Robust classification with convolutional prototype learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [190] S. Guerriero, B. Caputo, and T. Mensink, “Deepncm: Deep nearest class mean classifiers,” in *Int. Conf. Learn. Representations workshop*, 2018.

- [191] J. Snell, K. Swersky, and R. S. Zemel, “Prototypical networks for few-shot learning,” in *Advances Neural Inf. Process. Syst.*, 2017.
- [192] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, “icarl: Incremental classifier and representation learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017.
- [193] A. Quattoni, M. Collins, and T. Darrell, “Transfer learning for image classification with sparse prototype representations,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008.
- [194] M. Biehl, B. Hammer, P. Schneider, and T. Villmann, “Metric learning for prototype-based classification,” in *Innovations in Neural Information Paradigms and Applications*, 2009, pp. 183–199.
- [195] D. Erhan, Y. Bengio, A. Courville, and P. Vincent, “Visualizing higher-layer features of a deep network,” *University of Montreal*, vol. 1341, no. 3, p. 1, 2009.
- [196] A. Mahendran and A. Vedaldi, “Understanding deep image representations by inverting them,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015.
- [197] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, “Understanding neural networks through deep visualization,” *arXiv preprint arXiv:1506.06579*, 2015.
- [198] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, “On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation,” *PloS one*, vol. 10, no. 7, p. e0130140, 2015.
- [199] A. Shrikumar, P. Greenside, and A. Kundaje, “Learning important features through propagating activation differences,” in *Proc. Int. Conf. Mach. Learn.*, 2017.
- [200] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017.
- [201] M. T. Ribeiro, S. Singh, and C. Guestrin, “”why should i trust you?” explaining the predictions of any classifier,” in *Int. Conf. on knowl. Discovery & Data Mining*, 2016.
- [202] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling, “Visualizing deep neural network decisions: Prediction difference analysis,” in *Int. Conf. Learn. Representations*, 2017.
- [203] P. W. Koh and P. Liang, “Understanding black-box predictions via influence functions,” in *Proc. Int. Conf. Mach. Learn.*, 2017.

- [204] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in *Advances Neural Inf. Process. Syst.*, 2017.
- [205] C. Chen, O. Li, D. Tao, A. Barnett, C. Rudin, and J. K. Su, “This looks like that: deep learning for interpretable image recognition,” in *Advances Neural Inf. Process. Syst.*, 2019.
- [206] D. Alvarez Melis and T. Jaakkola, “Towards robust interpretability with self-explaining neural networks,” in *Advances Neural Inf. Process. Syst.*, 2018.
- [207] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas *et al.*, “Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav),” in *Proc. Int. Conf. Mach. Learn.*, 2018.
- [208] T. Wang, “Gaining free or low-cost interpretability with interpretable partial substitute,” in *Proc. Int. Conf. Mach. Learn.*, 2019.
- [209] A. Subramanian, D. Pruthi, H. Jhamtani, T. Berg-Kirkpatrick, and E. Hovy, “Spine: Sparse interpretable neural embeddings,” in *AAAI Conference on Artificial Intelligence*, 2018.
- [210] X. Chen, Y. Duan, R. Houthoofd, J. Schulman, I. Sutskever, and P. Abbeel, “Infogan: Interpretable representation learning by information maximizing generative adversarial nets,” in *Advances Neural Inf. Process. Syst.*, 2016.
- [211] S. You, D. Ding, K. Canini, J. Pfeifer, and M. Gupta, “Deep lattice networks and partial monotonic functions,” in *Advances Neural Inf. Process. Syst.*, 2017.
- [212] Y. Ming, P. Xu, H. Qu, and L. Ren, “Interpretable and steerable sequence learning via prototypes,” in *Int. Conf. on knowl. Discovery & Data Mining*, 2019.
- [213] S. O. Arik and T. Pfister, “Protoattend: Attention-based prototypical learning,” *Journal of Machine Learning Research*, vol. 21, pp. 1–35, 2020.
- [214] M. Nauta, R. van Bree, and C. Seifert, “Neural prototype trees for interpretable fine-grained image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [215] R. Short and K. Fukunaga, “The optimal distance measure for nearest neighbor classification,” *IEEE Transactions on Information Theory*, vol. 27, no. 5, 1981.
- [216] T. Hastie and R. Tibshirani, “Discriminant adaptive nearest neighbor classification and regression,” in *Advances Neural Inf. Process. Syst.*, 1995.

- [217] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, “Signature verification using a” siamese” time delay neural network,” *Advances Neural Inf. Process. Syst.*, 1993.
- [218] R. Hadsell, S. Chopra, and Y. LeCun, “Dimensionality reduction by learning an invariant mapping,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2006.
- [219] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, “Supervised contrastive learning,” in *Advances Neural Inf. Process. Syst.*, 2020.
- [220] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015.
- [221] W. Chen, X. Chen, J. Zhang, and K. Huang, “Beyond triplet loss: a deep quadruplet network for person re-identification,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017.
- [222] K. Sohn, “Improved deep metric learning with multi-class n-pair loss objective,” in *Advances Neural Inf. Process. Syst.*, 2016.
- [223] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krahenbuhl, “Sampling matters in deep embedding learning,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017.
- [224] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “Deepface: Closing the gap to human-level performance in face verification,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014.
- [225] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, “Cosface: Large margin cosine loss for deep face recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [226] Z. Cao, D. Liu, Q. Wang, and Y. Chen, “Towards unbiased label distribution learning for facial pose estimation using anisotropic spherical gaussian,” in *Proc. Eur. Conf. Comput. Vis.*, 2022.
- [227] Z. Cao, Z. Chu, D. Liu, and Y. Chen, “A vector-based representation to enhance head pose estimation,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021.
- [228] T. Xiao, S. Li, B. Wang, L. Lin, and X. Wang, “Joint detection and identification feature learning for person search,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017.

- [229] M. Kaya and H. Ş. Bilge, “Deep metric learning: A survey,” *Symmetry*, vol. 11, no. 9, p. 1066, 2019.
- [230] M. Gutmann and A. Hyvärinen, “Noise-contrastive estimation: A new estimation principle for unnormalized statistical models,” in *Int. Conf. on Artificial Intelligence and Statistics*, 2010.
- [231] A. v. d. Oord, Y. Li, and O. Vinyals, “Representation learning with contrastive predictive coding,” *arXiv preprint arXiv:1807.03748*, 2018.
- [232] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, “Learning deep representations by mutual information estimation and maximization,” in *Int. Conf. Learn. Representations*, 2019.
- [233] X. Chen, H. Fan, R. Girshick, and K. He, “Improved baselines with momentum contrastive learning,” *arXiv preprint arXiv:2003.04297*, 2020.
- [234] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in *Proc. Int. Conf. Mach. Learn.*, 2020.
- [235] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, “Momentum contrast for unsupervised visual representation learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [236] M. A. Bautista, A. Sanakoyeu, E. Tikhoncheva, and B. Ommer, “Cliquecnn: Deep unsupervised exemplar learning,” in *Advances Neural Inf. Process. Syst.*, 2016.
- [237] J. Xie, R. Girshick, and A. Farhadi, “Unsupervised deep embedding for clustering analysis,” in *Proc. Int. Conf. Mach. Learn.*, 2016.
- [238] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, “Deep clustering for unsupervised learning of visual features,” in *Proc. Eur. Conf. Comput. Vis.*, 2018.
- [239] X. Yan, I. Misra, A. Gupta, D. Ghadiyaram, and D. Mahajan, “Clusterfit: Improving generalization of visual representations,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [240] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, “Unsupervised learning of visual features by contrasting cluster assignments,” in *Advances Neural Inf. Process. Syst.*, 2020.
- [241] J. Li, P. Zhou, C. Xiong, and S. C. Hoi, “Prototypical contrastive learning of unsupervised representations,” in *Int. Conf. Learn. Representations*, 2020.

- [242] W. Van Gansbeke, S. Vandenhende, S. Georgoulis, M. Proesmans, and L. Van Gool, “Scan: Learning to classify images without labels,” in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [243] K. N. Yaling Tao, Kentaro Takagi, “Clustering-friendly representation learning via instance discrimination and feature decorrelation,” in *Int. Conf. Learn. Representations*, 2021.
- [244] Y. Sun, C. Cheng, Y. Zhang, C. Zhang, L. Zheng, Z. Wang, and Y. Wei, “Circle loss: A unified perspective of pair similarity optimization,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [245] R. Suzuki, S. Fujita, and T. Sakai, “Arc loss: Softmax with additive angular margin for answer retrieval,” in *Asia Information Retrieval Symposium*, 2019, pp. 34–40.
- [246] W. Wang, T. Zhou, F. Yu, J. Dai, E. Konukoglu, and L. Van Gool, “Exploring cross-image pixel contrast for semantic segmentation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [247] M. Biehl, B. Hammer, and T. Villmann, “Distance measures for prototype based classification,” in *International Workshop on Brain-Inspired Computing*, 2013.
- [248] X. Ji, J. F. Henriques, and A. Vedaldi, “Invariant information clustering for unsupervised image classification and segmentation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [249] X. Zhan, J. Xie, Z. Liu, Y.-S. Ong, and C. C. Loy, “Online deep clustering for unsupervised representation learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [250] B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong, “Towards k-means-friendly spaces: Simultaneous deep learning and clustering,” in *Proc. Int. Conf. Mach. Learn.*, 2017.
- [251] J. Wu, K. Long, F. Wang, C. Qian, C. Li, Z. Lin, and H. Zha, “Deep comprehensive correlation mining for image clustering,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [252] M. Tschannen, J. Djolonga, P. K. Rubenstein, S. Gelly, and M. Lucic, “On mutual information maximization for representation learning,” *arXiv preprint arXiv:1907.13625*, 2019.
- [253] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, “Unsupervised feature learning via non-parametric instance discrimination,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.

- [254] Y. Tao, K. Takagi, and K. Nakata, “Clustering-friendly representation learning via instance discrimination and feature decorrelation,” *arXiv preprint arXiv:2106.00131*, 2021.
- [255] T. W. Tsai, C. Li, and J. Zhu, “Mice: Mixture of contrastive experts for unsupervised image clustering,” in *Int. Conf. Learn. Representations*, 2020.
- [256] N. Tumanyan, M. Geyer, S. Bagon, and T. Dekel, “Plug-and-play diffusion features for text-driven image-to-image translation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023.
- [257] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017.
- [258] K. Kim, S. Park, E. Jeon, T. Kim, and D. Kim, “A style-aware discriminator for controllable image translation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [259] T. Park, A. A. Efros, R. Zhang, and J.-Y. Zhu, “Contrastive learning for unpaired image-to-image translation,” in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [260] T. Park, J.-Y. Zhu, O. Wang, J. Lu, E. Shechtman, A. Efros, and R. Zhang, “Swapping autoencoder for deep image manipulation,” *Advances Neural Inf. Process. Syst.*, 2020.
- [261] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017.
- [262] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in *Proc. Int. Conf. Mach. Learn.*, 2017.
- [263] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, “Improved training of wasserstein gans,” *Advances Neural Inf. Process. Syst.*, 2017.
- [264] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, “Unrolled generative adversarial networks,” *ICLR*, 2017.
- [265] S. Ravuri and O. Vinyals, “Classification accuracy score for conditional generative models,” *Advances Neural Inf. Process. Syst.*, 2019.
- [266] B. Li, K. Xue, B. Liu, and Y.-K. Lai, “BbDM: Image-to-image translation with brownian bridge diffusion models,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023.

- [267] K. Mei, N. G. Nair, and V. M. Patel, “Bi-noising diffusion: Towards conditional diffusion models with generative restoration priors,” *arXiv preprint arXiv:2212.07352*, 2022.
- [268] I. Donadello, L. Serafini, and A. D. Garcez, “Logic tensor networks for semantic image interpretation,” *International Joint Conferences on Artificial Intelligence*, 2017.
- [269] T. Zhou, S. Qi, W. Wang, J. Shen, and S.-C. Zhu, “Cascaded parsing of human-object interaction recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2021.
- [270] K. Yi, J. Wu, C. Gan, A. Torralba, P. Kohli, and J. Tenenbaum, “Neural-symbolic vqa: Disentangling reasoning from vision and language understanding,” *Advances Neural Inf. Process. Syst*, 2018.
- [271] S. Amizadeh, H. Palangi, A. Polozov, Y. Huang, and K. Koishida, “Neuro-symbolic visual reasoning: Disentangling,” in *Proc. Int. Conf. Mach. Learn.*, 2020.
- [272] F. Arabshahi, J. Lee, M. Gawarecki, K. Mazaitis, A. Azaria, and T. Mitchell, “Conversational neuro-symbolic commonsense reasoning,” in *AAAI Conference on Artificial Intelligence*, 2021.
- [273] A. B. Yildirim, V. Baday, E. Erdem, A. Erdem, and A. Dundar, “Inst-inpaint: Instructing to remove objects with diffusion models,” *arXiv preprint arXiv:2304.03246*, 2023.
- [274] J. Pfeiffer, G. Geigle, A. Kamath, J.-M. O. Steitz, S. Roth, I. Vulić, and I. Gurevych, “xGQA: Cross-Lingual Visual Question Answering,” in *Annual Meeting of the Association for Computational Linguistics*, 2022.
- [275] Y. Lu, Q. Wang, S. Ma, T. Geng, Y. V. Chen, H. Chen, and D. Liu, “Transflow: Transformer as flow learner,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023.
- [276] J. Liang, T. Zhou, D. Liu, and W. Wang, “Clustseg: Clustering for universal segmentation,” *arXiv preprint arXiv:2305.02187*, 2023.
- [277] D. Liu, Y. Cui, W. Tan, and Y. Chen, “Sg-net: Spatial granularity network for one-stage video instance segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [278] W. Wang, J. Liang, and D. Liu, “Learning equivariant segmentation with instance-unique querying,” *Advances Neural Inf. Process. Syst*, 2022.
- [279] Y. Cui, C. Han, and D. Liu, “Cml-mots: Collaborative multi-task learning for multi-object tracking and segmentation,” *arXiv preprint arXiv:2311.00987*, 2023.

- [280] Z. Qin, X. Lu, X. Nie, D. Liu, Y. Yin, and W. Wang, "Coarse-to-fine video instance segmentation with factorized conditional appearance flows," *IEEE/CAA Journal of Automatica Sinica*, 2023.
- [281] Y. Wang, J. Zhao, H. Xu, C. Han, Z. Tao, D. Zhao, D. Zhou, G. Tong, D. Liu, and Z. Ji, "A systematic evaluation of computation methods for cell segmentation," *bioRxiv*, 2024.
- [282] J. Beal, E. Kim, E. Tzeng, D. H. Park, A. Zhai, and D. Kislyuk, "Toward transformer-based object detection," *arXiv preprint arXiv:2012.09958*, 2020.
- [283] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [284] D. Liu, Y. Cui, Y. Chen, J. Zhang, and B. Fan, "Video object detection for autonomous driving: Motion-aid feature calibration," *Neurocomputing*, vol. 409, 2020.
- [285] X. Pan, Z. Xia, S. Song, L. E. Li, and G. Huang, "3d object detection with pointformer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [286] Z. Yuan, X. Song, L. Bai, Z. Wang, and W. Ouyang, "Temporal-channel transformer for 3d lidar-based video object detection for autonomous driving," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.
- [287] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," in *Int. Conf. Learn. Representations*, 2021.
- [288] L. Huang, J. Tan, J. Liu, and J. Yuan, "Hand-transformer: non-autoregressive structured modeling for 3d hand pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [289] L. Huang, J. Tan, J. Meng, J. Liu, and J. Yuan, "Hot-net: Non-autoregressive transformer for 3d hand-object pose estimation," in *ACM MultiMedia*, 2020.
- [290] K. Lin, L. Wang, and Z. Liu, "End-to-end human pose and mesh reconstruction with transformers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [291] S. Yang, Z. Quan, M. Nie, and W. Yang, "Transpose: Keypoint localization via transformer," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [292] H. Bao, L. Dong, S. Piao, and F. Wei, "Beit: Bert pre-training of image transformers," in *Int. Conf. Learn. Representations*, 2022.
- [293] X. He, C. Li, P. Zhang, J. Yang, and X. E. Wang, "Parameter-efficient fine-tuning for vision transformers," *arXiv preprint arXiv:2203.16329*, 2022.

- [294] Q. Fournier, G. M. Caron, and D. Aloise, “A practical survey on faster and lighter transformers,” *arXiv preprint arXiv:2103.14636*, 2021.
- [295] K. Islam, “Recent advances in vision transformer: A survey and outlook of recent work,” *arXiv preprint arXiv:2203.01536*, 2022.
- [296] C. Zhang, H. Wan, X. Shen, and Z. Wu, “Patchformer: An efficient point transformer with patch attention,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [297] Y. Li, C.-Y. Wu, H. Fan, K. Mangalam, B. Xiong, J. Malik, and C. Feichtenhofer, “Mvitv2: Improved multiscale vision transformers for classification and detection,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [298] X. Zhai, A. Kolesnikov, N. Houlsby, and L. Beyer, “Scaling vision transformers,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [299] C. Riquelme, J. Puigcerver, B. Mustafa, M. Neumann, R. Jenatton, A. Susano Pinto, D. Keysers, and N. Houlsby, “Scaling vision with sparse mixture of experts,” in *Advances Neural Inf. Process. Syst.*, 2021.
- [300] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, “Vivit: A video vision transformer,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [301] C.-F. R. Chen, Q. Fan, and R. Panda, “Crossvit: Cross-attention multi-scale vision transformer for image classification,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [302] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, “Pyramid vision transformer: A versatile backbone for dense prediction without convolutions,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [303] M. Noroozi and P. Favaro, “Unsupervised learning of visual representations by solving jigsaw puzzles,” in *Proc. Eur. Conf. Comput. Vis.*, 2016.
- [304] R. Zhang, P. Isola, and A. A. Efros, “Colorful image colorization,” in *Proc. Eur. Conf. Comput. Vis.*, 2016.
- [305] Y. He, S. Zheng, Y. Tay, J. Gupta, Y. Du, V. Aribandi, Z. Zhao, Y. Li, Z. Chen, D. Metzler *et al.*, “Hyperprompt: Prompt-based task-conditioning of transformers,” in *Proc. Int. Conf. Mach. Learn.*, 2022.
- [306] B. Lester, R. Al-Rfou, and N. Constant, “The power of scale for parameter-efficient prompt tuning,” in *Empirical Methods in Natural Language Process.*, 2021.
- [307] F. Ma, C. Zhang, L. Ren, J. Wang, Q. Wang, W. Wu, X. Quan, and D. Song, “Xprompt: Exploring the extreme of prompt tuning,” in *Empirical Methods in Natural Language Process.*, 2022.

- [308] Y. Gao, X. Shi, Y. Zhu, H. Wang, Z. Tang, X. Zhou, M. Li, and D. N. Metaxas, “Visual prompt tuning for test-time domain adaptation,” *arXiv preprint arXiv:2210.04831*, 2022.
- [309] Y. Xing, Q. Wu, D. Cheng, S. Zhang, G. Liang, and Y. Zhang, “Class-aware visual prompt tuning for vision-language pre-trained model,” *arXiv preprint arXiv:2208.08340*, 2022.
- [310] S. Park, J. Lee, S. Mo, and J. Shin, “Lookahead: A far-sighted alternative of magnitude-based pruning,” *arXiv preprint arXiv:2002.04809*, 2020.
- [311] Y. Guo, A. Yao, and Y. Chen, “Dynamic network surgery for efficient dnns,” *Advances Neural Inf. Process. Syst*, 2016.
- [312] X. Dong, S. Chen, and S. Pan, “Learning to prune deep neural networks via layer-wise optimal brain surgeon,” *Advances Neural Inf. Process. Syst*, 2017.
- [313] X. Ding, G. Ding, Y. Guo, and J. Han, “Centripetal sgd for pruning very deep convolutional networks with complicated structure,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019.
- [314] Z. You, K. Yan, J. Ye, M. Ma, and P. Wang, “Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks,” *Advances Neural Inf. Process. Syst*, 2019.
- [315] G. Fang, X. Ma, M. Song, M. B. Mi, and X. Wang, “Depgraph: Towards any structural pruning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023.
- [316] X. Ma, G. Fang, and X. Wang, “Llm-pruner: On the structural pruning of large language models,” *arXiv preprint arXiv:2305.11627*, 2023.
- [317] L. Beyer, X. Zhai, A. Royer, L. Markeeva, R. Anil, and A. Kolesnikov, “Knowledge distillation: A good teacher is patient and consistent,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [318] Z. Chen, Y. Yang, W. Qifan, L. Jiahao, W. Jingang, X. Yunsen, W. Wei, and S. Dawei, “Minidisc: Minimal distillation schedule for language model compression,” *arXiv preprint arXiv:2205.14570*, 2022.
- [319] S. Wu, H. Chen, X. Quan, Q. Wang, and R. Wang, “Ad-kd: Attribution-driven knowledge distillation for language model compression,” 2023.
- [320] L. Hou, Z. Huang, L. Shang, X. Jiang, X. Chen, and Q. Liu, “Dynabert: Dynamic bert with adaptive width and depth,” in *Advances Neural Inf. Process. Syst*, 2020.

- [321] A. Fan, E. Grave, and A. Joulin, “Reducing transformer depth on demand with structured dropout,” in *Int. Conf. Learn. Representations*, 2020.
- [322] P. Michel, O. Levy, and G. Neubig, “Are sixteen heads really better than one?” in *Advances Neural Inf. Process. Syst.*, 2019.
- [323] S. Han, H. Mao, and W. J. Dally, “Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding,” *arXiv preprint arXiv:1510.00149*, 2015.
- [324] R. Yu, A. Li, C.-F. Chen, J.-H. Lai, V. I. Morariu, X. Han, M. Gao, C.-Y. Lin, and L. S. Davis, “Nisp: Pruning networks using neuron importance score propagation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [325] S. W. Kim and H.-E. Kim, “Transferring knowledge to smaller network with class-distance loss,” 2017.
- [326] R. Müller, S. Kornblith, and G. E. Hinton, “When does label smoothing help?” *Advances Neural Inf. Process. Syst.*, 2019.
- [327] J. Gou, B. Yu, S. J. Maybank, and D. Tao, “Knowledge distillation: A survey,” *Int. J. Comput. Vis.*, 2021.
- [328] Z. Li, X. Li, L. Yang, B. Zhao, R. Song, L. Luo, J. Li, and J. Yang, “Curriculum temperature for knowledge distillation,” in *AAAI Conference on Artificial Intelligence*, 2023.
- [329] J. H. Cho and B. Hariharan, “On the efficacy of knowledge distillation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [330] B. Zhao, Q. Cui, R. Song, Y. Qiu, and J. Liang, “Decoupled knowledge distillation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [331] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2013.
- [332] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, “Fitnets: Hints for thin deep nets,” *arXiv preprint arXiv:1412.6550*, 2014.
- [333] B. Heo, M. Lee, S. Yun, and J. Y. Choi, “Knowledge transfer via distillation of activation boundaries formed by hidden neurons,” in *AAAI Conference on Artificial Intelligence*, 2019.
- [334] P. Passban, Y. Wu, M. Rezagholizadeh, and Q. Liu, “Alp-kd: Attention-based layer projection for knowledge distillation,” in *AAAI Conference on Artificial Intelligence*, 2021.

- [335] D. Chen, J.-P. Mei, Y. Zhang, C. Wang, Z. Wang, Y. Feng, and C. Chen, “Cross-layer distillation with semantic calibration,” in *AAAI Conference on Artificial Intelligence*, 2021.
- [336] X. Wang, T. Fu, S. Liao, S. Wang, Z. Lei, and T. Mei, “Exclusivity-consistency regularized knowledge distillation for face recognition,” in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [337] Z. Huang and N. Wang, “Like what you like: Knowledge distill via neuron selectivity transfer,” *arXiv preprint arXiv:1707.01219*, 2017.
- [338] J. Kim, S. Park, and N. Kwak, “Paraphrasing complex network: Network compression via factor transfer,” *Advances Neural Inf. Process. Syst.*, 2018.
- [339] S. Zagoruyko and N. Komodakis, “Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer,” *arXiv preprint arXiv:1612.03928*, 2016.
- [340] S. You, C. Xu, C. Xu, and D. Tao, “Learning from multiple teacher networks,” in *Special Interest Group on Knowledge Discovery and Data Mining*, 2017.
- [341] W. Park, D. Kim, Y. Lu, and M. Cho, “Relational knowledge distillation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019.
- [342] J. Yim, D. Joo, J. Bae, and J. Kim, “A gift from knowledge distillation: Fast optimization, network minimization and transfer learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017.
- [343] L. Yu, V. O. Yazici, X. Liu, J. v. d. Weijer, Y. Cheng, and A. Ramisa, “Learning metrics from teachers: Compact networks for image embedding,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019.
- [344] B. Peng, X. Jin, J. Liu, D. Li, Y. Wu, Y. Liu, S. Zhou, and Z. Zhang, “Correlation congruence for knowledge distillation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [345] H. Chen, Y. Wang, C. Xu, C. Xu, and D. Tao, “Learning student networks via feature embedding,” *IEEE Trans. Neural Netw. Learning Sys.*, 2020.
- [346] F. Tung and G. Mori, “Similarity-preserving knowledge distillation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [347] N. Passalis, M. Tzelepi, and A. Tefas, “Heterogeneous knowledge distillation using information flow modeling,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.

- [348] M. Gao, Y. Wang, and L. Wan, “Residual error based knowledge distillation,” *Neurocomputing*, 2021.
- [349] X. Jin, B. Peng, Y. Wu, Y. Liu, J. Liu, D. Liang, J. Yan, and X. Hu, “Knowledge distillation via route constrained optimization,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [350] Z. Yang, Z. Li, A. Zeng, Z. Li, C. Yuan, and Y. Li, “Vitkd: Practical guidelines for vit feature knowledge distillation,” *arXiv preprint arXiv:2209.02432*, 2022.
- [351] X. Chen, Q. Cao, Y. Zhong, J. Zhang, S. Gao, and D. Tao, “Deardk: data-efficient early knowledge distillation for vision transformers,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [352] X. Yao, Y. ZHANG, Z. Chen, J. Jia, and B. Yu, “Distill vision transformers to cnns via low-rank representation approximation,” 2022.
- [353] B. Zhao, R. Song, and J. Liang, “Cumulative spatial knowledge distillation for vision transformers,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2023.
- [354] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, “Training data-efficient image transformers & distillation through attention,” in *Proc. Int. Conf. Mach. Learn.*, 2021.
- [355] S. Ren, Z. Gao, T. Hua, Z. Xue, Y. Tian, S. He, and H. Zhao, “Co-advise: Cross inductive bias distillation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [356] D. Zhou, B. Kang, X. Jin, L. Yang, X. Lian, Z. Jiang, Q. Hou, and J. Feng, “Deepvit: Towards deeper vision transformer,” *arXiv preprint arXiv:2103.11886*, 2021.
- [357] D. Han, X. Pan, Y. Han, S. Song, and G. Huang, “Flatten transformer: Vision transformer using focused linear attention,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2023.
- [358] A. Vattani, “K-means requires exponentially many iterations even in the plane,” in *Annual Symposium on Computational Geometry*, 2009.
- [359] A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming, “K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data,” *Information Sciences*, 2023.
- [360] S. Balakrishnan, M. J. Wainwright, and B. Yu, “Statistical guarantees for the em algorithm: From population to sample-based analysis,” *Annals of Statistics*, 2017.

- [361] J. Liang, Y. Cui, Q. Wang, T. Geng, W. Wang, and D. Liu, “Clusterfomer: Clustering as a universal visual learner,” *Advances Neural Inf. Process. Syst.*, 2024.
- [362] T. Pang, K. Xu, Y. Dong, C. Du, N. Chen, and J. Zhu, “Rethinking softmax cross-entropy loss for adversarial robustness,” in *Int. Conf. Learn. Representations*, 2020.
- [363] X. Zhang, R. Zhao, Y. Qiao, and H. Li, “Rbf-softmax: Learning deep representative prototypes with radial basis function softmax,” in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [364] E. Rosch, “Cognitive representations of semantic categories.” *Journal of Experimental Psychology: General*, vol. 104, no. 3, p. 192, 1975.
- [365] B. J. Knowlton and L. R. Squire, “The learning of categories: Parallel brain systems for item memory and category knowledge,” *Science*, vol. 262, no. 5140, pp. 1747–1749, 1993.
- [366] P. A. Knight, “The sinkhorn–knopp algorithm: convergence and applications,” *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 1, pp. 261–275, 2008.
- [367] M. Boudiaf, J. Rony, I. M. Ziko, E. Granger, M. Pedersoli, P. Piantanida, and I. B. Ayed, “A unifying mutual information view of metric learning: cross-entropy vs. pairwise losses,” in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [368] D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Instance normalization: The missing ingredient for fast stylization,” *arXiv preprint arXiv:1607.08022*, 2016.
- [369] —, “Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017.
- [370] C. Cornelio, J. Stuehmer, S. X. Hu, and T. Hospedales, “Learning where and when to reason in neuro-symbolic inference,” in *Int. Conf. Learn. Representations*, 2022.
- [371] OpenAI, “Gpt-4 technical report,” *ArXiv*, vol. abs/2303.08774, 2023. [Online]. Available: <https://api.semanticscholar.org/CorpusID:257532815>
- [372] N. R. Jennings, “On agent-based software engineering,” *Artificial intelligence*, 2000.
- [373] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. Van Den Hengel, “Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.

- [374] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, “Masked-attention mask transformer for universal image segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [375] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. Van Gool, “Repaint: Inpainting using denoising diffusion probabilistic models,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [376] M. Xia, Z. Zhong, and D. Chen, “Structured pruning learns compact and accurate models,” in *Annual Meeting of the Association for Computational Linguistics*, 2022.
- [377] T. Kim, J. Oh, N. Kim, S. Cho, and S.-Y. Yun, “Comparing kullback-leibler divergence and mean squared error loss in knowledge distillation,” *arXiv preprint arXiv:2105.08919*, 2021.
- [378] T. Van Erven and P. Harremos, “Rényi divergence and kullback-leibler divergence,” *IEEE Trans. on Info. Theory*, 2014.
- [379] C. Liu, C. Tao, J. Feng, and D. Zhao, “Multi-granularity structural knowledge distillation for language model compression,” in *Annual Meeting of the Association for Computational Linguistics*, 2022.
- [380] A. Jaegle, S. Borgeaud, J.-B. Alayrac, C. Doersch, C. Ionescu, D. Ding, S. Koppula, D. Zoran, A. Brock, E. Shelhamer *et al.*, “Perceiver io: A general architecture for structured inputs & outputs,” *Int. Conf. Learn. Representations*, 2022.
- [381] D. Sun, D. Vlastic, C. Herrmann, V. Jampani, M. Krainin, H. Chang, R. Zabih, W. T. Freeman, and C. Liu, “Autoflow: Learning a better training set for optical flow,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [382] F. Zhang, O. J. Woodford, V. A. Prisacariu, and P. H. Torr, “Separable flow: Learning motion cost volumes for optical flow estimation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [383] S. Jiang, D. Campbell, Y. Lu, H. Li, and R. Hartley, “Learning to estimate hidden motions with global motion aggregation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [384] A. Luo, F. Yang, K. Luo, X. Li, H. Fan, and S. Liu, “Learning optical flow with adaptive graph reasoning,” in *AAAI Conference on Artificial Intelligence*, 2022.
- [385] A. Luo, F. Yang, X. Li, and S. Liu, “Learning optical flow with kernel patch attention,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.

- [386] Z. Zheng, N. Nie, Z. Ling, P. Xiong, J. Liu, H. Wang, and J. Li, “Dip: Deep inverse patchmatch for high-resolution optical flow,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [387] S. Zhao, L. Zhao, Z. Zhang, E. Zhou, and D. Metaxas, “Global matching with overlapping attention for optical flow estimation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [388] H. Xu, J. Zhang, J. Cai, H. Rezatofghi, and D. Tao, “Gmflow: Learning optical flow via global matching,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [389] M. Zhai, X. Xiang, N. Lv, S. M. Ali, and A. E. Saddik, “Skflow: Optical flow estimation using selective kernel networks,” *Advances Neural Inf. Process. Syst.*, 2022.
- [390] N. Zhang, F. Nex, G. Vosselman, and N. Kerle, “Lite-mono: A lightweight cnn and transformer architecture for self-supervised monocular depth estimation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023.
- [391] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, “Ccnnet: Criss-cross attention for semantic segmentation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [392] Q. Yu, H. Wang, S. Qiao, M. Collins, Y. Zhu, H. Adam, A. Yuille, and L.-C. Chen, “k-means mask transformer,” *Proc. Eur. Conf. Comput. Vis.*, 2022.
- [393] X. Ma, Y. Zhou, H. Wang, C. Qin, B. Sun, C. Liu, and Y. Fu, “Image as set of points,” in *Int. Conf. Learn. Representations*, 2023. [Online]. Available: <https://openreview.net/forum?id=awnvqZja69>
- [394] X. Wang, J. Gao, M. Long, and J. Wang, “Self-tuning for data-efficient deep learning,” in *Proc. Int. Conf. Mach. Learn.*, 2021.
- [395] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese, “Deep metric learning via lifted structured feature embedding,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016.
- [396] J. Plested and T. Gedeon, “Deep transfer learning for image classification: a survey,” *arXiv preprint arXiv:2205.09904*, 2022.
- [397] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015.
- [398] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” *arXiv preprint arXiv:1706.05587*, 2017.
- [399] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun, “Unified perceptual parsing for scene understanding,” in *Proc. Eur. Conf. Comput. Vis.*, 2018.

- [400] B. Cheng, A. G. Schwing, and A. Kirillov, “Per-pixel classification is not all you need for semantic segmentation,” in *Advances Neural Inf. Process. Syst.*, 2021.
- [401] L. Zhao and W. Tao, “Jsnet: Joint instance and semantic segmentation of 3d point clouds,” in *AAAI Conference on Artificial Intelligence*, 2020.
- [402] H. Lei, N. Akhtar, and A. Mian, “Seggcn: Efficient 3d point cloud segmentation with fuzzy spherical kernel,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [403] S. Fan, Q. Dong, F. Zhu, Y. Lv, P. Ye, and F.-Y. Wang, “Scf-net: Learning spatial contextual features for large-scale point cloud segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [404] M. Xu, R. Ding, H. Zhao, and X. Qi, “Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [405] H. Ran, J. Liu, and C. Wang, “Surface representation for point clouds,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [406] L. Tang, Y. Zhan, Z. Chen, B. Yu, and D. Tao, “Contrastive boundary learning for point cloud segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [407] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, “Point transformer,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [408] C. Park, Y. Jeong, M. Cho, and J. Park, “Fast point transformer,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [409] J. Choe, C. Park, F. Rameau, J. Park, and I. S. Kweon, “Pointmixer: Mlp-mixer for point cloud understanding,” in *Proc. Eur. Conf. Comput. Vis.*, 2022.
- [410] X. Wu, Y. Lao, L. Jiang, X. Liu, and H. Zhao, “Point transformer v2: Grouped vector attention and partition-based pooling,” in *Advances Neural Inf. Process. Syst.*, 2022.
- [411] X. Lai, J. Liu, L. Jiang, L. Wang, H. Zhao, S. Liu, X. Qi, and J. Jia, “Stratified transformer for 3d point cloud segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [412] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, “Randla-net: Efficient semantic segmentation of large-scale point clouds,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.

- [413] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui, "Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [414] H. Ran, W. Zhuo, J. Liu, and L. Lu, "Learning inner-group relations on point clouds," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [415] F. Zhang, J. Fang, B. Wah, and P. Torr, "Deep fusionnet for point cloud semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [416] Z. Hu, M. Zhen, X. Bai, H. Fu, and C.-l. Tai, "Jsenet: Joint semantic segmentation and edge detection network for 3d point clouds," in *Proc. Eur. Conf. Comput. Vis.*, 2020.
- [417] G. Qian, Y. Li, H. Peng, J. Mai, H. Hammoud, M. Elhoseiny, and B. Ghanem, "Pointnext: Revisiting pointnet++ with improved training and scaling strategies," *Advances Neural Inf. Process. Syst.*, 2022.
- [418] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, and J. Jia, "Pointgroup: Dual-set point grouping for 3d instance segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [419] M. Zhong, X. Chen, X. Chen, G. Zeng, and Y. Wang, "Maskgroup: Hierarchical point grouping and masking for 3d instance segmentation," in *IEEE Int. Conf. on Multimedia and Expo*, 2022.
- [420] Z. Liang, Z. Li, S. Xu, M. Tan, and K. Jia, "Instance segmentation in 3d scenes using semantic superpoint tree networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [421] T. He, C. Shen, and A. Van Den Hengel, "Dyco3d: Robust instance segmentation of 3d point clouds through dynamic convolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [422] S. Chen, J. Fang, Q. Zhang, W. Liu, and X. Wang, "Hierarchical aggregation for 3d instance segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [423] Y. Wu, M. Shi, S. Du, H. Lu, Z. Cao, and W. Zhong, "3d instances as 1d kernels," in *Proc. Eur. Conf. Comput. Vis.*, 2022.
- [424] F. Engelmann, M. Bokeloh, A. Fathi, B. Leibe, and M. Nießner, "3d-mpa: Multi-proposal aggregation for 3d semantic instance segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.

- [425] L. Han, T. Zheng, L. Xu, and L. Fang, “Occuseg: Occupancy-aware 3d instance segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [426] T. Vu, K. Kim, T. M. Luu, T. Nguyen, and C. D. Yoo, “Softgroup for 3d instance segmentation on point clouds,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [427] A. Milioto, J. Behley, C. McCool, and C. Stachniss, “Lidar panoptic segmentation for autonomous driving,” in *IROS*, 2020.
- [428] S. Gasperini, M.-A. N. Mahani, A. Marcos-Ramiro, N. Navab, and F. Tombari, “Panoster: End-to-end panoptic segmentation of lidar point clouds,” *IEEE Robotics and Automation Letters*, 2021.
- [429] Z. Zhou, Y. Zhang, and H. Foroosh, “Panoptic-polarnet: Proposal-free lidar point cloud panoptic segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [430] F. Hong, H. Zhou, X. Zhu, H. Li, and Z. Liu, “Lidar-based panoptic segmentation via dynamic shifting network,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [431] K. Sirohi, R. Mohan, D. Büscher, W. Burgard, and A. Valada, “Efficientlps: Efficient lidar panoptic segmentation,” *IEEE Transactions on Robotics*, 2021.
- [432] R. Razani, R. Cheng, E. Li, E. Taghavi, Y. Ren, and L. Bingbing, “Gp-s3net: Graph-based panoptic sparse semantic segmentation network,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [433] S. Xu, R. Wan, M. Ye, X. Zou, and T. Cao, “Sparse cross-scale attention network for efficient lidar panoptic segmentation,” in *AAAI Conference on Artificial Intelligence*, 2022.
- [434] J. Li, X. He, Y. Wen, Y. Gao, X. Cheng, and D. Zhang, “Panoptic-phnet: Towards real-time and high-precision lidar panoptic segmentation via clustering pseudo heatmap,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [435] M. Liu, Q. Zhou, H. Zhao, J. Li, Y. Du, K. Keutzer, L. Du, and S. Zhang, “Prototype-voxel contrastive learning for lidar point cloud panoptic segmentation,” in *ICRA*, 2022.
- [436] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012.

- [437] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, “Panoptic segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019.
- [438] M. Huh, P. Agrawal, and A. A. Efros, “What makes imagenet good for transfer learning?” *arXiv preprint arXiv:1608.08614*, 2016.
- [439] R. Zhang, L. Wang, Y. Wang, P. Gao, H. Li, and J. Shi, “Parameter is not all you need: Starting from non-parametric networks for 3d point cloud analysis,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023.
- [440] C. Meng, Y. He, Y. Song, J. Song, J. Wu, J.-Y. Zhu, and S. Ermon, “Sdedit: Guided image synthesis and editing with stochastic differential equations,” *Int. Conf. Learn. Representations*, 2022.
- [441] G. Kwon and J. C. Ye, “Diffusion-based image translation using disentangled style and content representation,” *Int. Conf. Learn. Representations*, 2023.
- [442] K. Crowson, S. Biderman, D. Kornis, D. Stander, E. Hallahan, L. Castriato, and E. Raff, “Vqgan-clip: Open domain image generation and editing with natural language guidance,” in *Proc. Eur. Conf. Comput. Vis.*, 2022.
- [443] O. Bar-Tal, D. Ofri-Amar, R. Fridman, Y. Kasten, and T. Dekel, “Text2live: Text-driven layered image and video editing,” in *Proc. Eur. Conf. Comput. Vis.*, 2022.
- [444] Y. Ren, S. Guo, W. Bae, and D. J. Sutherland, “How to prepare your task head for finetuning,” in *Int. Conf. Learn. Representations*, 2023.
- [445] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances Neural Inf. Process. Syst*, 2019.
- [446] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015.
- [447] M. V. Narkhede, P. P. Bartakke, and M. S. Sutaone, “A review on weight initialization strategies for neural networks,” *Artificial Intelligence Review*, 2022.
- [448] M. G. Atigh, J. Schoep, E. Acar, N. Van Noord, and P. Mettes, “Hyperbolic image segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [449] A. Ermolov, L. Mirvakhabova, V. Khrulkov, N. Sebe, and I. Oseledets, “Hyperbolic vision transformers: Combining improvements in metric learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.

- [450] O. Ganea, G. Bécigneul, and T. Hofmann, “Hyperbolic neural networks,” in *Advances Neural Inf. Process. Syst.*, 2018.
- [451] V. Khrulkov, L. Mirvakhobova, E. Ustinova, I. Oseledets, and V. Lempitsky, “Hyperbolic image embeddings,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [452] W. Peng, T. Varanka, A. Mostafa, H. Shi, and G. Zhao, “Hyperbolic deep neural networks: A survey,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2021.
- [453] L. McInnes, J. Healy, and J. Melville, “Umap: Uniform manifold approximation and projection for dimension reduction,” *arXiv preprint arXiv:1802.03426*, 2018.
- [454] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” in *Proc. Int. Conf. Mach. Learn.*, 2017.
- [455] Y. Tian, D. Krishnan, and P. Isola, “Contrastive representation distillation,” in *Int. Conf. Learn. Representations*, 2019.
- [456] X. Liu, L. Li, C. Li, and A. Yao, “Norm: Knowledge distillation via n-to-one representation matching,” in *Int. Conf. Learn. Representations*, 2023.
- [457] K. Binici, N. T. Pham, T. Mitra, and K. Leman, “Preventing catastrophic forgetting and distribution mismatch in knowledge distillation via synthetic data,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022.
- [458] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, “Flownet: Learning optical flow with convolutional networks,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015.
- [459] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, “A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016.
- [460] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The kitti dataset,” *The International Journal of Robotics Research*, 2013.
- [461] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, “A naturalistic open source movie for optical flow evaluation,” in *Proc. Eur. Conf. Comput. Vis.*, 2012.
- [462] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo *et al.*, “The many faces of robustness: A critical analysis of out-of-distribution generalization,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.

- [463] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *Proc. Eur. Conf. Comput. Vis.*, 2014.
- [464] G. Van Horn, S. Branson, R. Farrell, S. Haber, J. Barry, P. Ipeirotis, P. Perona, and S. Belongie, “Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015.
- [465] T. Gebru, J. Krause, Y. Wang, D. Chen, J. Deng, and L. Fei-Fei, “Fine-grained car detection for visual census estimation,” in *AAAI Conference on Artificial Intelligence*, 2017.
- [466] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, “Imagenet large scale visual recognition challenge,” *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
- [467] J. Hessel, A. Holtzman, M. Forbes, R. L. Bras, and Y. Choi, “Clipscore: A reference-free evaluation metric for image captioning,” *arXiv preprint arXiv:2104.08718*, 2021.
- [468] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, “Emerging properties in self-supervised vision transformers,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021.
- [469] R. Likert, “A technique for the measurement of attitudes.” *Archives of Psychology*, 1932.
- [470] I. E. Allen and C. A. Seaman, “Likert scales and data analyses,” *Quality Progress*, vol. 40, no. 7, pp. 64–65, 2007.
- [471] D. Bertram, “Likert scales,” *Retrieved November*, vol. 2, no. 10, pp. 1–10, 2007.
- [472] H. N. Boone Jr and D. A. Boone, “Analyzing likert data,” *The Journal of Extension*, vol. 50, no. 2, p. 48, 2012.
- [473] G. Norman, “Likert scales, levels of measurement and the “laws” of statistics,” *Advances in Health Sciences Education*, vol. 15, pp. 625–632, 2010.