Visual Intelligence Beyond Human Supervision

Xudong Wang

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2025-148
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2025/EECS-2025-148.html

August 10, 2025

Copyright © 2025, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

Visual Intelligence Beyond Human Supervision

by

XuDong Wang

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Trevor Darrell, Chair
Professor Jitendra Malik
Professor Kurt Keutzer
Dr. Ishan Misra

Summer 2025

Visual Intelligence Beyond Human Supervision

Copyright 2025
by
XuDong Wang

Abstract

Visual Intelligence Beyond Human Supervision

by

XuDong Wang

Doctor of Philosophy in Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Trevor Darrell, Chair

Achieving artificial general intelligence requires developing models capable of perceiving, understanding, and interacting with the world across diverse sensory modalities—beyond the confines of language alone. While self-supervised learning has enabled remarkable advances in large language models (LLMs), replicating this success in the visual domain remains a significant challenge, largely due to the continued reliance on human-annotated data. This dissertation explores how self-supervised learning can unlock visual intelligence beyond human supervision, enabling models to learn directly from the inherent structure and regularities of the visual world.

The thesis presents a series of efforts aimed at advancing this vision. First, it investigates self-supervised visual world understanding, demonstrating that models can achieve strong segmentation performance without the billions of labeled masks used in supervised approaches such as the Segment Anything Model (SAM). Instead, our work shows that models can "segment anything" by leveraging the rich semantics present in unlabeled data. Second, it introduces methods that unify generative and discriminative visual models through self-supervision and synthetic data, allowing these systems to complement one another and improve both visual understanding and generation. Third, the dissertation examines how to build robust visual models through self-supervised debiased learning, proposing techniques that mitigate bias and enhance generalization under imperfect data conditions, within a data-centric representation learning framework.

Together, these contributions serve a common goal: building scalable, multi-modality visual intelligence that learn not by mimicking human annotations, but by discovering the latent structure of the world itself!

To those I love, and who love me.

Contents

List of Figures

List of Tables

Acknowledgments

It's hard to believe that this chapter is nearing its close. The PhD journey at Berkeley has been one of the most demanding—and deeply rewarding—experiences of my life. Along the way, I've grown not just as a researcher, but as a person. This path would not have been possible without the guidance, support, and generosity of those who walked beside me. To each of you, I owe more than words can express.

First and foremost, I thank my PhD advisor, Professor Trevor Darrell, for the opportunity to work in one of the world's top research groups in computer vision and AI. His support, insight, and belief in me have been instrumental to my growth. He gave me the freedom to pursue what I cared about most, while offering thoughtful, grounded advice. When I felt discouraged, he reminded me to focus on long-term impact rather than short-term outcomes. I still remember him saying, "XuDong, it doesn't matter whether a paper is accepted or not—it's always more important to write papers you're proud of." That truth has stayed with me. Before starting any project, I now ask myself: will this be a paper I can be proud of? His ability to pair high-level vision with practical, empathetic mentorship has profoundly shaped how I think about research—and the kind of researcher I strive to become.

I am especially grateful to Professors Jitendra Malik and Kurt Keutzer, whose foundational work in computer vision and deep learning has long inspired me. I deeply appreciate their service on both my qualifying and thesis committees, and their thoughtful feedback and support throughout my time at Berkeley.

I am also thankful to my mentors and collaborators at Google DeepMind, Meta FAIR, and Meta GenAI—especially Professor Cordelia Schmid and Dr. Ishan Misra—for providing opportunities that allowed me to grow and contribute at the forefront of AI research. Ishan's mentorship during our collaborations at Meta made the summers of research both intellectually rewarding and personally memorable. I'm also grateful to Cordelia for her guidance during my internship at DeepMind—her extraordinary dedication and clarity elevated every project we worked on. I want to thank Professor Stella X. Yu for her mentorship and support during my time at the International Computer Science Institute (ICSI). In my first two years of the PhD program, she taught me how to think deeply and rigorously about research—lessons I continue to carry with me.

I feel fortunate to have been part of the Berkeley Artificial Intelligence Research (BAIR) Lab—a vibrant, intellectually rich environment that continually challenged and inspired me. I've learned so much from conversations and collaborations with many brilliant researchers, including Tony Long Lian, Baifeng Shi, Amir Bar, Boyi Li, David Chan, Jiaxin Ge, Stephanie Fu, Zhuang Liu, Max Fu, Renhao Wang, Roi Herzig, Dantong Niu, Devin Guillory, Yutong Bai, Jerome Quenum, Grace Luo, Junyi Zhang, Haven Haiwen Feng, Lisa Dunlap, Songwei Ge, Norman Mu, Ritwik Gupta, Rodolfo Corona, Sanjay Subramanian, Sheng Shen, Colorado Reed, Tsung-Han (Patrick) Wu, Suzie Petryk, Tete Xiao, Allan Jabri, Tyler Bonnen, Qianqian Wang, Chun Ming Kim, Tianjun Zhang, Ziyang Wu, Ruilong Li, Jiachen Lian, Justin Kerr, Yuexiang Zhai, Haozhi Qi, Konstantinos Kallidromitis, Haoran Geng, Ilija Radosavovic, Yi Ma, Sewon Min, Alane Suhr, Alexei A. Efros, Ken Goldberg, Angjoo Kanazawa—and many others.

At the vision group of ICSI, I was fortunate to work with and learn from a group of talented researchers and friends: Zhongqi Miao, Tsung-Wei Ke, Jyh-Jing Hwang, Yunhui Guo, Ziwei Liu, Ke Wang, Yubei Chen, Qian Yu, Chun-Hsiao (Daniel) Yeh, Baladitya Yellapragada, Zhihang Ren, Utkarsh Singhal, Sascha Hornauer and many others.

During my internships at Meta FAIR (New York office), Meta GenAI (Seattle office), and Google DeepMind (Mountain View office), I met many wonderful collaborators and mentors, including Xingyi Zhou, Alireza Fathi, Rohit Girdhar, Yue Zhao, Sherry Zihui Xue, Kaiwen Kevin Zha, Haozhe An, Zixuan Huang, Sachit Menon, Chen Wei, Huiyu Wang, Jiachen Zhu, Dinghuai Zhang, Mannat Singh, Andrew Brown, Quentin Duval, Samaneh Azadi, Sai Saketh Rambhatla, Xi Yin, Devi Parikh, Ian Huang, Lijun Yu, Xiuye Gu, David A. Ross, and Ming-Hsuan Yang.

I've also had the privilege of mentoring and collaborating with outstanding undergraduate students at Berkeley. Watching their growth has been one of the most fulfilling parts of my PhD. Thank you, Tony, Shufan, Ashish, Marlo, Jingfeng, Bomin, Ji, and Kehan.

To my parents, thank you for your boundless love, trust, and sacrifice. As the only child in our family, I know how much strength and courage it took for you to support my decision to pursue a PhD so far from home—over ten thousand kilometers away. Despite the distance and years apart, your unwavering belief in me has been my foundation. You have always stood behind me with quiet strength and unconditional love, never asking for anything in return. I carry your hopes and values with me in everything I do, and this dissertation is a testament to your selfless support. I am forever grateful.

To my wife, Ruolin Li, thank you for being my unwavering source of love, strength, and balance. We got married during the pandemic, in the midst of our PhD journeys at Berkeley, when the world felt uncertain and everything was in flux. As fellow PhD students, we could truly understand each other's highs and lows, celebrate each other's milestones, and offer support in the moments that felt overwhelming. Your empathy, encouragement, and presence made all the difference. You've been not only my partner, but my truest companion and soulmate on this long and winding road. This dissertation is as much yours as it is mine, and I look forward to a future filled with more laughter, and a lifetime of shared dreams—side by side.

Chapter 1

Introduction

Large language models (LLMs) [1]–[5], exemplified by GPTs, have fundamentally reshaped how we process and interact with information. Their breakthroughs are driven by two key factors: the vast availability of internet-scale data and the self-supervised next-token prediction paradigm — both operating without explicit human supervision. Yet, language is only a narrow projection of human intelligence. Achieving true artificial general intelligence (AGI) requires models to integrate richer sensory modalities — encompassing vision, physical interaction, spatial navigation, and social dynamics — that extend far beyond text alone. Among these, visual intelligence is essential: the ability to parse scenes, localize objects, and model compositional structure directly from raw pixels, enabling systems to perceive and reason about the visual world with minimal human intervention.

Unfortunately, while large language models have achieved impressive generalization by leveraging the compositionality and structure of language, extending these gains to the visual domain remains a major open challenge. Unlike text, visual data is continuous and high-dimensional, lacking inherent tokenization or semantic grammar. Moreover, supervised learning in vision heavily depends on large-scale labeled datasets, which are expensive to collect and often biased in content and distribution.

This dissertation tries to answer: ***Can we build general-purpose visual systems that learn from the natural structure of visual data—with minimal or even without human supervision?*** I aim to establish both theoretical foundations and practical self-supervised learning (SSL) approaches that process diverse real-world visual data and apply them to complex downstream tasks. By enhancing SSL techniques for multimodal models—*minimizing reliance on human-annotated datasets*—I strive to develop intelligent systems that can comprehend and interact with the environment in ways that mirror or even surpass human sensation, perception, and reaction.

Toward this goal, I explore a suite of techniques rooted in self-supervised learning to acquire object-centric, semantic, and compositional representations directly from unlabeled images and videos. I focus on exploiting visual inductive biases, leveraging synthetic signals for supervision, and closing the loop between discriminative and generative models under a unified SSL framework. All these contributions reflect a broader vision: ***visual intelligence should emerge from the structure of data—not from the structure of labels.*** This perspective challenges conventional su-

pervised paradigms and advocates for building vision systems that learn in a label-free, data-driven, and scalable manner.

1.1 Unsupervised Visual World Understanding

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

— Marcel Proust

My research addresses the long-standing challenge of learning directly from the inherent structure and patterns of the visual world without human supervision, through three key perspectives:

1) Self-supervised representation learning: we have developed a series of self-supervised learning algorithms [6]–[11] that extract rich and informative features from unlabeled data, eliminating the need for large annotated datasets. These methods uncover meaningful representations and structural regularities, providing a strong foundation for diverse downstream tasks. ***2) Segment anything without supervision.*** As a domain studied for decades, object localization and segmentation lies at the heart of visual understanding and are crucial for enabling AI systems to perceive, reason, and interact in an object-centric manner. However, existing approaches often rely on large-scale human-labeled datasets, which constrain their scalability and often introduces significant biases based on the annotators' perceptions of "what constitutes an instance". These challenges raise a crucial question: *Can we "segment anything" without supervision?* In response, we have introduced a series of research efforts–CutLER [12], U2Seg [13], VideoCutLER [14], and UnSAM [6]–that enables both interactive and whole-image segmentation without human supervision. Via automatically capturing the hierarchical structure of visual scenes with a "divide-and-conquer" strategy for parsing complex scenes, unsupervised UnSAM [6] delivers impressive results, rivaling the performance of the supervised SAM [15]—a foundation model trained on over 1.1 billion labeled masks—while relying only on self-supervised pseudo-masks. ***3) SSL for robust decision-making models in multimodal environments.*** In vision-based reinforcement learning (RL), domain transfer remains a challenge. We decouple vision from action, extracting universal visual features while maintaining RL policy intact, enabling reliable performance in dynamic environments [9].

1.2 Analysis-by-Synthesis: Generative Models for Visual Scene Understanding

"To understand is to invent."

— Jean Piaget

Visual scene understanding improves generative models' ability to interpret the visual world within a compact and informative feature space. Conversely, ***a model capable of generating realistic and meaningful content must have a deep understanding of the complex structures and patterns in its training data***. The body of research in Sec. 1.2 and Sec. 1.3 aims to bridge generative and discriminative models using synthetic data generation, allowing both models to complement and enhance each other, thereby enriching both the analysis and synthesis of contents and improving how AI systems interpret and interact with environments without human supervision.

A model capable of creating realistic and meaningful content must have a deep understanding of the complex structures and patterns within its training data. Motivated by this, in Sec. 1.2, we intend to answer can we learn good representations for discriminative tasks using generative models? The answer is Yes! We present *VisualLexicon (or ViLex)* [16], a new visual "language" learned from text-to-image diffusion models. Using a self-supervised learning pipeline, ViLex generates embeddings optimized for reconstructing input images through a frozen text-to-image (T2I) diffusion model, preserving the detailed information necessary for high-fidelity semantic-level reconstruction. As visual embeddings in the text space, ViLex embeddings can be used independently as "text" tokens or combined with natural language tokens for prompting diffusion models with both visual and textual cues for multimodal image generation. ViLex is also compatible with downstream vision-language tasks like visual question answering, image captioning, and referring expression segmentation, significantly enhancing performance. After training, ViLex can be directly used as a new "language", *i.e.*, the "text-prompt", to a frozen text encoder like CLIP [17], enabling the re-creation of semantically similar images without the need for actual text prompts.

We also introduce LLM-based segmentation models, including *SESAME*[18], *HIPIE*[19], and *SegLLM* [20]. In particular, *SegLLM* reformulates visual understanding as a next-token prediction problem, enabling segmentation models to engage in natural language dialogue and reason about complex user intentions.

1.3 Analysis-for-Synthesis: Understanding Models for Visual Scene Generation

"Without craftsmanship, inspiration is a mere reed shaken in the wind."
— Johannes Brahms

Generative models learn by simulating the world — but to simulate it accurately, they must first understand it. I introduce *InstanceDiffusion* [21], which injects fine-grained, object-level control into text-to-image diffusion models using machine-generated annotations from perception models as pseudo-labels. By pairing visual structure with natural language prompts, *InstanceDiffusion* enables precise manipulation of instance attributes, layouts, and semantics — all without manual supervision.

While most generative models synthesize pixels from abstract embeddings, structured generation demands models capable of interpreting, controlling, and manipulating scenes at the object

level. To this end, I develop models that produce controllable visual outputs conditioned on layout, attributes, and spatial constraints, paving the way for grounded generation and interactive design. Architectural innovations such as *UniFusion* and *ScaleU* allow flexible integration of diverse location formats — including points, boxes, scribbles, and masks — into the generation pipeline. The result is a set of systems that not only produce photorealistic images but also faithfully capture user intent with high compositional consistency.

1.4 Self-supervised Debiased Learning

"Everything we hear is an opinion, not a fact. Everything we see is a perspective, not the truth."

— Marcus Aurelius

While self-supervised learning promises label-free scalability, it often inherits and amplifies the biases embedded in real-world data. Such biases arise not only from imbalanced datasets, but also from the intrinsic similarity structure of visual representations — causing models to favor certain predictions even when trained on balanced inputs. This leads to shortcut learning, spurious correlations, and poor generalization under distribution shift.

My series of research for debiasing deep learning models is based on the insight that ***model predictions are naturally imbalanced due to intrinsic data similarity, even when a model is trained on balanced source data.*** I investigate self-supervised approaches to robustify learning from imperfect data. I explore this across three settings:

1) Class-imbalanced learning: To address the challenge of bias in models trained on imbalanced data, our RIDE [22] takes a dynamic view of training data and provides a principled bias-variance analysis as the data distribution fluctuates. We found that existing classifiers suffer from increased variance and confusion in tail classes, even when designed for long-tailed distributions. We found that existing classifiers, even when designed for long-tailed distributions, suffer from increased variance, and the bias gap between head and tail classes remains large, primarily due to increased confusion with hard negatives for the tail classes. Motivated by this observation, we propose Routing Diverse Experts (RIDE) [22]. RIDE reduces model variance by employing multiple experts, mitigates bias with a distribution-aware diversity loss, and lowers computational costs using a dynamic expert routing module, without relying on manual reweighting; ***2) Out-of-domain generalization:*** Although it is well-known that a model trained on biased data has biased predictions, surprisingly, our DebiasPL [10] found that even CLIP's [17] predictions are significantly biased, despite both source and target data being class-balanced. Based on this observation, we study the issues of naturally imbalanced pseudo-labels, which are confident predictions made on unlabeled target data by a classifier trained on labeled source data. Our [10] proposes an adaptive debiasing module with **counterfactual reasoning** and an adaptive marginal loss, aiming at dynamically alleviating biased pseudo labels' influence on a student model-without leveraging any prior knowledge on marginal class distribution; ***3) Visual policy learning:*** Current RL models are

highly sensitive to unexpected environmental changes, even if those changes are irrelevant to the learning task, *i.e.*, distractions in the background. To address this, we propose unsupervised Visual Attention and Invariance (VAI) [9] for RL. VAI works by analyzing action videos from a training environment and extracting foregrounds through unsupervised keypoint detection, followed by unsupervised visual attention that generates a foreground mask for each video frame. During testing, the learned model is used to provide distraction-free visual input to the RL policy learner. VAI not only learns domain-invariant vision without supervision but also removes visual distractions, enabling the RL policy to be more focused and significantly more effective.

These contributions share a common theme: ***debiasing should emerge from the model's understanding of why a prediction is made — not just what the prediction is.*** By embedding causal reasoning into self-supervised learning, I enable models to overcome the limitations of their training data and generalize more reliably to unseen environments and tasks.

1.5 Thesis Organization

The remainder of the dissertation is organized as follows:

- **Part I** focuses my body of research on unsupervised visual world understanding, proposing scalable pipelines for extracting instance, semantic, and panoptic structures from static and dynamic scenes.
- **Part II** explores analysis-by-synthesis approaches for building structured representations using generative models aligned with visual semantics.
- **Part III** investigates analysis-for-synthesis with structure—developing controllable image generation models conditioned on fine-grained object-level prompts, produced by perception models.
- **Part IV** presents my research efforts for robust self-supervised learning under data imbalance, noisy labels, and domain shifts.
- **Part V** concludes by summarizing the core insights and proposing future directions in extending visual self-supervision to broader multimodal and embodied intelligence systems.

In closing, this thesis contends that ***visual intelligence should arise from the intrinsic structure of data, rather than the imposed structure of labels.*** It advances the foundations of self-supervised visual learning through the design of scalable, compositional, and robust systems capable of learning directly from raw, unlabeled data. Collectively, these contributions establish a foundation for general-purpose visual intelligence — enabling both perception and interaction — that is vital for autonomous agents operating in the open world.

Part I

Unsupervised Visual World Understanding

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

— Marcel Proust

Chapter 2

CutLER: Cut and Learn for Unsupervised Instance Segmentation

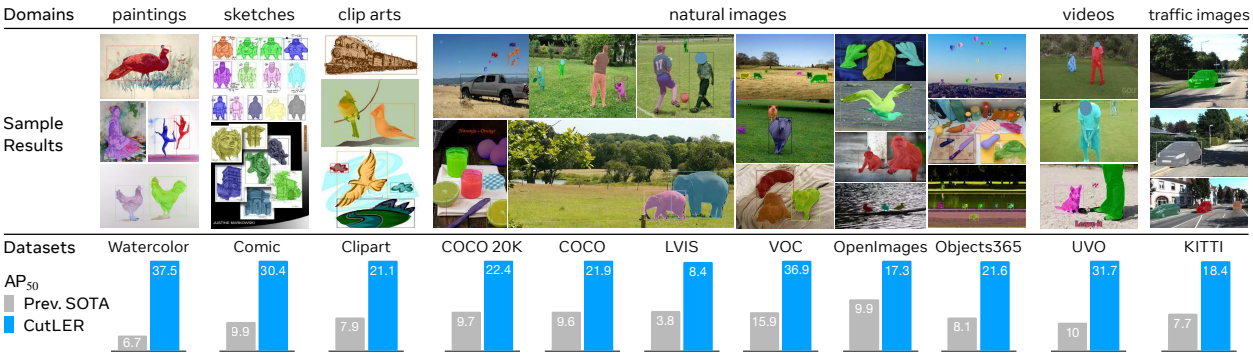


Figure 2.1: Zero-shot unsupervised object detection and instance segmentation using our CutLER model, which is trained without human supervision. We evaluate the model using the standard detection AP^{box}_{50}. CutLER gives a strong performance on a variety of benchmarks spanning diverse image domains - video frames, paintings, clip arts, complex scenes, *etc*. Compared to the previous state-of-the-art method, FreeSOLO [23] with a backbone of ResNet101, CutLER with a backbone of ResNet50 provides strong gains on all benchmarks, increasing performance by more than $2\times$ on 10 of the 11 benchmarks. We evaluate [23] with its official code and checkpoint.

We propose **Cut**-and-**LE**a**R**n (CutLER), a simple approach for training unsupervised object detection and segmentation models. We leverage the property of self-supervised models to 'discover' objects without supervision and amplify it to train a state-of-the-art localization model without any human labels. CutLER first uses our proposed MaskCut approach to generate coarse masks for multiple objects in an image, and then learns a detector on these masks using our robust loss function. We further improve performance by self-training the model on its predictions. Compared to prior work, CutLER is simpler, compatible with different detection architectures, and detects multiple objects. CutLER is also a zero-shot unsupervised detector and improves detection perfor-

mance AP_{50} by over $2.7\times$ on 11 benchmarks across domains like video frames, paintings, sketches, *etc*With finetuning, CutLER serves as a low-shot detector surpassing MoCo-v2 by 7.3% AP^{box} and 6.6% AP^{mask} on COCO when training with 5% labels.

2.1 Introduction

Object localization is a critical task in computer vision that enables AI systems to perceive, reason, plan and act in an object-centric manner. Training models for localization require special annotations like object boxes, masks, localized points, *etc*which are both difficult and resource intensive to collect. Without accounting for overhead, annotating $\sim$164K images in the COCO dataset [24] with masks for just 80 classes took more than 28K human hours of annotation time. In this work, we study unsupervised object detection and instance segmentation models that can be trained without any human labels. Our key insight is that simple probing and training mechanisms can amplify the innate localization ability of self-supervised models [25], leading to state-of-the-art unsupervised zero-shot detectors.

Our method **Cut**-and-**LE**a**R**n (CutLER) consists of three simple, architecture- and data-agnostic mechanisms. Consistent with prior self-supervised learning methods [25]–[28], CutLER is trained exclusively on unlabeled ImageNet data without needing additional training data, but contrary to these methods, CutLER can be directly employed to perform complex segmentation and detection tasks over a wide range of domains. *First*, we propose MaskCut that can automatically produce *multiple* initial coarse masks for each image, using the pretrained self-supervised features. *Second*, we propose a simple loss dropping strategy to train detectors using the coarse masks while being robust to objects missed by MaskCut. *Finally*, we observe that despite learning from these coarse masks, the detectors 'clean' the ground truth and produce masks (and boxes) that are better than the coarse masks used to train them. Therefore, we further show that multiple rounds of self-training on the models' own predictions allow it to evolve from capturing the similarity of local pixels to capturing the global geometry of the object, thus producing finer segmentation masks.

Prior work shows that a self-supervised vision transformer (ViT) [29] can automatically learn patch-wise features that detect a single *salient* object in an image [25], [30]–[33]. However, unlike CutLER, such salient object detection methods only locate a single, usually the most prominent, object and cannot be used for real world images containing multiple objects. While some recent methods, *e.g.*, FreeSOLO [23] and DETReg [34], also aim at unsupervised multi-object detection (or multi-object discovery), they rely on a particular detection architecture, *e.g.*, SOLO-v2 [35] or DDETR [36], [37]. Additionally, apart from self-supervised features trained on ImageNet [38], the current state-of-the-art methods FreeSOLO and MaskDistill [39] also require 'in-domain' unlabeled data for model training.

In contrast, CutLER works with various detection architectures and can be trained solely on ImageNet, without requiring in-domain unlabeled data. Thus, during model training, CutLER does not see any images from any target dataset and yields a zero-shot model capable of detecting and segmenting multiple objects in diverse domains.

	DINO	LOST	TokenCut	FreeSOLO	Ours
detect multiple objects	✗	✓	✗	✓	✓
zero-shot detector	✓	✗	✓	✗	✓
compatible with various detection architectures	-	✓	-	✗	✓
pretrained model for supervised detection	✓	✗	✗	✓	✓

Table 2.1: We compare previous methods on unsupervised object detection, including DINO [25], LOST [32], TokenCut [33] and FreeSOLO [23], with our CutLER in term of key properties. Our CutLER is the only method with all these desired properties.

Features of CutLER. ***1) Simplicity:*** CutLER is simple to train and agnostic to the choice of detection and backbone architectures. Thus, it can be integrated effortlessly into existing object detection and instance segmentation works. ***2) Zero-shot detector:*** CutLER trained solely on ImageNet shows strong zero-shot performance on 11 different benchmarks where it outperforms prior work trained with additional in-domain data. We **double the** $\mathbf{AP}^{\mathbf{box}}_{50}$ performance on 10 of these benchmarks, as shown in Fig. 2.1, and even outperform supervised detectors on the UVO video instance segmentation benchmark. ***3) Robustness:*** CutLER exhibits strong robustness against domain shifts when tested on images from different domains such as video frames, sketches, paintings, clip arts, *etc*. ***4) Pretraining for supervised detection:*** CutLER can also serve as a pretrained model for training fully supervised object detection and instance segmentation models and improves performance on COCO, including on few-shot object detection benchmarks.

2.2 Related Work

Self-supervised feature learning involves inferring the patterns within the large-scale unlabeled data without using human-annotated labels. *Contrastive learning based* [26], [28], [40], [41] methods learn such representations that similar samples or various augmentations of the same instance are close to each other, while dissimilar instances are far apart. *Similarity-based self-supervised learning* methods [42], [43] learn representations via minimizing the distance between different augmentations of the same instance and use only positive sample pairs. *Clustering-based feature learning* [9], [44]–[46] automatically discovers the natural grouping of data in the latent representation space. Recently, [47], [48] have shown that *masked autoencoders*, which learn representations via masking out a large random subset of image patches and reconstructing the missing pixels or patches [47]–[50], are scalable self-supervised learners for computer vision [47].

In contrast to these unsupervised representation learning efforts, our work aims to automatically discover natural pixel groupings and locate instances within each image.

Unsupervised object detection and instance segmentation. The main comparisons to previous works are listed in Table 2.1 and are elaborated as follows:

DINO [25] observes that the underlying semantic segmentation of images can emerge from

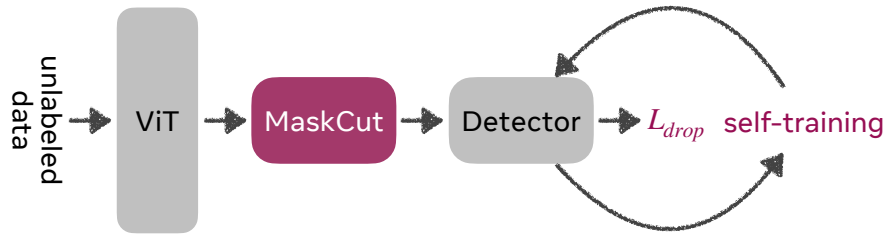


Figure 2.2: Overview of CutLER. We propose a simple yet effective method to train an object detection and instance segmentation model without using any supervision. We first propose MaskCut to extract initial coarse masks from the features of a self-supervised ViT. We then learn a detector using our loss dropping strategy that is robust to objects missed by MaskCut. We further improve the model using multiple rounds of self-training.

the self-supervised Vision Transformer (ViT) [29], which does not appear explicitly in either supervised ViT or ConvNets [25], [51]. Based on this observation, LOST [32] and TokenCut [33] leverage self-supervised ViT features and propose to segment *one single* salient object [32], [33], [52] from each image based on a graph that is constructed with DINO's patch features.

These previous works either can not detect more than one object from each image, *e.g.*, DINO and TokenCut, or can not improve the quality of features for better transfer to downstream detection and segmentation tasks, *e.g.*, TokenCut and LOST. Unlike these works, CutLER can locate multiple objects and serve as a pretrained model for label-efficient and fully-supervised learning.

FreeSOLO [23] achieves unsupervised instance segmentation by extracting coarse object masks in an unsupervised manner, followed by mask refinement through a self-training procedure. While FreeSOLO's FreeMask stage can generate multiple coarse masks per image, the quality of these masks is often rather low [23]. MaskDistill [39] distills class-agnostic initial masks from the affinity graph produced by a self-supervised DINO [25]. However, it utilizes *one single* mask per image in the distillation stage, which greatly limits the model's ability to detect multiple objects.

By contrast, the initial masks generated by our MaskCut are usually better in quality and quantity than the initial masks used by [23], [39]. Therefore, CutLER achieves $2\times\sim4\times$ higher AP^{box} and AP^{mask} than FreeSOLO [23] and MaskDistill [39] on almost all experimented detection and segmentation benchmarks, even when FreeSOLO and MaskDistill are trained and tested on the same domain.

2.3 Method

We tackle the problem of unsupervised object detection and segmentation with a simple cut-and-learn pipeline. Our method builds upon insights from recent work [25], [33], showing that self-supervised representations can discover objects. While these methods often find a single object per image, we propose a simple approach that can discover multiple objects and significantly improves segmentation and detection performance. The overview of our cut-and-learn pipeline is illustrated

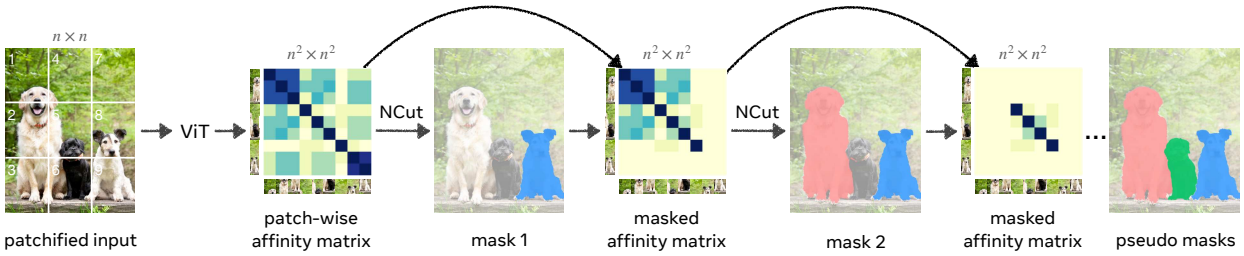


Figure 2.3: MaskCut can discover multiple object masks in an image without supervision. We build upon [25], [33] and create a patch-wise similarity matrix for the image using a self-supervised DINO [25] model's features. We apply Normalized Cuts [53] to this matrix and obtain a single foreground object mask of the image. We then mask out the affinity matrix values using the foreground mask and repeat the process, which allows MaskCut to discover multiple object masks in a single image. In this pipeline illustration, we set $n=3$.

in Fig. 2.2. *First*, we propose MaskCut that generates multiple binary masks per image using self-supervised features from DINO [25] (Sec. 2.3). *Second*, we show a dynamic loss dropping strategy, called DropLoss, that can learn a detector from MaskCut's initial masks while encouraging the model to explore objects missed by MaskCut (Sec. 2.3); *Third*, we further improve the performance of our method through multiple rounds of self-training (Sec. 2.3).

Preliminaries

Normalized Cuts (NCut) treats the image segmentation problem as a graph partitioning task [53]. We construct a fully connected undirected graph via representing each image as a node. Each pair of nodes is connected by edges with weights W_{ij} that measure the similarity of the connected nodes. NCut minimizes the cost of partitioning the graph into two sub-graphs, *i.e.*, a bipartition, by solving a generalized eigenvalue system

$$(D-W)x = \lambda Dx \tag{2.1}$$

for finding the eigenvector x that corresponds to the second smallest eigenvalue λ, where D is a $N \times N$ diagonal matrix with $d(i) = \sum_j W_{ij}$ and W is a $N \times N$ symmetrical matrix.

DINO and TokenCut. DINO [25] finds that the self-supervised ViT can automatically learn a certain degree of perceptual grouping of image patches. TokenCut [33] leverages the DINO features for NCut and obtaining foreground/background segments in an image. The authors use the similarity of the patches in the DINO feature space as the similarity weight W_{ij} in NCut. Specifically, following multiple recent methods [32], [33], [39], we use the cosine similarity of 'key' features from the last attention layer of DINO-pretrained model, *i.e.*, $W_{ij} = \frac{K_i K_j}{\|K_i\|_2 \|K_j\|_2}$ where K_i is the 'key' feature of patch i, and solve Eq. (2.1) for finding the second smallest eigenvector x.

A limitation of TokenCut is that it only computes a single binary mask for an image and thus only finds one object per image. Although we can use the other N–2 smallest eigenvectors to locate

more than one instance, this significantly degrades the performance for multi-object discovery, as demonstrated in Sec. 2.5.

MaskCut for Discovering Multiple Objects

As we discussed in Sec. 2.3, vanilla NCut is limited to discovering a single object in an image. We propose MaskCut that extends NCut to discover multiple objects per image by iteratively applying NCut to a *masked* similarity matrix (illustrated in Fig. 2.3). After getting the bipartition x^t from NCut at stage t, we get two disjoint groups of patches and construct a binary mask M^t, where

$$M^t_{ij} = \begin{cases} 1, & \text{if } M^t_{ij} \geq \text{mean}(x^t) \\ 0, & \text{otherwise.} \end{cases} \tag{2.2}$$

To determine which group corresponds to the foreground, we make use of two criteria: *1)* intuitively, the foreground patches should be more prominent than background patches [25], [30], [33]. Therefore, the foreground mask should contain the patch corresponding to the maximum *absolute* value in the second smallest eigenvector M^t; *2)* we incorporate a simple but empirically effective object-centric prior [54]: the foreground set should contain less than two of the four corners. We reverse the partitioning of the foreground and background, *i.e.*, $M^t_{ij} = 1 - M^t_{ij}$, if the criteria 1 is not satisfied while the current foreground set contains two corners or the criteria 2 is not satisfied. In practice, we also set all $W_{ij} < \tau^{\text{ncut}}$ to $1e^{-5}$ and $W_{ij} \geq \tau^{\text{ncut}}$ to 1.

To get a mask for the $(t+1)^{\text{th}}$ object, we update the node similarity W^{t+1}_{ij} via masking out these nodes corresponding to the foreground in previous stages:

$$W^{t+1}_{ij} = \frac{(K_i \prod_{s=1}^{t} \hat{M}^s_{ij})(K_j \prod_{s=1}^{t} \hat{M}^s_{ij})}{\|K_i\|_2 \|K_j\|_2} \tag{2.3}$$

where $\hat{M}^s_{ij} = 1 - M^s_{ij}$. Using the updated W^{t+1}_{ij}, we repeat Eqs. (2.1) and (2.2) to get a mask M^{t+1}. We repeat this process t times and set $t = 3$ by default.

DropLoss for Exploring Image Regions

A standard detection loss penalizes predicted regions r_i that do not overlap with the 'ground-truth'. Since the 'ground-truth' masks given by MaskCut may miss instances, the standard loss does not enable the detector to discover new instances not labeled in the 'ground-truth'. Therefore, we propose to ignore the loss of predicted regions r_i that have a small overlap with the 'ground-truth'. More specifically, during training, we drop the loss for each predicted region r_i that has a maximum overlap of τ^{IoU} with any of the 'ground-truth' instances:

$$\mathcal{L}_{\text{drop}}(r_i) = \mathbb{1}(\text{IoU}^{\text{max}}_i > \tau^{\text{IoU}}) \mathcal{L}_{\text{vanilla}}(r_i) \tag{2.4}$$

where $\text{IoU}^{\text{max}}_i$ denotes the maximum IoU with all 'ground-truth' for r_i and $\mathcal{L}_{\text{vanilla}}$ refers to the vanilla loss function of detectors. $\mathcal{L}_{\text{drop}}$ does not penalize the model for detecting objects missed in the 'ground-truth' and thus encourages the exploration of different image regions. In practice, we use a low threshold $\tau^{\text{IoU}} = 0.01$.

Multi-Round Self-Training

Empirically, we find that despite learning from the coarse masks obtained by MaskCut, detection models 'clean' the ground truth and produce masks (and boxes) that are better than the initial coarse masks used for training. The detectors refine mask quality, and our DropLoss strategy encourages them to discover new object masks. Thus, we leverage this property and use multiple rounds of self-training to improve the detector's performance.

We use the predicted masks and proposals with a confidence score over $0.75{-}0.5t$ from the t^{th}-round as the additional pseudo annotations for the $(t+1)^{\text{th}}$-round of self-training. To de-duplicate the predictions and the ground truth from round t, we filter out ground-truth masks with an IoU > 0.5 with the predicted masks. We found that three rounds of self-training are sufficient to obtain good performance. Each round steadily increases the number of 'ground-truth' samples used to train the model.

Implementation Details

Training data. We only use the images from the ImageNet [38] dataset (1.3 million images) for all parts of the CutLER model and do not use any type of annotations either for training or any supervised pretrained models.

MaskCut. We use MaskCut with three stages on images resized to 480×480 pixels and compute a patch-wise affinity matrix using the ViT-B/8 [29] DINO [25] model. We use Conditional Random Field (CRF) [55] to post-process the masks and compute their bounding boxes.

Detector. While CutLER is agnostic to the underlying detector, we use popular Mask R-CNN [56] and Cascade Mask R-CNN [57] for all experiments, and use Cascade Mask R-CNN by default, unless otherwise noted. We train the detector on ImageNet with initial masks and bounding boxes for 160K iterations with a batch size of 16. When training the detectors with a ResNet-50 backbone [58], we initialize the model with the weights of a self-supervised pretrained DINO [25] model. We explored other pre-trained models, including MoCo-v2 [27], SwAV [46], and CLD [9], and found that they gave similar detection performance. We also leverage the copy-paste augmentation [59], [60] during the model training process. Rather than using the vanilla copy-paste augmentation, to improve the model's ability to segment small objects, we randomly downsample the mask with a scalar uniformly sampled between 0.3 and 1.0. We then optimize the detector for 160K iterations using SGD with a learning rate of 0.005, which is decreased by 5 after 80K iterations, and a batch size of 16. We apply a weight decay of 5×10^{-5} and a momentum of 0.9.

Self-training. We initialize the detection model in each stage using the weights from the previous stage. We optimize the detector using SGD with a learning rate of 0.01 for $80K$ iterations. Since the self-training stage can provide a sufficient number of pseudo-masks for model training, we don't use the DropLoss during the self-training stages.

We provide more details on model implementation and training in Sec. 2.6.

Datasets →	Avg.		COCO		COCO20K		VOC		LVIS		UVO		Clipart		Comic		Watercolor		KITTI		Objects365		OpenImages	
Metrics →	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR
Prev. SOTA [23]	9.0	13.4	9.6	12.6	9.7	12.6	15.9	21.3	3.8	6.4	10.0	14.2	7.9	15.1	9.9	16.3	6.7	16.2	7.7	7.1	8.1	10.2	9.9	14.9
CutLER	24.3	35.5	21.9	32.7	22.4	33.1	36.9	44.3	8.4	21.8	31.7	42.8	21.1	41.3	30.4	38.6	37.5	44.6	18.4	27.5	21.6	34.2	17.3	29.6
vs. SOTA	+15.3	+22.1	+12.3	+20.1	+12.7	+20.5	+21.0	+23.0	+4.6	+15.4	+21.7	+28.6	+13.2	+26.2	+20.5	+22.3	+30.8	+28.4	+10.7	+20.4	+13.5	+24.0	+7.4	+14.7

Table 2.2: State-of-the-art **zero-shot unsupervised object detection** performance on 11 different datasets spanning a variety of domains. We report class-agnostic multi-object detection performance and the averaged results for 11 datasets using AP_{50}^{box} and AR_{100}^{box}. Our CutLER is trained in an unsupervised manner solely on ImageNet. While the previous SOTA method [23] is typically fine-tuned on extra data, *e.g.*, ∼241k unlabeled COCO images, CutLER significantly outperforms it. Results of [23] are produced with official code and checkpoint.

Methods	Pretrain	Detector	Init.	COCO 20K						COCO val2017					
				AP_{50}^{box}	AP_{75}^{box}	AP^{box}	AP_{50}^{mask}	AP_{75}^{mask}	AP^{mask}	AP_{50}^{box}	AP_{75}^{box}	AP^{box}	AP_{50}^{mask}	AP_{75}^{mask}	AP^{mask}
non zero-shot methods															
LOST [32]	IN+COCO	FRCNN	DINO	-	-	-	2.4	1.0	1.1	-	-	-	-	-	-
MaskDistill [39]	IN+COCO	MRCNN	MoCo	-	-	-	6.8	2.1	2.9	-	-	-	-	-	-
FreeSOLO* [23]	IN+COCO	SOLOv2	DenseCL	9.7	3.2	4.1	9.7	3.4	4.3	9.6	3.1	4.2	9.4	3.3	4.3
zero-shot methods															
DETReg [34]	IN	DDETR	SwAV	-	-	-	-	-	-	3.1	0.6	1.0	8.8	1.9	3.3
DINO [25]	IN	-	DINO	1.7	0.1	0.3	-	-	-	-	-	-	-	-	-
TokenCut [33]	IN	-	DINO	-	-	-	-	-	-	5.8	2.8	3.0	4.8	1.9	2.4
CutLER (ours)	IN	MRCNN	DINO	21.8	11.1	10.1	18.6	9.0	8.0	21.3	11.1	10.2	18.0	8.9	7.9
CutLER (ours)	IN	Cascade	DINO	22.4	12.5	11.9	19.6	10.0	9.2	21.9	11.8	12.3	18.9	9.7	9.2
vs. SOTA				**+12.7**	**+9.3**	**+7.8**	**+9.9**	**+6.6**	**+4.9**	**+12.3**	**+8.7**	**+8.1**	**+9.5**	**+6.4**	**+4.9**

Table 2.3: Unsupervised object detection and instance segmentation on COCO `20K` and COCO `val2017`. We report the detection and segmentation metrics and note the pretraining data (Pretrain), detectors, and backbone initialization (Init.). Methods in the top half of the table train on extra unlabeled images from the downstream datasets, while zero-shot methods in the bottom half only train on ImageNet. Despite using an older detector, CutLER outperforms all prior works on all evaluation metrics. *: results obtained with the official code and checkpoint. IN, Cascade, MRCNN, and FRCNN denote ImageNet, Cascade Mask R-CNN, Mask R-CNN, and Faster R-CNN, respectively.

2.4 Experiments

We evaluate CutLER on various detection and segmentation benchmarks. In Sec. 2.4, we show that CutLER can discover objects without any supervision on completely unseen images. Despite being evaluated in a zero-shot manner on eleven benchmarks, CutLER outperforms prior methods that use in-domain training data. Sec. 2.4 shows that finetuning CutLER further improves detection performance, outperforming prior work like MoCo-V2 and FreeSOLO.

Unsupervised Zero-shot Evaluations

We conduct extensive experiments on eleven different datasets, covering various object categories, image styles, video frames, resolutions, camera angles, *etc*to verify the effectiveness of CutLER

as a universal unsupervised object detection and segmentation method. We describe the different datasets used for zero-shot evaluation in detail in Sec. 2.6. CutLER is trained solely using images from ImageNet and evaluated in a zero-shot manner on all downstream datasets without finetuning on any labels or data.

Evaluating unsupervised object detectors poses two unique challenges. *First*, since the model is trained without any notion of semantic classes, it cannot be evaluated using the class-aware detection setup. Thus, like prior work [32], [34], [35] we use the class-agnostic detection evaluation. *Second*, object detection datasets often only annotate a subset of the objects in the images. For example, while COCO and LVIS use the same images, COCO only labels 80 object classes, and LVIS labels 1203 object classes. In this partially labeled setup, Average Recall (AR) is a valuable metric for unsupervised detection as it does not penalize the models for detecting novel objects unlabeled in the dataset. Thus, we additionally report AR for all datasets.

Zero-shot detection on 11 benchmarks. We evaluate CutLER on a variety of datasets and report the detection performance using AP^{box}_{50} and AR^{box}_{100} metrics in Fig. 2.1 and Table 2.2. CutLER uses a *smaller* model size and *less* training data than prior work. Compared to the previous SOTA approach, FreeSOLO [23] with a backbone of ResNet101, CutLER, with the smaller ResNet50 backbone, significantly outperforms it in each of these benchmarks spanning various image distributions, more than doubling performance on 10 of them. Also note that, FreeSOLO requires FreeMask pre-training using approximately 1.3M ImageNet images and model fine-tuning using additional data in test benchmarks.

We observe that on different domains, *e.g.*watercolor or frames from videos (UVO dataset), CutLER improves performance by over $4\times$ and $2\times$, respectively. Fig. 2.1 shows some qualitative examples of CutLER's predictions.

Detailed comparisons on COCO20K and COCO. Table 2.3 presents detailed detection and segmentation evaluations (also referred to as 'multi-object' discovery) on two popular benchmarks: COCO `val2017` [24] and COCO `20K`, which contains a subset of 20K images of COCO [23], [32]. CutLER consistently surpasses prior works by a large margin (often gets $2\sim3\times$ higher AP) on both the segmentation and detection tasks. Although CutLER is not trained on any images from COCO, it surpasses existing methods trained on COCO by more than 10% in terms of AP^{mask}_{50} and AP^{box}_{50}.

Fig. 2.4 shows the qualitative comparisons between [23] and our CutLER on COCO `val2017`, along with human annotations. Surprisingly, *CutLER can often detect novel instances that human annotators miss*.

We present detailed comparisons on COCO `20K`, COCO `val2017` and LVIS [61] benchmarks in Sec. 2.6.

Detailed comparisons on UVO and VOC. For a comprehensive comparison with existing unsupervised multi-object detection methods, we report the results for UVO `val` [66] and VOC `trainval07` [67]. Table 2.4 shows that CutLER yields significant performance gains over previous SOTA, obtaining over $3\times$ higher AP, with the most considerable improvement coming from AP_L. On UVO, Table 2.5 shows that CutLER more than quadruples the AP of previous SOTA and almost triples the AP^{box}_{50}. *Our AP^{mask}_{50} is even 4.8% higher than the fully-supervised SOLOv2 [35]*

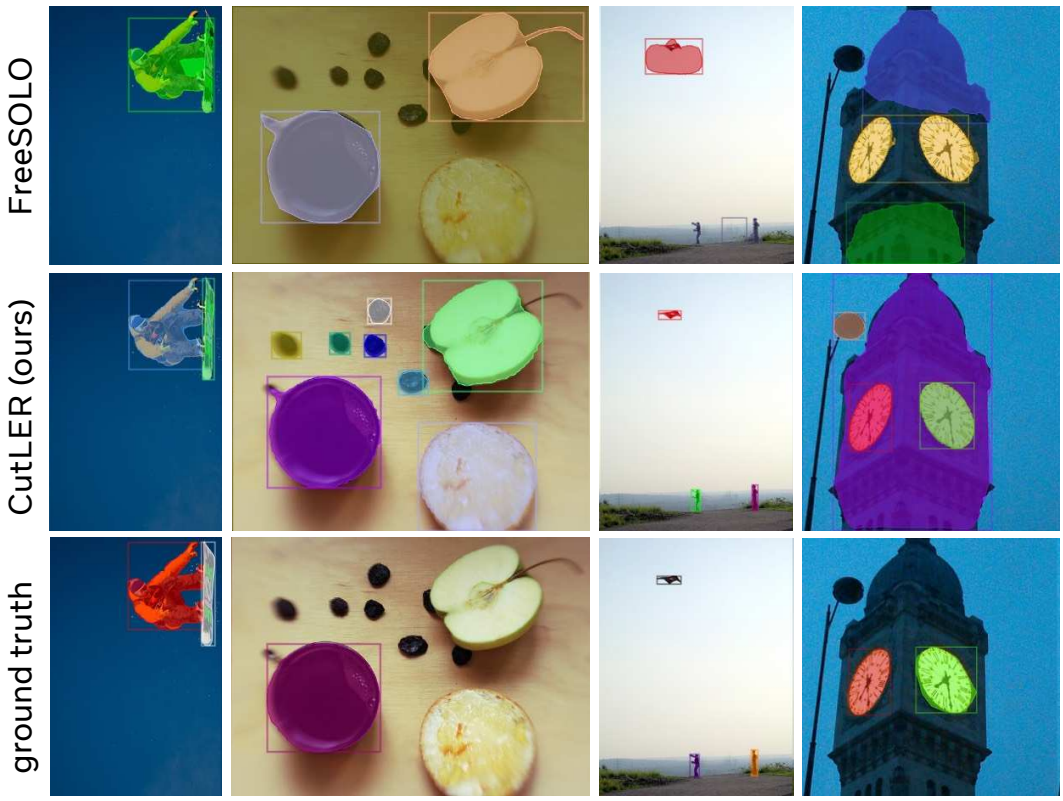


Figure 2.4: *Compared to the previous state-of-the-art* [23], our CutLER can better discriminate instances (*e.g.*person and skis in col. 1), discover more objects (*e.g.*apple and raisins in col. 2), and produce higher quality segmentation masks even for small objects (*e.g.*kite in col. 3); *compared to human annotations*, CutLER can locate novel instances that are overlooked by human annotators, such as the streetlight and clock tower in col. 4. **Qualitative comparisons** between previous SOTA methods (row 1) and our CutLER (row 2) on COCO, as well as ground truth annotations by human annotators (row 3), are visualized.

Methods	AP_{50}	AP_{75}	AP	AP_S	AP_M	AP_L
rOSD [30]	13.1	-	4.3	-	-	-
LOD [31]	13.9	-	4.5	-	-	-
LOST [32]	19.8	-	6.7	-	-	-
FreeSOLO* [23]	15.9	3.6	5.9	0.0	2.0	9.3
CutLER (ours)	36.9	19.2	20.2	1.3	6.5	32.2
vs. prev. SOTA	**+17.1**	**+15.6**	**+13.5**	**+1.3**	**+4.5**	**+22.9**

Table 2.4: Zero-shot unsupervised object detection on **VOC**. *: reproduced results with official code and checkpoint.

trained on LVIS with 100% annotations, significantly narrowing the gap between supervised and unsupervised learning.

Methods	AP^{box}_{50}	AP^{box}_{75}	AP^{box}	AP^{mask}_{50}	AP^{mask}_{75}	AP^{mask}
fully-supervised methods:						
SOLO-v2 (w/ COCO)[35]	-	-	-	38.0	20.9	21.4
Mask R-CNN (w/ COCO)[56]	-	-	-	31.0	14.2	15.9
SOLO-v2 (w/ LVIS)[35]	-	-	-	14.8	5.9	7.1
unsupervised methods:						
FreeSOLO* [23]	10.0	1.8	3.2	9.5	2.0	3.3
CutLER (ours)	31.7	14.1	16.1	22.8	8.0	10.1
vs. prev. SOTA	**+21.7**	**+12.3**	**+12.9**	**+13.3**	**+6.0**	**+6.8**

Table 2.5: Zero-shot unsupervised object detection and instance segmentation on the **UVO `val` video benchmark**. CutLER outperforms prior unsupervised methods and achieves better performance than the supervised SOLO-v2 model trained on the LVIS dataset. *: reproduced results with official code and checkpoint.

Label-Efficient and Fully-Supervised Learning

We now evaluate CutLER as a pretraining method for training object detection and instance segmentation models. While CutLER can discover objects without any supervision, finetuning it on a target dataset aligns the model output to the same set of objects labeled in the dataset.

Setup. We use CutLER to initialize a standard Cascade Mask R-CNN [57] detector with a ResNet50 [58]. Prior work uses more advanced detectors, SOLOv2 [35] used in [23] and DDETR [37] used in [34], that perform better. However, we choose Cascade Mask R-CNN for its simplicity and show in Sec. 2.5 that CutLER's performance improves with stronger detectors. We train the detector on the COCO [24] dataset using the bounding box and instance mask labels. To evaluate label efficiency, we subsample the training set to create subsets with varying proportions of labeled images. We train the detector, initialized with CutLER, on each of these subsets. As a baseline, we follow the settings from MoCo-v2 [27] and train the same detection architecture initialized with a MoCo-v2 ResNet50 model, given its strong performance on object detection tasks. Both MoCo-v2 and our models are trained for the $1\times$ schedule using Detectron2 [68], except for extremely low-shot settings with 1% or 2% labels. Following previous works [23], when training with 1% or 2% labels, we train both MoCo-v2 and our model for 3,600 iterations with a batch size of 16.

Results. Fig. 2.5 shows the results of fine-tuning the detector on different subsets of COCO. When tested with low-shot settings, *e.g.*, 2% and 5% labeled data, our approach achieves 5.4% and 7.3% higher AP^{box} than the MoCo-v2 baseline, respectively. Even when training with full annotations, CutLER still consistently gives more than 2% improvements, outperforming MoCo-v2 for both object detection and segmentation. More impressively, CutLER outperforms prior SOTA methods - FreeSOLO [23] and DETReg [34] despite using an older detection architecture.

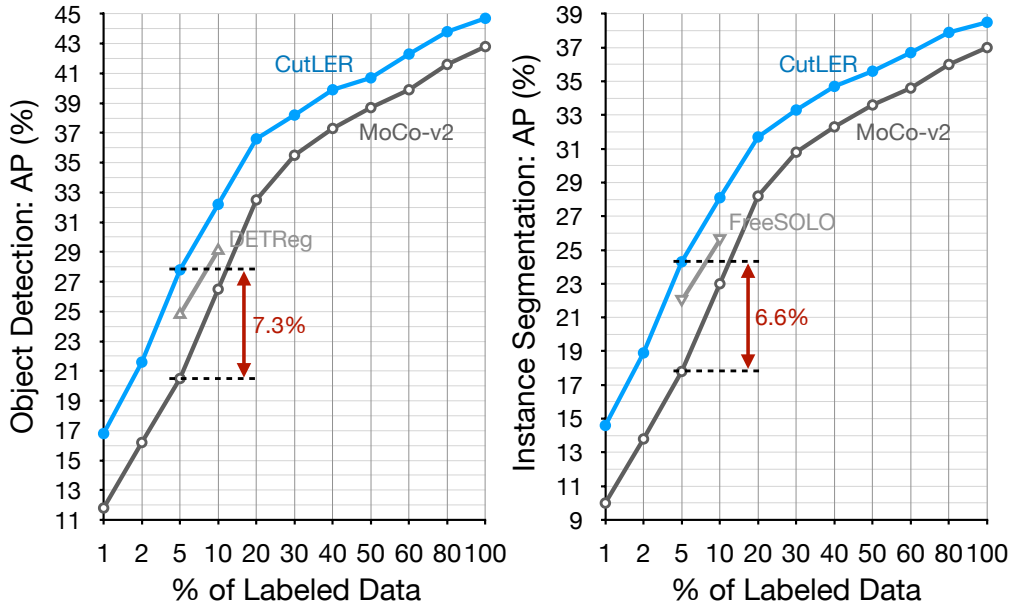


Figure 2.5: Finetuning CutLER for low-shot and fully supervised detection and instance segmentation. We fine-tune a Cascade Mask R-CNN model initialized with CutLER or MoCo-v2 on varying amounts of labeled data on the COCO dataset. We use the same schedule as the self-supervised pretrained MoCo-v2 counterpart and report the detection and instance segmentation performance. CutLER consistently outperforms the MoCo-v2 baseline: in the low-shot setting with 1% labels and the fully supervised setting using 100% labels. CutLER also outperforms FreeSOLO [23] and DETReg [34] on this benchmark despite using an older detection architecture. Results with Mask R-CNN are in the appendix.

2.5 Ablations

We analyze the design decisions in CutLER. We use similar settings to Sec. 2.4 and train CutLER only on ImageNet. We use the Cascade Mask R-CNN detection architecture and evaluate our model primarily on the COCO and UVO unsupervised detection benchmarks. All ablation studies are conducted without self-training unless otherwise noted.

Importance of each component. We analyze the main components of CutLER and report their relative contribution in Table 2.6. We report results on the popular COCO [24] dataset and a densely annotated video instance segmentation dataset UVO [66]. We also report the performance of running TokenCut [33] on the COCO dataset. Next, we use TokenCut's official codes to generate masks on ImageNet and use them for training a Cascade Mask R-CNN [57]. This base model provides substantial gains over just using TokenCut on COCO. We add each of our proposed components to this strong base model. Using MaskCut increases AP_{50}^{mask} and AP^{mask} by 4.7% and 2.7%, respectively. Also, the improvements to AP_{50}^{mask} is larger for densely annotated dataset UVO, *i.e.*4.7% vs. 2.7%. These results prove that MaskCut's ability to segment multiple instances per image is vital for densely annotated datasets. Adding DropLoss brings another 1.6% and 0.9% improvements to AP_{50}^{mask} for UVO and COCO, respectively. Multi-round of self-training increases

Methods	UVO		COCO	
	AP_{50}^{mask}	AP^{mask}	AP_{50}^{mask}	AP^{mask}
TokenCut [33]	-	-	4.9	2.0
Base	14.6	5.4	13.5	5.7
+ MaskCut	19.3	8.1	15.8	7.7
+ DropLoss	20.9	9.0	16.6	8.2
+ copy-paste [59], [60]	21.5	9.9	17.7	8.8
+ self-train (CutLER)	22.8	10.1	18.9	9.7

Table 2.6: Ablation study on the contribution of each component. Results reported on COCO and video segmentation dataset UVO.

Methods	AP_{50}^{box}	AP^{box}	AR_{100}^{box}	AP_{50}^{mask}	AP^{mask}	AR_{100}^{mask}
TokenCut (1 eigenvec.)	5.2	2.6	5.0	4.9	2.0	4.4
TokenCut (3 eigenvec.)	4.7	1.7	8.1	3.6	1.2	6.9
MaskCut ($t = 3$)	6.0	2.9	8.1	4.9	2.2	6.9
CutLER	21.9	12.3	32.7	18.9	9.7	27.1

Table 2.7: CutLER achieves much higher results even when compared to a modified TokenCut that can produce more than one mask per image. Compared to TokenCut, MaskCut gets a higher recall without reducing precision. We report results on COCO.

Size →	240	360	480	640
AP_{50}^{mask}	15.1	16.6	17.7	17.9

(a) Image size

τ^{ncut} →	0	0.1	0.15	0.2	0.3
AP_{50}^{mask}	17.1	17.5	17.7	17.6	17.5

(b) τ^{ncut}

N →	2	3	4
AP_{50}^{mask}	16.9	17.7	17.7

(c) # masks

τ^{IoU} →	0	0.01	0.1	0.2
AP_{50}^{mask}	17.4	17.7	14.4	12.7

(d) τ^{IoU}

Table 2.8: Ablations for MaskCut and DropLoss used for training CutLER. We report CutLER's detection and instance segmentation performance on COCO `val2017`, without adding the self-training stage. **(a)** We vary the size of the image used for MaskCut. **(b)** We vary the threshold τ^{ncut} in MaskCut, which controls the sparsity of the affinity matrix used for Normalized Cuts. **(c)** We vary the number of masks extracted using MaskCut and train different CutLER models. **(d)** We vary τ^{IoU} in DropLoss, *i.e.*, the maximum overlap between the predicted regions and the ground truth beyond which the loss for the predicted regions is ignored. Default settings are highlighted in gray.

the quantity and quality of pseudo-masks, leading to 1.3% improvements. These results show that each simple proposed component is critical for strong performance.

Comparison with TokenCut. TokenCut [33] is also a zero-shot segmentation method. However, it only segments a single instance per image, as discussed in Sec. 2.3. In order to generate more than one segmentation mask per image, we use a modified TokenCut by using more of the smaller eigenvectors and combining all produced masks. Table 2.7 shows the object detection performance on COCO's validation set for vanilla TokenCut, our modified TokenCut and CutLER. Although

	UVO			COCO		
	AP^{mask}_{50}	AP^{mask}	AP^{mask}_{75}	AP^{mask}_{50}	AP^{mask}	AP^{mask}_{75}
1 round	20.6	9.0	7.0	17.7	8.8	8.0
2 rounds	22.2	9.6	7.5	18.5	9.5	8.8
3 rounds	22.8	10.1	8.0	18.9	9.7	9.2
4 rounds	22.8	10.2	8.2	18.9	9.8	9.3

Table 2.9: Number of self-training rounds used in CutLER. We find that 3 rounds of self-training are sufficient. Self-training provides larger gains for the densely labeled UVO dataset.

	Mask R-CNN	Cascade Mask R-CNN	ViTDet
AP^{box}_{50}/ AP^{box}	20.3 / 10.6	20.8 / 11.5	21.5 / 11.8
AP^{mask}_{50}/ AP^{mask}	17.2 / 8.5	17.7 / 8.8	18.0 / 9.0

Table 2.10: CutLER with different detection architectures. We report results on COCO and observe that CutLER is agnostic to the detection architecture and improves performance using stronger detection architectures such as ViTDet with a backbone of ViT-B.

using more eigenvectors increases the recall AR^{box}_{100}, it significantly reduces the precision AP^{box}. CutLER not only improves the average recall AR^{box}_{100} by $4\times$ but also surpasses TokenCut's average precision AP^{box} by $4.8\times$, *i.e.*480% relative improvements.

Design choices in MaskCut and DropLoss and their impact on the final localization performance is presented in Table 2.8. We first study the effect of the image size used by MaskCut for generating the initial masks. As expected, Table 2.8a shows that MaskCut benefits from using higher resolution images presumably as it provides a higher resolution similarity between pixels. We pick a resolution of 480px for a better trade-off between the speed of MaskCut and its performance. In Table 2.8b, we study the effect of the threshold used in MaskCut for producing a binary W matrix (Sec. 2.3). Overall, CutLER seems to be robust to the threshold values. We understand the impact of the number of masks per image generated by MaskCut in Table 2.8c. Increasing the number improves the performance of the resulting CutLER models. This shows that MaskCut generates high-quality masks that directly impact the overall performance. Finally, in Table 2.8d, we vary the IOU threshold used for DropLoss. With a high threshold, we ignore the loss for a higher number of predicted regions while encouraging the model to explore. 0.01 works best for the trade-off between exploration and detection performance.

Self-training and its impact on the final performance is analyzed in Table 2.9. Self-training consistently improves performance across the UVO and COCO benchmarks and all metrics. UVO, which has dense object annotations, benefits more from the multi-round of self-training. By default, CutLER uses 3 rounds of self-training. Fig. 2.6 shows qualitative examples of how self-training improves both the quality of predictions and the number of objects predicted.

Figure 2.6: Multiple rounds of self-training can improve the pseudo-masks in terms of quality and quantity. We show **qualitative visualizations and the number of pseudo-masks** for all three rounds.

Pre-train	CutLER	AP^{box}_{50}	AP^{box}_{75}	AP^{box}	AP^{mask}_{50}	AP^{mask}_{75}	AP^{mask}
IN1K	IN1K	20.8	10.8	11.5	17.7	8.0	8.8
YFCC1M	YFCC1M	19.4	10.4	10.9	16.3	7.4	8.1
IN1K	YFCC1M	14.9	7.6	8.2	12.1	5.4	5.9
YFCC1M	IN1K	14.8	7.2	8.0	11.8	5.2	5.8

Table 2.11: Impact of datasets used to pre-train DINO and train CutLER. CutLER's detection performance is similar when pretraining both DINO and CutLER with the same dataset: the object-centric ImageNet dataset or the non-object-centric YFCC dataset.

Generalization to different detection architectures. We use different detector architectures for training CutLER and measure their performance in Table 2.10. We observe that CutLER works with various architectures, and its performance is improved with stronger architectures.

Impact of the pretraining dataset. We now study the impact of the dataset used for 1) pretraining the self-supervised DINO model and 2) training the CutLER model. The commonly used ImageNet dataset has a well-known object-centric bias [38] which may affect the unsupervised detection performance. Thus, we also use YFCC [69], a non-object-centric dataset. We control for the number of images in both ImageNet and YFCC for a fair comparison and use them for training DINO and CutLER. As Table 2.11 shows, CutLER's performance on COCO is robust to the choice of object-centric or non-object-centric datasets as long as the same dataset is used to train DINO and CutLER. This shows the generalization of CutLER to different data distributions. However, training DINO and CutLER with different data leads to worse performance, suggesting the importance of using the same image distribution for learning both DINO and CutLER models.

datasets	domain	testing data	#images	instance segmentation label
COCO [24]	natural images	`val2017` split	5,000	✓
COCO20K [24]	natural images	a subset of COCO	20,000	✓
UVO [66]	video frames	`val` split	7,356	✓
LVIS [61]	natural images	`val` split	19,809	✓
KITTI [70]	traffic images	`trainval` split	7,521	✗
Pascal VOC [67]	natural images	`trainval07` split	9,963	✗
Clipart [71]	clip arts	`traintest` split	1,000	✗
Watercolor [71]	paintings	`traintest` split	2,000	✗
Comic [71]	sketches	`traintest` split	2,000	✗
Objects365-V2 [72]	natural images	`val` split	80,000	✗
OpenImages-V6 [73]	natural images	`val` split	41,620	✗

Table 2.12: Summary of datasets used for zero-shot evaluation.

2.6 Appendix

Training details

While CutLER is agnostic to the underlying detector, we use popular Mask R-CNN [56] and Cascade Mask R-CNN [57] for all experiments, and use Cascade Mask R-CNN by default, unless otherwise noted. We train the detector on ImageNet with initial masks and bounding boxes for 160K iterations with a batch size of 16. When training the detectors with a ResNet-50 backbone [58], we initialize the model with the weights of a self-supervised pretrained DINO [25] model. We explored other pre-trained models, including MoCo-v2 [27], SwAV [46], and CLD [9], and found that they give similar detection performance. Therefore, we initialize model weights with DINO by default.

We also leverage the copy-paste augmentation [59], [60] during the model training process. Rather than using the vanilla copy-paste augmentation to improve the model's ability to segment small objects, we randomly downsample the mask with a scalar uniformly sampled between 0.3 and 1.0. We then optimize the detector for $160K$ iterations using SGD with a learning rate of 0.005, which is decreased by 5 after $80K$ iterations and a batch size of 16. We apply a weight decay of 5×10^{-5} and a momentum of 0.9.

For the multi-round of self-training, in each stage, we initialize the detection model using the weights from the previous stage. We optimize the detector using SGD with a learning rate of 0.01 for $80K$ iterations. Since the self-training stage can provide a sufficient number of pseudo-masks for model training, we don't use the exploration loss during the self-training stage.

Datasets	AP^{box}_{50}	AP^{box}_{75}	AP^{box}	AP^{box}_{S}	AP^{box}_{M}	AP^{box}_{L}	AR^{box}_{1}	AR^{box}_{10}	AR^{box}_{100}	AP^{mask}_{50}	AP^{mask}_{75}	AP^{mask}	AP^{mask}_{S}	AP^{mask}_{M}	AP^{mask}_{L}	AR^{mask}_{1}	AR^{mask}_{10}	AR^{mask}_{100}
COCO	21.9	11.8	12.3	3.7	12.7	29.6	6.8	19.6	32.8	18.9	9.2	9.7	2.4	8.8	24.3	5.8	16.5	27.1
COCO20K	22.4	11.9	12.5	4.1	12.7	29.5	6.8	19.7	33.1	19.6	9.2	10.0	2.8	8.9	24.3	5.8	16.6	27.4
UVO	31.7	14.1	16.1	3.7	11.3	25.3	6.8	24.5	42.5	31.6	14.1	16.1	3.7	11.3	25.3	4.6	18.0	32.2
LVIS	8.4	3.9	4.5	2.7	9.1	15.1	2.4	9.2	21.8	6.7	3.2	3.5	1.9	6.1	12.5	2.1	7.9	18.7
KITTI	18.4	6.7	8.5	0.5	5.6	19.2	6.2	16.6	27.8	-	-	-	-	-	-	-	-	-
Pascal VOC	36.9	19.2	20.2	1.3	6.5	32.2	16.5	32.8	44.0	-	-	-	-	-	-	-	-	-
Clipart	21.1	6.0	8.7	1.1	5.8	11.6	6.6	27.0	40.7	-	-	-	-	-	-	-	-	-
Watercolor	37.5	10.9	15.7	0.1	1.1	20.0	19.4	37.8	44.2	-	-	-	-	-	-	-	-	-
Comic	30.4	7.7	12.2	0.0	1.3	16.0	8.5	28.2	38.4	-	-	-	-	-	-	-	-	-
Objects365	21.6	10.3	11.4	3.0	10.4	20.4	3.0	15.4	34.2	-	-	-	-	-	-	-	-	-
OpenImages	17.3	9.5	9.7	0.4	2.3	14.9	6.5	17.6	29.6	-	-	-	-	-	-	-	-	-

Table 2.13: Detailed zero-shot evaluation results on all benchmarks used in this work.

Datasets used for zero-shot evaluation

COCO and COCO20K [24] is a large-scale object detection and instance segmentation dataset, containing about $115K$ and $5K$ images in the training and validation split, respectively. Additionally, COCO has an unannotated split of $123K$ images. We test our model in a class-agnostic manner on COCO `val2017` and COCO `20K`, without fine-tuning on any images in COCO. COCO $20K$ is a subset of the COCO `trainval2014` [24], containing 19817 randomly sampled images, used as a benchmark in [30], [32], [33]. We report class-agnostic COCO style averaged precision and averaged recall for object detection and segmentation tasks.

Pascal VOC [67] is another popular benchmark for object dtetection. We evaluate our model on its `trainval07` split in COCO style evaluation matrics.

UVO [66]. Unidentified Video Objects (UVO) is an exhaustively annotated dataset for video object detection and instance segmentation. We evaluate our model on UVO `val` by frame-by-frame inference and report results in COCO style evaluation matrics.

LVIS [61] collected 2.2 million high-quality instance segmentation masks for over 1000 entry-level object categories, which naturally constitutes the long-tailed data distribution. We report class-agnostic object detection and instance segmentation results on LVIS `val` split, containing about $5K$ images.

CrossDomain [71] contains three subsets of watercolor, clipart, and comics, in which objects are depicted in watercolor, sketch and painting styles, respectively. We evaluate our model on all annotated images from these three datasets, *i.e.*, `traintest`.

Objects365 *V2* [72] presents a supervised object detection benchmark with a focus on diverse objects in the wild. We evaluate CutLER on the 80K images from its `val` split.

OpenImages *V6* [73] unifies image classification, object detection, and instance segmentation, visual relationship detection, *etc*in one dataset. We evaluate CutLER on its 42K images from the `val` split.

KITTI [70] presents a dataset captured from cameras mounted on mobile vehicles used for autonomous driving research. We evaluate CutLER on 7521 images from KITTI's `trainval` split.

We provide the summary of these datasets used for zero-shot evaluation in Table 2.12.

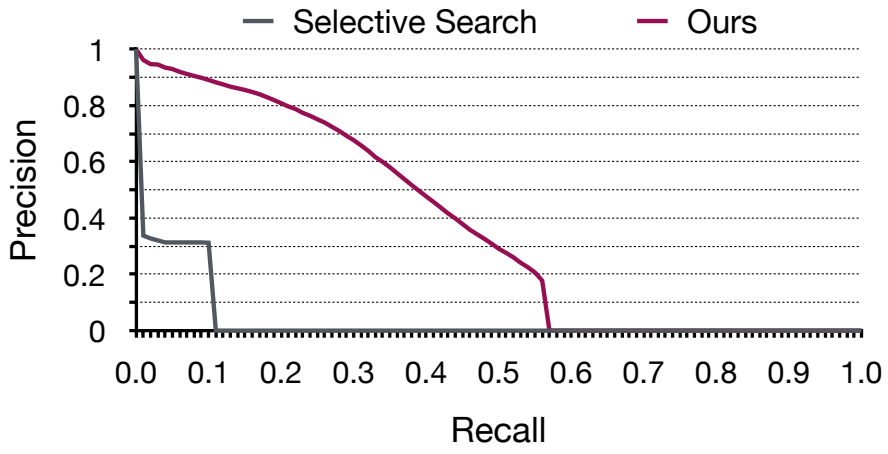


Figure 2.7: Precision-recall curve for comparing selective search and CutLER on VOC07 `trainval`.

Additional results for zero-shot detection & segmentation

In this section, we use official COCO API and provide more results with standard COCO metrics, including AP across various IoU thresholds - AP (averaged over IoU thresholds from 0.5 to 0.95 with a step size of 0.05), AP_{50} (IoU@0.5) and AP_{75} (IoU@0.75), and AP across scales - AP_S (small objects), AP_M (medium objects) and AP_L (large objects). We provide detailed results on all these benchmarks listed in Table 2.12 and report these results in Table 2.13. We report the performance of object detection for all datasets. In addition, for those datasets that provide annotations for instance segmentation, we also present the performance of the instance segmentation task. It is worth noting that on these datasets without segmentation labels, CutLER can still predict instance segmentation masks, but since we do not have ground truth masks to be compared, we cannot evaluate the results.

CutLER vs. Selective Search

Selective Search [74] is a popular unsupervised object discovery method, used in many early state-of-the-art detectors such as R-CNN [75] and Fast R-CNN [76]. However, generating possible object locations with sliding windows greatly reduces inference speed (please refer to [74] for more details on selective search). We compare CutLER's performance to selective search in Fig. 2.7 and observe that CutLER provides a significant improvement in both precision and recall, which indicates that CutLER is a better performing unsupervised method for region proposal generation with real-time inference speed.

Training details for label-efficient and fully-supervised learning

We train the detector on the COCO [24] dataset using the bounding box, and instance mask labels. To evaluate label efficiency, we subsample the training set to create subsets with varying propor-

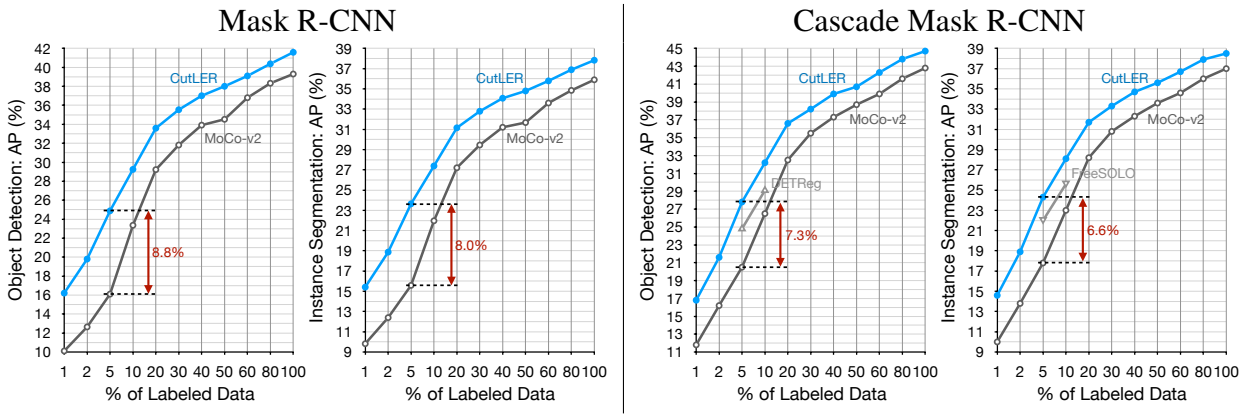


Figure 2.8: Fine-tuning on MS-COCO with various annotation ratios. We report results using Mask R-CNN and Cascade Mask R-CNN with a backbone of ResNet-50 as the detector.

tions of labeled image We train the detector, initialized with CutLER, on each of these subsets.

As a baseline, we follow the settings from MoCo-v2 [27] and train the same detection architecture initialized with a MoCo-v2 ResNet50 model, given its strong performance on object detection tasks. MoCo-v2 and our models use the same training pipeline and hyper-parameters and are trained for the $1\times$ schedule using Detectron2 [68], except for extremely low-shot settings with 1% or 2% labels. Following previous works [23], when training with 1% or 2% labels, we train both MoCo-v2 and our model for 3,600 iterations with a batch size of 16.

Our detector weights are initialized with ImageNet-1K pre-trained CutLER, except for the weights of the final bounding box prediction layer and the last layer of the mask prediction head, which are randomly initialized with values taken from a normal distribution. For experiments on COCO with labeling ratios below 50%, during model training, we use a batch size of 16, and learning rates of 0.04 and 0.08 for model weights loaded from the pre-trained CutLER and randomly initialized, respectively. For experiments on COCO with labeling ratios between 50% and 100%, the learning rates of all layers decay by a factor of 2.

For a fair comparison, baselines and CutLER use the same hyper-parameters and settings.

2.7 Summary

Object localization is a fundamental task in computer vision. In this paper, we have shown that a simple yet effective cut-and-learn approach can achieve extraordinary performance on challenging object detection and instance segmentation tasks without needing to train with human annotations. As a zero-shot unsupervised detector, CutLER, trained solely on ImageNet, outperforms the detection performance of previous works by over $2.7\times$ on 11 benchmarks across various domains.

Chapter 3

VideoCutLER: Unsupervised Video Instance Segmentation

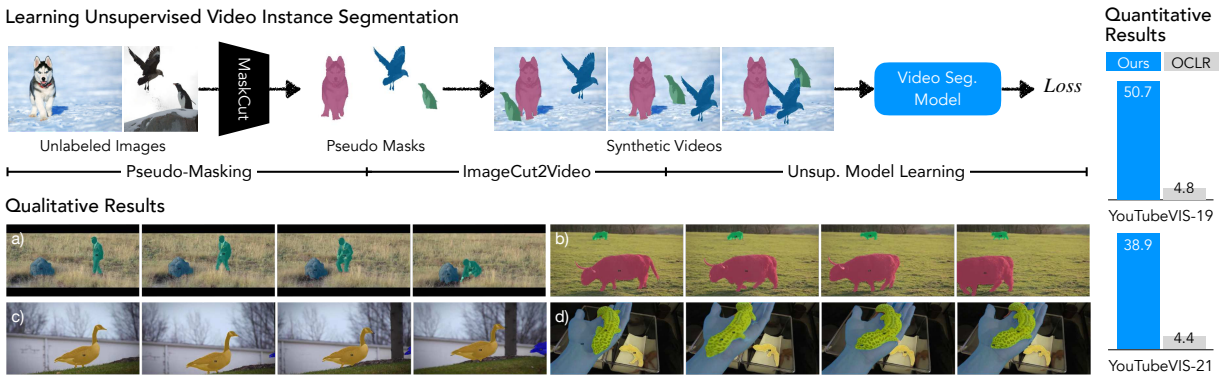


Figure 3.1: VideoCutLER is a simple unsupervised video instance segmentation method (UnVIS). We show the first competitive unsupervised results on the challenging YouTubeVIS benchmark. Moreover, unlike most prior approaches, we demonstrate that UnVIS models can be learned without relying on natural videos and optical flow estimates. ***Row 1:*** We propose **VideoCutLER**, a simple cut-synthesis-and-learn pipeline that involves three main steps. Firstly, we generate pseudo-masks for multiple objects in an image using MaskCut [12]. Then, we convert a random pair of images in the minibatch into a video with corresponding pseudo mask trajectories using ImageCut2Video. Finally, we train an unsupervised video instance segmentation model using these mask trajectories. ***Row 2:*** Despite being trained only on unlabeled images, at inference time VideoCutLER can be directly applied to unseen videos and can segment and track multiple instances across time (Fig. 3.1a), even for small objects (Fig. 3.1b), objects that are absent in specific frames (Fig. 3.1c), and instances with high overlap (Fig. 3.1d). ***Column 2:*** Our method surpasses the previous SOTA method OCLR [77] by a factor of 10 in terms of class-agnostic AP_{50}^{video}.

Existing approaches to unsupervised video instance segmentation typically rely on motion estimates and experience difficulties tracking small or divergent motions. We present VideoCutLER, a simple method for unsupervised multi-instance video segmentation without using motion-based

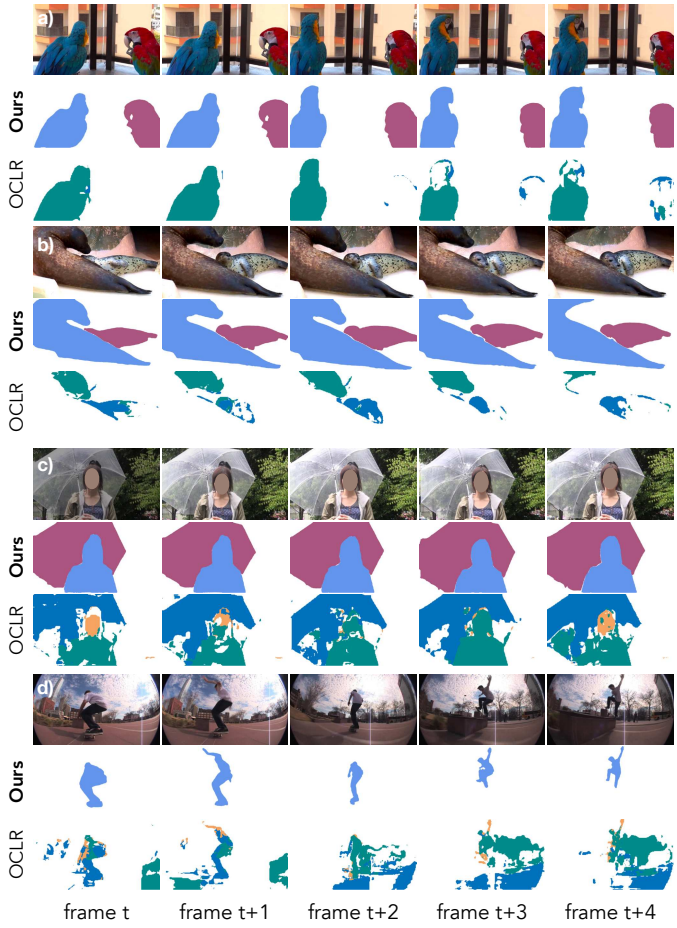


Figure 3.2: Challenges encountered by the previous state-of-the-art OCLR: Within the framework of OCLR [77], a method that heavily relies on optical flows as model inputs, several distinct failure cases emerge. These include situations where the method struggles to accurately segment both moving and static objects (as demonstrated in Fig. 3.2a), struggles to effectively track non-rigid objects as a coherent unit (Fig. 3.2b), encounters difficulties in distinguishing overlapping instances (Fig. 3.2c), and fails to maintain consistent predictions under varying illumination conditions (Fig. 3.2d). Nonetheless, many of these challenges can be effectively addressed through the application of our proposed approach, VideoCutLER, without being reliant on the optical estimations used by various prior works [77], [78]. We present qualitative comparisons using the YouTubeVIS dataset [79].

learning signals like optical flow or training on natural videos. Our key insight is that using high-quality pseudo masks and a simple video synthesis method for model training is surprisingly sufficient to enable the resulting video model to effectively segment and track multiple instances across video frames. We show the first competitive unsupervised learning results on the challenging YouTubeVIS-2019 benchmark, achieving 50.7% AP^{video}_{50}, surpassing the previous state-of-the-art by a large margin. VideoCutLER can also serve as a strong pretrained model for supervised video instance segmentation tasks, exceeding DINO by 15.9% on YouTubeVIS-2019 in terms of AP^{video}.

3.1 Introduction

Video instance segmentation is vital for various computer vision applications, *e.g.*video surveillance, autonomous driving, and video editing, yet labeled videos are costly to obtain. Hence, there is a pressing need to devise an unsupervised video instance segmentation approach that can comprehend video content comprehensively and operate in general domains without labels.

Prior work in this area typically relies on an optical flow network as an off-the-shelf motion estimator [77], [78], [80]. Although optical flow can be informative in detecting pixel motion

between frames, it is not always a reliable technique, particularly in the presence of occlusions, motion blur, complex motion patterns, changes in illuminations, *etc*. As a result, models that heavily rely on optical flow estimations may fail in several common scenarios. For example, stationary or slowly moving objects may have flow estimates similar to the background, causing them to be omitted in the segmentation process (*e.g.*, the parrot with negligible motion is missed in Fig. 3.2a). Similarly, non-rigid objects with non-consistent motions for several parts have varying optical flows, leading to a failure in segmenting all parts cohesively as a unit if object motion is presumed constant (Fig. 3.2b). Also, objects with similar motion patterns and high overlap are complex for optic flow methods to accurately distinguish between them, especially in boundary regions (Fig. 3.2c). Finally, objects with illumination changes across frames can cause optical-flow based models to produce non-consistent and blurred segmentation masks (Fig. 3.2d). Given the limitations above, we advocate for unsupervised video segmentation models which do not depend on optical flow estimates. We propose a method to train a video segmentation model by generating simple synthetic videos from individual images, without relying on explicit motion estimates or requiring labeled natural videos.

Our method, **VideoCutLER**, is an unsupervised **Video** instance segmentation model that employs a **Cu**t-syn**t**hesis-and-**LEaR**n pipeline (Fig. 3.1). First, given unlabeled images, we extract pseudo-masks for multiple objects in an image using MaskCut [12], leveraging a self-supervised DINO [25] and a spectral clustering method Normalized Cuts [53] (details in Sec. 3.3). Second, given unlabeled images and their pseudo-masks in a minibatch, we propose ImageCut2Video, a surprisingly simple video synthesis scheme that generates a video from those with corresponding pseudo mask trajectories (details in Sec. 3.3). Finally, those mask trajectories are used to train a video instance segmentation model, aiming to perform object segmentation with temporal consistency across video frames (details in Sec. 3.3). Our model learns to segment and track object instances based on their appearance (feature) similarities across video frames.

Despite being learned from only unlabeled images (and the temporally simple synthetic video sequences we construct from them), VideoCutLER succeeds at multi-instance video segmentation, achieving a new state-of-the-art (SOTA) performance of 50.7% AP_{50}^{video} on YouTubeVIS-2019. This result surpasses the previous SOTA [77] by substantial margins of 45.9% (50.7% vs. 4.8%). This result also considerably narrows the performance gap between supervised and unsupervised learning, reducing it from 29.1% to 11.0% in terms of the AP_{50}^{video}.

Moreover, most prior works on self-supervised representation learning [9], [25], [26], [28], [42] are limited to providing initializations only for the model backbones, with the remaining layers being randomly initialized. In contrast, our pretraining strategy takes a more comprehensive approach that allows all model weights to be pretrained, resulting in a stronger pretrained model better suited for supervised learning. As a result, our method outperforms DINO's [25] AP^{video} on YoutubeVIS-2019 by 15.9%.

Contributions. ***Insights:*** We found that a simple video synthesis method yield surprisingly effective results for training unsupervised multi-instance video segmentation models. This efficacy is achieved without the necessity of explicit motion estimates or the utilization of natural videos (relying solely on unlabeled ImageNet data suffices), a novel aspect that has not been previously demonstrated in the field. ***Methods:*** We propose a simple yet effective cut-synthesize-and-learn pipeline

	CRW	DINO	OCLR	Ours
Segment multiple objects	✓	✗	✓	✓
Track objects across frames	✓	✗	✓	✓
No need for optical flow	✓	✓	✗	✓
No 1st-frame ground-truth	✗	✗	✓	✓
No human labels at any stage	✗	✗	✓†	✓
Pretrained model for sup. learning	✗	✓	✗	✓

Table 3.1: We compare previous methods on unsupervised video instance segmentation, including CRW [86], DINO [25], and OCLR [77], with our VideoCutLER in term of key properties. Our VideoCutLER is the only approach that fulfills all these desired properties. †: The optical flow estimator OCLR employs (RAFT [80]) is pretrained on both synthetic data and human-annotated data like KITTI-2015 [91] and HD1K [92].

VideoCutLER for learning video instance segmentation models, given unlabeled images. ***Results:*** Our method shows the first successfully results on challenging unsupervised multi-instance video segmentation benchmark YouTubeVIS, outperforming the previous SOTA model's AP_{50}^{video} by a large margin.

3.2 Related Work

Unsupervised video instance segmentation (VIS) requires not only separating and tracking the main moving foreground objects from the background, but also differentiating between different instances, without any human annotations [81]. Previous works [33], [78], [82]–[84] on unsupervised video segmentation has primarily centered on ***unsupervised video object segmentation*** (VOS), aiming to detect all moving objects as the foreground and to generate a pixel-level binary segmentation mask, regardless of whether the scene contains a single instance or multiple instances. Despite some works exploring ***unsupervised video instance segmentation*** (VIS), many of these approaches have resorted to either utilizing first frame annotations [25], [85], [86] to propagate label information throughout the video frames or leveraging supervised learning using large amounts of external labeled data [87]–[90]. Furthermore, prior studies typically utilized optical flow networks that were pretrained with human supervision using either synthetic data or labeled natural videos [77], [78], [83], [87].

The properties deemed necessary for an unsupervised learning method to excel in video instance segmentation tasks are presented and discussed in Table 3.1. Our proposed method, VideoCutLER, is the only approach that satisfies all these properties, making it an effective and promising solution for unsupervised video instance segmentation.

Unsupervised object discovery aims to automatically discover and segment objects in an image in an unsupervised manner [12], [23], [33], [82]. LOST [32] and TokenCut [33] focus on salient object detection and segmentation via leveraging the patch features from a pretrained DINO [25] model. For multi-object discovery, FreeSOLO [23] first generates object pseudo-masks for unla-

beled images, then learns an unsupervised instance segmentation model using these pseudo-masks. CutLER [12] presents a straightforward cut-and-learn pipeline for unsupervised detection and segmentation of multiple instances. It has demonstrated promising results on more than eleven different benchmarks, covering a wide range of domains.

In contrast to previous approaches, our unsupervised learning method focuses on simultaneously tracking objects in a video sequence while identifying correspondences between instances across multiple frames.

Self-supervised representation learning generates its own supervision signal by exploiting the implicit patterns or structures present in the input data [25], [26], [46], [47]. Unlike most previous self-supervised learning models, which still require fine-tuning on labeled data to be operative on complex computer vision tasks, such as detection and segmentation, VideoCutLER can tackle these complex, challenging tasks with purely unsupervised learning methods.

3.3 VideoCutLER

We present VideoCutLER, a simple cut-synthesis-and-learn pipeline consisting of three main steps. First, we generate pseudo-masks for multiple objects in an image using MaskCut (Sec. 3.3). Next, we convert a random pair of images in the minibatch into a synthetic video with corresponding pseudo mask trajectories using ImageCut2Video (Sec. 3.3). Finally, we train an unsupervised video instance segmentation model using these mask trajectories. As the model inputs do not contain explicit motion estimates, it learns to track objects based on their appearance similarity (Sec. 3.3).

Single-image unsupervised segmentation

We employ the MaskCut method, introduced in the CutLER [12] method. MaskCut is an efficient spectral clustering approach for unsupervised image instance segmentation and object detection and can discover multiple object masks in a single image without human supervision. MaskCut builds upon a self-supervised DINO model [36] with a backbone of ViT [29] and a cut-based clustering method Normalized Cuts (NCut) [53]. MaskCut first generates a patch-wise affinity matrix $W_{ij} = \frac{K_i K_j}{\|K_i\|_2 \|K_j\|_2}$ using the 'key' features K_i for patch i from DINO's last attention layer. Subsequently, the NCut algorithm [53] is employed on the affinity matrix by solving a generalized eigenvalue problem

$$(D - W)x = \lambda D x \tag{3.1}$$

where D is a diagonal matrix with $d(i) = \sum_j W_{ij}$ and x is the eigenvector that corresponds to the second smallest eigenvalue λ. Then, the foreground masks M^s can be extracted via bi-partitioning x, which segments a single object within the image. To segment more than one object, MaskCut uses an iterative process that masks out the values in the affinity matrix using the extracted

foreground mask:

$$W_{ij}^{t} = \frac{(K_i \prod_{s=1}^{t} M_{ij}^{s})(K_j \prod_{s=1}^{t} M_{ij}^{s})}{\|K_i\|_2 \|K_j\|_2} \tag{3.2}$$

and repeats the NCut algorithm. We set $t = 3$ by default.

Although MaskCut can effectively locate and segment multiple objects in an image, it operates only on a single image, lacking temporal consistency in the instance segmentation masks produced across video frames.

ImageCut2Video Synthesis for Training

We propose a learning-based approach to ensuring temporal consistency in video segmentation masks, based on generating synthetic videos from pairs of individual images and MaskCut masks. Surprisingly, we found that an extremely simple synthetic video generation method yields sufficient training data to learn a powerful video segmentation model that can operate on videos with much greater complexity of motion than is present in the training data.

Given unlabeled images in the minibatch and their pseudo-masks, our ImageCut2Video method synthesizes corresponding videos and pseudo-mask trajectories, thereby allowing us to train the model in an unsupervised manner while offering the necessary supervision for simultaneous detection, segmentation, and tracking of objects in videos.

First, given an image and its corresponding pseudo-masks in the mini-batch, we duplicate the image t times and connect its MaskCut pseudo-masks to form the initial trajectories. This synthetic video, however, only contains static foreground objects. To generate additional trajectories with mobile objects, a second image is randomly selected from the mini-batch, and its objects are cropped using its MaskCut pseudo-masks. These objects are then randomly resized, repositioned, and augmented before being pasted onto the first image. The resulting masks are connected along the temporal dimension to generate additional trajectories with mobile objects.

Specifically, given a target image I_1, a random source image I_2 in the mini-batch and its corresponding set of binary pseudo-masks $\{M_2^1, ..., M_2^s\}$, we first apply a transformation function $\mathcal{T}$ to resize and shift these pseudo-masks randomly. This gives us a new set of pseudo-masks $\{\hat{M}_2^1, ..., \hat{M}_2^s\}$, where $\hat{M}_2^s = \mathcal{T}(M_2^s)$. Next, we synthesize a video with t frames by duplicating image I_1 for t times and pasting the augmented masks onto I_1 using:

$$I_1^t = I_1 \times \Pi_{i=1}^{s}(1 - \hat{M}_2^i) + I_2 \times (1 - \Pi_{i=1}^{s}(1 - \hat{M}_2^i)) \tag{3.3}$$

where $\times$ refers to element-wise multiplication.

Video Segmentation Model

During training, the synthetic videos produced by ImageCut2Video, comprising both mobile and stationary objects, are used as the inputs to train a video instance segmentation model. The segmentation mask trajectories corresponding to each object in the video serve as 'ground-truth' labels.

Methods	Training settings			YouTubeVIS-2019							YouTubeVIS-2021						
	flow	videos	sup.	AP_{50}	AP_{75}	AP	AP_S	AP_M	AP_L	AR_{10}	AP_{50}	AP_{75}	AP	AP_S	AP_M	AP_L	AR_{10}
MotionGroup* [78]	✓	✓	✗	1.3	0.1	0.3	0.2	0.3	0.5	1.7	1.1	0.1	0.2	0.1	0.2	0.5	1.5
OCLR* [77]	✓	✓	✗†	4.8	0.4	1.3	0.0	1.2	5.5	11.0	4.4	0.3	1.2	0.1	1.6	7.1	9.6
CutLER‡	✗	✗	✗	37.5	14.6	17.1	3.3	13.9	27.6	30.4	29.2	10.4	12.8	3.1	12.8	27.8	22.6
VideoCutLER	✗	✓*	✗	50.7	24.2	26.0	5.6	20.9	37.9	42.4	38.9	19.0	17.1	5.3	18.3	37.5	31.3
vs. prev. SOTA				**+12.8**	**+9.6**	**+8.9**	**+2.3**	**+7.0**	**+10.3**	**+12.0**	**+9.7**	**+8.6**	**+4.3**	**+2.2**	**+5.5**	**+9.7**	**+8.7**

Table 3.2: Zero-shot unsupervised multi-instance video segmentation on YouTubeVIS-2019 and YouTubeVIS-2021. We report the instance segmentation metrics (AP and AR) and training settings. *: reproduced MotionGroup [78] and OCLR [77] results with the official code and checkpoints. †: the optical flow estimator OCLR employs (RAFT [80]) is pretrained on both synthetic data [95], [96] and human-annotated data, such as KITTI-2015 [91] and HD1K [92]. ‡: We train a CutLER [12] model with Mask2Former as a detector on ImageNet-1K, following CutLER's official training recipe, and use it as a strong baseline. *: VideoCutLER is trained on synthetic videos generated using ImageNet. Sup and flow denote human supervision and optical flow information, respectively. We evaluate results on YouTubeVIS's `train` splits in a class-agnostic manner (note: we never train on YouTubeVIS).

We utilize VideoMask2Former [93], [94] with a backbone of ResNet50 [58] as our video instance segmentation (VIS) model. It operates by attending to the 3D spatiotemporal features of our synthetic videos and generating 3D volume predictions of pseudo-mask trajectories using shared queries across frames. The shared queries across frames enable the model to segment and track object instances based on their appearance (feature) similarities, making it a powerful framework for analyzing video sequences.

Implementation Details

VideoCutLER. We first employ the MaskCut approach on images preprocessed to a resolution of 480×480 pixels. We then compute a patch-wise cosine similarity matrix using the pretrained ViT-Base/8 DINO [25] model, which serves as input to the MaskCut algorithm for initial segmentation mask generation. We set $t = 3$, which is the maximum number of masks per image. To refine the segmentation masks, we employ a post-processing step using Conditional Random Fields (CRFs) [55], which enforces smoothness constraints and preserves object boundaries, resulting in improved segmentation masks.

Next, we use ImageCut2Video to synthetic videos given images and their pseudo-masks in a mini-batch. We found that synthetic videos with two frames are sufficient to train a video instance segmentation model; therefore, we use $s = 2$ by default. We randomly change the brightness, contrast, and rotation of the masks to create new variations of pseudo-masks. Additionally, we randomly resize the pseudo-masks (scale∈[0.8,1.0]), and shift their positions.

Training and test data. Our model is trained solely on the unlabeled images from ImageNet [38], which comprises approximately 1.3 million images. Without further fine-tuning on any video datasets, we test our model's zero-shot unsupervised video instance segmentation performance on

four multi-instance video segmentation benchmarks, including YouTubeVIS-2019 [79], YouTubeVIS-2021 [79], DAVIS2017 [97], and DAVIS2017-Motion [97], [98].

YoutubeVIS-2019 and YouTube-VIS2021 contain 2,883 high-resolution YouTube videos and 3,859 high-resolution YouTube videos, respectively. We evaluate the zero-shot unsupervised learning performance on their training splits in a class-agnostic manner. For DAVIS-2017, we evaluate our model's performance on the 30 videos from its val set.

Training settings. 1) Unsupervised Image Model Pretraining: We first pretrain a Mask2Former [93] model with a backbone of ResNet50 [58] on ImageNet using MaskCut's pseudo-masks. The model is optimized for 160k iterations, with a batch size of 16 and a learning rate of 0.00002. The learning rate is decayed by a factor of 20 at iteration 80,000. To prevent overfitting, a dropout layer with a rate of 0.3 is added after the self-attention layers of transformer decoders. **2) Unsupervised Video Model Learning:** We initialize the VideoMask2Former model [94] with model weights from the previous stage, and then fine-tune it on the synthetic videos we construct from ImageNet. We train VideoCutLER on 8 A100 GPUs for 80k iterations, using the AdamW optimizer [99]. We set the initial learning rate to 0.000005 and apply a learning rate multiplier of 0.1 to the backbone. A dropout layer with a rate of 0.3 is added after the self-attention layers of transformer decoders.

Evaluation metric AP^{video} and AR^{video}: The evaluation metrics used in YouTubeVIS are Averaged Precision (AP) and Averaged Recall (AR), which are similar to those used in COCO [24]. The evaluation is specifically conducted at 10 intersection-over-union (IoU) thresholds ranging from 50% to 95% with a step of 5% [79]. However, unlike in image instance segmentation, each instance in a video comprises a sequence of masks, so the IoU computation is performed not only in the spatial domain, but also in the temporal domain by summing the intersections at every single frame over the unions at every single frame.

Evaluation metric $\mathcal{J}$ and $\mathcal{F}$: For DAVIS [98], we report results using their official evaluation metrics $\mathcal{J}\&\mathcal{F}$, $\mathcal{J}$ and $\mathcal{F}$. The region measure ($\mathcal{J}$) [98] is the intersection-over-union (IoU) score between the algorithm's mask and the ground-truth mask. The boundary measure ($\mathcal{F}$) [98] is the average precision of the boundary of the algorithm's mask. The evaluation metrics are computed separately for each instance, and then the results are averaged over all instances to get the final score. $\mathcal{J}\&\mathcal{F}$ is the mean of $\mathcal{J}$ and $\mathcal{F}$.

3.4 Experiments

We evaluate the performance of VideoCutLER on several video instance segmentation benchmarks. In Sec. 3.4, we demonstrate that our approach can effectively perform segmentation and tracking of multiple objects in videos, even when trained on unlabeled ImageNet images without any form of supervision. Our experimental results reveal that our method can drastically reduce the performance gap between unsupervised and supervised learning methods for video instance discovery and tracking. Furthermore, Sec. 3.4 demonstrates that fine-tuning VideoCutLER leads to further performance gains in video instance segmentation, surpassing previous works such as DINO in both fully supervised learning and semi-supervised learning tasks. In Sec. 3.4, we conduct an ablation study to examine the impact of key components and their hyperparameters.

Methods	Training settings				DAVIS2017			DAVIS2017-Motion		
	flow	videos	sup.	training data	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$(Mean)	$\mathcal{F}$(Mean)	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$(Mean)	$\mathcal{F}$(Mean)
MotionGroup (sup.) [78]	✓	✓	✗	IN-1K+synthetic	-	-	-	39.5	44.9	34.2
Mask R-CNN (w/ flow)* [56], [77]	✓	✓	✗	IN-1K+synthetic	-	-	-	50.3	50.4	50.2
OCLR (w/ flow)* [77]	✓	✓	✗	IN-1K+synthetic	39.6	38.2	41.1	55.1	54.5	55.7
VideoCutLER	✗	✗	✗	IN-1K	43.6	41.7	45.5	57.3	57.4	57.2
vs. prev. SOTA					**+4.0**	**+3.5**	**+4.4**	**+2.2**	**+2.9**	**+1.5**

Table 3.3: Zero-shot unsupervised single/few-instance segmentation. VideoCutLER also outperforms the previous state-of-the-arts on DAVIS2017 and DAVIS2017-Motion. *Note: 12 out of 30 videos from DAVIS2017 and 26 out of 30 videos from DAVIS2017-Motion contain only 1 moving instance. Additionally, DAVIS datasets focus solely on the performance of moving prominent objects, even in videos where multiple objects are present.* This disadvantages our model since it can segment both static and moving objects and has not been exposed to any downstream videos during training. *: utilize optical flow predictions from RAFT [80], which is pretrained on external videos. All methods are evaluated in a zero-shot manner, *i.e.*no fine-tuning on target videos.

Unsupervised Zero-shot Evaluations

In this section, we evaluate the performance of our method against previous state-of-the-art approaches on various video instance segmentation benchmarks.

Evaluating unsupervised video instance segmentation poses two main challenges. Firstly, as unsupervised learning methods train the model without semantic classes, the class-aware video segmentation setup cannot be used directly for an evaluation. As a result, following previous works, we evaluate video instance segmentation results in a class-agnostic manner. Secondly, video instance segmentation datasets often annotate only a subset of the objects in the video, which makes Average Recall (AR) a valuable metric that does not penalize models for detecting novel objects not labeled in the dataset [12]. Therefore, we report both AR and AP for YouTubeVIS. Regarding DAVIS, we use the official unsupervised learning metrics $\mathcal{J}$, $\mathcal{F}$, and $\mathcal{J}\&\mathcal{F}$. All these metrics assess the performance of unsupervised video instance segmentation in a class-agnostic manner. Sec. 3.3 lists more details on evaluation metrics.

Detailed comparisons on YouTubeVIS. Table 3.2 presents a summary of the results for unsupervised zero-shot video instance segmentation on the YouTubeVIS-2019 and YouTubeVIS-2021 datasets. We compare our method's results with the previous state-of-the-art methods OCLR [77] and motion grouping [78]. We reproduce their results using their official code and checkpoints to ensure fairness.

Although OCLR [77] is also trained on synthetic videos, it relies on the off-the-shelf optical flow estimator RAFT [80] to compute optical flows for RGB sequences. It is worth noting that RAFT is pretrained on a combination of synthetic videos [95], [96] and human-annotated videos such as KITTI-2015 [91] and HD1K [92]. Our approach, VideoCutLER, despite not using any optical flow estimations like many previous works on unsupervised video segmentation, achieves over 10× higher AP_{50} and 18× higher AP than OCLR [77] on YouTubeVIS-2019. Additionally, we achieve over 30% higher recall. Furthermore, unlike the previous state-of-the-art

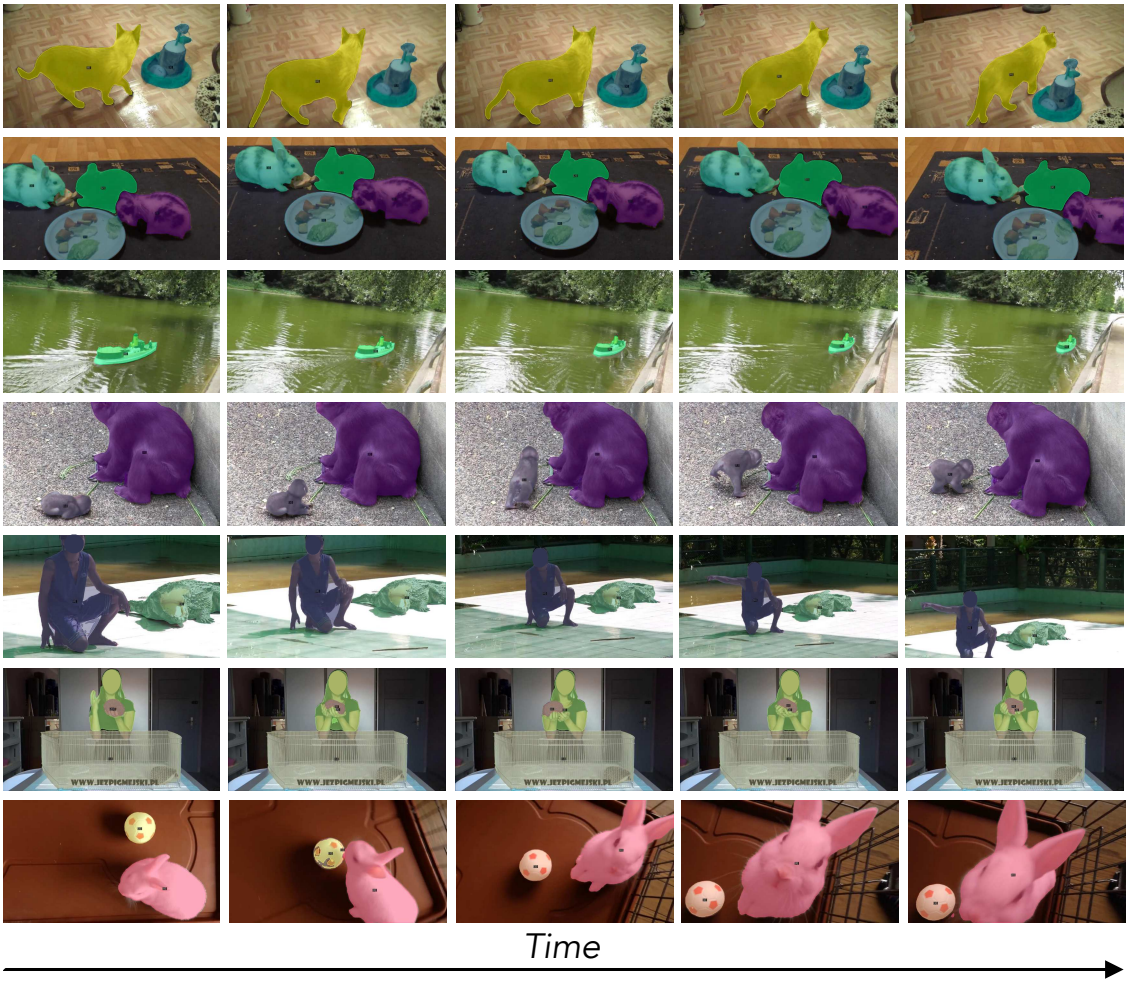


Figure 3.3: We present **qualitative visualizations** illustrating the zero-shot unsupervised video instance segmentation outcomes of VideoCutLER on YouTubeVIS dataset. It's noteworthy that VideoCutLER is solely pretrained on image dataset ImageNet-1K, and its evaluation is conducted directly on the video dataset YouTubeVIS (no further fine-tuning required). The visual results provided effectively highlight that VideoCutLER is capable of segmenting and tracking multiple instances, delivering consistent tracking results across video frames, and successfully distinguishing between various instances, even when significant overlapping occurs. We show more demo results in appendix.

method OCLR [77], which exhibits poor performance in segmenting small objects (with 0.0% AP_S), our approach significantly outperforms it. Similar performance gains can be observed on YouTubeVIS-2021. Finally, the performance gains to CutLER [12] demonstrates the effectiveness of VideoCutLER in training unsupervised multi-instance video segmentation models, surpassing CutLER by over 12.8% on YouTubeVIS-2019.

In Fig. 3.3, we present qualitative visualizations illustrating the zero-shot unsupervised video instance segmentation outcomes of VideoCutLER on YouTubeVIS dataset.

Methods	Training settings				YouTubeVIS-2021\YouTubeVIS-2019						
	flow	videos	sup.	training data	AP_{50}	AP_{75}	AP	AP_S	AP_M	AP_L	AR_{100}
Mask2Former [94]	✓	✓	✗	IN-1K+YT2019	48.9	22.2	24.9	-	-	-	-
MaskTrack R-CNN* [79]	✓	✓	✓	IN-1K+YT2019	32.4	13.0	15.0	8.4	24.9	39.0	20.3
MaskTrack R-CNN* [79]	✓	✓	✓	IN-1K+COCO+YT2019	35.8	18.7	18.7	10.5	31.3	46.8	24.5
OCLR* [77]	✓	✓	✗	IN-1K+synthetic	3.3	0.2	1.0	0.3	2.7	7.5	5.4
VideoCutLER	✗	✗	✗	IN-1K	21.4	7.1	9.0	4.9	13.3	29.6	17.1
vs. prev. SOTA					**+18.1**	**+6.9**	**+8.0**	**+4.6**	**+10.6**	**+22.1**	**+11.7**

Table 3.4: VideoCutLER greatly narrows the **gap between fully-supervised learning and unsupervised learning** for multi-instance video segmentation. Results are evaluated in a class-agnostic manner on the relative complement of the set of videos from YouTubeVIS-2021 and the set of videos from YouTubeVIS-2019. VideoCutLER and Mask2Former use a backbone of ResNet50. *: reproduced results with the official code and checkpoints. IN-1K refers to ImageNet-1K.

Detailed comparisons on DAVIS. To provide a comprehensive evaluation and comparison with existing unsupervised video instance segmentation approaches, we also assess the performance of our model on the validation sets of DAVIS-2017 and DAVIS2017-Motion [77], [98]. Note that both DAVIS2017 and DAVIS2017-Motion datasets focus only on the performance of instance segmentation on ***prominent moving objects***, even in videos with multiple objects. As a result, only a single or a few objects of interest per video are annotated, which may not reflect the challenges that arise when multiple objects are present.

Although the evaluation of DAVIS is an unfair assessment for us since VideoCutLER is supposed to segment both static and moving objects, whereas DAVIS focuses on moving prominent objects, with only a single or a few moving objects of interest per video annotated. However, Table 3.3 shows that VideoCutLER yields approximinately 4% higher $\mathcal{J}$, $\mathcal{F}$, and $\mathcal{J}\&\mathcal{F}$. The additional results on DAVIS demonstrate that VideoCutLER achieves superior performance not only on static or minimally moving objects but also on dynamic objects, where prior methods relying on optical flow estimates can benefit from additional cues.

Comparison of supervised and unsupervised learning in object discovery and tracking abilities is presented in Table 3.4. We train a supervised MaskTrack R-CNN [79] model on the human-annotated training set of YouTubeVIS-2019 dataset, and evaluate it in a class-agnostic manner on the videos that are not shared between YouTubeVIS-2019 and YouTubeVIS-2021 datasets [79]. Table 3.4 shows that our VideoCutLER model significantly narrows the gap between supervised learning and unsupervised learning methods in terms of the averaged precision AP_{50} (gaps: 29.1%→11.0%) and the averaged recall AR_{100} (gaps: 14.9%→3.2%), particularly for the AR_{100}.

Label-Efficient and Fully-Supervised Learning

In this section, we investigate VideoCutLER as a pretraining approach for supervised video instance segmentation models, and evaluate its effectiveness in label-efficient and fully-supervised learning scenarios.

Methods	Architecture	YouTubeVIS-2019						YouTubeVIS-2021					
		AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
DINO [25]	Mask2Former [94]	23.0	39.0	23.7	6.0	28.0	34.2	24.6	41.4	25.9	8.7	34.0	39.9
VideoCutLER	Mask2Former [94]	38.9	56.7	43.3	22.1	43.1	51.8	33.4	53.8	36.3	15.7	40.9	54.8
vs. prev. SOTA		**+15.9**	**+17.7**	**+19.6**	**+16.1**	**+15.1**	**+17.6**	**+8.8**	**+12.4**	**+10.4**	**+7.0**	**+6.9**	**+14.9**

Table 3.5: VideoCutLER can serve as a strong pretrained model for the **supervised video instance segmentation** task. The video segmentation model, Mask2Former, is initialized with various pretrained models, *i.e.*, DINO or VideoCutLER, and fine-tuned on the training set with human annotations. We report the instance segmentation metrics and evaluate the model performance on the `val` splits.

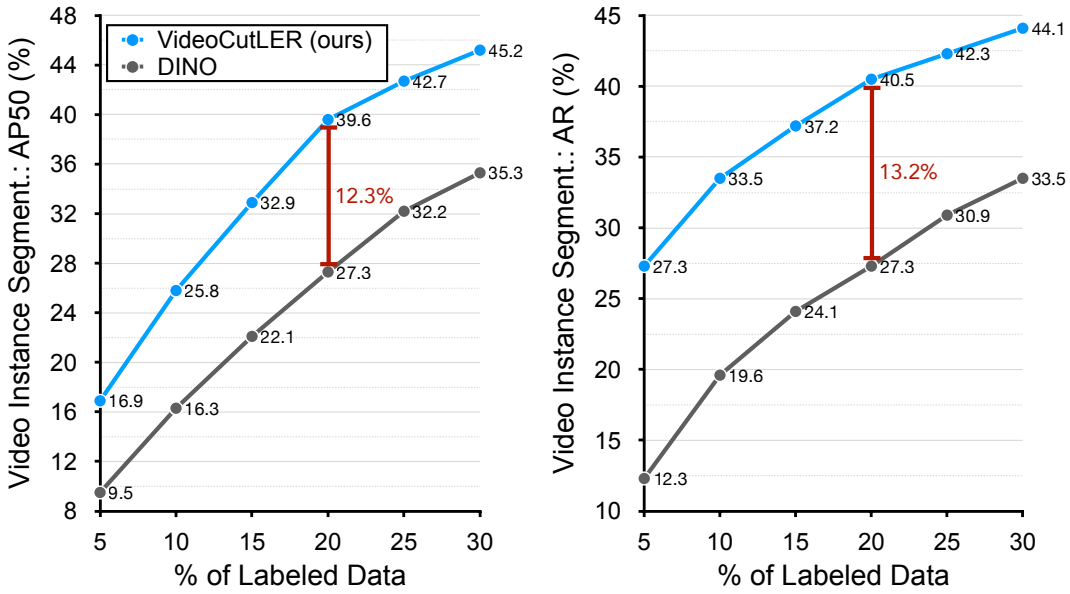


Figure 3.4: We fine-tune VideoCutLER for **semi-supervised video instance segmentation** on the YouTubeVIS-2019 dataset, using different percentages of labeled training data. We evaluate the performance of our method by reporting the average precision and recall on the validation set of YouTubeVIS-2019. To establish a strong baseline, we use the self-supervised DINO [25] model and initialize the weights of VideoMask2Former with DINO. To ensure a fair comparison, both baselines and VideoCutLER are trained using the same schedule and recipe.

Setup. We use VideoMask2Former with a backbone of ResNet50 for all experiments in this section unless otherwise noted. For our experiments on semi-supervised learning, we randomly sample a subset of videos from the training split with different proportions of labeled videos. After pretraining our VideoCutLER model on ImageNet, we fine-tune the model on the YouTubeVIS-2019 [79] dataset with its human annotations. For our experiments on the fully-supervised learning task, we fine-tune the VideoCutLER model on all available labeled data from the training sets of YouTubeVIS. For baselines, we initialize a VideoMask2Former model with a DINO [25] model pre-trained on ImageNet and fine-tuned on labeled videos. Since DINO has shown strong performance in

detection and segmentation tasks, it serves as a strong baseline for our experiments.

For semi-supervised learning, both the baselines and our models are trained for $2\times$ schedule, with a learning rate of 0.0001 for all model weights, except for the final classification layers, which use a learning rate of 0.0016. We train the models using a batch size of 16 and 8 GPUs. For fully-supervised learning, we use the $1\times$ schedule and a learning rate of 0.0002 for the final classification layers. We evaluate their performance on the `val` split of the YouTubeVIS-2019, and report results from its official evaluation server.

Data for fully-/semi-supervised VIS. We fine-tune the pretrained VideoCutLER model on all or a subset of the training split of YouTubeVIS-2019. We then evaluate the resulting models on the validation set. To ensure a fair comparison, we use the same amount of human annotations to train our model and baselines. Specifically, we initialize the baselines with the DINO-pretrained model and fine-tune them on the training set of the respective dataset. We evaluate the model performance on their validation sets and report results from its official evaluation server.

Results. Most prior approaches on self-supervised representation learning [9], [25], [26], [28], [42] are limited to providing initializations only for the model backbones, with the remaining layers, such as Mask2Former's decoders, being randomly initialized. In contrast, VideoCutLER takes a more comprehensive approach that allows all model weights to be pretrained, resulting in a stronger pretrained model better suited for supervised learning. As a result, as shown in Fig. 3.4 and Table 3.5, our method outperforms these prior works significantly, offering a strong pretrained model for fully-/semi-supervised learning tasks.

Fig. 3.4 shows that VideoCutLER consistently outperforms the strong baseline method DINO [25] across all label-efficient learning settings with varying proportions of labeled YouTubeVIS-2019 videos. The most significant performance gains are observed when 20% labeled data is provided, where VideoCutLER exceeds DINO by over 12% AP_{50} and 13.2% AR. As demonstrated in Table 3.5, training the model with all available labeled videos from YouTubeVIS yields considerable performance gains, surpassing DINO by more than 15.9% AP on YouTubeVIS-2019 and 8.8% on YouTubeVIS-2021, respectively.

Ablation Study

Hyper-parameters and design choices. We present an ablation study on several key hyper-parameters and design choices of VideoCutLER in Table 3.6. First, we analyze the impact of varying the size of video frames used for training VideoCutLER. From Table 3.6a, we observe that the shortest edge length of 240 pixels yields the best performance. Using a larger resolution does not always lead to better results. Next, Table 3.6b shows the effect of the number of frames used for training video instance segmentation models. We found that synthetic videos with three frames are optimal for learning an unsupervised video instance segmentation model. Increasing the number of frames does not result in a further improved performance, aligning with the findings reported in [94]. Furthermore, Table 3.6c investigates the contribution of several augmentation methods, including brightness, rotation, contrast, and random cropping, which are used as default during model training. We found that compared to ImageCut2Video without any data augmentations, adding these augmentations can bring about 3% performance gains.

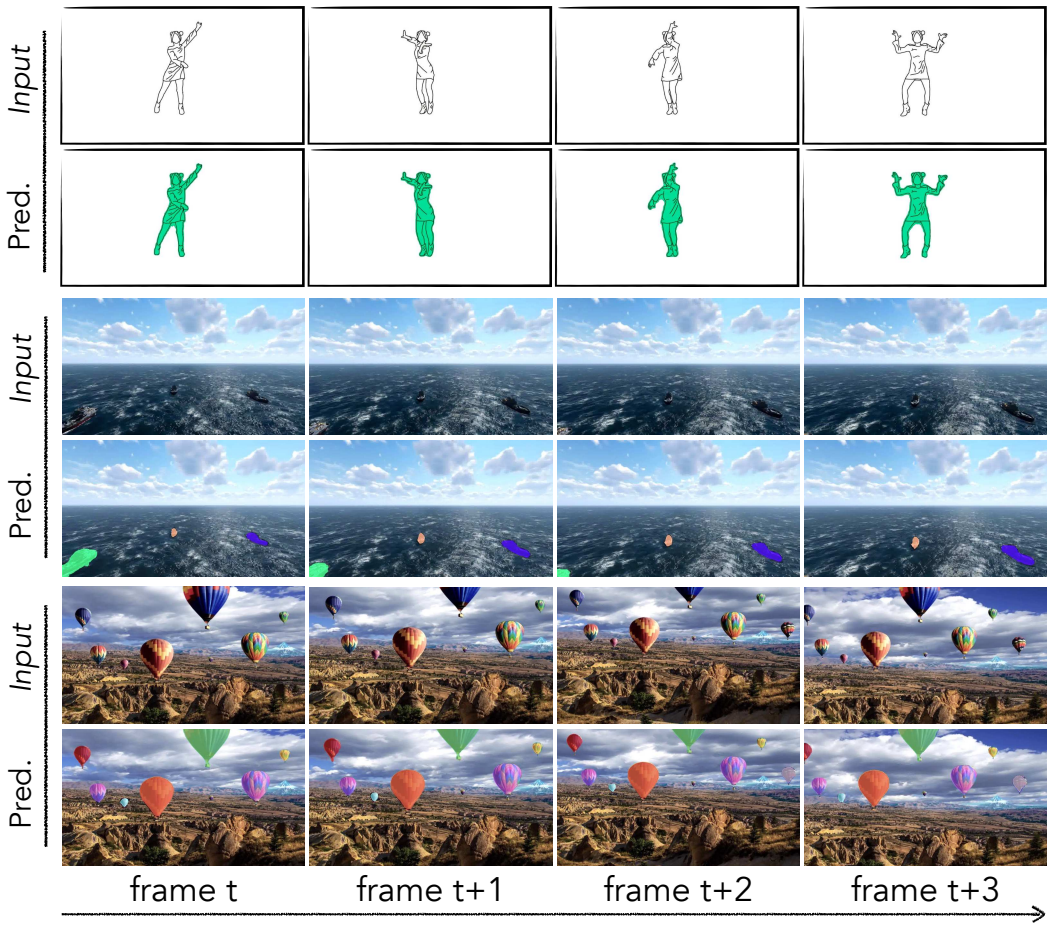


Figure 3.5: We present qualitative results on videos covering a range of **out-of-domain** sources, *e.g.*, sketches, 3D computer-generated imagery (CGI) and hybrid (CGI + realistic). VideoCutLER can produce high-quality segmentation and tracking results for small objects that are often difficult to distinguish from the background, as well as for object sketches that lack textual information.

Generalizability. The results presented in Fig. 3.5 demonstrate that VideoCutLER can effectively perform video instance segmentation on out-of-domain data sources, *e.g.*sketches, 3D computer-generated imagery, and hybrid videos that combine CGI. These results shows that our model can be applied to a broad range of videos beyond the domains it was initially trained on, *i.e.*, ImageNet.

3.5 Summary

We presented a simple unsupervised approach to segment multiple instances in a video. Our approach, VideoCutLER, does not require labels, and does not rely on motion-based learning signals like optical flow. In fact, VideoCutLER does not need real videos for training as we synthesize videos using natural images from the ImageNet-1K. Despite being simpler, VideoCutLER outper-

Size →	180	360	480
AP_{50}^{video}	49.9	50.7	50.4

(a) Frame size.

# frames →	CutLER† [12]	2	3	4
AP_{50}^{video}	37.5	49.8	50.7	50.4

(b) # frames. z

Augmentations →	none	+bright	+rotation	+contrast	+crop	all
AP_{50}^{video}	47.8	48.1	48.9	48.3	48.7	50.7

(c) Data augmentations for ImageCut2Video.

Table 3.6: Ablations for VideoCutLER. We report video instance segmentation result AP_{50}^{video} on YoutubeVIS-2019. **(a)** The impact of varying video frame sizes on training VideoCutLER. **(b)** The effect of the number of frames used for model training. **(c)** The impact of several augmentation methods, including brightness, rotation, contrast, and random cropping, which are used as default during model training. Default settings are highlighted in gray.

forms models that use additional learning signals or video data, achieving $10\times$ their performance on benchmarks like YouTubeVIS. Moreover, VideoCutLER is a strong pretrained model for supervised learning. We hope that our approach enables both a wide range of applications in video recognition, as well as its simplicity enables easy future research.

Limitations: while VideoCutLER demonstrates its capability to achieve the state-of-the-art performance without relying on optical flow estimations, potential further improvements may be obtained by leveraging natural videos and integrating joint training with optical flow estimations.

Chapter 4

Unsupervised Universal Image Segmentation

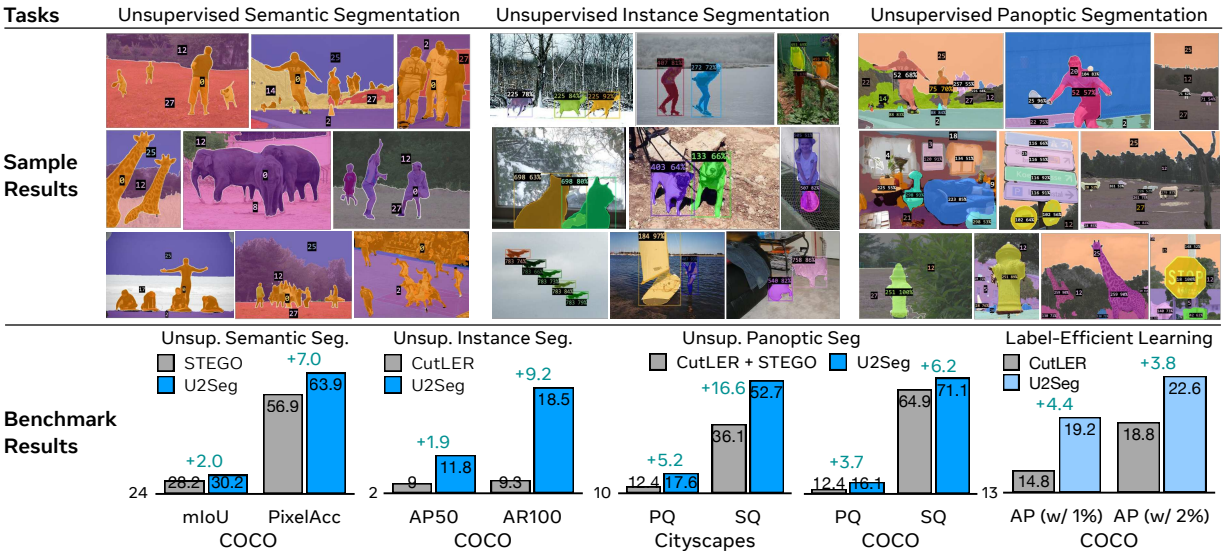


Figure 4.1: We present **U2Seg**, a unified framework for **U**nsupervised **U**niversal image **Seg**mentation that consistently outperforms previous state-of-the-art methods designed for individual tasks: CutLER [12] for unsupervised instance segmentation, STEGO [100] for unsupervised semantic segmentation, and the naive combination of CutLER and STEGO for unsupervised panoptic segmentation. We visualize instance segmentation results with "semantic label" + confidence score and semantic predictions with "semantic label". Zoom in for the best view.

Several unsupervised image segmentation approaches have been proposed which eliminate the need for dense manually-annotated segmentation masks; current models separately handle either semantic segmentation (*e.g.*, STEGO) or class-agnostic instance segmentation (*e.g.*, CutLER), but not both (*i.e.*, panoptic segmentation). We propose an Unsupervised Universal Segmentation model (U2Seg) adept at performing various image segmentation tasks—instance, semantic and

panoptic—using a novel unified framework. U2Seg generates pseudo semantic labels for these segmentation tasks via leveraging self-supervised models followed by clustering; each cluster represents different semantic and/or instance membership of pixels. We then self-train the model on these pseudo semantic labels, yielding substantial performance gains over specialized methods tailored to each task: a +2.6 AP^{box} boost (*vs.* CutLER) in unsupervised instance segmentation on COCO and a +7.0 PixelAcc increase (*vs.* STEGO) in unsupervised semantic segmentation on COCOStuff. Moreover, our method sets up a new baseline for unsupervised panoptic segmentation, which has not been previously explored. U2Seg is also a strong pretrained model for few-shot segmentation, surpassing CutLER by +5.0 AP^{mask} when trained on a low-data regime, *e.g.*, only 1% COCO labels. We hope our simple yet effective method can inspire more research on unsupervised universal image segmentation.

4.1 Introduction

The field of image segmentation has witnessed significant advancements in the recent years [15], [36], [56], [57], [101]–[104]. Nonetheless, the effectiveness of these segmentation methods heavily depends on the availability of extensive densely human-labeled data for training these models, which is both labor-intensive and costly and thus less scalable. In this paper, our objective is to explore the extent to which unsupervised image segmentation can be achieved without relying on any human-generated labels.

Several recent works such as CutLER [12] and STEGO [100] have emerged as promising approaches for unsupervised image segmentation. CutLER leverages the property of the self-supervised model DINO [25] to 'discover' objects without supervision, and learns a state-of-the-art localization model on pseudo instance segmentation masks produced by MaskCut [12] (based on Normalize Cuts [53]). Similarly leveraging DINO [25], STEGO [100] introduces a novel framework that distills unsupervised features into discrete semantic labels. This is achieved using a contrastive loss that encourages pixel features to form compact clusters while preserving their relationships across the corpora [100]. However, these methods have limitations:

- The output of ***unsupervised instance segmentation*** methods such as CutLER [12] comprises *class-agnostic* segments for "things", ignoring the "stuff" categories that represent pixel semantics. Moreover, CutLER often treats several *overlapping instances* as one instance, especially when these instances belong to the same semantic class.
- On the other hand, ***unsupervised semantic segmentation*** methods such as STEGO [100] focus on the segmentation of semantically coherent regions, lacking the capability to distinguish between individual instances.
- ***Unsupervised panoptic segmentation*** has not been addressed. Supervised panoptic segmentation methods [103], [105], [106] predict both "stuff" and "things" classes simultaneously; to the best of our knowledge there has not been work on unsupervised panoptic segmentation heretofore.

To address these limitations, we propose **U2Seg**, a novel ***U***nsupervised ***U***niversal image ***Seg***mentation model. U2Seg offers comprehensive scene understanding–instance, semantic and panoptic–without

relying on human annotations, segmenting semantically meaningful regions in the image as well as identifying and differentiating between individual instances within those regions.

U2Seg is comprised of three steps. First, we create high-quality, discrete semantic labels for instance masks obtained from MaskCut and DINO, by clustering semantically similar instance masks into distinct fine-grained clusters, as in Sec. 4.3. Next, we amalgamate the semantically pseudo-labeled "things" pixels (from the first step) with "stuff" pixels (from STEGO) to produce pseudo semantic labels for each pixel in the image. Lastly, a universal image segmentation model is trained using these pseudo-labels, resulting in a model capable of simultaneously predicting pixel-level (*i.e.*, semantic and class-agnostic instance segmentation) and instance-level semantic labels, detailed in Sec. 4.3.

Despite the inherent noise in these pseudo-labels, self-training the model with them yields substantial performance gains over specialized methods tailored to each task: U2Seg achieves a **+2.6 AP^{box}** boost (*vs.* CutLER) in unsupervised instance segmentation on COCO and a **+7.0 PixelAcc** increase (*vs.* STEGO) in unsupervised semantic segmentation on COCOStuff. Moreover, our method sets up a new baseline for unsupervised panoptic segmentation. We also find that the multi-task learning framework and learning unsupervised segmentor with semantic labels enable our model to generate a more discriminative feature space, which makes it a superior representation for downstream supervised detection and segmentation tasks. When trained on a low-data regime, such as 1% COCO labels, U2Seg surpasses CutLER by **+5.0 AP^{mask}**.

Contributions. Our main contribution is the first universal unsupervised image segmentation model that can tackle unsupervised semantic-aware instance, semantic and panoptic segmentation tasks using a unified framework. We establish a suite of benchmarks on unsupervised semantic-aware instance segmentation and panoptic segmentation, areas previously unexplored. Despite using a single framework, we demonstrate that U2Seg surpasses previous methods specialized for each task across all experimented benchmarks (instance, semantic, panoptic, *etc*) and datasets (COCO, Cityscapes, UVO, VOC, *etc*).

4.2 Related Work

Self-supervised Representation Learning focuses on feature learning from a large amount of unlabeled data without using human-made labels. *Contrastive Learning-Based Methods* [26], [28], [40], [41] learn representation by comparing similar instances or different versions of a single instance while separating dissimilar ones. *Similarity-Based Self-Supervised Learning* [42], [43] mainly reduces differences between different augmented versions of the same instance. *Clustering-Based Feature Learning* [9], [44]–[46] finds natural data groups in the hidden space. *Masked Autoencoders* [47]–[49] learn by masking and then reconstructing masked parts of the image.

Unsupervised Object Detection and Instance Segmentation. DINO [25] shows that self-supervised learning (SSL) Vision Transformers (ViT) [29] can reveal hidden semantic segmentation in images, which is not obvious in supervised counterparts [25], [51]. Extending this, LOST [32], TokenCut [33] and MaskDistill [39] use DINO's patch features to identify main objects in images. FreeSOLO [23] performs unsupervised class-agnostic instance segmentation by creating coarse

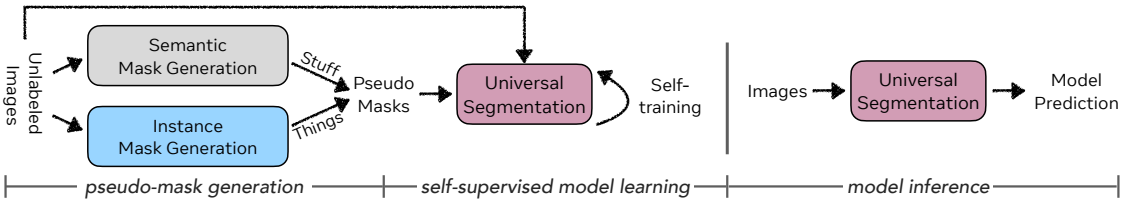


Figure 4.2: Overview of the training and inference pipeline for the proposed Unsupervised Universal Segmentation model (U2Seg) adept at performing various image segmentation tasks—instance, semantic and panoptic—using a novel unified framework.

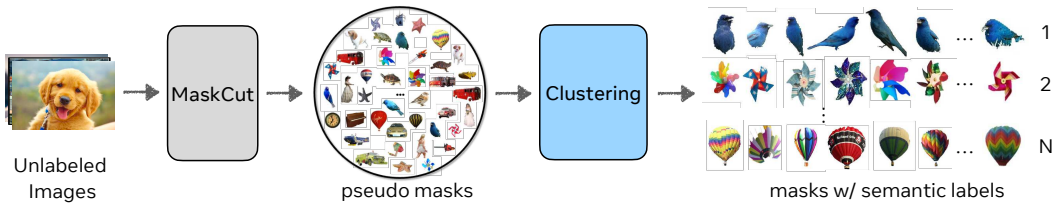


Figure 4.3: Pipeline overview for generating masks and their semantically meaningful pseudo labels in semantic-aware instance segmentation. We first use MaskCut to generate class-agnostic instance masks, which are then grouped into semantically meaningful clusters. These pseudo semantic labels are used for training a semantic-aware instance segmentor.

masks first, which are later improved through self-training. Meanwhile, CutLER [107] introduces the MaskCut method, which aims to identify multiple instances in a single image. Yet, MaskCut frequently consolidates overlapping instances into a single segment and lacks the capability to assign semantic labels to each instance.

Unsupervised Semantic Segmentation. IIC [108] maximizes mutual information for clustering, while PiCIE [109] uses invariance to photometric effects and equivariance to geometric transformations for segmentation. MaskContrast [110] learns unsupervised semantic segmentation by contrasting features within saliency masks. STEGO [100] refines pretrained SSL visual features to distill correspondence information embedded within these features, thereby fostering discrete semantic clusters.

Universal Segmentation has been introduced to deliver instance, semantic and panoptic segmentation tasks using a unified architecture [94], [103], [106], [111]–[117]. In this work, we propose U2Seg to tackle this challenging task without relying on human-annotated data.

Unsupervised Image Classification methods mainly focus on providing a semantic label for each query image that can be mapped to ground truth classes by hungarian matching. SCAN [118] proposes a three-stage pipeline that includes representation learning, deep clustering, and self-labeling. NNM [119] enhances SCAN by incorporating local and global nearest neighbor matching. RUC [120] further improves SCAN using a robust loss as training objective. However, these approaches only provide one classification prediction per image, whereas our method provides classification per-instance for instance segmentation and per-pixel for semantic segmentation.

4.3 Unsupervised Universal Segmentation

Preliminaries

We first explain the previous Unsupervised Instance Segmentation method CutLER [12], and Unsupervised Semantic Segmentation method STEGO [100].

CutLER [12] exploits self-supervised learning models like DINO [25] to 'discover' objects and train a state-of-the-art detection and segmentation model using a cut-and-learn pipeline. It first uses MaskCut to extract multiple initial masks from DINO [25] features. MaskCut first generates a patch-wise affinity matrix $W_{ij} = \frac{K_i K_j}{\|K_i\|_2 \|K_j\|_2}$ using the "key" features K_i for patch i from DINO's last attention layer. Subsequently, the cut-based clustering method Normalized Cut [53] is employed on the affinity matrix by finding the eigenvector x that corresponds to the second smallest eigenvalue. A foreground instance mask M^s is derived through bi-partitioning of the vector x, enabling segmentation of individual objects in the image. For multi-instance segmentation, MaskCut iteratively refines the affinity matrix by masking out already segmented objects, allowing for subsequent extractions

$$W_{ij}^t = \frac{(K_i \prod_{s=1}^t M_{ij}^s)(K_j \prod_{s=1}^t M_{ij}^s)}{\|K_i\|_2 \|K_j\|_2} \tag{4.1}$$

and repeating above steps by N times. CutLER then refines detection and segmentation through a loss-dropping strategy and iterative self-training.

STEGO [100] harnesses the semantically rich feature correlations produced by unsupervised methods like DINO [25] for segmentation. It trains a segmentation head to refine these correlations within an image, with its K-Nearest Neighbors (KNNs), and across randomly chosen images. Specifically, STEGO distills DINO's unsupervised features into distinct semantic labels by optimizing a correspondence loss. This loss function measures the feature correspondences F^{SC} between image feature pairs generated by DINO and the feature correspondence S_{hwij} derived from a trainable, lightweight segmentation head [100]:

$$L_{\text{corr}}(x, y, b) = -\sum_{hwij} (F_{hwij}^{SC} - b) \max(S_{hwij}, 0) \tag{4.2}$$

Unsupervised Instance Segmentation

Although CutLER [12] provides high-quality instance segmentation masks without human annotations, the predicted masks are class-agnostic, and thus do not include semantic labels for each instance. Our method addresses this issue by grouping the detected instances with a clustering method. In this way, instances assigned to the same cluster are associated with identical or closely related semantic information, while instances residing in separate clusters exhibit semantic dissimilarity.

Pseudo Semantic Labels. To train a detection and instance segmentation model, we vector quantize the model targets (pseudo semantic labels) by clustering the instance-level features of the entire

dataset, under constraints derived from self-supervision. Specifically, our approach starts with the generation of instance segmentation masks using MaskCut [12]. Subsequently, we utilize the efficient K-Means clustering method as implemented in USL [11] to cluster all segmentation masks into semantically meaningful clusters.

We employ K-Means clustering to partition n instances into $C(\leq n)$ clusters, where each cluster is represented by its centroid c [121], [122]. Each instance is assigned to the cluster with the nearest centroid. Formally, we conduct a C-way node partitioning, denoted as $\mathcal{S} = S_1, S_2, \ldots, S_C$, that minimizes the within-cluster sum of squares [123]:

$$\min_{\mathcal{S}} \sum_{i=1}^{C} \sum_{V \in S_i} |V - c_i|^2 = \min_{\mathcal{S}} \sum_{i=1}^{C} |S_i| \text{Var}(S_i) \tag{4.3}$$

This optimization process is carried out iteratively using the EM algorithm [124], starting from selecting random samples as initial centroids. As a result, this process assigns pseudo semantic labels, denoted as y_i, to each instance i, with y_i falling within the range of $[1, C]$.

The resulting semantic labels serve multiple purposes: ***1) Semantic-aware copy-paste augmentation***, which significantly improves CutLER's capability to differentiate overlapping instances, especially when they share similar semantic information. ***2) Training instance segmentation models***: They serve as pseudo ground-truth labels for training a non-agnostic instance segmentor.

Semantic-aware Copy-Paste Augmentation. In cluttered natural scenes, previous unsupervised instance segmentation model often fail to distinguish instances from the same semantic class. This results in multiple instances being captured in the same mask. To distinguish multiple overlapping objects and small objects in existing unsupervised detectors, we employ semantic-aware copy-paste augmentation, which includes several steps:

1) We begin by randomly selecting two instances, denoted as I_1 and I_2, both belonging to the same pseudo-category (or group/cluster). ***2)*** One of these instances undergoes a transformation function $\mathcal{T}$, which randomly resizes and shifts the associated pseudo-masks. ***3)*** The resized instance is then pasted onto another image, creating synthetic overlapping cases using the following equation:

$$I_3 = I_1 \cdot (1 - \mathcal{T}(M_c)) + I_2 \cdot \mathcal{T}(M_c) \tag{4.4}$$

where $\cdot$ denotes element-wise multiplication.

Learning Unsupervised Instance Segmentor. Traditionally, unsupervised segmentation community focused primarily on class-agnostic instance segmentation [12], [23], [33], whose outputs lack class labels. However, by incorporating clustering information obtained from pseudo-labels on ImageNet, as discussed above, our method allows the model to predict not only the location and segmentation of an object but also its pseudo semantic labels.

As observed by [12], "ground-truth" masks may miss instances. However, a standard detection loss penalizes predicted regions r_i that do not overlap with the "ground-truth". Therefore, following [12], we drop the loss for each predicted region r_i that has a maximum overlap of τ^{IoU} with any of the 'ground-truth' instances: $\mathcal{L}_{\text{drop}}(r_i) = \mathbb{1}(\text{IoU}_i^{\text{max}} > \tau^{\text{IoU}})\mathcal{L}_{\text{vanilla}}(r_i)$, where $\text{IoU}_i^{\text{max}}$ denotes the maximum IoU with all 'ground-truth' for r_i and $\mathcal{L}_{\text{vanilla}}$ is the vanilla loss function of detectors. $\mathcal{L}_{\text{drop}}$ encourages the exploration of image regions missed in the "ground-truth".

Task →	Agn Instance Seg.		Instance Seg.						Semantic Seg.		Panoptic Seg.					
Datasets →	COCO		COCO		VOC		UVO		COCO		COCO			Cityscapes		
Metric →	AP^{box}	AP^{box}_{50}	AP^{box}_{50}	AR^{box}_{100}	AP^{box}_{50}	AR^{box}_{100}	AP^{box}_{50}	AR^{box}_{100}	PixelAcc	mIoU	PQ	SQ	RQ	PQ	SQ	RQ
FreeSOLO [23]	9.6	4.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-
TokenCut [33]	5.8	3.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-
DINO [25]	-	-	-	-	-	-	-	-	30.5	9.6	-	-	-	-	-	-
PiCIE + H [109]	-	-	-	-	-	-	-	-	48.1	13.8	-	-	-	-	-	-
STEGO [100]	-	-	-	-	-	-	-	-	56.9	28.2	-	-	-	-	-	-
CutLER [107]	21.9	12.3	-	-	-	-	-	-	-	-	-	-	-	-	-	-
CutLER+	-	-	9.0	10.3	26.8	27.2	10.6	11.8	-	-	-	-	-	-	-	-
CutLER+STEGO	-	-	-	-	-	-	-	-	-	-	12.4	64.9	15.5	12.4	36.1	15.2
U2Seg	**22.8**	**13.0**	**11.8**	**21.5**	**31.0**	**48.1**	**10.8**	**25.0**	**63.9**	**30.2**	**16.1**	**71.1**	**19.9**	**17.6**	**52.7**	**21.7**
vs.prev.SOTA	**+0.9**	**+0.7**	**+2.8**	**+11.2**	**+4.2**	**+20.9**	**+0.2**	**+13.2**	**+7.0**	**+2.0**	**+3.7**	**+6.2**	**+4.4**	**+5.2**	**+16.6**	**+6.5**

Table 4.1: With a unified framework, U2Seg outperforms previous state-of-the-art methods tailored for individual tasks across various datasets, including CutLER for unsupervised instance segmentation, STEGO for unsupervised semantic segmentation, and CutLER+STEGO for unsupervised panoptic segmentation. "Agn Instance Seg" denotes class-agnostic instance segmentation.

Unsupervised Universal Image Segmentation

Pseudo Labels for Panoptic Segmentation. For each pixel (i, j) in the image, we vector quantize pixels with different semantics or instance membership, generating pseudo semantic labels for panoptic segmentation. We assign each pixel a semantic label based on "stuff" or "things" identity. This results in an instance label $(I(i, j))$ for "things" or a semantic label $(S(i, j))$ for "stuff". The critical challenge in this process is distinguishing between pixels associated with "things" (countable, often foreground) and "stuff" (uncountable, usually background) [125].

To resolve this problem, our method unfolds in three steps: ***1) Semantic Labeling for "Things":*** Utilizing the class-agnostic instance segmentation capabilities of CutLER [12], we first identify "things" within an image, generating class-agnostic instance masks. These masks then undergo deep clustering to attribute a semantic label $I_C(i, j)$ to each instance, detailed in Sec. 4.3. ***2) Semantic Labeling for "Stuff":*** For "stuff" pixels, we deploy the unsupervised semantic segmentation model STEGO [100], which distills DINO's unsupervised features into discrete semantic labels, as outlined in 4.3. This step assigns a "stuff" semantic label to all pixels, including those of "Things" identified earlier. ***3) Integrating Labels for "Things" and "Stuff".*** We determine a pixel's classification as "things" or "stuff" using the following logic:

$$I(i,j) = \begin{cases} I_C(i,j), & \text{if } I_C(i,j) \neq 0 \\ S_S(i,j), & \text{if } I_C(i,j) = 0 \ \& \ S_S(i,j) \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.5}$$

This process merges the semantic labels, assigning priority to "things" labels over "stuff" where applicable. We then train a universal segmentation model on these pseudo-labels for instance, semantic and panoptic segmentation tasks.

Learning Unsupervised Universal Image Segmentor. After we obtain the pseudo labels for panoptic segmentation, following [113], we construct an unsupervised universal image segmen-

tation model, that has two branches: instance segmentation branch and semantic segmentation branch, to address corresponding segmentation tasks. The model is trained jointly for both branches, employing the following loss function: $\mathcal{L} = \lambda_i(\mathcal{L}_c+\mathcal{L}_b+\mathcal{L}_m)+\lambda_s\mathcal{L}_s$, where $\mathcal{L}_c$ represents the classification loss, $\mathcal{L}_b$ is the detection loss, $\mathcal{L}_m$ is the segmentation loss, and $\mathcal{L}_s$ signifies the semantic loss. The $\mathcal{L}_s$ is computed as a per-pixel cross-entropy loss between the predicted and ground-truth labels. The hyperparameters λ_i and λ_s balance these two parts.

4.4 Experiments and Results

Experimental Setup

Training Data. Our model is trained on 1.3M unlabeled images from ImageNet [126] and is evaluated directly across various benchmarks, unless otherwise noted. For unsupervised semantic segmentation comparisons with STEGO [100], we additionally fine-tune our model using MSCOCO's unlabeled images, following STEGO [100].
Test Data. For unsupervised instance segmentation, we test our model on COCO `val2017`, PASCAL VOC `val2012` [67] and UVO `val` [127]. For unsupervised panoptic segmentation, we evaluate our model on COCO `val2017` and Cityscapes `val` [128].
Evaluation Metrics. We use AP, AP_{50}, AP_{75} and AR_{100} to evaluate the unsupervised instance segmentation; PixelAcc and mIoU for unsupervised semantic segmentation; PQ, RQ, SQ for unsupervised universal image segmentation. After predicting the instance with its semantic labels, we use Hungarian matching to map the semantic labels to class names in the real dataset (details in 4.5). It evaluates the consistency of the predicted semantic segments with the ground truth labels, remaining unaffected by any permutations in the predicted class labels.
Implementation Details. Following [129], we employ Panoptic Cascade Mask R-CNN [57], [129] with a ResNet50 backbone [58]. Following CutLER's training recipe [12], our model, initialized with DINO pre-trained weights, is trained on unlabeled ImageNet for two epochs. It starts with an initial learning rate of 0.01, which then decreases after the first epoch to 5×10^{-5}, with a batch size of 16 for all models. For unsupervised panoptic segmentation, we maintain the same training schedule as unsupervised instance segmentation for zero-shot evaluation. In non-zero-shot scenarios, the models undergo training on a combination of unlabeled COCO and ImageNet datasets, beginning with a learning rate of 0.01 over 90k steps.

Unsupervised Universal Image Segmentation

To the best of our knowledge, U2Seg represents the first effort in addressing unsupervised *semantic-aware* instance, semantic and panoptic segmentation, all unified under a single framework. Due to the absence of benchmarks for unsupervised semantic-aware instance segmentation and panoptic segmentation, we establish comprehensive benchmarks and baselines for both tasks.

In Table 4.1, we demonstrate that U2Seg, utilizing a unified framework, significantly outperforms all previous approaches across various benchmarks and datasets. For ***class-agnostic un-***

Metric	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AR^{box}_{100}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	AR^{mask}_{100}
CutLER+	5.9	9.0	6.1	10.3	5.3	8.6	5.5	9.3
U2Seg	7.3	11.8	7.5	21.5	6.4	11.2	6.4	18.5
Δ	**+1.4**	**+2.8**	**+1.4**	**+11.2**	**+1.1**	**+2.6**	**+0.9**	**+9.2**

Table 4.2: The results for **zero-shot unsupervised object detection and instance segmentation** on COCO `val2017`. The model is trained on ImageNet with a cluster number of 800. We compare it with CutLER+, a combination of CutLER and offline clustering.

Methods	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AR^{box}_{100}
CutLER+	17.1	26.8	18.1	27.2
U2Seg	19.0	31.0	19.5	48.1
Δ	**+1.9**	**+4.2**	**+1.4**	**+20.9**

Table 4.3: The results for **zero-shot unsupervised object detection on PASCAL VOC `val2012`**. The model is trained on ImageNet with a cluster number of 800. We compare it with CutLER+, a combination of CutLER and offline clustering.

Metric	AP^{box}	AP^{box}_{50}	AR^{box}_{100}	AP^{mask}	AP^{mask}_{50}	AR^{mask}_{100}
CutLER+	6.3	10.6	11.8	6.0	9.0	10.4
U2Seg	6.8	10.8	25.0	6.2	9.5	21.0
Δ	**+0.5**	**+0.2**	**+13.2**	**+0.2**	**+0.5**	**+10.6**

Table 4.4: The results for **zero-shot unsupervised object detection and instance segmentation on UVO `val`**. The model is trained on ImageNet with a cluster number of 800. We compare with CutLER+, a combination of CutLER and offline clustering.

supervised instance segmentation, our method achieves an increase of **+0.9** in AP^{box} compared to CutLER [12]. This improvement is largely attributed to our novel semantic-aware copy-paste augmentation, as detailed in Sec. 4.3. For ***unsupervised semantic-aware instance segmentation***, we benchmark against the advanced baseline CutLER+, derived from CutLER, and record a substantial gain of over 11.2% in AR. A more comprehensive analysis of these results is provided in Sec. 4.4. For ***unsupervised semantic segmentation***, our approach surpasses the state-of-the-art STEGO with impressive margins of **+7.0** in **PixelAcc** and **+2.0** in **mIoU**. Lastly, for ***unsupervised panoptic segmentation***, we compare against the strong baseline of CutLER+STEGO, a hybrid of CutLER+ and STEGO, and observe performance gains of over 6.2% in SQ on MSCOCO and a notable 16.6% improvement in SQ on Cityscapes. Further comparisons and discussions on this task are elaborated in Sec. 4.4.

Unsupervised Instance Segmentation

We performed extensive experiments for zero-shot unsupervised instance segmentation. Given that prior methods [12], [23], [31], [33] are limited to class-agnostic instance segmentation, we

Figure 4.4: Universal image segmentation visualization in COCO `val2017`. We present the results with cluster IDs predicted by U2Seg, categorizing athletes playing hockey (left columns) as "139", those playing badminton (middle columns) as "52" and gentlemen (right columns) as "132". After Hungarian matching, the IDs are automatically matched to the category "person" for quantitative evaluations.

Methods	Pretrain	PQ	SQ	RQ
	zero-shot methods			
U2Seg	IN	15.7	46.6	19.8
	non zero-shot methods			
CutLER+STEGO	COCO	12.4	36.1	15.2
U2Seg	COCO	15.4	51.5	19.0
U2Seg	COCO+IN	17.6	52.7	21.7
Δ		**+5.2**	**+16.6**	**+6.5**

Table 4.5: Unsupervised Panoptic image segmentation on Cityscapes `val`. We show PQ, SQ and RQ on zero-shot and non-zero shot settings with the cluster number of 800. We compare with CutLER+STEGO, a combination of CutLER+ and STEGO.

developed CutLER+, a strong baseline for unsupervised semantic-aware instance segmentation, building upon the current state-of-the-art CutLER [12]. CutLER+ operates in two steps: it first uses the pre-trained CutLER to generate class-agnostic instance masks, and subsequently assigns semantic labels to all instance masks through offline clustering.

Table 4.2 demonstrates that U2Seg markedly improves performance in both unsupervised object detection and instance segmentation on MSCOCO, delivering a **+2.8** boost in AP^{box}_{50} and a **+2.6** rise in AP^{mask}_{50} over CutLER+. Additionally, our method sees a substantial increase of approximately **+10.0** in AR_{100}. Results on PASCAL VOC `val2012` and UVO `val` are detailed in Table 4.3 and Table 4.4, respectively. Notably, we achieve gains exceeding **+20%** in AR for PASCAL VOC and **+10%** for UVO.

Unsupervised Panoptic Segmentation

For unsupervised panoptic/universal image segmentation, our experiments span two scenarios. In the zero-shot setting, the model is trained exclusively on unlabeled ImageNet images. For non-zero-shot (in-domain) scenarios, we train on unlabeled COCO images or a mix of COCO and ImageNet. With no existing benchmarks for unsupervised panoptic segmentation, we establish a new baseline by integrating the state-of-the-art unsupervised semantic segmentation from STEGO [100]

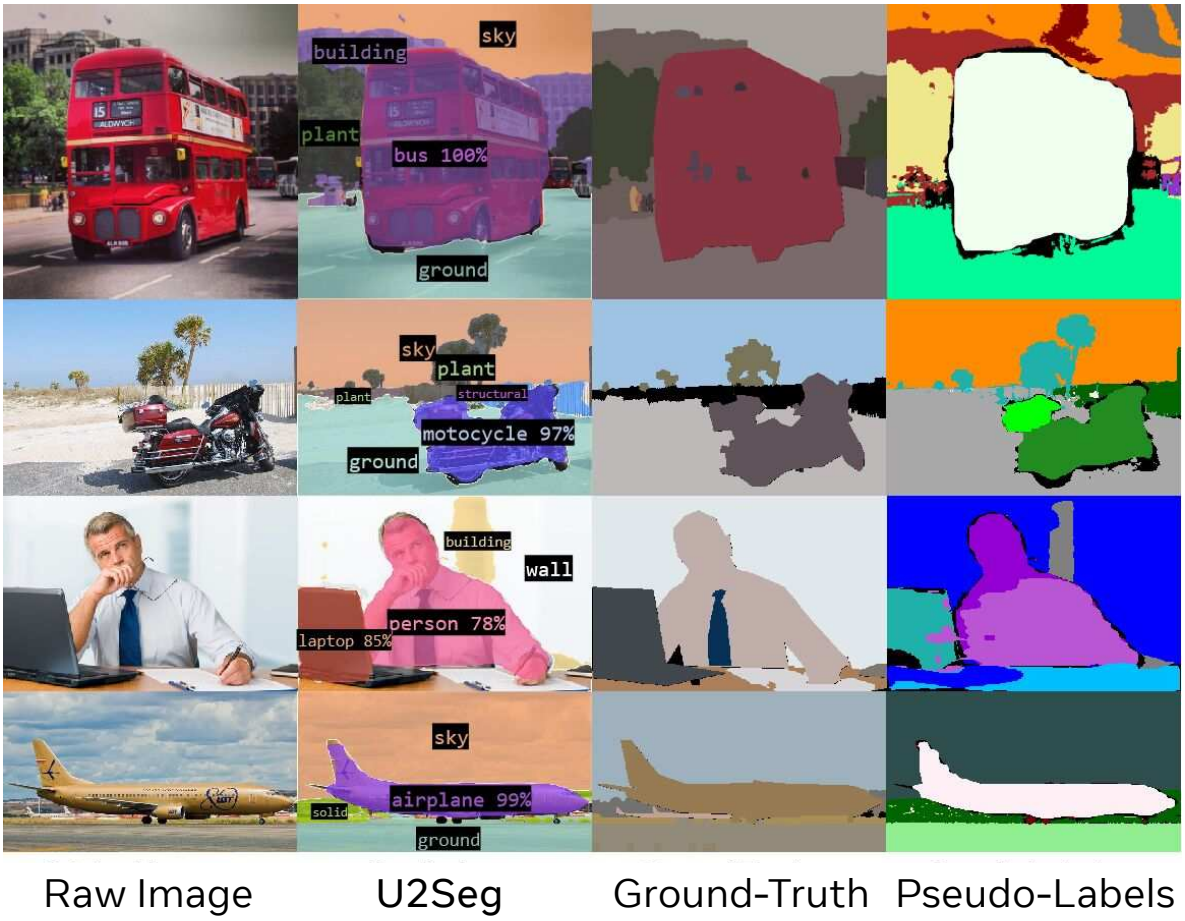


Figure 4.5: Visualizations of U2Seg's **unsupervised Panoptic segmentation** results on COCO `val2017` (**after Hungarian matching**). The pseudo label is the naive combination of previous state-of-the-art instance segmentation, *i.e.*CutLER [12], and semantic segmentation, *i.e.*, STEGO [100], results.

Methods	Pretrain	PQ	SQ	RQ
	zero-shot methods			
U2Seg	IN	11.1	60.1	13.7
	non zero-shot methods			
CutLER+STEGO	COCO	12.4	64.9	15.5
U2Seg	COCO	15.3	66.5	19.1
U2Seg	COCO+IN	16.1	71.1	19.9
Δ		**+3.7**	**+6.2**	**+4.4**

Table 4.6: Unsupervised Panoptic image segmentation on COCO `val2017`. We show PQ, SQ and RQ on zero-shot and non-zero shot settings. We use CutLER+STEGO, a combination of CutLER+ and STEGO, as a strong baseline.

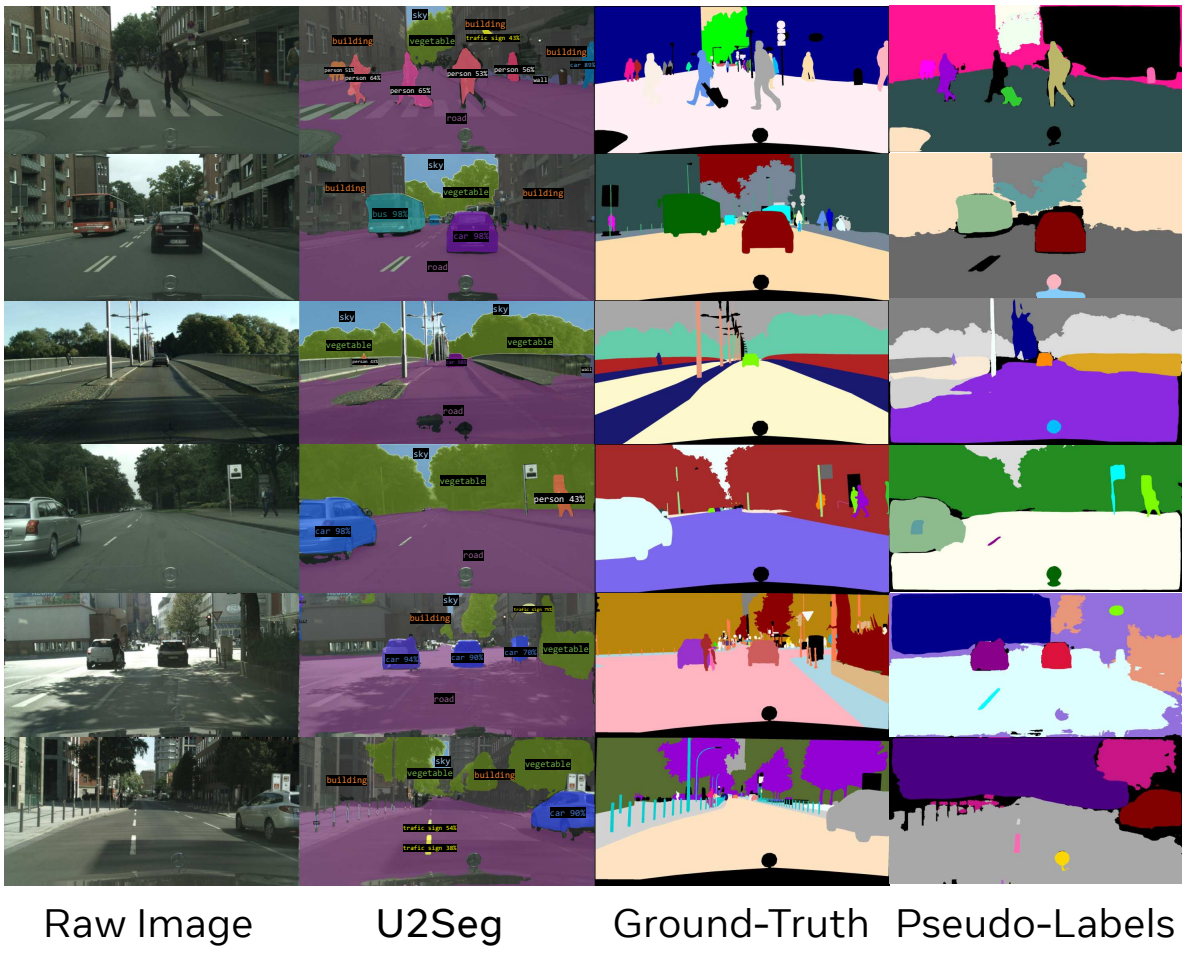


Figure 4.6: Qualitative results of U2Seg's **Panoptic image segmentation** results on Cityscapes `val` (**after Hungarian matching**).

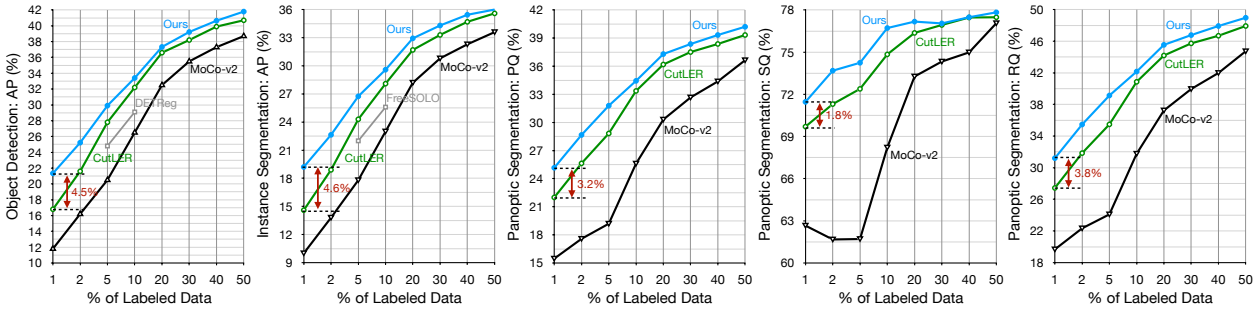


Figure 4.7: We evaluate the **label-efficient learning** performance on 3 different tasks: object detection (the left), instance segmentation (the second left) and panoptic image segmentation (the last three).

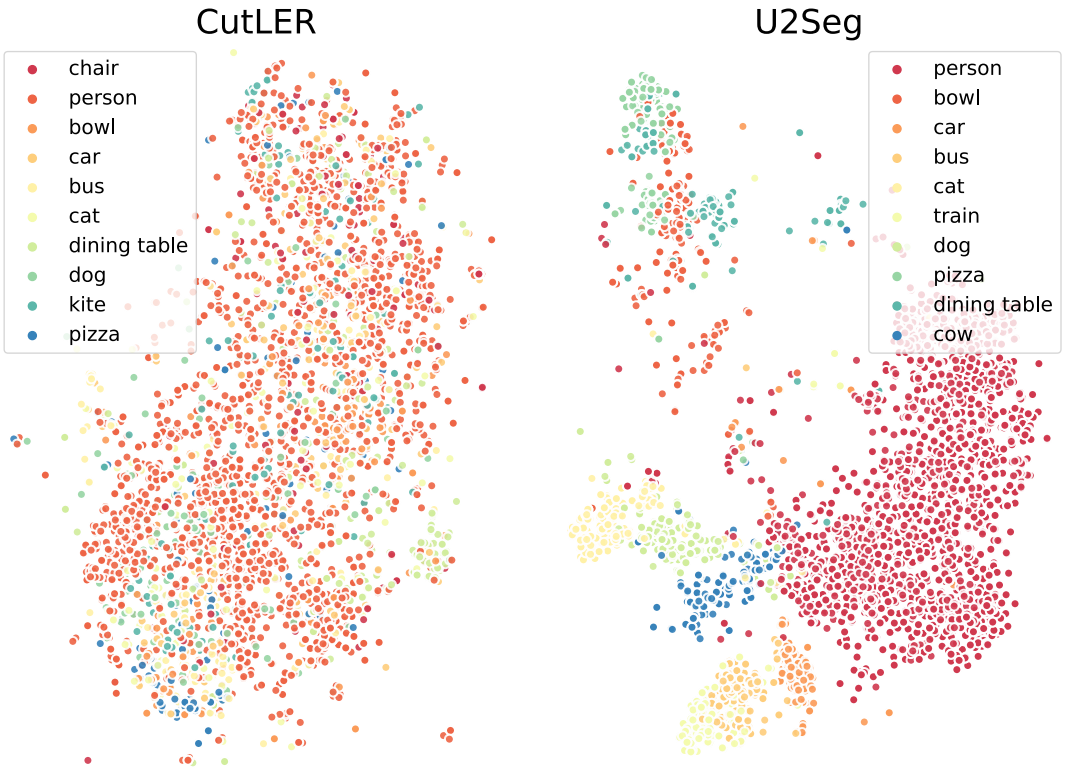


Figure 4.8: U2Seg learns features that are more discriminative than those learned by CutLER. The **t-SNE** [130] visualization of the features from the model's FC layer. We color-code each dot based on its ground-truth category.

with semantic-aware instance segmentation from CutLER+ (discussed in Sec. 4.4), which are then merged to create panoptic/universal segmentation outcomes, referred to as CutLER+STEGO.

Table 4.6 presents the PQ, SQ, and RQ scores of U2Seg on COCO `val2017`. U2Seg surpasses the strong baseline CutLER+STEGO with a **+3.5** improvement in PQ and an increase of over **+4.0** in RQ. Qualitative results of U2Seg's performance is provided in Fig. 4.4, with the predicted semantic labels visualized. The qualitative results suggest that an over-clustering strategy in pseudo-label generation, *e.g.*setting the number of clusters to 300 or 800, leads to highly granular model predictions. For instance, as in Fig. 4.4, the model distinctly categorizes hockey players as "139", badminton players as "52", and gentlemen in suits as "132", showcasing its potent discriminative capabilities.

To quantitatively measure the quality of segmentation masks and their corresponding semantic labels, we use Hungarian matching (detailed in Sec. 4.5) to align semantic labels with the category names from the test dataset; for instance, all three sub-clusters depicted in Fig. 4.4 are assigned to the "person" category. The qualitative outcomes post-Hungarian matching are shown in Fig. 4.5, where our model demonstrates superior panoptic segmentation mask quality. For instance, while the baseline tends to segment parts separately (as seen with the man's head and torso being treated

# Cluster	COCO		UVO		VOC	
	AP^{box}_{50}	AR^{box}_{100}	AP^{box}_{50}	AR^{box}_{100}	AP^{box}_{50}	AR^{box}_{100}
300	9.3	20.1	9.8	22.6	29.6	45.7
800	11.8	21.5	10.8	25.0	31.0	48.0
2911	13.3	22.1	15.1	25.8	31.6	48.3

Table 4.7: Over-clustering can improve the model performance. We show results on different datasets for the unsupervised object detection using **different cluster numbers**.

as separate entities in the third row), our model correctly identifies them as parts of a single object. This level of recognition is also evident with the "trunks of the motorcycle" example in the second row. For additional results, please see Sec. 4.5. We also present results of the more challenging dataset Cityscapes in Table 4.5 and Fig. 4.6.

Efficient Learning

Specifically, for object detection and instance segmentation, we employ our unsupervised instance segmentation model, with cluster count set to 300, to initialize the model weights. We adopt the recipe from [107], [131] for model fine-tuning across various annotation splits. For label-efficient panoptic segmentation, we fine-tune the model initialized with our zero-shot unsupervised framework on the same data splits.

The results are depicted in Fig. 4.7, where our model's instance segmentation performance is benchmarked against MoCo-V2, DETReg, and CutLER. Our model consistently surpasses the state-of-the-art with consistent gains in both AP^{box} and AP^{mask}. In scenarios with panoptic image segmentation as the downstream task, we contrast our results with MoCo-V2 and CutLER in terms of PQ, SQ, and RQ metrics. The results illustrate a remarkable improvement, effectively doubling the performance boost from MoCo-V2 to CutLER, especially in few-shot contexts with limited annotations (1% or 2% labeled samples). This highlights the practical value of our approach in real-world unsupervised learning applications, where annotations are often scarce.

We attribute the performance gains primarily to the discriminative features our model learns, as in Fig. 4.8, obtaining effective model initialization for few-shot learning.

Ablation Studies

In this section, we conduct ablation study on U2Seg.

Numbers of clusters. The choice of cluster quantity significantly affects the model's representation granularity. Our ablation study on various cluster counts, as detailed in Table 4.7, reveals their impact on model performance. Over-clustering generally leads to a finer level of detail, prompting the model to learn more discriminative features.

Hungarian matching. As our trained model could predict the instance with corresponding semantic labels, we are able to go further beyond unsupervised class-agnostic instance segmentation. To quantitatively evaluate the performance, Hungarain matching is employed to match the predicted

conf	# matched	AP^{box}_{50}	AR^{box}_{100}
0.9	109	10.9	13.1
0.7	225	11.6	18.0
0.6	282	11.8	19.7
0.4	**389**	**11.8**	**21.5**
0.2	513	11.3	21.8
0.0	718	8.6	18.4

(a) Conf's effect on accuracy.

IoU	# matched	AP^{box}_{50}	AR^{box}_{100}
0.9	295	10.8	19.7
0.8	348	11.4	20.7
0.4	414	11.5	21.6
0.2	450	11.5	21.1
0.0	494	9.2	17.7
0.6	**389**	**11.8**	**21.5**

(b) IoU's effect on accuracy.

Table 4.8: Impact of Confidence and IoU on Hungarian Matching Performance: The left table illustrates the outcomes at a fixed IoU of 0.6 while varying the confidence scores. Conversely, the right table displays the results with a constant confidence of 0.4, altering the IoU values. The cluster number is 800.

semantic labels to the ground-truth dataset categories. See Sec. 4.5 for details of the adopted Hungarian matching used in our evaluation. As shown in Table 4.8, the two parameters conf threshold and IoU threshold also affect the precision and recall.

4.5 Appendix Materials

Datasets	Domain	Testing Data	#Images	Instance Segmentation Label
COCO [24]	natural images	`val2017` split	5,000	✓
UVO [66]	video frames	`val` split	21,235	✓
PASCAL VOC [67]	natural images	`trainval07` split	9,963	✗
Cityscapes [128]	urban scenes	`val` split	500	✓

Table 4.9: Summary of datasets used for evaluation.

Datasets used for Evaluation

We provide information about the datasets used in this work as shown in Table 4.9

COCO. The COCO dataset, introduced by [24], is used for object detection and instance segmentation. It has 115,000 training images, 5,000 validation images, and a separate batch of 123,000 unannotated images. We test our unsupervised instance segmentation on the COCO `val2017` set with zero-shot setting. We report results using standard COCO metrics, including average precision and recall for detection and segmentation. Also, for unsupervised universal image segmentation, we test the performance on COCO `val2017`. We report results using panoptic segmentation COCO metrics.

PASCAL VOC. The PASCAL VOC dataset [67] is a widely-used benchmark for object detection. We test our model using the `trainval07` split and adopt COCO-style evaluation metrics.

UVO. The UVO dataset [66] is designed for video object detection and instance segmentation. We test our unsupervised instance segmentation on the UVO `val` split, which includes 256 videos with each one annotated at 30 fps. We remove the extra 5 non-COCO categories which are marked as "other" in their official annotations. For evaluation, we employ COCO-style metrics.
Cityscapes. Cityscapes is a dataset dedicated to semantic urban scene understanding, focusing primarily on semantic segmentation of urban scenes. In our research, we tested our unsupervised universal image segmentation on the Cityscapes `val` splits, using COCO-stype panoptic evaluation metrics.

Hungarian Matching for Unsupervised Segmentation Evaluation

In unsupervised object detection and instance segmentation, category IDs are predicted without referencing any predefined labels. For convenience, we differentiate the predicted category ID of U2Seg as "cluster ID" while keep the ground truth category ID as "category ID" in the following analysis. To evaluate the segmentation performance, particularly concerning category accuracy, an optimal correspondence between the cluster ID and the ground truth category ID is essential. We leverage a multi-to-one Hungarian matching for evaluation of U2Seg.
Hungarain Matching. Given a set of predicted bounding boxes, masks associated with predicted cluster IDs and the corresponding ground truth, the objective is to find the best match from "cluster ID" to "category ID". To do this, we first use the predicted confidence score conf as a threshold to filter the predicted instance, removing the ones with low confidence. Then, for each predicted instance with its cluster ID, we calculate the IoU of the predicted bounding box or mask with all ground truth instances, then select the one whose IoU is bigger than the predefined threshold, regarding it as the ground truth category ID for this cluster ID. After we get these cluster ID and ground truth category ID pairs, we form a histogram for each kind of cluster ID based on its overlap with all kinds of ground truth category ID. The ground truth category ID that appears most frequently in this histogram becomes the mapping for this cluster ID. This process may result in multiple predicted cluster IDs being mapped to the same ground truth category ID, leading to a multi-to-one matching scenario.

In our experiment, the confidence score threshold conf to filter the predicted instance and the IoU threshold to match predicted instance with its ground truth instance are both hyperparameters, some ablations can be found in Sec. 4.6.
Evaluation Implications. The multi-to-one Hungarian matching method provides a systematic and efficient way to assess the performance of unsupervised segmentation models. By mapping predicted cluster ID to their most likely ground truth counterparts, the method ensures that the evaluation reflects the true categorization capability of the model. This, in turn, allows for a fair and consistent comparison across different unsupervised segmentation techniques.

Unsupervised Instance Segmentation

In this section, we provide complete results for the unsupervised instance segmentation of U2Seg. The results are presented over various datasets and classes to furnish a comprehensive evaluation

Datasets	# cluster	IoU	Conf	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{box}_{S}	AP^{box}_{M}	AP^{box}_{L}	AR^{box}_{1}	AR^{box}_{10}	AR^{box}_{100}
UVO	2911	0.6	0.1	9.7	15.1	9.3	0.6	5.2	14.4	18.0	25.3	25.8
	800	0.4	0.1	6.8	10.8	7.2	0.6	2.9	10.2	17.2	24.5	25.0
	300	0.7	0.1	6.5	9.8	6.5	0.8	2.6	9.2	16.0	22.2	22.6
VOC	2911	0.5	0.2	19.2	31.6	19.7	1.0	6.4	26.6	28.6	44.9	48.3
	800	0.8	0.2	19.0	31.0	19.5	0.6	4.8	26.6	28.8	45.2	48.1
	300	0.8	0.4	18.4	29.6	18.8	0.3	3.8	26.0	27.1	41.0	42.8
COCO	2911	0.5	0.3	8.2	13.3	8.4	1.4	7.0	18.2	14.1	21.4	22.1
	800	0.6	0.4	7.3	11.8	7.5	1.2	5.8	15.8	13.3	20.8	21.5
	300	0.6	0.3	5.7	9.3	5.9	0.5	4.6	12.9	11.9	19.5	20.1

Table 4.10: Complete results for unsupervised object detection. We show results on UVO `val`, PASCAL VOC `val2012` and COCO `val2017`, with corresponding clustering numbers. The IoU and Conf are the Hungarian matching parameter we use for evaluation.

Datasets	# cluster	IoU	Conf	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	AP^{mask}_{S}	AP^{mask}_{M}	AP^{mask}_{L}	AR^{mask}_{1}	AR^{mask}_{10}	AR^{mask}_{100}
UVO	2911	0.6	0.1	8.8	13.9	8.4	0.5	6.4	14.4	16.0	21.7	22.1
	800	0.4	0.1	6.2	9.5	6.0	0.5	2.1	9.8	15.7	20.6	21.0
	300	0.7	0.1	6.1	9.5	5.8	0.7	1.0	8.8	14.1	19.2	19.4
COCO	2911	0.5	0.3	7.3	12.4	7.4	0.8	4.9	17.9	12.8	18.7	19.2
	800	0.6	0.4	6.4	11.2	6.4	0.7	3.7	15.0	11.9	18.0	18.5
	300	0.6	0.3	4.9	8.6	5.0	0.3	2.6	11.8	10.7	16.9	17.3

Table 4.11: Complete results for **unsupervised instance segmentation.** We show results on UVO `val` and COCO `val2017`, with corresponding clustering numbers. The IoU and Conf is the Hungarian matching parameter we use for evaluation.

Datasets	Pretrain	# Cluster	PQ	PQ^{St}	PQ^{Th}	SQ	SQ^{Th}	SQ^{St}	RQ	RQ^{Th}	RQ^{St}
COCO	IN	300	11.1	9.5	19.3	60.1	60.3	59.0	13.7	11.6	25.0
	IN	800	11.9	10.5	19.6	65.9	67.4	58.2	14.8	12.8	25.3
	COCO	300	15.3	14.2	21.6	66.5	67.2	62.4	19.1	17.5	27.5
	COCO	800	15.5	14.6	20.5	69.7	71.1	62.6	19.1	17.8	26.1
	IN+COCO	300	15.5	14.4	21.2	67.1	67.7	64.3	19.2	17.8	26.9
	IN+COCO	800	16.1	15.1	21.2	71.1	72.5	63.8	19.9	18.6	26.8
Cityscapes	IN	300	15.3	4.1	23.4	48.8	54.7	44.6	19.5	5.4	29.7
	IN	800	15.7	4.3	24.0	46.6	47.5	45.9	19.8	5.5	30.2
	COCO	300	18.4	7.8	26.1	47.4	47.3	47.4	22.6	9.8	31.9
	COCO	800	15.4	5.8	22.3	51.5	62.9	43.2	19.0	7.5	27.4
	IN+COCO	300	16.5	6.2	24.1	44.1	45.2	43.3	20.5	7.9	29.7
	IN+COCO	800	17.6	8.4	24.2	52.7	67.5	42.0	21.7	10.5	29.9

Table 4.12: Complete results for **unsupervised universal image segmentation.** We show results for different models pretrained on various dataset and test on COCO `val2017`, Cityscapes `val`, with corresponding cluster numbers.

Model	AP^{box}	AP^{box}_{50}	AP^{mask}	AP^{mask}_{50}
CutLER+	5.9	9.0	5.3	8.6
Panoptic	6.1	9.8	5.8	9.0
Instance	7.3	11.8	6.4	11.2

Table 4.13: Limitation of U2Seg. We show the zero-shot unsupervised instance segmentation results on COCO `val2017`. CutLER+ is evaluated on the combination of CutLER and offline clustering, Panoptic is trained on both *"stuff"* and *"things"* pseudo labels, Instance is trained solely on *"things"* labels.

of our model's capability.

Table 4.10 and Table 4.11 display the results for unsupervised object detection and instance segmentation on different datasets. One trend can be observed across the different datasets: as the number of the predicted cluster ID increases (*e.g.*, moving from 300 to 2911), there is a consistent increase for most of the metrics. This trend can be succinctly attributed to the intrinsic properties of the multi-to-one Hungarian matching approach (we also show the parameter IoU and Conf used for Hungarian matching). With an increase of the cluster numbers, the Hungarian matching has a broader set of predictions to associate with a single label. This inherently increases the chances of having at least one correct prediction for the given label, making the matching process more amenable. In essence, larger cluster numbers afford easier matching, thereby boosting the evaluation metrics.

Furthermore, the qualitative results are shown in Fig. 4.9, with the samples selected in COCO `val2017` and PASCAL VOC `val2012`. After Hungarian matching, we are able to get the real categories of the predicted instances.

Unsupervised Universal Image Segmentation

Our model's performance for unsupervised universal image segmentation closely mirrors the trends observed in instance segmentation. Specifically, as the number of the predicted clusters increases, the performance of the panoptic segmentation also improves. Detailed universal segmentation results are shown in Table 4.12.

Limitation

The primary goal of our research is to develop a comprehensive model capable of excelling in all areas of unsupervised segmentation. As shown in Table 4.13, in terms of the individual sub-task, the universal model exhibits a slight underperformance compared to its counterpart model trained with task-specific annotations. This suggests that U2Seg is adaptable to various tasks, yet it requires task-specific training to achieve the best outcomes for a specific sub-task. Looking ahead, we aim to develop a more versatile model that can be trained once to effectively handle multiple tasks.

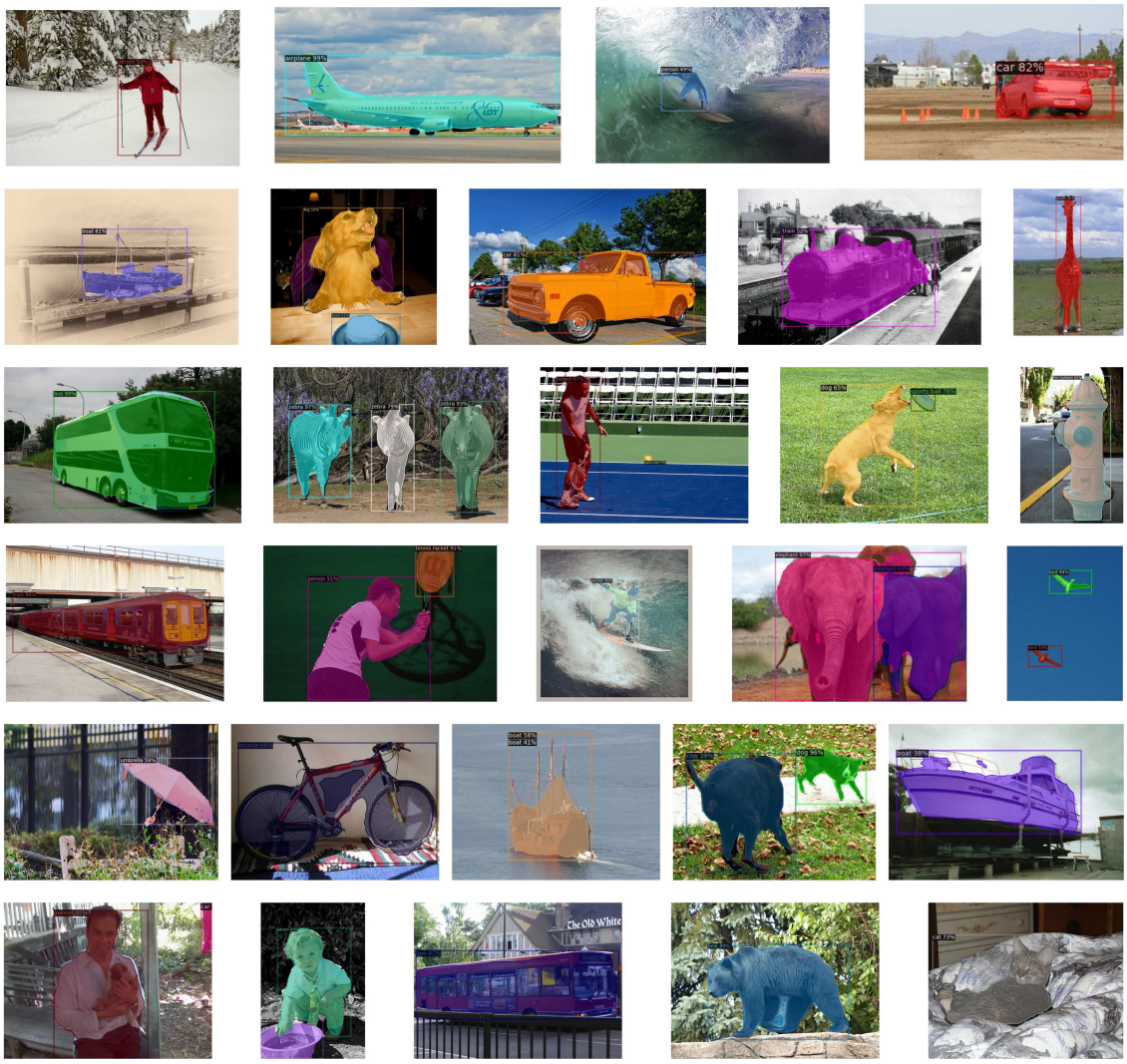


Figure 4.9: Unsupervised object detection and instance segmentation visualization of COCO `val2017` and PASCAL VOC `val2012` (**after Hungarian matching**).

4.6 Summary

We present **U2Seg**, a novel **U**nsupervised **U**niversal Image **Seg**mentation model, adept at performing unsupervised instance, semantic, and panoptic segmentation tasks within a unified framework. Evaluated on extensive benchmarks, U2Seg consistently outperforms previous state-of-the-art methods designed for individual tasks. Additionally, U2Seg achieves the new state-of-the-art for label-efficient panoptic segmentation and instance segmentation. We anticipate that U2Seg, free from the constraints of human annotations, will demonstrate enhanced performance when scaled up with more training data, representing an important direction for our future research.

Acknowledgement: Trevor Darrell, Dantong Niu and Xudong Wang were funded by DoD including DARPA LwLL and the Berkeley AI Research (BAIR) Commons.

Chapter 5

Segment Anything without Supervision

The Segmentation Anything Model (SAM) requires labor-intensive data labeling. We present Unsupervised SAM (UnSAM) for promptable and automatic whole-image segmentation that does not require human annotations. UnSAM utilizes a divide-and-conquer strategy to "discover" the hierarchical structure of visual scenes. We first leverage top-down clustering methods to partition an unlabeled image into instance/semantic level segments. For all pixels within a segment, a bottom-up clustering method is employed to iteratively merge them into larger groups, thereby forming a hierarchical structure. These unsupervised multi-granular masks are then utilized to supervise model training. Evaluated across seven popular datasets, UnSAM achieves competitive results with the supervised counterpart SAM, and surpasses the previous state-of-the-art in unsupervised segmentation by 11% in terms of AR. Moreover, we show that supervised SAM can also benefit from our self-supervised labels. By integrating our unsupervised pseudo masks into SA-1B's ground-truth masks and training UnSAM with only 1% of SA-1B, a lightly semi-supervised UnSAM can often segment entities overlooked by supervised SAM, exceeding SAM's AR by over 6.7% and AP by 3.9% on SA-1B.

5.1 Introduction

Trained on massive unlabeled data using self-supervised learning methods, Large Language Models (LLMs) [2]–[4], [133]–[135] in natural language processing have revolutionized our world and redefined human-computer interactions. In the domain of computer vision, the recent introduction of the Segment Anything Model (SAM) [132] has dramatically transformed the field with its exceptional ability to handle diverse image segmentation tasks. However, the need for comprehensive manual labeling of training data—over 20 minutes per image [132]—limits SAM from following the scaling laws that benefit LLMs [136]. As a result, despite SA-1B [132] being the most extensive segmentation dataset available, it contains only about 11 million images. Moreover, human-annotated data often introduces significant biases based on the annotators' perceptions of "what constitutes an instance", which frequently leads to the oversight of small entities within the images.

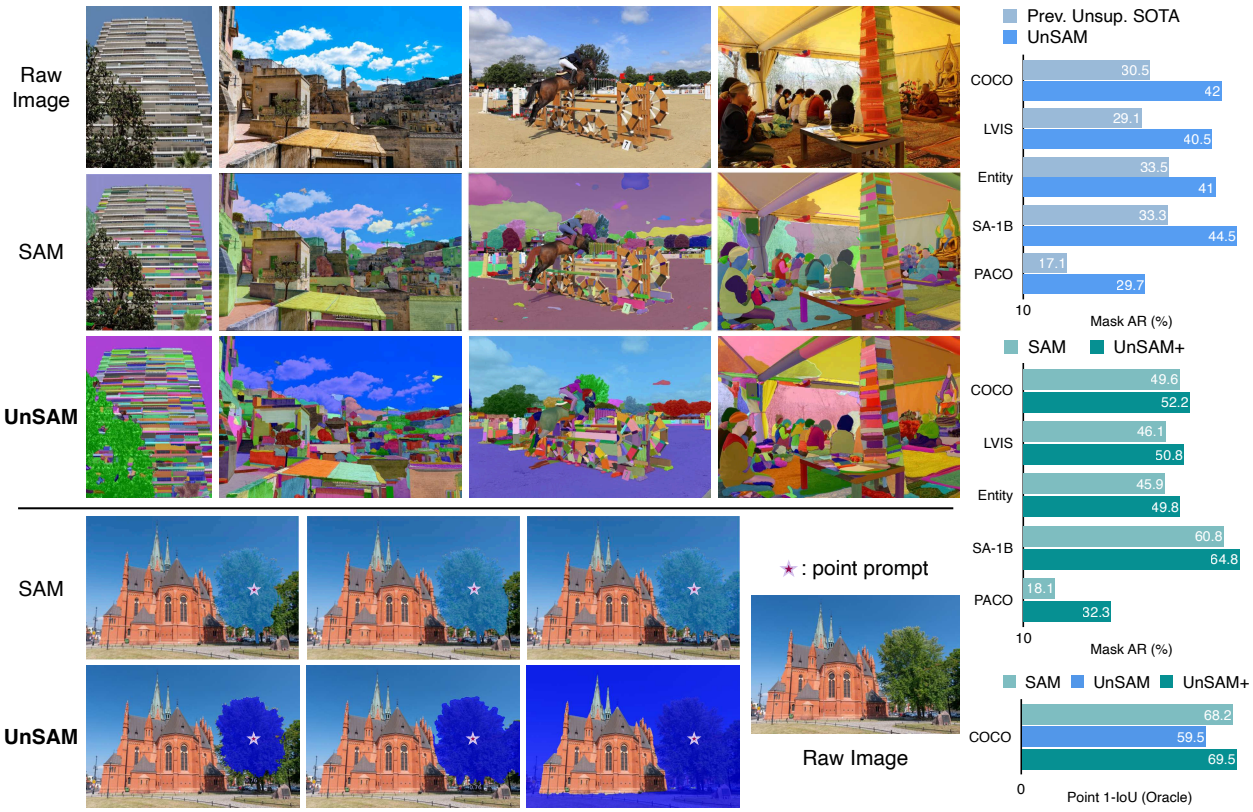


Figure 5.1: UnSAM significantly surpasses the performance of the previous SOTA methods in unsupervised segmentation, and delivers impressive whole image and promptable segmentation results, rivaling the performance of the supervised SAM [132]. This comparative analysis features our unsupervised UnSAM, the supervised SAM, and an enhanced version, UnSAM+, across a variety of datasets. The top section displays raw images (row 1) alongside whole image segmentation outputs from UnSAM (row 3), and SAM (row 2). The bottom section highlights our promptable segmentation results using a point prompt (*i.e.*, the star mark). The right panel quantitatively compares the performance across models, including metrics like Mask AR (%) and Point IoU.

This challenge raises a crucial question addressed in this paper: *Can we "segment anything" without supervision?* In response, we present **UnSAM**, an innovative unsupervised learning method capable of performing both interactive and whole-image segmentation without the need for supervision.

How can we achieve fine-grained and multi-granular segmentation masks comparable to those in SA-1B [132] without supervision? Insights from neuroscience suggest that the human visual system exploits the structure of visual scenes by decomposing dynamic scenes into simpler parts or motions. This perception of hierarchically organized structures implies a powerful "divide-and-conquer" strategy for parsing complex scenes [137], [138]. Drawing inspiration from this, we introduce a divide-and-conquer approach designed to generate hierarchical image segmentation

results directly from raw, unlabeled images. The divide-and-conquer approach is a crucial element of UnSAM, enabling it to effectively parse and segment images at multiple levels of granularity.

Our pseudo-mask generation pipeline initiates with a top-down clustering approach (*i.e.*, the divide stage), to extract initial semantic and instance-level masks using a Normalized Cuts-based method CutLER [12], [53]. Subsequently, UnSAM refines these masks using a bottom-up clustering method (*i.e.*, the conquer stage): within each mask, we iteratively merge semantically similar pixels into larger segments based on various similarity thresholds. The resulting masks at different thresholds in the conquer stage, along with the masks produced in the divide stage, create a hierarchical structure. Technically, we can generate a vast range of granularities with minimal extra cost! Furthermore, UnSAM captures more subtle details that pose challenges for human annotators, significantly enriching the granularity and utility of unsupervised segmentation models.

Equipped with these sophisticated multi-granular pseudo masks as "ground-truth" labels, UnSAM is adeptly trained to perform both interactive and automatic whole-image segmentation, demonstrating remarkable versatility across various segmentation scenarios. We have observed that our UnSAM model frequently identifies objects that SAM [132] overlooks, particularly types of objects or parts typically missed by ground-truth annotations of SA-1B [132], such as human ears, animal tails, *etc*.

The capabilities of UnSAM are rigorously tested across seven major whole-entity and part segmentation datasets, *e.g.*, MSCOCO [24], LVIS [61], SA-1B [132], ADE [139], Entity [140], PartImageNet [141] and PACO [142]. As illustrated in Fig. 5.1, we demonstrate some noteworthy behaviors:

- The performance gap between unsupervised segmentation models and SAM can be significantly reduced: By training on just 1% of SA-1B's unlabeled images with a ResNet50 backbone, UnSAM not only advances the state-of-the-art in unsupervised segmentation by 10% but also achieves comparable performance with the labor-intensive, fully-supervised SAM.
- The supervised SAM can also benefit from our self-supervised labels: integrating our unsupervised pseudo masks with SA-1B's ground-truth data and retraining UnSAM on this combined data enables UnSAM+ to outperform SAM's AR by over 6.7% and AP by 3.9%. We observed that UnSAM and UnSAM+ can often discover entities missed by SAM.

5.2 Related Works

Self-supervised Image Segmentation

Recent advances in unsupervised image segmentation [12], [23], [25], [30], [35], [52], [143]–[148] have leveraged the emergent segmentation capabilities of self-supervised Vision Transformers (ViT) [25], [29], [47] to "discover" objects within images. Initial efforts, such as TokenCut [144] and LOST [32], have produced semantically meaningful pixel groupings for salient objects by utilizing the class-attention mechanism of self-supervised ViTs. As a representative work in the unsupervised segmentation domain, CutLER [12] introduced a cut-and-learn pipeline for unsupervised object detection and image segmentation. CutLER initially generates high-quality pseudo masks

for multiple objects using MaskCut [12], followed by learning a detector on these masks using a loss dropping strategy. Extending this approach, VideoCutLER [14] employs a cut-synthesis-and-learn strategy for segmenting and tracking multiple instances across video frames without supervision. Additionally, SOHES [145] introduced the global-local self-exploration method to cluster image features from high to low cosine similarity, obtaining pseudo masks that cover multiple hierarchical levels.

In contrast, UnSAM introduces a divide-and-conquer pipeline that generates more pseudo masks per image at the same processing speed, but with enhanced quality and broader coverage across hierarchical levels. Furthermore, UnSAM captures more subtle details that pose challenges for human annotators, significantly enriching the granularity and utility of unsupervised segmentation models.

Promptable Image Segmentation

Tradition segmentation models have focused on predicting masks for all instances or semantic parts within a single image simultaneously. Recently, however, models have begun to interact with users, generating segmentation masks based on user inputs such as points [132], [149]–[152], text descriptions [153], or bounding boxes [132]. Moreover, some approaches now frame segmentation tasks within an in-context learning framework [154], [155], utilizing in-context examples to define distinct segmentation tasks. For example, the Segment Anything model [132] can produce masks in a zero-shot manner based on different types of prompts. One limitation of SAM is that it only produces three class-agnostic masks. An extension, Semantic-SAM [149], aims to segment and recognize objects at multiple granularities through a multi-choice learning scheme, allowing each click point to produce masks at multiple levels along with their semantic labels. Nevertheless, both models are supervised and rely on large-scale, human-annotated data, which introduces issues of annotator bias and scalability limitations.

In contrast, our unsupervised UnSAM and lightly semi-supervised UnSAM+ model demonstrate superior performance in the promptable segmentation task, offering a robust alternative to these fully-supervised approaches.

5.3 Preliminaries

Cut and Learn (CutLER) and MaskCut

CutLER [12] introduces a cut-and-learn pipeline to precisely segment instances without supervision. The initial phase, known as the cut stage, uses a normalized cut-based method, MaskCut [12], to generate high-quality instance masks given the patch-wise cosine similarity matrix $W_{ij} = \frac{K_i K_j}{|K_i|_2 |K_j|_2}$, where K_i is "key" features of patch i in the last attention layer of unsupervised ViT. To extract multiple instance masks from a single image, MaskCut repeats this operation but adjusts by masking out patches from previously segmented instances in the affinity matrix: $W^t_{ij} = \frac{\left(K_i \sum_{s=1}^{t} M^s_{ij}\right)\left(K_j \sum_{s=1}^{t} M^s_{ij}\right)}{\|K_i\|_2 \|K_j\|_2}$ Subsequently, CutLER's learning stage trains a segmen-

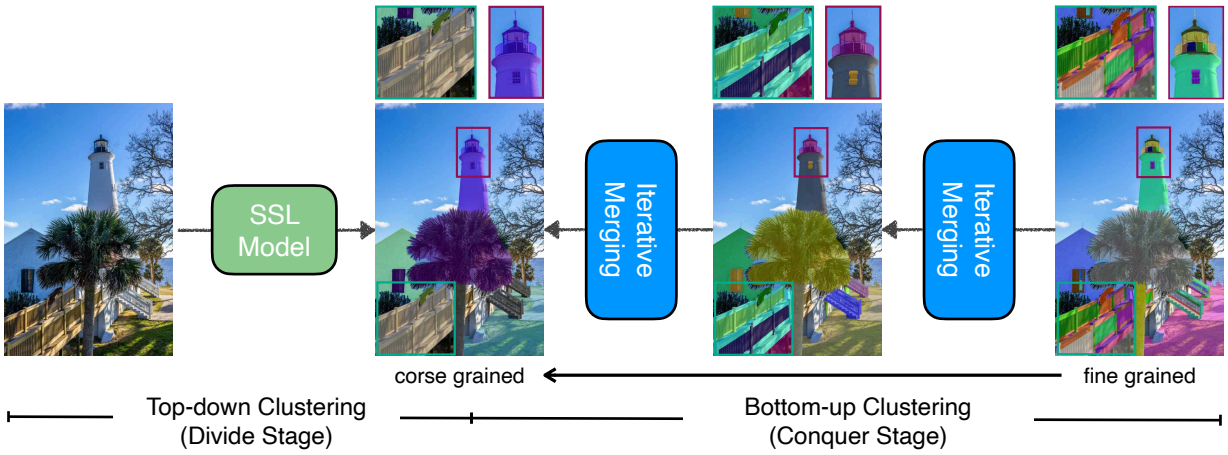


Figure 5.2: Our divide-and-conquer pipeline for generating the "ground-truth" pseudo masks used for training UnSAM without human supervision begins with a top-down clustering approach (*i.e.*, the divide stage), to extract initial semantic/instance-level masks using a Normalized Cuts [53]-based CutLER [12]. Subsequently, we refine these masks using a bottom-up clustering method (*i.e.*, the conquer stage): within each mask, we iteratively merge semantically similar pixels into larger segments using various similarity thresholds. The resulting masks at different thresholds create a hierarchy. We zoom-in selected regions to visualize details.

tation/detection model on these pseudo-masks with drop-loss. Please check Sec. 5.6 for more details on CutLER.

Segment Anything Model (SAM) and SA-1B

Segment Anything [132] tackles the promptable segmentation task. At its core lies the Segment Anything Model (SAM), which is capable of producing segmentation masks given user-provided points, boxes, and masks in a zero-shot manner. One significant contribution of SAM is the release of the SA-1B dataset [132], which comprises 11M high-resolution images and 1.1 billion segmentation masks, providing a substantial resource for training and evaluating segmentation models. While SAM significantly accelerates the labeling of segmentation masks, annotating an image still requires approximately 14 seconds per mask. Given that each image contains over 100 masks, this equates to more than 30 minutes per image, posing a substantial cost and making it challenging to scale up the training data effectively. For more details on SAM and SA-1B, please check Sec. 5.6.

Algorithm 1 Divide and Conquer

$I_{\text{resized}} \leftarrow$ input image I resized to 1024×1024
$M \leftarrow \{m : m \in \text{CutLER}(I_{\text{resized}}) \wedge m_{\text{score}} > \tau\}$
for $m \in M$ **do**
  Add m into S_0
  $bbox \leftarrow$ bounding box $[x_1, y_1, x_2, y_2]$ of m
  $I_{\text{local}} \leftarrow I_{\text{resized}}$ cropped by $bbox$, resized to 256×256
  $K \leftarrow \text{DINO}(I_{\text{local}})$
  for $\theta_t \in \theta_l, \ldots, \theta_1$ **do**
    if $t = l$ **then**
      Initialize $k_i^t \leftarrow K_i$, $C_i^t \leftarrow\ p_i\ \forall i$, $a \leftarrow 1$
      where p_i is corresponding patch of K_i, add p_i into $S_l\ \forall i$
    else
      Initialize $S_t \leftarrow S_{t+1}$, $k_i^t \leftarrow k_i^{t+1}$, $C_i^t \leftarrow\ C_i^{t+1}\ \forall i$
    end if
    while $a \geq \theta_t$ **do**
      Identify adjacent p_i, p_j with $i, j \leftarrow \underset{i,j}{\text{argmax}} \frac{k_i^t k_j^t}{||k_i^t||_2 ||k_j^t||_2}$, $a \leftarrow \max_{i,j} \frac{k_i^t k_j^t}{||k_i^t||_2 ||k_j^t||_2}$
      Identify cluster C_m^t, C_n^t, where $p_i \in C_m^t, p_j \in C_n^t$
      Remove C_m^t and C_n^t from S_t
      $C^t \leftarrow C_m^t \cup C_n^t$, add C^t into S_t
      $\forall p_z \in C^t, k_z^t \leftarrow \frac{a_m k_i^t + a_n k_j^t}{a_m + a_n}$, where a_m is the size of cluster C_m^t and $p_i \in C_m^t$
    end while
  end for
end for

5.4 UnSAM: Segment Anything without Supervision

Divide-and-Conquer for Hierarchical Image Segmentation

Our segment anything without supervision model starts by generating pseudo masks that respect the hierarchical structure of visual scenes without supervision. This approach is motivated by the observation that the "divide and conquer" strategy is a fundamental organizational principle employed by the human visual system to efficiently process and analyze the vast complexity of visual information present in natural scenes [137], [138]. Our pseudo-mask generation pipeline **divide-and-conquer**, which is summarized in Alg. 1 and illustrated in Fig. 5.2, consists of two stages:

Divide stage: we leverage a Normalized Cuts (NCuts)-based method, CutLER [12], [53], to obtain semantic and instance-level masks from unlabeled raw images. CutLER's cut-and-learn pipeline and its MaskCut method are discussed in Sec. 5.3. However, the coarser-granularity masks predicted by CutLER can be noisy. To mitigate this, we filter out masks with a confidence score

below a threshold τ. Empirically, salient semantic and instance-level entities typically encompass richer part-level entities (for example, a person has identifiable parts such as legs, arms, and head, whereas a background sky contains few or no sub-level entities). To extract these part-level entities with a hierarchical structure, we employ a conquer phase.

Conquer stage: for each instance-/semantic-level mask discovered in the previous stage, we employ iterative merging [145], [156] to decompose the coarse-grained mask into simpler parts, forming a hierarchical structure.

More specifically, we first crop local patches using the masks we obtained in the divide phase, and bi-linearly interpolate local patches to the resolution of 256×256. We then feed them into DINO pre-trained ViT-B/8 [25] encoder $f(\cdot)$, and extract 'key' features $k_i = f(p_i)$ from the last attention layer as patch-wise features for local patches p_i. Subsequently, the conquer phase employs iterative merging [145], [156] to group patches into larger clusters, with pre-defined cosine similarity thresholds at $\theta \in \{\theta_1, ..., \theta_l\}$, where l is the predefined granularity levels.

In iteration t, our method finds two adjacent patches (p_i, p_j) from two separate clusters (C_m^t, C_n^t) with the highest cosine similarity $\frac{k_i^t k_j^t}{||k_i^t||_2 ||k_j^t||_2}$, merges them into one cluster, and updates k_i^t and k_j^t to $\frac{a_m k_i^t + a_n k_j^t}{a_m + a_n}$, where a_m is the number of patches in cluster $C_m^t (p_i \in C_m^t)$. The conquer stage repeats this step until the maximum cosine similarity is less than θ_t, collects all merged clusters as new part-level pseudo masks, and uses smaller threshold θ_{t+1} to iterate again. Each coarse-grained mask discovered in the divide stage can form a hierarchical structure H after the conquer stage:

$$H = \{S_0, S_1, ..., S_t, ..., S_l\}, \text{where } S_t = \{C_1^t, ..., C_{n_t}^t\}, n_i \leq n_j \text{ if } i < j \tag{5.1}$$

n_t is the number of clusters/masks belonging to granularity level t and $n_0 = 1$.

Mask merging: The new part-level pseudo masks discovered in the conquer stage are added back to the semantic and instance-level masks identified in the divide stage. We then use Non-Maximum Suppression (NMS) to eliminate duplicates. Following previous works in unsupervised image segmentation [12], [143], [145], we also employ off-the-shelf mask refinement methods, such as Conditional Random Fields (CRF) [157] and CascadePSP [158], to further refine the edges of the pseudo masks. Finally, we filter out the post-processed masks that exhibit significant differences in Intersection-over-Union (IoU) before and after refinement.

Preliminary results: The divide-and-conquer pipeline achieves a pseudo mask pool with more entities, a broader range of granularity levels, and superior quality compared to previous work, *e.g.*, CutLER [12], U2Seg [143] and SOHES [145]. As shown in Table 5.3, its pseudo masks reach 23.9% AR on 1000 randomly selected validation images from the SA-1B dataset [132], representing a 45.7% improvement over the state-of-the-art.

Key distinctions over prior works on pseudo-mask generation: The divide-and-conquer strategy employed by UnSAM sets it apart from previous works:

[12], [143] rely solely on top-down clustering methods, providing only instance and semantic-level masks, and thereby missing the hierarchical structure present in complex images. In contrast, our pipeline captures this hierarchical structure by identifying more fine-grained pixel clusters.

While [145] does incorporate some hierarchical structure through bottom-up clustering with iterative merging, it still misses many fine-grained instances and some large-scale instance masks.

Additionally, the iterative merging in [145] focuses on small regions below a certain mask size threshold, primarily to refine noisy small masks, limiting its ability to detect a full range of entity sizes. Our experimental results demonstrate qualitatively and quantitatively superior performance compared to prior works, particularly in producing high-quality, detailed pseudo-masks that better capture the hierarchical complexity of visual scenes.

Model Learning and Self-Training

Although the pseudo masks generated by our pipeline are qualitatively and quantitatively superior to those from prior works, they can still be somewhat noisy. Our self-supervised pipeline has limitations in identifying certain types of instances. For example, iterative merging sometimes fails to correctly associate disconnected parts of the same entity. To address this, we utilize a self-training strategy to further enhance UnSAM's model performance. UnSAM learns an image segmentation model using the masks discovered by the divide-and-conquer strategy. It has been observed that self-training enables the model to "clean" the pseudo masks and predict masks of higher quality [12]. Once we have prepared the pseudo-masks, UnSAM can be integrated with any arbitrary whole-image or promptable image segmentation models during the model learning or self-training stage.

Whole-image segmentation. We choose the vanilla Masked Attention Mask Transformer (Mask2Former) [94] for simplicity. The key innovation of Mask2Former is the introduction of a masked attention mechanism in the transformer's cross-attention block, defined as $\text{softmax}(M + QK^T)V$, where the attention mask M at feature location (x, y) is given by: $M(x,y) = \begin{cases} 0 & \text{if } M(x,y) = 1 \\ -\infty & \text{otherwise} \end{cases}$. This mechanism constrains attention within the region of the predicted mask. UnSAM is then trained using the following mask prediction loss:

$$\mathcal{L} = \lambda_{\text{ce}}\mathcal{L}_{\text{ce}} + \lambda_{\text{dice}}\mathcal{L}_{\text{dice}} \tag{5.2}$$

where $\mathcal{L}_{\text{ce}}$ and $\mathcal{L}_{\text{dice}}$ is the cross-entropy and Dice loss, with λ_{ce} and λ_{dice} as their respective weights.

After one round of self-training UnSAM on the pseudo-masks, we perform a second round of self-training by merging high-confidence mask predictions (with a confidence score greater than $\tau_{\text{self-train}}$) as the new 'ground-truth' annotations. To avoid duplication, we filter out ground truth masks that have an IoU greater than 0.5 with the predicted masks.

Promptable Image Segmentation. Similar to SAM [132], our unsupervised SAM can also produce high-quality object masks from input prompts such as points. We utilize Semantic-SAM [149] as the base model for predicting multiple granularity levels of masks from a single click. During the learning process, we randomly sample points within an inner circle (radius $\leq 0.1 \cdot \min(\text{Mask}_{\text{width}}, \text{Mask}_{\text{height}})$) of the mask to simulate user clicks.

UnSAM+: Improving Supervised SAM with Unsupervised Segmentation

The supervised SAM model's [132] reliance on human-annotated data introduces a significant bias based on the annotator's perception of *'what constitutes an instance'*, frequently missing some

entities within the image. In contrast, since our mask generation pipeline does not rely on human supervision, it can often identify valid objects or parts that are overlooked by SA-1B's [132] ground-truth annotations.

Motivated by this observation, we leverage UnSAM to improve the performance of the supervised SAM [132] by implementing a straightforward yet effective strategy: merging SA-1B's ground-truth masks $D_{\text{SA-1B}}$ with our unsupervised segmentation masks D_{UnSAM} based on the IoU, formulated as:

$$D^i_{\text{UnSAM+}} = D^i_{\text{SA-1B}} \cup \{\forall C_m \in D^i_{\text{UnSAM}} \text{ if } \text{IoU}^{\max}(C_m, \forall C_n \in D^i_{\text{SA-1B}}) \leq \tau_{\text{UnSAM+}}\} \tag{5.3}$$

$\tau_{\text{UnSAM+}}$ is the IoU threshold, $\text{IoU}^{\max}$ is the maximum IoU between C_m and any mask C_n in $D^i_{\text{SA-1B}}$, and $D^i_{\text{SA-1B}}$ and $D^i_{\text{UnSAM+}}$ is the set of SA-1B and unsupervised masks within image i, respectively. We then train UnSAM+ on $D_{\text{UnSAM+}}$ for promptable image segmentation and whole-image segmentation. The fusion approach leverages the strengths of both supervised and unsupervised annotations, addressing the limitations inherent in human-annotated datasets while significantly enriching the diversity and comprehensiveness of the training data. This results in a more robust and generalizable segmentation model UnSAM+, surpassing the performance of SAM.

5.5 Experiments

Model Training Settings

We provide a brief overview of the model training settings and include more details in Sec. 5.6.

Pseudo mask generation. In the divide stage, we set the confidence threshold τ=0.3; in the conquer stage, we choose threshold $\theta_{merge} = [0.6, 0.5, 0.4, 0.3, 0.2, 0.1]$. When merging the pseudo masks with the ground truths for training UnSAM+, we select $\tau_{\text{UnSAM+}} = 0.02$. **Whole-image segmentation.** UnSAM picks DINO [25] pre-trained ResNet-50 [58] as the backbone and Mask2former [94] as the mask decoder. The default learning rate is 5×10^{-5} with a batch size of 16 and a weight decay of 5×10^{-2}. We train the model for 8 epochs. **Promptable segmentation.** UnSAM uses the self-supervised pre-trained Swin-Transformer [159] Tiny model as the backbone, and leverages Semantic-SAM [149] as the base model. We set the number of hierarchy levels to 6, which is also the number of predicted masks UnSAM generates per prompt during inference. One can easily train with a different number of granularity levels as needed. For all experiments, we train UnSAM with 1$\sim$4% unlabeled images from SA-1B dataset [132].

Evaluation Datasets and Metrics

Whole-image segmentation. To evaluate our model's performance, we test our models on various datasets in a zero-shot manner to evaluate the performance of segmenting entities from all granularity levels. We choose COCO [24], LVIS [61], ADE20K [139], EntitySeg [140], and SA-1B [132] that mainly encompass semantic-/instance-level entities; PartImageNet [141] and PACO [142] that

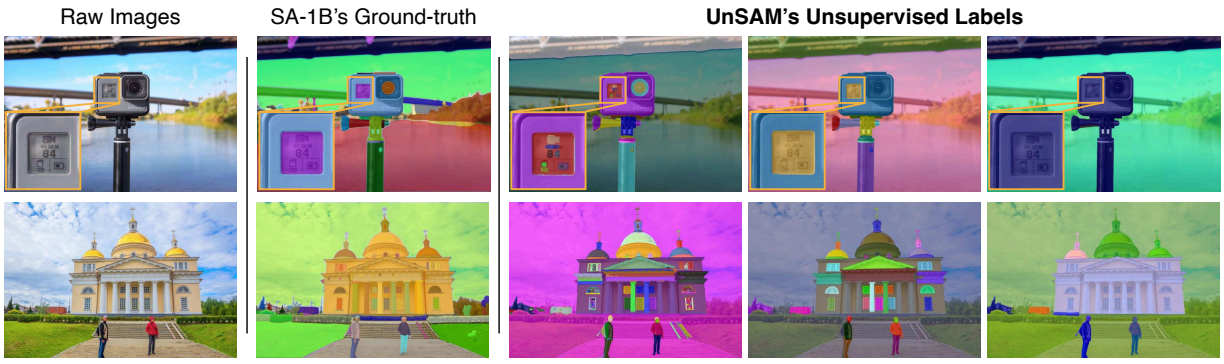


Figure 5.3: Unsupervised pseudo-masks generated by our divide-and-conquer pipeline not only contain precise masks for coarse-grained instances (column 5), *e.g.*, cameras and persons, but also capture fine-grained parts (column 3), *e.g.*, digits and icons on a tiny camera monitor that are missed by SA-1B's [132] ground-truth labels.

Methods	Backbone (# params)	# images	Avg.	Datasets with Whole Entities					Datasets w/ Parts	
				COCO	LVIS	ADE	Entity	SA-1B	PtIn	PACO
SAM (supervised)	ViT-H/16 (85M)	11M	42.1	49.6	46.1	45.8	45.9	60.8	28.3	18.1
FreeSOLO [23]	RN-101 (45M)	1.3M	7.3	11.6	5.9	7.3	8.0	2.2	13.8	2.4
CutLER [12]	RN-50 (23M)	1.3M	21.8	28.1	20.2	26.3	23.1	17.0	28.7	8.9
SOHES [145]	ViT-B/8 (85M)	0.2M	30.1	30.5	29.1	31.1	33.5	33.3	36.0	17.1
UnSAM	RN-50 (23M)	0.1M	**39.2**	**40.5**	**37.7**	**35.7**	**39.6**	**41.9**	**51.6**	**27.5**
UnSAM	RN-50 (23M)	0.2M	**40.4**	**41.2**	**39.7**	**36.8**	**40.3**	**43.6**	**52.1**	**29.1**
UnSAM	RN-50 (23M)	0.4M	**41.1**	**42.0**	**40.5**	**37.5**	**41.0**	**44.5**	**52.7**	**29.7**
vs. prev. SOTA			**+11.0**	**+11.5**	**+11.4**	**+6.4**	**+7.5**	**+11.2**	**+16.7**	**+12.6**

Table 5.1: UnSAM achieves the state-of-the-art results on unsupervised image segmentation, using a backbone of ResNet50 and training with only 1% of SA-1B [132] data. We perform a zero-shot evaluation on various image segmentation benchmarks, including whole entity datasets, *e.g.*, COCO and ADE, and part segmentation datasets, *e.g.*, PACO and PartImageNet. The evaluation metric is average recall (AR).

cover part-level entities. The SA-1B test set consists of randomly selected 1000 images not included in our training set. Notably, each dataset only covers entities from certain hierarchical levels and certain pre-defined classes, while our model generates masks from all levels and all classes. Hence, the COCO Average Precision (AP) metric could not reflect our model's authentic performance in segmenting all entities in the open-world. Following prior work [12], [145], we mainly consider Average Recall (AR) to compare with different models.

Point-based promptable segmentation. We evaluate our point-based interactive segmentation model on MSCOCO `Val2017` [24]. Following the previous work on promptable image segmentation [132], [149], we pick two metrics for model evaluation MaxIoU and OracleIoU. For each point prompt, UnSAM predicts 6 masks representing different granularity levels. MaxIoU calculates the IoU between the mask with the highest confidence score among 6 masks, whereas

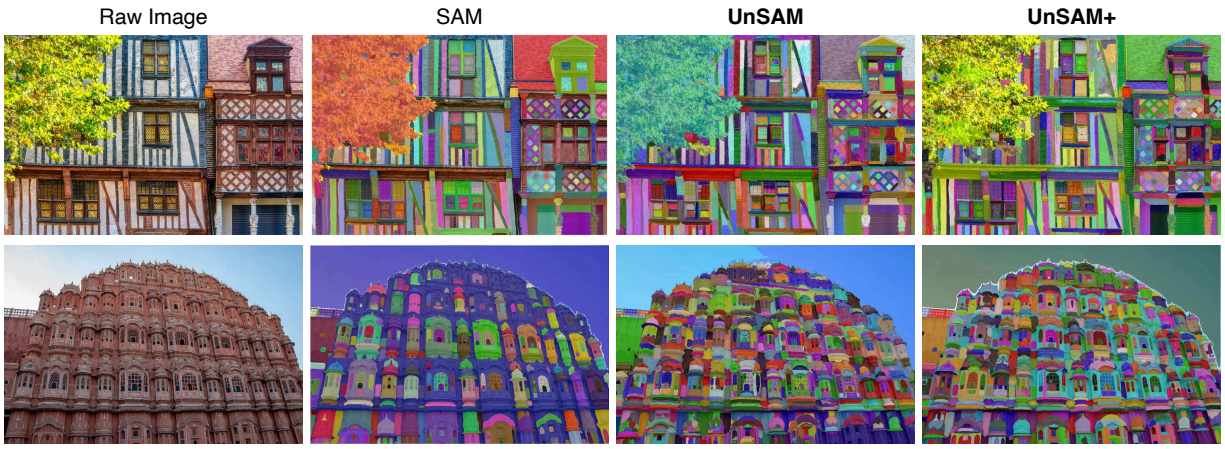


Figure 5.4: UnSAM has competitive dense object segmentation results compared to the supervised SAM [132].

OracleIoU picks the highest IoU between 6 predicted masks and the ground truth. For each mask in a test image, we select its center as the point prompt.

Evaluation Results

Unsupervised pseudo-masks. Unsupervised pseudo-masks generated by our divide-and-conquer pipeline not only contain precise masks for coarse-grained instances, but also capture fine-grained parts that are often missed by SA-1B's [132] ground-truth labels, as shown in Fig. 5.3.

Whole-image segmentation. Remarkably, UnSAM outperforms the previous state-of-the-art methods across **all** evaluation datasets as summarized in Table 5.1. UnSAM demonstrates superior performance compared to the SOTA method even when trained with only 1% SA-1B training data and a backbone of ResNet-50 with only 23M parameters, while the SOTA utilizes twice training data and a backbone with nearly four times the parameters. This implies that UnSAM is a lightweight, easier to train, and less data-hungry model with better zero-shot performance in segmenting entities in the open-world as shown in Figs. 5.4 and 5.5. On average, UnSAM surpasses the previous SOTA by 11.0% in AR. When evaluated on PartImageNet [141] and PACO [142] benchmarks, UnSAM exceeds the SOTA by 16.6% and 12.6 %, respectively.

When compared to the supervised SAM [132], UnSAM's AR across all datasets is already very close, with only a 1% difference. On PartImageNet [141] and PACO [142], UnSAM surpasses SAM by 24.4% and 11.6%. This further demonstrates the excellent capability of our divide-and-conquer pipeline in discovering details that human annotators tend to miss.

Furthermore, our UnSAM+, trained with integrated unsupervised pseudo masks and SA-1B [132] ground truth, outperforms SAM's [132] AR by over 6.7% and AP by 3.9% as shown by Table 5.2 and 5.4. UnSAM+ demonstrates superior average recall compared to SAM across all evaluation datasets except for ADE20K [139], which is dominated by semantic-level annotations. UnSAM+'s

Methods	Backbone (# params)	Sup. Labels	Unsup. Labels	# images	Avg.	Datasets with Whole Entities					Datasets w/ Parts	
						COCO	LVIS	ADE	Entity	SA-1B	PtIn	PACO
SAM	ViT-B/8 (85M)	✓	✗	11M	42.1	49.6	46.1	**45.8**	45.9	60.8	28.3	18.1
UnSAM	RN-50 (23M)	✗	✓	0.1M	39.2	40.5	37.7	35.7	39.6	41.9	**51.6**	27.5
UnSAM+	RN-50 (23M)	✓	✓	0.1M	**48.8**	**52.2**	**50.8**	45.3	**49.8**	**64.8**	46.0	**32.3**
vs. SAM					**+6.7**	**+2.6**	**+4.7**	-0.5	**+3.9**	**+4.0**	**+17.7**	**+14.2**

Table 5.2: UnSAM+ can outperform SAM [132] on most experimented benchmarks (including SA-1B [132]), when training UnSAM on 1% of SA-1B with both ground truth masks and our unsupervised labels. This demonstrates that our unsupervised pseudo masks can serve as a powerful add-on to the densely annotated SA-1B masks!

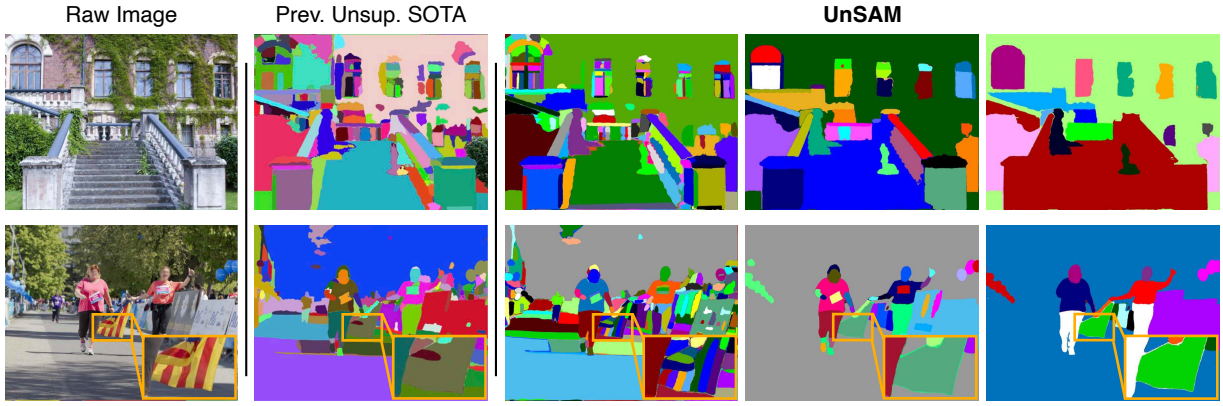


Figure 5.5: UnSAM not only discovers more fine-grained masks than the previous state-of-the-art unsupervised segmentation method [145], but also provides segmentation masks with a wide range of granularity. We show qualitative comparisons between UnSAM (with 3 levels of granularity) and baseline models on SA-1B [132].

Methods	AR_{1000}	AR_S	AR_M	AR_L
SOHES (CRF [157])	12.0	3.5	9.5	20.7
SOHES (CascadePSP [158])	16.4	6.0	15.8	22.6
UnSAM (CRF [157])	15.3	2.3	11.9	27.7
UnSAM (CascadePSP [158])	**23.9**	**7.9**	**22.4**	**34.0**
vs. prev. SOTA	**+7.5**	**+1.9**	**+6.6**	**+11.4**

Table 5.3: Evaluation on unsupervised pseudo masks using SA-1B's [132] ground-truth annotations.

Methods	AP	AR_S	AR_M	AR_L	AR_{1000}
SAM	38.9	20.0	59.9	**82.8**	60.8
UnSAM+	**42.8**	**36.2**	**65.9**	76.5	**64.8**
vs. sup. SAM	**+3.9**	**16.2**	**+6.0**	-6.3	**+4.0**

Table 5.4: Quantitative comparisons between our lightly semi-supervised SAM, UnSAM+, and the fully-supervised SAM [132] on SA-1B [132].

significantly 16.2 % higher AR on small entities further confirms that our pseudo masks can effectively complement the SA-1B datasets with more details it ignores and the UnSAM+ can often

Figure 5.6: Qualitative comparisons of promptable image segmentation between the fully-supervised SAM [132], our unsupervised UnSAM, and the lightly semi-supervised UnSAM+. Both UnSAM and UnSAM+ consistently deliver high-quality, multi-granular segmentation masks in response to the point prompts (*i.e.*, the star mark).

Methods	Backbone (# params)	Sup. Labels	Unsup. Labels	% of SA-1B	Point (Max) 1-IoU	Point (Oracle) 1-IoU
SAM (B)	ViT-B/8 (85M)	✓	✗	100%	52.1	68.2
UnSAM	Swin-Tiny (25M)	✗	✓	1%	40.3	59.5
UnSAM+	Swin-Tiny (25M)	✓	✓	1%	**52.4**	**69.5**

Table 5.5: Despite using a backbone that is 3× smaller and being trained on only 1% of SA-1B, our lightly semi-supervised UnSAM+ surpasses the fully-supervised SAM in promptable segmentation task on COCO.

discover entities missed by SAM as demonstrated in Fig. 5.4 and Fig. 5.7.

Point-based promptable segmentation. As shown in Table 5.5, UnSAM trained with our pseudo masks achieve 40.3% MaxIoU and 59.5% OracleIoU on COCO. Notably, we train the model with only 1% of the data that SAM [132] uses and a backbone with 4× fewer parameters. Moreover, the UnSAM+ trained with integrated pseudo masks and SA-1B ground truths outperforms SAM on both MaxIoU and OracleIoU with 0.3% and 1.3% respectively. Qualitative results are shown in Fig. 5.6.

5.6 Appendix

Training Details

Pseudo mask preparation details. Empirically, in the divide stage, we set the confidence threshold $\tau = 0.3$; in the conquer stage, we choose threshold $\theta_{merge} = [0.6, 0.5, 0.4, 0.3, 0.2, 0.1]$. For each image, the divide-and-conquer pipeline generates on average 334 pseudo masks. In the self-training phase, the $\tau_{\text{self-train}} = 0.7$, and each image has 448 pseudo masks per image after merging

Figure 5.7: More visualizations on SA-1B [132]. From top to bottom are raw images, segmentation by SAM, segmentation by UnSAM, and segmentation by UnSAM+.

high-confidence mask predictions generated by UnSAM. When merging the pseudo masks with the ground truths for training UnSAM+, we select $\tau_{\text{UnSAM+}} = 0.02$.

Whole-image segmentation. UnSAM picks DINO [25] pre-trained ResNet-50 [58] as the backbone and Mask2former [94] as the mask decoder. Given the abundant number of pseudo masks generated, UnSAM augments data only by cropping a 1024×1024 region from the original image. To cope with a large amount of 'ground-truth' masks per image, we find that having 2000 learnable queries produces the best result. We randomly select at most 200 'ground-truth' masks per image to speed up the training process. The default learning rate is 5×10^{-5} with batch size equals 16 and weight decay 5×10^{-2}. We train the model for 8 epochs. All model training in this paper was conducted using either 4 A100 GPUs or 8 RTX 3090 GPUs.

Promptable segmentation. We use the self-supervised pretrained Swin-Transformer, specifically the Swin-Tiny model [159], as the backbone and leverage Semantic-SAM [149] as the base model. Given at most 6 levels of masks corresponding to one input point in SA-1B [132], we set the number of hierarchy levels to 6, which is also the number of predicted masks UnSAM generates per prompt during inference. However, one can easily train with a different number of granularity levels as needed. The default learning rate is 1×10^{-4} with a batch size of 8. The learning rate decreases by a factor of 10 at 90% and 95% of the training iterations. We train the model for 4 epochs.

Preliminary: Cut and Learn (CutLER) and MaskCut

CutLER [12] introduces a cut-and-learn pipeline to precisely segment instances without supervision. The initial phase, known as the cut stage, uses a normalized cut-based method, MaskCut [12], to generate high-quality instance masks that serve as pseudo-labels for subsequent learning phases. MaskCut begins by harnessing semantic information extracted from "key" features K_i of patch i in the last attention layer of unsupervised vision transformers. It then calculates a patch-wise cosine similarity matrix $W_{ij} = \frac{K_i K_j}{|K_i|_2 |K_j|_2}$. To extract multiple instance masks from a single image, MaskCut initially applies Normalized Cuts [53], which identify the eigenvector x corresponding to the second smallest eigenvalue. The vector x is then bi-partitioned to extract the foreground instance mask M^s. Subsequent iterations repeat this operation but adjust by masking out patches from previously segmented instances in the affinity matrix: $W^t_{ij} = \frac{\left(K_i \sum_{s=1}^{t} M^s_{ij}\right)\left(K_j \sum_{s=1}^{t} M^s_{ij}\right)}{\|K_i\|_2 \|K_j\|_2}$ Subsequently, CutLER's learning stage trains a segmentation/detection model with drop-loss, which encourages the model to explore areas not previously identified by MaskCut. An iterative self-training phase is employed for continuously refining the model's performance.

Preliminary: Segment Anything Model (SAM) and SA-1B

Inspired by achievement in the NLP field, the Segment Anything project [132] introduces the novel *promptable segmentation task*. At its core lies the Segment Anything Model (SAM) [132], which is capable of producing segmentation masks given user-provided text, points, boxes, and masks in a zero-shot manner. SAM comprises three key components: an MAE [47] pre-trained Vision

Transformer [29] that extracts image embeddings, the prompt encoders that embed various types of prompts, and a lightweight Transformer [160] decoder that predicts segmentation masks by integrating image and prompt embeddings.

One significant contribution of SAM [132] is the release of the SA-1B dataset, which comprises 11 million high-resolution images and 1.1 billion segmentation masks, providing a substantial resource for training and evaluating segmentation models. In particular, annotators interactively used SAM to annotate images, and this newly annotated data was then utilized to iteratively update SAM. This cycle was repeated multiple times to progressively enhance both the model and the dataset.

While SAM [132] significantly accelerates the labeling of segmentation masks, annotating an image still requires approximately 14 seconds per mask. Given that each image contains over 100 masks, this equates to more than 30 minutes per image, posing a substantial cost and making it challenging to scale up the training data effectively.

Evaluation Datasets

COCO (Common Objects in Context) [24] is a widely utilized object detection and segmentation dataset. It consists of 115,000 labeled training images, 5,000 labeled validation images, and more than 200,000 unlabeled images. Its object segmentation covers 80 categories and is mainly on the instance-level. We evaluate our model on COCO `Val2017` with 5000 validation images without training or fine-tuning on any images from the COCO training set. The metrics we choose are class-agnostic COCO style averaged precision and averaged recall for the whole-image inference task, and MaxIoU and OracleIoU for the promptable segmentation task.

SA-1B [132] consists of 11 million high-resolution (1500 on average) images and 1.1 billion segmentation masks, approximately 100 masks per image. All masks are collected in a class-agnostic manner with various subject themes including locations, objects, and scenes. Masks cover a wide range of granularity levels, from large-scale objects to fine-grained details. In the whole-image inference task, we randomly selected 1000 SA-1B images that are not used to generate pseudo labels as the validation set.

LVIS (Large Vocabulary Instance Segmentation) [61] has 164,000 images with more than 1,200 categories and more than 2 million high-quality instance-level segmentation masks. It has a long tail distribution that naturally reveals a large number of rare categories. In the whole-image inference task, we evaluate our model using its 5000 validation images in a zero-shot manner.

EntitySeg [140] is an open-world, class-agnostic dataset that consists of 33277 images in total. There are on average 18.1 entities per image. More than 80% of its images are of high resolution with at least 1000 pixels for the width. EntitySeg also has more accurate boundary annotations. In the whole-image inference task, we evaluate our model with 1314 low-resolution version images (800×1300 on average) in a zero-shot manner.

PACO (Parts and Attributes of Common Objects) [142] is a detection dataset that provides 641,000 masks for part-level entities not included in traditional datasets. It covers 75 object categories and 456 object-part categories. In the whole-image inference task, we evaluate our model with 2410 validation images in a zero-shot manner.

Figure 5.8: Failure cases of UnSAM. From left to right are raw images, segmentation by SAM, and segmentation by UnSAM.

PartImageNet [141] is a large-scale, high-quality dataset with rich part segmentation annotations on a general set of classes with non-rigid, articulated objects. It includes 158 classes and 24,000 images from ImageNet [161]. In the whole-image inference task, we evaluate our model with 2956 validation images in a zero-shot manner.

ADE20K [139] is composed of 25,574 training and 2,000 testing images spanning 365 different scenes. It mainly covers semantic-level segmentation with 150 semantic categories and 707,868 objects from 3,688 categories. In the whole-image inference task, we evaluate our model with 2000 testing images in a zero-shot manner.

More Visualizations

We provide more qualitative results of UnSAM and UnSAM+ in a zero-shot manner in Figure 5.9, and Figure 5.10.

Limitations

In images with very dense fine-grained details, UnSAM tends to miss repetitive instances with similar texture. As shown in Figure 5.8, in the first row, although UnSAM accurately segments the leaves in the center of the picture, it misses some leaves located at the top of the image. Additionally, UnSAM occasionally over-segment images. In the second row, the right sleeve cuff of the dancer has meaningless segmentation masks. This issue mainly arises because the unsupervised

clustering method mistakenly considers some information, such as folds and shadows on clothing, as criteria for distinguishing different entities. In contrast, human annotators can use prior knowledge to inform the model that such information should not be valid criteria. In this regard, unsupervised methods still need to close the gap with supervised methods.

Ethical Considerations

We train UnSAM and UnSAM+ on ground truths of and pseudo masks generated on SA-1B [132]. SA-1B contains licensed images that are filtered for objectionable content. It is geographically diverse, but some regions and economic groups are underrepresented. Downstream use of UnSAM and UnSAM+ may create their own potential biases.

5.7 Summary

Image segmentation is a fundamental task in computer vision, traditionally relying on intensive human annotations to achieve a detailed understanding of visual scenes. We propose UnSAM, an unsupervised segmentation model that significantly surpasses the performance of previous state-of-the-art methods in unsupervised image segmentation. Additionally, our unsupervised UnSAM model delivers impressive results, rivaling the performance of the cutting-edge supervised SAM, and exceeding it in certain semi-supervised settings.

Acknowledgement. We thank helpful discussions with Jitendra Malik, Cordelia Schmid, Ishan Misra, Xinlei Chen, Xingyi Zhou, Alireza Fathi, Renhao Wang, Stephanie Fu, Qianqian Wang, Baifeng Shi, Max Letian Fu, Tony Long Lian, Songwei Ge, Bowen Cheng and Rohit Girdhar. We thank Shengcao Cao and Hao Zhang for their help in reproducing baseline results. XuDong Wang and Trevor Darrell were funded by DoD including DARPA LwLL and the Berkeley AI Research (BAIR) Commons.

Figure 5.9: More visualizations on COCO [24]. From top to bottom are raw images, segmentation by SAM, segmentation by UnSAM, and segmentation by UnSAM+.

Figure 5.10: More visualizations on PACO [142]. From top to bottom are raw images, segmentation by SAM, segmentation by UnSAM, and segmentation by UnSAM+.

Part II

Analysis-by-Synthesis: Generative Models for Visual Scene Understanding

"To understand is to invent."

— Jean Piaget

Chapter 6

Visual Lexicon: Rich Image Features in Language Space

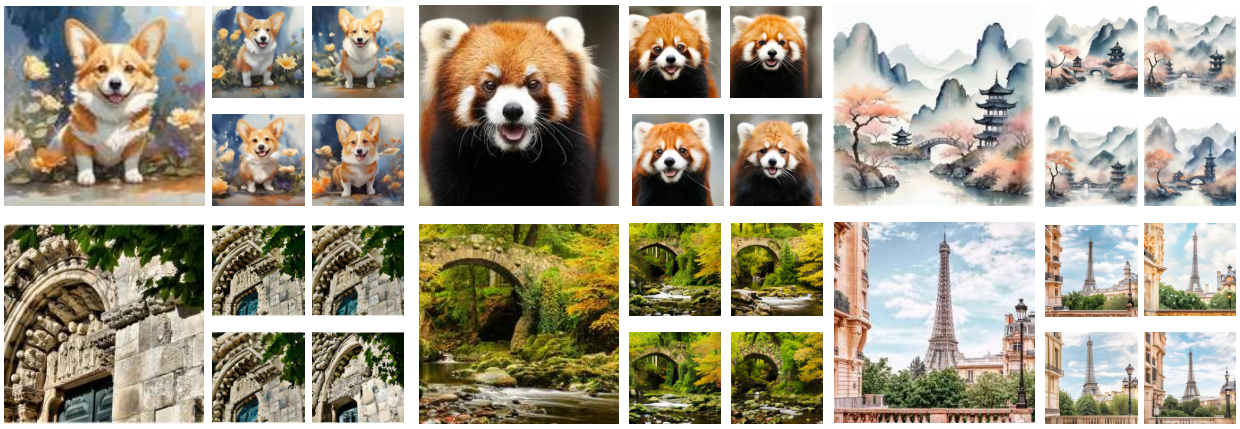


Figure 6.1: Given the cute corgi painting in the top left corner, how can we extract a visual representation that captures semantic-level information – such as object categories and layouts – while preserving rich visual details like image styles, textures and colors? **We introduce ViLex model that generates image representations in the text vocabulary space**, acting as a new visual "language", while retaining intricate visual details that are difficult, if not impossible, to convey in natural language. The set of images (generated under different diffusion noises) in the 2×2 grid, which are highly semantically and visually similar to each other, is created by using ViLex as "text" prompts for text-to-image diffusion models.

We present Visual Lexicon, a novel visual language that encodes rich image information into the text space of vocabulary tokens while retaining intricate visual details that are often challenging to convey in natural language. Unlike traditional methods that prioritize either high-level semantics (e.g., CLIP) or pixel-level reconstruction (e.g., VAE), ViLex simultaneously captures rich semantic content and fine visual details, enabling high-quality image generation and comprehensive visual scene understanding. Through a self-supervised learning pipeline, ViLex generates tokens

optimized for reconstructing input images using a frozen text-to-image (T2I) diffusion model, preserving the detailed information necessary for high-fidelity semantic-level reconstruction. As an image embedding in the language space, ViLex tokens leverage the compositionality of natural languages, allowing them to be used independently as "text tokens" or combined with natural language tokens to prompt pretrained T2I models with both visual and textual inputs, mirroring how we interact with vision-language models (VLMs). Experiments demonstrate that ViLex achieves higher fidelity in image reconstruction compared to text embeddings—even with a single ViLex token. Moreover, ViLex successfully performs various DreamBooth tasks in a zero-shot, unsupervised manner without fine-tuning T2I models. Additionally, ViLex serves as a powerful vision encoder, consistently improving vision-language model performance across 15 benchmarks relative to a strong SigLIP baseline.

6.1 Introduction

How should we represent an image? This is a fundamental question in computer vision. Over decades of progress, there have been two primary approaches: representations optimized for understanding high-level semantics [17], [38], [162], or for high-fidelity image reconstruction [47], [163], [164], often used in image generation [165], [166]. Understanding-focused models like CLIP [17] and DINO [25] capture high-level semantics but lose pixel-level details. Conversely, reconstruction-focused models, such as VAEs [163], retain fine visual details but lack semantic richness, making them less effective for tasks like vision-language modeling. In this paper, we aim to address the question: *"can a single representation excel in both image reconstruction and semantic understanding?"*

To bridge this gap, we introduce **ViLex** that encodes images into a **Vi**sual **Lex**icon within the text space. ViLex model is designed to capture both high-level semantics – such as object categories and spatial layouts – while preserving rich visual details like styles, and textures that are difficult or even impossible to articulate in natural language.

We achieve this, as illustrated in Fig. 6.2, by leveraging a self-supervised learning strategy based on a frozen, pretrained text-to-image (T2I) diffusion model, which acts as the source of supervisory signals. Although initially developed for generative purposes, many recent works [167]–[170] have discovered that diffusion models [165], [171], [172] inherently capture both semantic and detailed visual information through their denoising process.

To incorporate rich visual information into our ViLex model, we repurpose diffusion models as decoders within an autoencoder [163], [173]–[175] framework. ViLex embeddings are mapped into the latent space of the T2I diffusion model's vocabulary tokens—specifically, the index-to-embedding lookup matrix of the text encoder [17], [176], which converts text token IDs into embeddings. Using diffusion models as decoders for *semantic-level image reconstruction*, rather than traditional VAE decoders [163] or MAE [47] designed for *pixel-level reconstruction*, encourages the model to learn meaningful semantic representations that are highly transferable to diverse visual scene understanding tasks. This design enables ViLex to harness the rich visual knowledge

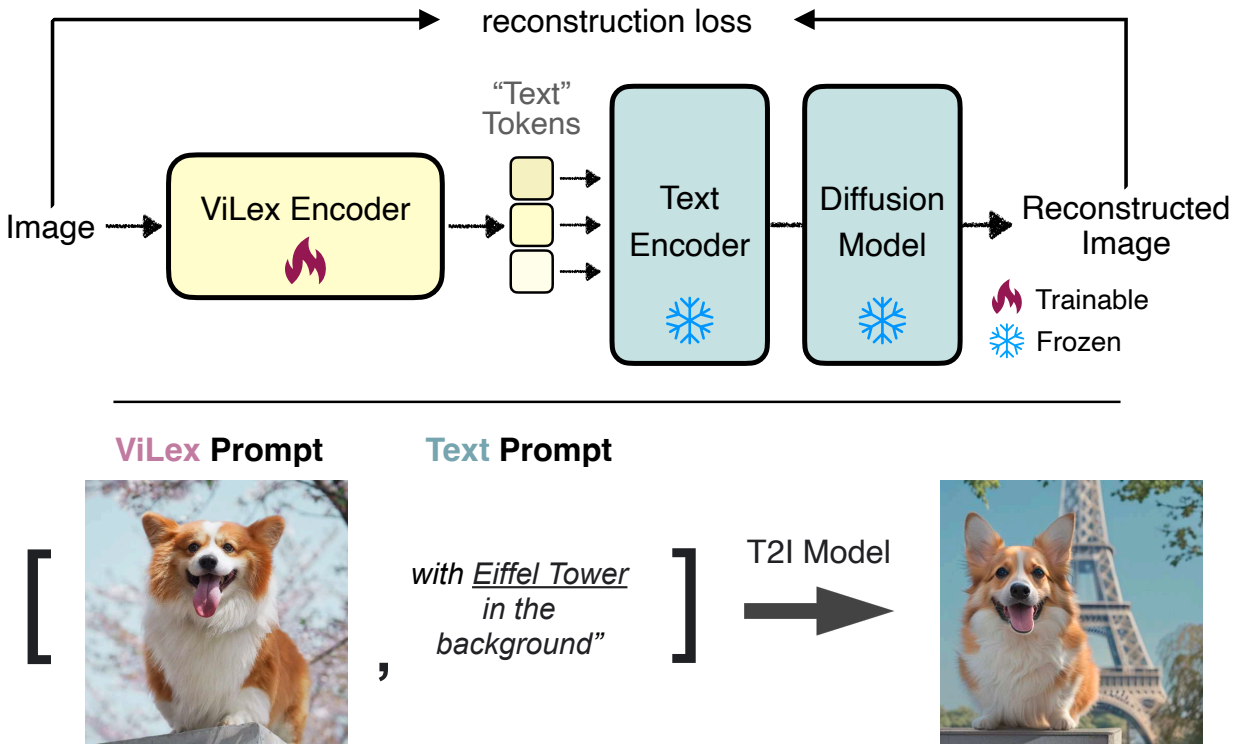


Figure 6.2: top) ViLex empowers linguistic space to capture visual richness. We propose **ViLex**, an image encoder that maps images into the vocabulary space, effectively preserving semantic information and intricate visual details. The embeddings from ViLex function as a *Visual Lexicon* that preserve semantic and intricate visual details of the image. ViLex is trained with a frozen text-to-image diffusion model and can be utilized independently as "text" tokens for image generation. **bottom) Linguistic space empowers ViLex to enjoy compositionality**. ViLex can be combined with natural language tokens for prompting a pretrained T2I diffusion models with both visual and textual cues.

embedded in diffusion models while maintaining a lightweight structure, making it well-suited for a broad range of understanding tasks beyond diffusion models' original generative applications.

The ViLex model consists of a vision encoder that extracts visual representations from the input image and an attention pooling layer that transforms the visual representation into visual lexicon tokens. During training, ViLex model is optimized with an image reconstruction loss, receiving gradients from the frozen diffusion model and its text encoder to fine-tune the visual lexicon tokens for accurately reconstructing images with similar appearance. Additionally, we propose the TailDrop strategy during training, where the last k visual lexicon tokens are randomly dropped to encourage the earlier tokens to encapsulate richer semantic information. During inference, the number of tokens can be dynamically adjusted to meet user requirements.

Our ViLex model is designed to support both image generation and understanding tasks. ***For image generation:*** ViLex tokens can be directly used as "text-prompts", enabling the re-creation of semantically and visually similar images. Experiments on COCO image reconstruction demon-

strate that our ViLex significantly outperforms its counterparts image-guided DALL·E 3 [177] and DeDiffusion [178] in terms of the layout, semantic, and style consistency with the input image, based on human studies. Notably, even with just a single ViLex token, the FID score of ViLex remains lower than that of DeDiffusion [178], showcasing the representational power of Visual Lexicon. Additionally, ViLex embeddings can seamlessly integrate with text prompts, for example, *"an image similar to [ViLex tokens], in Van Gogh style"*, enabling ***multimodal image generation*** and DreamBooth [179] tasks in a zero-shot fashion by prompting a frozen T2I model with both visual and textual inputs. ***For image understanding:*** replacing the strong semantic-pretrained backbone SigLIP [180] with ViLex's vision encoder in vision-language models [181] leads to improvements across various vision-language tasks, including image and video captioning [182], [183], visual question answering [184], and referring segmentation [185].
The main contributions of our work are:

- We propose ViLex, an image encoder that maps images into the text space of text-to-image diffusion models. The resulting image embeddings capture both high-level semantics and intricate visual details that are otherwise challenging to convey in natural language.
- ViLex enables zero-shot multimodal image generation by prompting T2I models with both ViLex tokens and text prompts, without requiring fine-tuning a T2I model or modifying its architectures. ViLex also improves the performance of generating semantically similar images compared to previous baselines, significantly reducing FID.
- ViLex enhances the understanding capability of the image embeddings. Replacing the image encoder in vision-language models with ViLex yields performance gains on various visual understanding tasks, including image/video captioning, VQA, and referring segmentation.

6.2 Related Work

Image Representation Learning is a fundamental task in computer vision. There are two popular approaches: representing an image with features optimized for visual scene understanding [9], [17], [38], [40], [162], [180], [186]–[188] or with features optimized for high-fidelity image reconstruction [47], [163], [164], [189], which is often used in image generation [165], [190]. Understanding-focused representations like those in CLIP [17], SigLIP [180], DINO [25], and DINOv2 [25], [162] capture high-level semantic information but lose pixel-level details. Conversely, reconstruction-focused features, commonly from AutoEncoder-based models (AEs), like VAE [163], MAE [47], and BEiT [191], retain fine image details but often lack semantic richness, limiting their utility in downstream tasks like vision-language modeling [181], [192]. AutoEncoders, while effective for pixel-level fidelity, often struggle with discriminative tasks. Their focus on reconstructing local, semantically agnostic details leads to suboptimal performance in tasks demanding rich, discriminative representations [191], [193], [194]. We intend to propose a new vision encoder that provides image representations for both image understanding and generation tasks.

Image Inversion for Diffusion Models aims to determine the text prompt that can be used for generating a specific source image. Prompt-inversion [195], [196] uses gradient descent to move from the pixel space to the text-embedding space. Techniques like Dreambooth [179], [197] and textual-inversion [198] learn special text tokens for given instances, but require gradient-based training for each individual image, making them slow at inference time. Also, DreamBooth needs to determine the LORA adapters for model architecture changes and is not generic for visual understanding. Recently, DeDiffusion [178] proposed training a model to generate the inverse text using Gumbel softmax, but the quality of reconstruction is limited by what can be represented by text tokens. Our approach bypasses discrete language-based text representations, enabling higher-quality reconstructions and efficient single-pass inference.

Representation Learning with Diffusion Models has been explored by several previous works [170], [199]–[203]. l-DAE [199] uses the diffusion loss as a self-supervised learning objective, while ODISE [170] employs diffusion-pretrained features for zero-shot panoptic segmentation. DIVA [201] shows that finetuning a CLIP backbone [17] with gradients from diffusion models enhances localization capabilities. In contrast, we harness the built-in knowledge of T2I models to learn visual features, effectively framing the generation of text embeddings that reconstruct an image as a powerful representation learning objective.

Image Tokenization, commonly associated with variational autoencoders (VAEs) [163], is crucial for compressing images into a lower-dimensional space for diffusion model training. VQ-VAEs [171], [204] utilizes a discrete codebook for quantizing latent representations, while recent works like MagViTv2 [205], FSQ [206], and BSQ [207] improve quantization with direct binary encoding. More recent image tokenizers [164], [208], [209] propose new encoding strategies, including scale prediction [208], [209] and 1D compression [164]. Unlike these tokenizers, which predict features as noise, our model predicts features conditioned on the diffusion model. Thus, instead of aiming for lossless reconstruction, our focus is on recreating images with high semantic fidelity.

6.3 Describe an Image with a Visual Lexicon

In this subsection, we introduce ViLex that maps images directly into the text space, while effectively preserving complex visual details that are difficult to express in natural language. Our representation effectively acts as a new "language" for text-to-image generation and a strong vision encoder for downstream vision-language tasks.

Approach overview. ViLex aims to capture high-level semantic representations – such as object categories and layouts – while also preserving rich visual details like styles, patterns, and textures that are difficult or even impossible to describe in natural language.

To accomplish this, we train ViLex through a self-supervised learning pipeline, with a frozen pretrained text-to-image (T2I) diffusion model serving as the source of supervisory signals. This approach enables ViLex to extract visual representations that encompass both semantic-level understanding and intricate visual features. Unlike previous works that directly use diffusion models as feature extractors [167]–[170], we take a different approach. In our framework, diffusion mod-

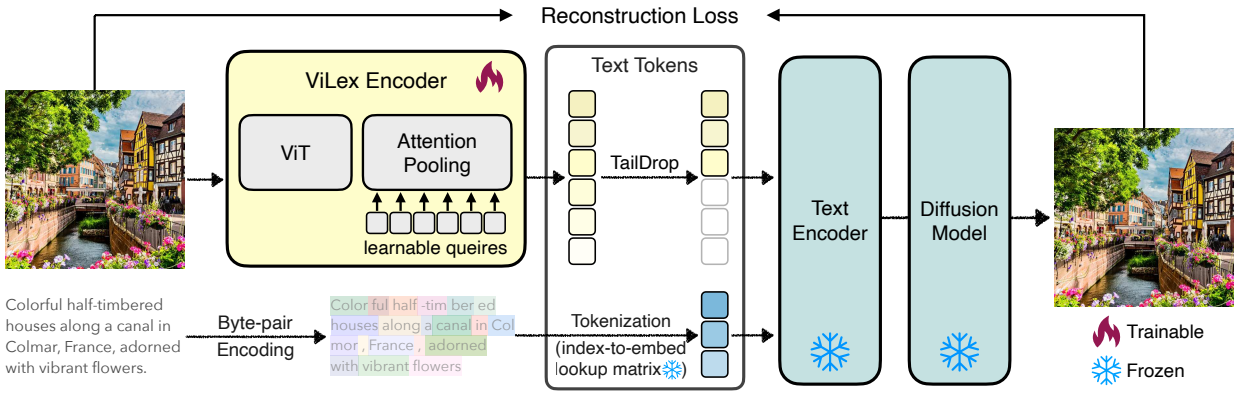


Figure 6.3: The pipeline of ViLex: We learn a Visual Lexicon from a frozen diffusion model using an image reconstruction loss. After training, ViLex can be directly used as the "text-prompt" to a frozen text encoder, *e.g.*, CLIP or T5, enabling the re-creation of semantically similar images without the need for actual text prompts. In addition, during training, we implement the TailDrop strategy, where the last k tokens are randomly dropped, encouraging earlier tokens in ViLex to carry richer semantic information. ViLex tokens can be utilized independently as "text" tokens for image generation or combined with natural language tokens for prompting T2I diffusion models with both visual and textual cues for multimodal image generation.

els act as decoders in an autoencoder [163], [173]–[175] pipeline, which allows us to bake the semantic richness learned by these models into a vision encoder. As a result, ViLex benefits from the detailed and rich visual representations of diffusion models while being significantly more lightweight, making it suitable for a broader range of visual scene understanding applications beyond diffusion models' generative origins.

Training approach. As depicted in Fig. 6.3, we employ an autoencoder framework to learn ViLex from a pre-trained text-to-image diffusion model. The overall approach consists of three key components: 1) Image Encoder: A vision transformer (ViT) [210] based image encoder that extracts visual representations from the input image. 2) Image-to-text projection: An attention pooling module that transforms the visual representation into ViLex embeddings within the text space. These embeddings can be independently used as inputs for the frozen text encoder and the subsequent diffusion model, or concatenated with text tokens derived from natural language. 3) Decoder: A pretrained text-to-image diffusion model serves as the decoder in the autoencoder pipeline, generating images from the "text" tokens.

During training, ViLex encoder – which comprises the ViT and the attention pooling module – is optimized via the gradients from image reconstruction loss, while the text-to-image (T2I) model remains frozen throughout the process. After training, ViLex can function as a new "language", effectively serving as a "text prompt" for frozen text encoders such as CLIP [17] or T5 [176]. This enables the generation of semantically similar images without traditional text-based prompts, capturing intricate visual details.

Represent images as text embeddings. Since ViLex is designed to serve as text tokens for text-to-image (T2I) diffusion models, we project the k patch-level representations p_i of an image i into

Figure 6.4: ViLex retains more visual details in image-to-image generation compared to DALL·E 3 [177] and DeDiffusion [178], accurately capturing elements such as image style (*e.g.*, the oil painting style in row 1), layout (*e.g.*, the relative position of the corgi and the lighthouse), pose (*e.g.*, the corgi's stance), and object shapes (*e.g.*, the shape of Van Gogh's hat). This enables ViLex to produce images that are both semantically and visually consistent with the original input. Even models with text embeddings in a shared language-vision space, like DALL·E 3, capable of generating semantic variations of an image, struggle to faithfully reconstruct the original appearance of the input image. For image-guided DALL·E results, we provide the input images along with the text prompt, *"generate an image exactly the same as the input image"*. For DeDiffusion, we follow its official image-to-image generation pipeline and use SDXL [190] as the T2I model.

the text space using a multi-head cross-attention layer [211], denoted as $f(\cdot)$. This layer contains n learnable queries and uses the k output patch tokens from the image encoder as inputs. These k patch tokens function as both keys and values within the cross-attention mechanism. Through this setup, the model learns to pool the k patch tokens into ViLex embeddings, denoted as v, consisting of n tokens such that $v_i = f(p_i)$, as shown in Fig. 6.3.

To illustrate how to implicitly align ViLex embeddings v with text tokens c compatible with a pretrained T2I model, let's first examine how actual text prompts are tokenized. Using Byte-Pair Encoding (BPE) [2], [212], [213] as an example – a tokenizer employed by CLIP [17] – BPE tokenizes text into sub-word units, effectively managing large vocabularies and handling rare or unseen words. The process has two steps: *1. Tokenization with BPE*: Each input sentence is tokenized using BPE, yielding a sequence of sub-word tokens. For instance, the phrase "hello world" might be tokenized as: tokens $=$ ["hel", "lo", "wor", "ld"]. *2. Embedding Lookup*: Each sub-word token is mapped to a learned embedding vector via a pre-trained vocabulary lookup matrix $\mathcal{V}$. If e_i denotes the embedding for token i, each sub-word token with an index t_i is given by $e_i = \mathcal{V}[t_i]$.

The ViLex embeddings v are trained to be implicitly aligned with the latent space of the lookup matrix $\mathcal{V}$, ensuring compatibility with T2I diffusion models. Before feeding ViLex embeddings v independently or as part of the concatenated token sequence $[v, c]$ to the T2I model, where c is the text tokens, we add [BOS] and [EOS] tokens at the beginning and end of the sequence, respectively.

Text-free guidance for multimodal image generation. Classifier-Free Guidance (CFG) [214] has

been a popular technique to enhance the quality of generated samples of diffusion models by controlling the trade-off between adhering to a given prompt and producing diverse outputs. In CFG, two sets of samples are generated: one conditioned on the input (*e.g.*, text prompt) and one unconditioned, allowing flexible guidance during the sampling process. Let $\epsilon_\theta(x_t, c)$ denote the noise predicted by the model conditioned on a prompt c at time t, and $\epsilon_\theta(x_t)$ denote the unconditioned noise prediction. In CFG, the final prediction ϵ_{guided} is computed as:

$$\epsilon_{\text{guided}} = \epsilon_\theta(x_t) + w \cdot (\epsilon_\theta(x_t, c) - \epsilon_\theta(x_t)) \tag{6.1}$$

Increasing the guidance scale w intensifies the prompt adherence, improving fidelity to c at the cost of diversity.

Inspired by CFG, we introduce Text-Free Guidance (TFG) for our multimodal image generation, balancing the influence of a given text prompt c and ViLex embedding v. TFG controls the trade-off by incorporating visual representations from ViLex alongside the text prompt, enabling finer control over generated images. In TFG, we modify the noise prediction by combining the conditioned prediction on the visual and text prompts, denoted as $\epsilon_\theta(x_t, [v, c])$, and the prediction conditioned on ViLex alone, *i.e.*$\epsilon_\theta(x_t, v)$. The TFG noise prediction ϵ_{tfg} is then computed as:

$$\epsilon_{\text{tfg}} = \epsilon_\theta(x_t, v) + w_{\text{tfg}} \cdot (\epsilon_\theta(x_t, [v, c]) - \epsilon_\theta(x_t, v)) \tag{6.2}$$

where w_{tfg} is the guidance scale, allowing us to control the impact of ViLex tokens relative to the text prompt. TFG enables multimodal image generation by incorporating both textual and visual cues, *without* requiring T2I model architectural changes or using LoRA [215] adapters.

TailDrop for dynamic visual token compression. Different images contain varying amounts of information, this creates a trade-off between representation compactness and detail richness. We propose a flexible token budget method using a similar masking strategy as in SoundStream [216], which we refer to as TailDrop. Specifically, during training, we randomly drop the last k ViLex tokens. Since the early tokens are more frequently independently used for image generation, the earlier tokens in ViLex are encouraged to carry richer semantic information. After the model training, during the inference time, users can dynamically adjust the number of tokens in ViLex to suit the needs.

Training loss. We adopt the standard diffusion [172], [217], [218] training objective to optimize ViLex, backpropagating the reconstruction loss to update its parameters. In a diffusion model, the denoising objective aims to learn a model $\epsilon_\theta(x_t, t)$ that predicts the noise ϵ added to data x_0 at timestep t. Given a noisy sample x_t, the objective minimizes the difference between the predicted and true noise:

$$\mathcal{L}_{\text{denoise}} = \mathbb{E}_{x_0, \epsilon, t}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|^2\right], \tag{6.3}$$

where $x_t = \sqrt{\alpha_t} x_0 + \sqrt{1 - \alpha_t}\epsilon$, with α_t controlling the noise schedule. This loss enables the model to reverse the diffusion process, gradually reconstructing x_0 from x_t.

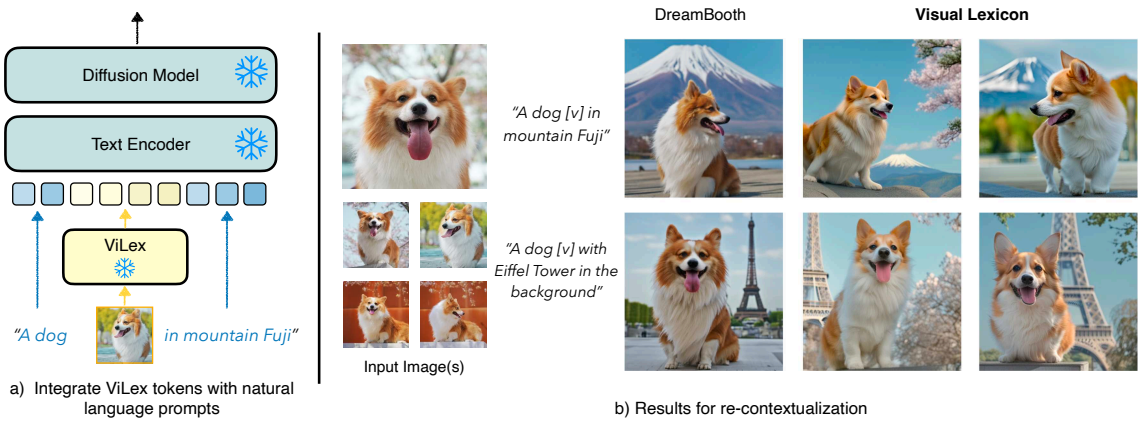


Figure 6.5: ViLex can be seamlessly integrated with natural language prompts for **zero-shot unsupervised image re-contextualization** using a frozen text-to-image (T2I) diffusion model. Unlike DreamBooth [179], ViLex requires no fine-tuning of the T2I model on a set of input images from the same object or modifications to the model architecture (*e.g.*, adding a LoRA [215] adapter). Instead, ViLex is a universal model that enables zero-shot, unsupervised re-contextualization by simply prompting the T2I model with ViLex tokens and corresponding text prompt tokens, just like how we use real words. **a)** The inference pipeline demonstrating image re-contextualization. **b)** Qualitative comparisons with DreamBooth, with DreamBooth results taken from their project page.

6.4 Experiments

Technical Details

Text-to-image diffusion model. Following DeDiffusion [178], we use Imagen [165] as the base text-to-image diffusion model, adapting the U-Net architecture from [219], [220] with 600M parameters, an embedding dimension of 256, and an input resolution of 64×64. The text encoder of Imagen is OpenCLIP ViT-H/14 [221], [222] with a vocabulary size of 49408. The U-Net conditions on text embeddings via a pooled embedding vector, which is added to the diffusion timestep embedding. Imagen is further conditioned on the full sequence of text embeddings by incorporating cross-attention over the text embeddings at multiple resolutions.

Model architecture of ViLex. ViLex consists of two components: a ViT-based image encoder and a transformer-based attention pooling module. Both components are unfrozen during the training process. For the image encoder, we use a pretrained SigLIP@224 [180]. SigLIP utilizes ViT-base as the backbone and is pretrained on the WebLI dataset [223] using a sigmoid loss and trained on English image-text pairs, with input images resized to 224×224. Attention pooling contains n learnable queries, where $n \leq 75$, along with [SOS] and [EOS] tokens to ensure the total token count remains within the 77-context length limit defined by the CLIP text encoder [17], [221]. We randomly initialize the attention pooling layer (5 transformer blocks).

Model training. The training data is obtained from WebLI [223], enabling training on either images alone or with image-text pairs. We found that joint image-text training and our TFG are essential for enabling multimodal image generation. However, training without text captions does

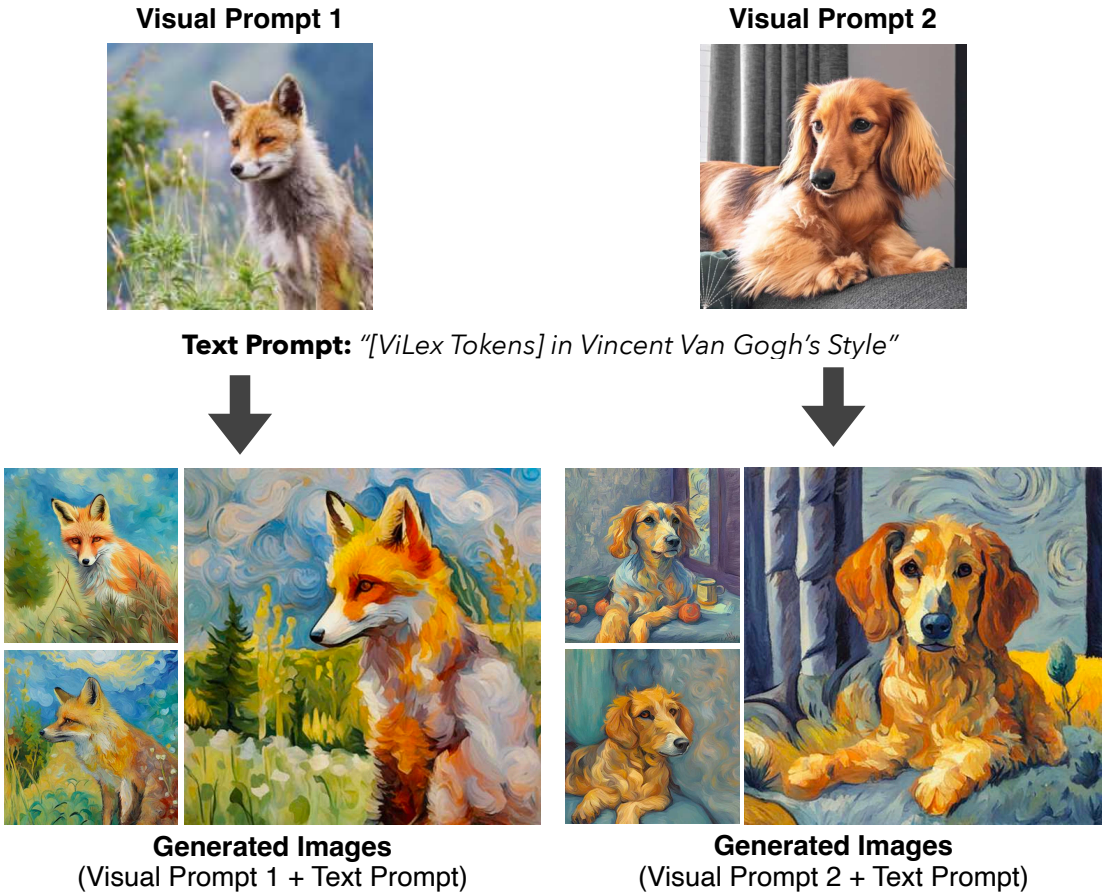


Figure 6.6: ViLex can also support zero-shot unsupervised art rendition via prompting T2I models with ViLex and text prompts.

not negatively impact performance on downstream vision-language tasks. Following [165], [178], we use Adafactor optimizer [224] and a weight decay of 0.01. Training is performed with a batch size of 2048 over 300K steps, which takes approximately 2.5 days on 64 TPUv5 chips. The ViT is initialized with a pretrained SigLIP model. We use learning rate 1×10^{-5} for the image encoder and 3×10^{-4} for the attention pooling layers, with a cosine learning rate decay and a 10K-step linear warmup. More training details are in the supplement. After training, our ViLex encoder maps an image to ViLex representations. We next evaluate two capabilities of these frozen ViLex representations: image generation and visual understanding.

Experiments on Image Generation

Image-to-image generation aims to generate similar images given an input image. Feeding our extracted ViLex features on the input image to our T2I model [165] with different diffusion random noise yields re-created images. We compare with two popular models: DALL·E 3 [177] and DeDiffusion [178], both of which convert the input image to explicit texts. DeDiffusion [178] used

	T2I	DeDiffusion	ViLex			
			1	4	16	75
FID (↓)	6.52	3.89	3.65	2.91	2.38	**2.07**
IS (↑)	14.06	14.68	15.33	15.42	15.51	**15.88**

Table 6.1: Even when only using just one continuous token, ViLex outperform the discrete tokens from both DeDiffusion [178] (image→text→image) and the vanilla Imagen [165] (text→image). FID scores on MSCOCO-64×64 were used for image reconstruction comparison across various image generation pipelines, all of which employ Imagen [165] as the base text-to-image diffusion model for fair comparisons. IS refers to inception score.

	DeDiffusion			DALL·E 3		
	layout	semantic	style	layout	semantic	style
vs. ViLex (↑)	98%	95%	98%	91%	76%	90%

Table 6.2: Human studies on generating semantically and visually similar images using the image-to-image pipeline. We report the percentage of ratings favoring ViLex over DeDiffusion [178] and the image-guided DALL·E 3 [177] in terms of layout, semantic, and style consistency with the input image.

		Image Captioning					Visual Question Answering						Image Segmentation						Video
Backbone	FID (↓)	*COCOcap*	*COCO-35L*	*TextCap*	*SciCap-Val*	*SciCap-Test*	*VQAv2-Val*	*TextVQA*	*OKVQA*	*SciQA*	*VizWizQA*	*GQA*	*RC-val*	*RC-testA*	*RC-testB*	*RCp-testA*	*RCp-testB*	*RCg-test*	*MSRVTT*
Original SigLIP	2.54	139.7	138.6	122.1	131.7	135.5	81.4	51.9	57.1	85.9	74.3	64.8	66.2	69.0	63.6	63.3	55.3	59.6	69.4
ViLex SigLIP	**2.38**	**141.5**	**139.4**	**124.0**	**134.3**	**136.2**	**81.6**	**52.9**	**58.4**	**87.9**	**74.9**	**65.3**	**67.6**	**70.0**	**65.1**	**65.3**	**57.3**	**62.6**	**70.7**

Table 6.3: ViLex improves both image understanding and reconstruction capabilities of vision encoders by fine-tuning them using ViLex's training approach. Compared with the official SigLIP model [180], ViLex SigLIP, fine-tuned with ViLex approach, demonstrates superior image reconstruction quality (evidenced by a lower FID score) and enhanced visual scene understanding (as shown by improved results on numerous vision-language tasks). We utilize PaliGemma's [181] framework for linear evaluation, replacing the vision encoder with either the fine-tuned SigLIP in ViLex or the official one, and freeze vision encoder and fine-tune the model on downstream tasks. We use the same hyper-parameters and model architecture for a fair comparison. RC refers to RefCOCO dataset.

exactly the same encoder architecture and T2I model as us, giving us an apple-to-apple comparison between our ViLex feature and texts. We conducted both human studies (Table 6.2) and quantitative metrics on Fréchet Inception Distance (FID) [225] and Inception Score (IS) [226] scores (Table 6.1).

The quantitative results in Table 6.1 show that our ViLex efficiently preserves visual information, surpassing the explicit text counterpart [178] even with only 1 token in both FID and IS metrics. We further evaluate the effectiveness of ViLex through human assessments on image-to-image generation tasks. Results in Table 6.2 indicate that ViLex significantly outperforms the baselines, achieving 98% win rates in layout alignment, 95% in semantic fidelity, and 98% in style preservation against [178]. ViLex also outperforms DALL·E 3. We present qualitative compar-

Figure 6.7: Qualitative results of semantic-level image reconstruction with varying numbers of ViLex tokens. ViLex tokens enable the reconstruction of semantically similar images even with a single token, effectively capturing high-level semantics such as categories, object counts, and overall poses. As the number of tokens increases to 16, finer details begin to emerge: mid-level features like image styles, colors, and object textures become apparent. With 75 ViLex tokens, the reconstructions achieve high appearance fidelity, incorporating fine-grained details that blend both low-level and high-level information, such as precise object shapes, sizes, and cross-instance relationships.

isons in Fig. 6.4.

Zero-shot unsupervised multimodal image generation with identity preserving. One advantage of mapping images to the word space is to embed images directly into a sentence in an interleaved way. This enables multimodal image generation using a pure text-to-image model without finetuning. In Fig. 6.5, we demonstrate the ability of ViLex to serve as a "language" for multimodal image generation while preserving the object identity in a given image. The conventional approach to this task like DreamBooth [179] requires LoRA [215] adapters and a test-time finetuning with a set of images to learn an embedding for object identity which is slow and computationally expensive, while ours directly infers the target identity feature using our encoder. Figs. 6.5 and 6.6 shows that ViLex improves the resulting images by embedding detailed semantic and visual context, producing coherent multimodal results that reflect both the textual prompt and the visual cues provided. We highlight again that our model is not finetuned for this task and does not require a text-time-finetuning like DreamBooth, but simply prompting a standard T2I model with ViLex embeddings and corresponding text prompts.

When do we need more ViLex tokens? Fig. 6.7 presents qualitative results of semantic-level image reconstruction with varying numbers of ViLex tokens, illustrating the gradual refinement of visual details as the token count increases. With just 1 token, ViLex effectively captures high-level semantic information, such as object categories, counts, and poses. As the the token count increases to 16, mid-level features, including image styles, colors, and textures, start to emerge, enhancing the overall visual representation. Finally, with 75 tokens, ViLex captures fine-grained details such as precise object shapes, sizes, and intricate cross-instance relationships. For instance, in the third-row image, the two cats' poses are accurately reconstructed, including their specific positioning and interaction as they cuddle together. These results demonstrate ViLex' adaptability, providing users with the flexibility to balance semantic richness and visual detail based on the application requirements.

Experiments on Image Understanding

We next verify that our frozen ViLex features can be directly used for understanding tasks, by feeding them to a large language model [227]. We use PaliGemma [181] as our visual-language model (VLM) architecture. PaliGemma [181] is an open-source VLM with a SigLIP-So400m [180] vision encoder and a Gemma-2B [227] language model. To use ViLex in PaliGemma, we replace the SigLIP [180] vision encoder with our ViLex encoder. In all our following experiments, we don't finetune our ViLex encoder, and only finetune the following large language model [227] to adapt to the tasks following the PaliGemma transfer design [181].

Improving vision encoders over SigLip. We first conduct a comparison between the visual encoder learned from our ViLex pretraining and the popular SigLip [180]. To ensure a fair comparison in terms of the architecture and the number of tokens, we use our features right after the ViT-encoder without the pooling layer. Table 6.3 shows the results. We also show the image reconstruction FID. The results show that ViLex feature consistently improves the strong SigLip model by over 1 point margin on a variety number of vision-language tasks, including image captioning, referring segmentation, and video understanding. This shows that ViLex can effectively upgrade existing vision encoders to perform better in both reconstruction and understanding.

Image Encoder	COCOcap	TextCaps	TextVQA	OKVQA	SciQA	RC-val	RCp-testA	RCg-test
VAE	61.4	15.0	12.9	33.3	82.1	19.5	12.1	13.8
ViLex SigLIP	**141.5**	**124.0**	**52.9**	**57.5**	**87.9**	**67.2**	**65.3**	**62.6**
ViLex	**142.8**	**137.7**	**59.4**	**58.3**	**88.6**	**70.9**	**69.2**	**65.9**

Table 6.4: Improving vision-language models with ViLex tokens. Concatenating ViLex tokens with SigLIP patch tokens enhances the performance of vision-language models across diverse downstream tasks, such as image captioning, visual question answering, and image segmentation, with a modest token increase of only 25% (from 256 to 336). **Pixel-reconstruction or semantic-reconstruction?** Although VAE can preserve pixel-level details during image reconstruction, its lack of semantic richness results in significantly poorer performance in vision-language modeling compared to ViLex. We use PaliGemma as the base VLM model.

Vision-language tasks. ViLex can also serve as a strong vision encoder for vision-language models. ViLex tokens are inherently ready to use for vision-language frameworks, therefore, they can be directly utilized without requiring fine-tuning the image encoder. Despite freezing the image encoder—including both the ViT and attention pooling layers—we observed significant performance improvements across various tasks, such as visual question answering, image captioning, and referring expression segmentation, as demonstrated in Table 6.4 and Table 6.5. Following prior works like QwenVL-7B, LLaVA1.5-13B, LLaVA1.6-13B [134], [228], [229], we adopt a multi-grid input strategy for extracting visual representations to ensure a fair comparison with these methods. However, unlike these methods, which increase the overall number of tokens by 2$\sim$5 times, we use only 16 ViLex tokens per grid. This results in only 25% increase in token count (adding just 80 tokens) while achieving substantial performance improvements and setting new state-of-the-art results on multiple VQA benchmarks.

Pixel-reconstruction or semantic-reconstruction? Although VAE [163], [171], [173] excels at preserving pixel-level details during image reconstruction, its [171] lack of semantic richness leads to substantially poorer performance (often $>$8 times lower) in vision-language modeling compared to ViLex, as demonstrated in Table 6.4.

Appendix Materials

Technical Details

We introduced the main technical and implementation details of our ViLex model in the main paper, here we provide a more comprehensive explanation.

Text-to-image diffusion model. Following DeDiffusion [178], we use Imagen [165] as the base text-to-image diffusion model, adapting the U-Net architecture from [219], [220] with 600M parameters, an embedding dimension of 256, and an input resolution of 64×64. The text encoder of Imagen is OpenCLIP ViT-H/14 [221], [222] with a vocabulary size of 49408. The U-Net condi-

Model	#toks	VQAv2	SciQA	VizWizQA	GQA
Freeze image encoder					
PaliGemma-2B (SigLIP@224)	256	81.4	85.9	74.3	64.8
PaliGemma-2B (ViLex@224)	336	**83.6**	**88.6**	**75.8**	**65.7**
Fine-tune image encoder					
LLaVA1.5-13B (CLIP@224)[228]	576	80.0*	71.6	53.6	62.0*
VILA-13B (CLIP@336)[230]	576	80.8*	73.7	60.6	63.3*
QwenVL-7B (CLIP@448)[134]	1024	78.8*	67.1	38.9	57.5*
LLaVA1.5-13B (CLIP@336)^{HD}[228]	1280	81.8*	71.0	57.5	63.3*
LLaVA1.6-13B (CLIP@384)^{HD}[229]	1280	81.8*	70.2	-	64.2*

Table 6.5: ViLex can be a strong vision encoder for vision-language tasks. Using the *lowest image resolution*, *fewer tokens* per image, *the smallest LLM model*, and *without fine-tuning image encoder*, our ViLex —integrated into PaliGemma as the image encoder—achieves SOTA performance across multiple visual question answering tasks. *: The training images from the datasets are utilized during model training or for fine-tuning the model.

tions on text embeddings via a pooled embedding vector, which is added to the diffusion timestep embedding. Imagen is further conditioned on the full sequence of text embeddings by incorporating cross-attention over the text embeddings at multiple resolutions. The Imagen model uses v-prediction [231] as its objective, with a batch size of 2048, and is trained for 3 million steps. As a baseline model, Imagen achieves an FID of 6.52 on 30K 64×64 MS-COCO 2014 validation images [165]. During image generation inference, we use a super-resolution model, such as an SDXL upsampler, to upsample the image resolution from 64×64 to 512×512 for better visualizations.

Model architecture of ViLex. ViLex consists of two components: a ViT-based image encoder and a transformer-based attention pooling module. Both components are unfrozen during the training process. For the image encoder, we use a pretrained SigLIP-So400M@224 [180]. SigLIP utilizes ViT-base as the backbone and is pretrained on the WebLI dataset [223] using a sigmoid loss and trained on English image-text pairs, with input images resized to 224×224. The model architecture of the ViT-base is shape-optimized on 400M training samples for improving the model efficiency and speed. In our method, the attention pooler is implemented as a single multi-head attention layer with learnable queries, using the encoder output as both keys and values. This allows the attention pooling module to effectively aggregate embeddings of varying lengths. The attention pooling module contains n learnable queries, where $n \leq 75$, along with [SOS] and [EOS] tokens to ensure the total token count remains within the 77-context length limit defined by the CLIP text encoder [17], [221]. The attention pooling layer comprises 5 transformer blocks, which are always randomly initialized.

Model training. The training data is obtained from WebLI [223], enabling training on either images alone or with image-text pairs. We found that joint image-text training and our TFG are essential for enabling multimodal image generation. However, training without text captions does

		Image Captioning					Visual Question Answering						Image Segmentation						Video
Backbone	#steps	*COCOcap*	*COCO-35L*	*TextCap*	*SciCap-Val*	*SciCap-Test*	*VQAv2-Val*	*TextVQA*	*OKVQA*	*SciQA*	*VizWizQA*	*GQA*	*RC-val*	*RC-testA*	*RC-testB*	*RCp-testA*	*RCp-testB*	*RCg-test*	*MSRVTT*
Original SigLIP	-	139.7	138.6	122.1	131.7	135.5	81.4	51.9	57.1	85.9	74.3	64.8	66.2	69.0	63.6	63.3	55.3	59.6	69.4
ViLex SigLIP	150k	**140.5**	**138.8**	**122.3**	**132.6**	**135.5**	**81.4**	**52.1**	**57.3**	**86.1**	**74.5**	**65.1**	**66.5**	**69.3**	**64.2**	**64.1**	**55.6**	**60.2**	**70.6**
ViLex SigLIP	600k	**141.5**	**140.0**	**124.0**	**134.3**	**136.1**	**81.8**	**52.7**	**58.3**	**89.3**	**75.0**	**65.4**	**67.5**	**69.7**	**65.6**	**65.2**	**57.2**	**62.6**	**71.4**

Table 6.6: ViLex improves both image understanding and reconstruction capabilities of vision encoders by fine-tuning them using ViLex's training approach. **Extending the fine-tuning of SigLIP with the ViLex approach from 150k to 600k steps results in improved overall model performance across evaluated benchmarks.** We use PaliGemma's [181] framework for linear evaluation, replacing the vision encoder with either the fine-tuned SigLIP in ViLex or the official one, and freeze vision encoder and fine-tune the model on downstream tasks. We use the same hyper-parameters and model architecture for a fair comparison.

not negatively impact performance on downstream vision-language tasks. Following [165], [178], we use Adafactor optimizer [224] and a weight decay of 0.01. Training is performed with a batch size of 2048 over 300K steps, which takes approximately 2.5 days on 64 TPUv5 chips. We found that double the training steps (from 300k to 600k) can further improve the model performance on increasing the performance of a pretrained vision encoder. The ViT is initialized with a pretrained SigLIP model and the attention pooling layers are randomly initialized. We use learning rate 1×10^{-5} for the image encoder and 3×10^{-4} for the attention pooling layers, with a cosine learning rate decay and a 10K-step linear warmup, and a weight decay of 0.01. After training, our ViLex encoder maps an image to ViLex representations. We next evaluate two capabilities of these frozen ViLex representations: image generation and visual understanding.

PaliGemma experiments. To evaluate the effectiveness of the proposed ViLex approach in enhancing a pretrained vision encoder for vision-language tasks, we integrate our vision encoder into the PaliGemma [181] framework and replace the vision encoder with either the fine-tuned SigLIP-So400M [180] from ViLex or the official version without model fine-tuning, freezing the vision encoder and fine-tuning the model on downstream tasks. Following PaliGemma's official pipeline, we transfer the model to a variety of individual academic benchmarks using a unified transfer approach with minimal hyperparameter tuning. To ensure fair comparison, we applied the same hyperparameter sweeping strategy for both the baseline and our fine-tuned vision encoder, reporting the best results for each. This structured approach allows us to fairly assess the impact of the proposed ViLex method on a wide range of vision-language tasks. The sweeping parameters for these tasks are as follows: COCOCap [24] (COCO image captioning task) and COCO-35L [232] (COCO captions translated in 35 languages): learning rate (4e-6, 5e-6, 6e-6), epochs (5, 10), dropout (0, 0.02, 0.05). TextCaps [233] (image captioning with reading comprehension): learning rate (4e-6, 6e-6), and training epochs (5, 10). For SciCaps [234] (captions for scientific figures): learning rate (6e-5, 7e-5), dropout (0.1, 0.2), and label smoothing (0.1, 0.2). For VQAv2 [184] (visual question answering): label smoothing (0.0, 0.1), dropout (0.0, 0.1), and

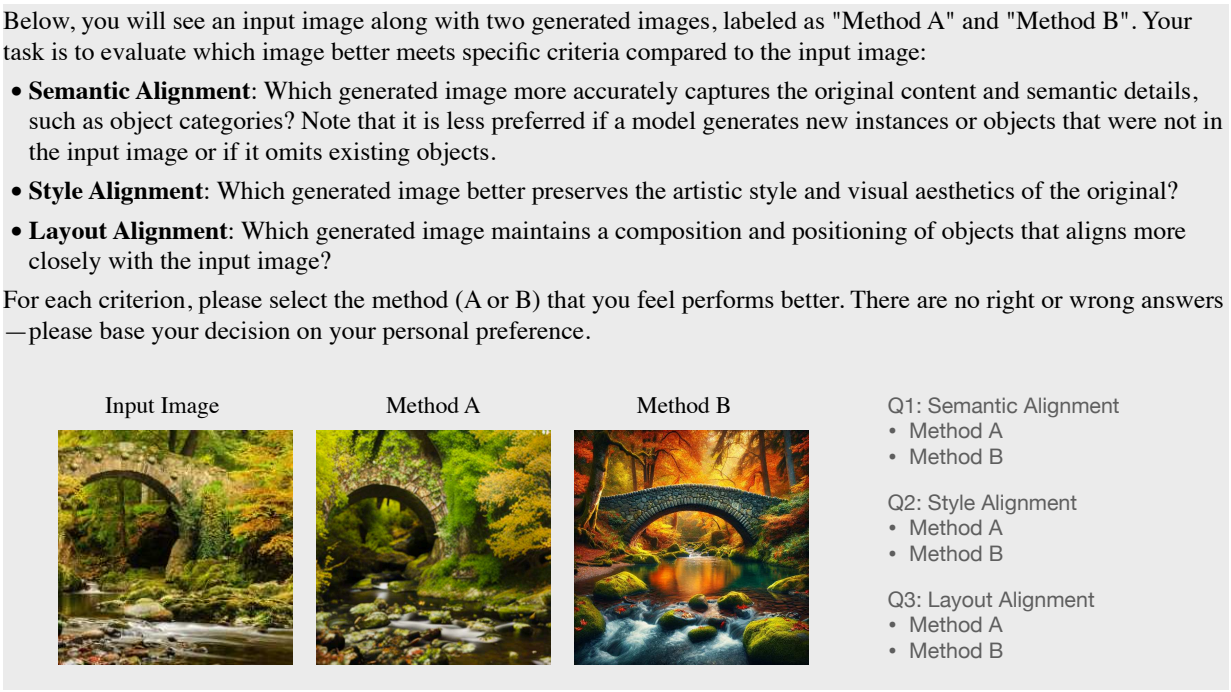


Figure 6.8: The instructions and question format used for human study.

weight decay (0, 1e-6). For TextVQA [235] (visual reasoning based on text in images): learning rate (4e-6, 6e-6). For OKVQA [236] (outside knowledge VQA), ScienceQA [237] (science question answering), and VizWizVQA [238] (VQA from people who are blind): learning rate (8e-6, 1e-5), and dropout (0.0, 0.02). For GQA [239] (VQA on image scene graphs): learning rate (5e-6, 1e-5), and dropout (0.0, 0.02, 0.05). For RefCOCO [185], [240], [241] (referring expression segmentation): label smoothing (0.1, 0.2), epochs (60, 100), and dropout (0, 0.05). For MSRVTT-Caps [183] (open-domain short video captioning): weight decay (0, 1e-6), dropout (0, 0.2), and epochs (20, 40).

Human Study

We conduct human studies to evaluate the quality of generated images using an image-to-image pipeline, focusing on three criteria: Semantic Alignment, Style Alignment, and Layout Alignment. For Semantic Alignment, participants judge which generated image more accurately captures the original content and semantic details, such as object categories. Introducing new instances or omitting existing ones from the input image is considered less desirable. For Style Alignment, participants assess which generated image best retains the artistic style and visual aesthetics of the original. For Layout Alignment, participants evaluate which generated image maintains a composition and positioning of objects that closely matches the input image.

The results of this evaluation are reported in Table 6.2 of the main paper. Detailed instructions

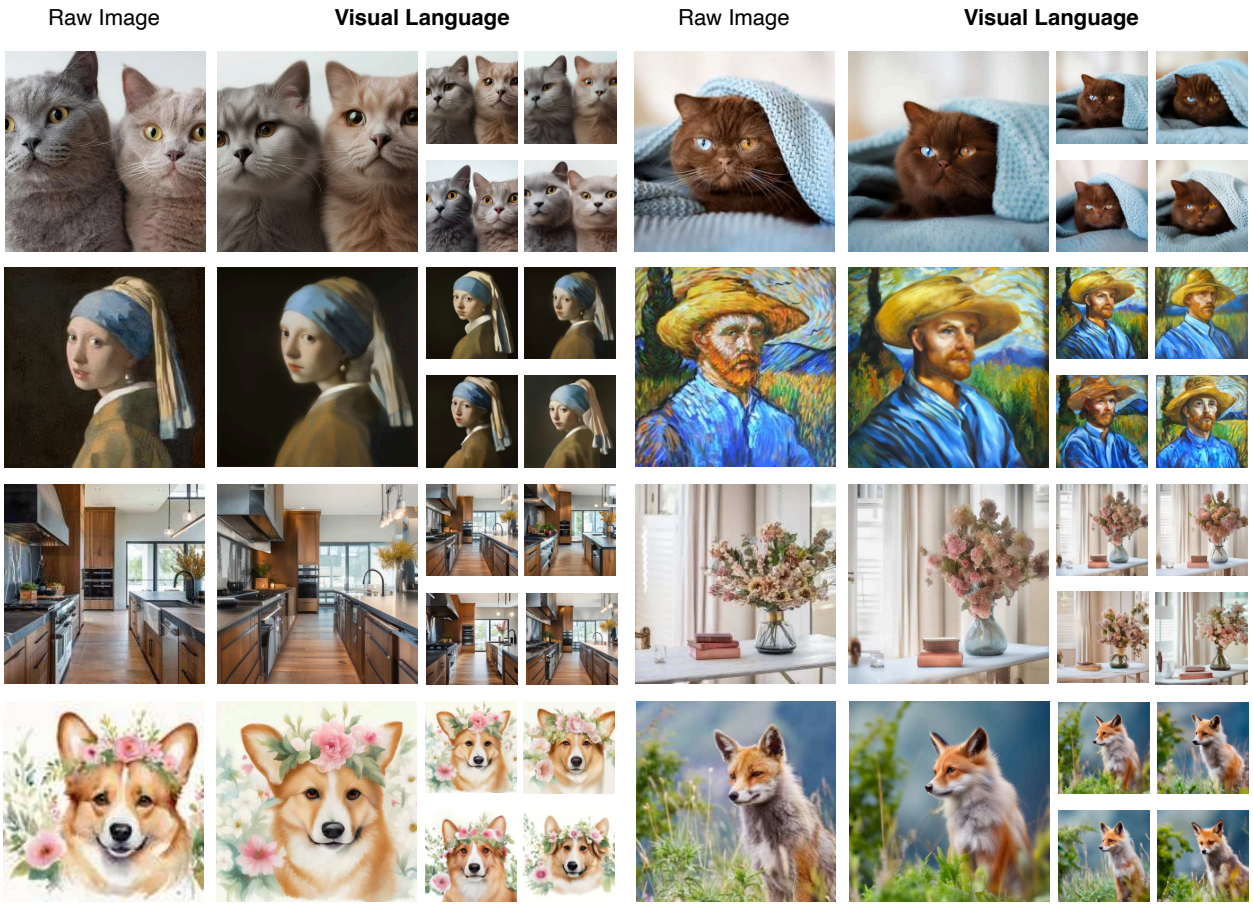


Figure 6.9: More demo results of generating a set of images (generated under different diffusion noises), which are highly semantically and visually similar to each other, by using ViLex tokens as "text" prompts for text-to-image diffusion models.

and the question format for the human study are shown in Fig. 6.8.

Ablation Study

Training Steps. We observed that extending the fine-tuning steps of the vision encoder using our ViLex pipeline leads to improved performance across nearly all evaluated benchmarks, as shown in Table 6.6. Specifically, increasing the training steps from 150k to 300k yields significant gains. Further extending the training to 600k steps provides marginal improvements compared to the 300k-step results. The largest improvements are observed in datasets that demand stronger spatial understanding, such as the referring expression segmentation datasets RefCOCO/+/g.

Number of attention pooling layers. Although increasing the number of attention pooling layers improves image reconstruction performance (as indicated by a lower FID score), it also introduces

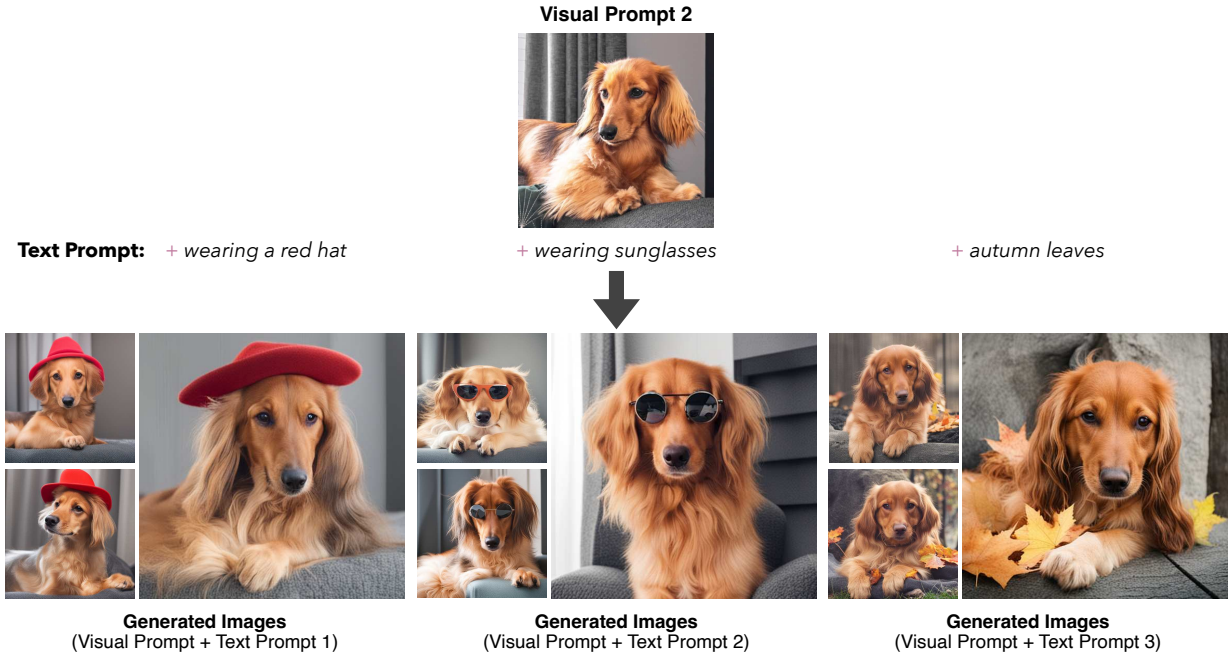


Figure 6.10: More demo results of zero-shot accessorization via prompting a frozen text-to-image generation model with our visual prompts (*i.e.*, ViLex tokens) and text prompts from natural language.

a trade-off with image understanding capabilities. As shown in Table 6.7, we found that using 5 attention pooling layers provides the optimal balance between image generation quality and developing an effective vision encoder for visual scene understanding.

#layers	FID	COCOCaps
2	2.62	140.7
5	2.58	141.5
8	2.52	141.0

Table 6.7: Ablation study on number of attention pooling layers.

Vision encoders. The ViLex approach effectively enhances various vision encoders for downstream visual scene understanding tasks. We initialize the vision encoder of ViLex with either the CoCa [211] pretrained ViT or the SigLIP [180] pretrained ViT-So400M. Similar to our experiments in previous sections, We observed consistent performance improvements for both image understanding tasks, such as COCOCaps [24], and video understanding tasks, such as MSRVTT-Caps [183]. Compared to a roughly 2% improvement for SigLIP in terms of CIDEr score on COCOCaps, the gains for CoCa were even more substantial, reaching approximately 4%. The flexibility to consistently improve different pretrained models demonstrates ViLex's generalizability across various types of vision encoders.

Datasets	CoCa	F.T. w/ ViLex	SigLIP	F.T. w/ ViLex
COCOCaps	131.6	**135.8**	139.7	**141.5**
MSRVTTCaps	56.1	**60.2**	69.4	**71.4**

Table 6.8: Fine-tuning vision encoders with the ViLex approach enhances image understanding performance across various pretrained models, including CoCa [211] and SigLIP [180]. F.T. denotes fine-tuning the vision encoder, such as CoCa, during the ViLex model pretraining stage.

Demo Results

Semantic-level image reconstruction. In this section, we present additional demo results in Fig. 6.9, showcasing a set of images generated with varying diffusion noises and different random seeds. These images demonstrate high semantic and visual consistency, leveraging ViLex tokens as "text" prompts for text-to-image diffusion models. However, as shown in the results, our model occasionally misses small objects in the scene. This limitation primarily stems from using a low-resolution text-to-image diffusion model as the base during the ViLex model's pretraining phase. We hypothesize that this issue could potentially be mitigated by employing a higher-resolution T2I model as the base model.

Prompting a frozen T2I model with both visual and textual prompts. In the main paper, we have demonstrated that ViLex tokens can serve as a novel visual "language" for multimodal image generation. Unlike methods such as DreamBooth [179], [197] and textual inversion [198], which require: (1) learning specialized text tokens for specific instances, (2) gradient-based training for each individual image, and (3) the use of LORA adapters [215] to modify the model architecture, DreamBooth must be fine-tuned separately for each object (or each set of images corresponding to the same object). In contrast, ViLex enables several DreamBooth tasks like image re-contextualization, artistic rendition and accessorization, as illustrated in Fig. 6.10, Fig. 6.5 and Fig. 6.6, by simply prompting a frozen T2I model with a combination of our visual prompts (*i.e.*, ViLex tokens) and natural language text prompts. This approach does not require changes to the architecture of a pretrained text-to-image generation model or any fine-tuning of the T2I model itself. All tasks are performed in a zero-shot and unsupervised manner.

6.5 Conclusions

We introduce ViLex, a visual lexicon that maps images directly into the text space, while effectively preserving complex visual details that are difficult to express in natural language. Our representation can be seamlessly integrated into text prompts from natural language for both multimodal image generation and downstream vision-language tasks. ViLex can also improve both image understanding and reconstruction capabilities of pretrained vision encoders by fine-tuning them using ViLex's training approach.

Part III

Analysis-for-Synthesis: Understanding Models for Visual Scene Generation

"Without craftsmanship, inspiration is a mere reed shaken in the wind."
— Johannes Brahms

Chapter 7

InstanceDiffusion: Instance-level Control for Image Generation

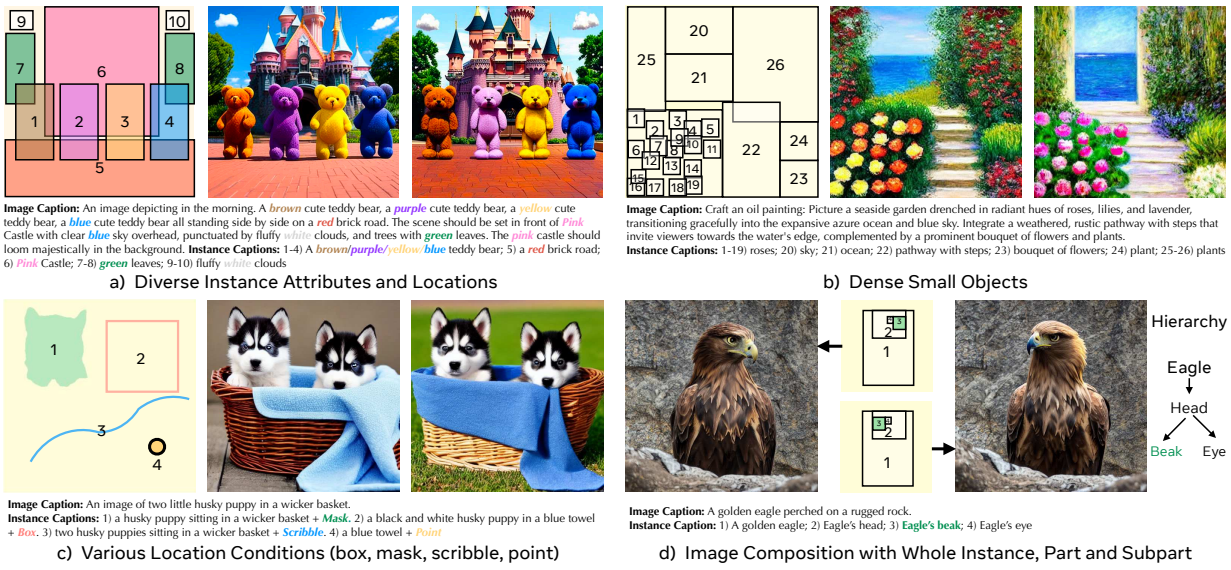


Figure 7.1: InstanceDiffusion's generations using instance-level text prompts and location conditions for image generation. Our model can respect: a) a variety of instances with diverse attributes (8 colors) and boxes, b) densely-packed instances (>25 objects), c) mixed location conditions (such as boxes, masks, scribbles, and points), and d) compositions with granularity spanning from entire instances to parts and subparts. The instance inputs and their global text prompts are displayed, with the location conditions displayed on the left image. Numbers in the box/mask/scribble/point refer to the instance id.

Text-to-image diffusion models produce high quality images but do not offer control over individual instances in the image. We introduce InstanceDiffusion that adds precise instance-level control to text-to-image diffusion models. InstanceDiffusion supports free-form language conditions per instance and allows flexible ways to specify instance locations such as simple single points,

scribble, bounding boxes or intricate instance segmentation masks, and combinations thereof. We propose three major changes to text-to-image models that enable precise instance-level control. Our UniFusion block enables instance-level conditions for text-to-image models, the ScaleU block improves image fidelity, and our Multi-instance Sampler improves generations for multiple instances. InstanceDiffusion significantly surpasses specialized state-of-the-art models for each location condition. Notably, on the COCO dataset, we outperform previous state-of-the-art by 20.4% AP^{box}_{50} for box inputs, and 25.4% IoU for mask inputs.

7.1 Introduction

Image generation models [165], [166], [172], [242]–[249] trained on web-scale data have made tremendous progress in the recent years. Notably, text conditioned diffusion models now produce high quality images that contain the free form concepts specified in the text [165], [166], [172], [248], [250], [251]. While text-based control is useful, it does not always allow for precise and intuitive control over the output image. Thus, many different forms of conditioning, *e.g.*, edges, normal maps, semantic layouts have been proposed for better control [252]–[261]. These richer controls enable a broader range of applications for the generative models in design, data generation [262], [263] *etc*. In this work, we focus on precise control over the *instances* in terms of their location and attributes in the output image.

We propose and study instance-conditioned image generation whereby a user can specify *every* instance in terms of its location and an instance-level text prompt to generate an image. The location can be specified using either a bounding box, an instance mask, a single point or a scribble. Practically, this allows for a flexible input where some instance locations maybe specified more precisely using masks, and others less precisely using points. The per instance text prompts allow for fine-grained control over the instance's attributes such as color, texture, *etc*. Our proposed instance-conditioned generation is a generalization of settings studied in prior work [253], [254], [264] that consider only one location format and do not use per instance captions.

Our model presents several design choices that enable more precise yet flexible control for instances in the output image. Since locations can be specified in a variety of formats, we present a unified way to parameterize and fuse their information during the generation process. Our unified modeling is simpler than prior work that uses separate architectures and strategies to model different location formats. Moreover, the unified modeling of location formats allows the model to exploit the shared underlying structure of instance locations which improves performance.

Through comprehensive evaluations, our method InstanceDiffusion outperforms state-of-the-art models specialized for particular instance conditions. We achieve a 20.4% increase in AP^{box}_{50} over GLIGEN [253] when evaluating with bounding box inputs on COCO [24] `val`. For mask-based inputs, we obtain a 25.4% boost in IoU compared to DenseDiffusion [265] and a 17.0% gain in AP^{mask}_{50} over ControlNet [254]. As prior methods do not study point or scribble inputs for image generation, we introduce evaluation metrics for these settings. InstanceDiffusion also demonstrates superior ability to adhere to attributes specified by instance-level text prompts. We

obtain a substantial 25.2 point gain in instance color accuracy and a 9.2 point improvement in texture accuracy compared to GLIGEN.

Contributions. (1) In this paper, we propose and study instance-conditioned image generation that allows flexible location and attribute specification for multiple instances. (2) We propose three key modeling choices that improve results – (i) *UniFusion* (Sec. 7.3), which projects various forms of instance-level conditions into the same feature space, and injects the instance-level layout and descriptions into the visual tokens; (ii) *ScaleU* (Sec. 7.3), which re-calibrates the main features and the low-frequency components within the skip connection features of UNet, enhancing the model's ability to precisely adhere to the specified layout conditions; (iii) *Multi-instance Sampler* (Sec. 7.3), which reduces information leakage and confusion between the conditions on multiple instances (text+layout). (3) A dataset with instance-level captions generated using pretrained models (Sec. 7.3) and a new set of evaluation benchmarks and metrics for measuring the performance of location grounded image generation (Sec. 7.4). (4) Our unified modeling of different location formats significantly improves results over prior work (Sec. 6.4). We also show that our findings can be applied to previous approaches and boost their performance.

7.2 Related Work

Image Diffusion Models [172], [217], [251] learn the process of text-to-image generation through iterative denoising steps initiated from an initial random noise map. Latent diffusion models (LDMs) [266], [267] perform the diffusion process in the latent space of a Variational AutoEncoder [267], [268], for computational efficiency, and encode the textual inputs as feature vectors from pretrained language models [17]. DALL-E 2 [166] synthesizes images using the image space of CLIP [17]. In contrast, Imagen [165] diffuses pixels directly, without the need for latent images. In addition, it demonstrates that generic large language models, such as T5 [176], trained solely on text corpora, are surprisingly effective at encoding text for image generation.

Image Generation with Spatial Controls is a form of conditional image synthesis task [247], [253]–[256], [269]–[277], which introduces spatial conditioning controls to guide the image generation process. *Make-a-Scene*, *SpaText* [264], *GLIGEN* [253], and *ControlNet* [254] add finer grained spatial control, such as semantic segmentation masks, to large pretrained diffusion models by allowing users to include additional images that explicitly define their desired image composition. *GLIGEN* [253] can also support controlled image generation using discrete conditions such as bounding boxes. MultiDiffusion [278], DenseDiffusion [265], Attend-and-Excite [279], ReCo [280], StructureDiffusion [281], Layout-Guidance [282], and BoxDiff [283] add location controls to diffusion models without fine-tuning the pretrained text-to-image models. **Discussions.** ControlNet and GLIGEN require training separate models for each type of controllable input, which increases the overall complexity of the system and not effectively capture interactions across various controllable inputs. Moreover, while ControlNet focuses solely on spatial conditions and GLIGEN employs *object category* as the text prompt, the lack of training the models with detailed instance-level prompts not only limits user control but also hinders the model from effectively leveraging instance descriptions.

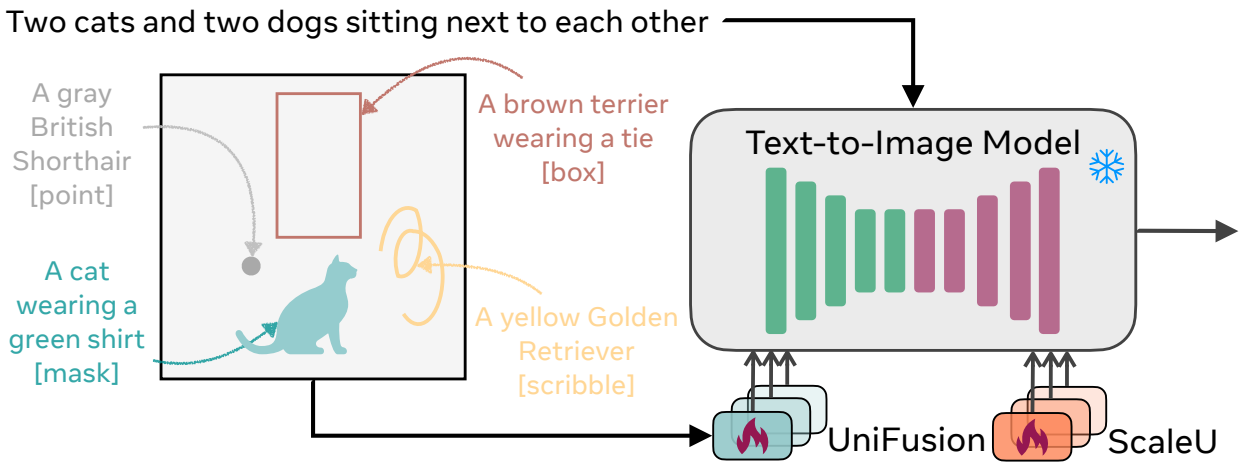


Figure 7.2: InstanceDiffusion enhances text-to-image models by providing additional instance-level control. In additon to a global text prompt, InstanceDiffusion allows for paired instance-level prompts and their locations to be specified when generating images. InstanceDiffusion is versatile, supporting a range of location forms, from the simplest points, boxes, and scribbles to more complex masks, and their flexible combinations.

7.3 Instance Diffusion

We study adding precise, versatile instance-level control for text-based image generation.
Problem definition. We aim to improve instance-level control in image generation by focusing on two conditioning inputs for each instance, namely, its location and a text caption describing the instance. More formally, we want to learn an image generation model $f(\mathbf{c}_g, \{(\mathbf{c}_1, \mathbf{l}_1), \ldots, (\mathbf{c}_n, \mathbf{l}_n)\})$ that is conditioned on a global text caption $\mathbf{c}_g$ and the per-instance conditions $(\mathbf{c}_i, \mathbf{l}_i)$ containing caption $\mathbf{c}_i$ and location $\mathbf{l}_i$ for n instances. This problem is similar to [264] and is a generalization of the 'open-set grounded text-to-image' [253] problem which does not consider per-instance captions. Our generalization allows for a generic and flexible way to control the scene-layout in terms of locations and attributes of the instances, as well as scene-level control via the global caption.

Approach overview

We introduce InstanceDiffusion (Fig. 7.2) for instance-conditioned image generation using a diffusion model. We consider a variety of different and flexible ways to specify an object's location, *e.g.*, a single point, a scribble, a bounding box, and an instance mask. Since obtaining large-scale paired (text, image) data is much easier compared to (instance, image) data, we use a pretrained text-to-image UNet model that is kept frozen. We add our proposed learnable **UniFusion** blocks to handle the additional per-instance conditioning. UniFusion fuses the instance conditioning with the backbone and modulate its features to enable instance-conditioned image generation. Additionally, we propose **ScaleU** blocks that improve the UNet's ability to respect instance-conditioning by

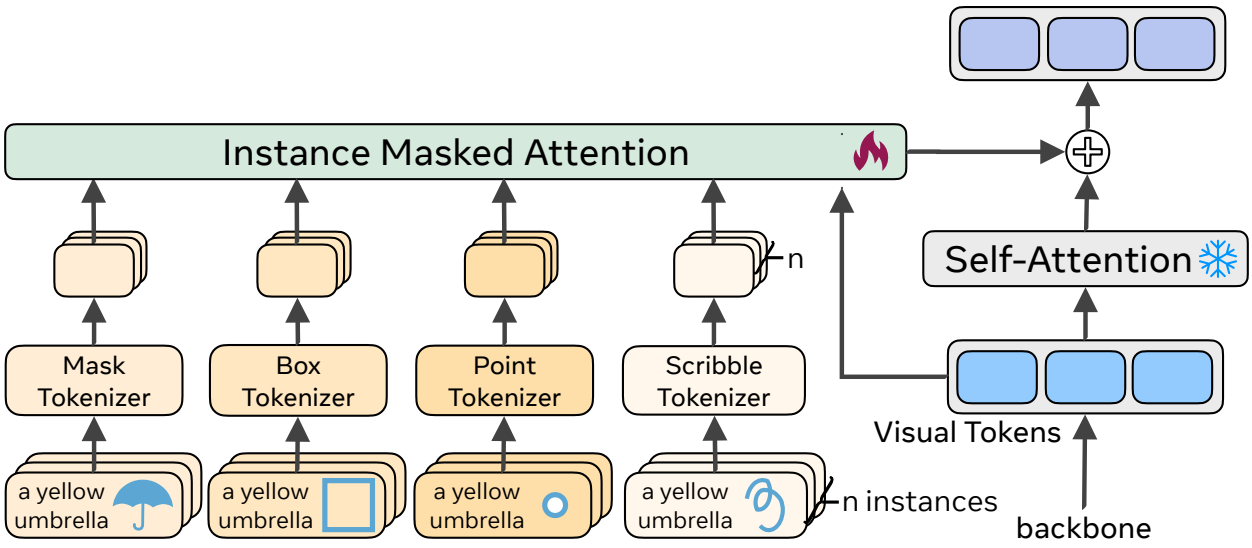


Figure 7.3: UniFusion projects various forms of instance-level conditions into the same feature space, seamlessly incorporating instance-level locations and text-prompts into the visual tokens from the diffusion backbone.

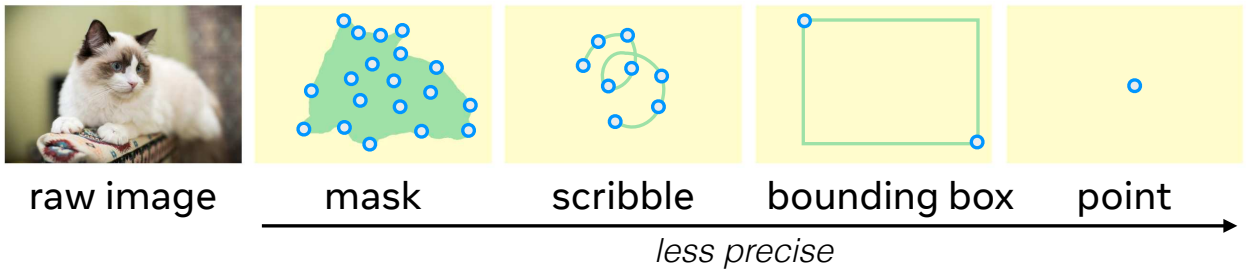


Figure 7.4: We represent different location condition formats as sets of points, with each format having varying quantities of points. Masks are represented as sparsely sampled points within the mask and uniformly sampled points from boundary polygons, bounding boxes by the top-right and bottom-right corners, and scribble are converted into uniformly sampled points.

rescaling the skip-connection and backbone feature maps produced in the UNet. At inference, we propose **Multi-instance Sampler** which reduces information leakage across multiple instances.

Since obtaining a large paired (instance, image) dataset is difficult, we automatically generate a dataset with instance-level location and text captions using state-of-the-art recognition systems. Finally, we propose a new and comprehensive benchmark to evaluate the model's performance for instance-conditioned generation.

UniFusion block

The UniFusion block, illustrated in Fig. 7.3, tokenizes the per-instance conditions $(\mathbf{c}_i, \mathbf{l}_i)$ and fuses them with the features, *i.e.*, visual tokens from the frozen text-to-image model. Similar to [253], [284], the UniFusion block is added between the self-attention and cross-attention layers of the backbone. The per-instance location $\mathbf{l}_i$ can be specified in *one or more* location formats such as masks, boxes, *etc*. We now describe the key operations in the UniFusion block.

Location parameterization. As shown in Fig. 7.4, we convert the four location formats - masks, boxes, scribbles, single point - into 2D points (denoted as $\mathbf{p}_i = \{(x_k, y_k)\}_{k=1}^{n}$ for instance i), with each 'format' having varying quantities of points n. A scribble is converted into a set of uniformly sampled points along the curve. We parameterize bounding boxes by their top-left and bottom-right corners and convert masks into a set of points sampled from within the mask and from boundary polygons. Prior work resizes semantic masks [253], [254] or boxes [264] into the diffusion latent space of size 64×64. We found that this design hurts performance for overlapping instances and small objects, and use our point-based parameterization instead.

Instance Tokenizer. We convert the 2D point coordinates $\mathbf{p}_i$ for each location using a Fourier mapping [285] $\gamma(\cdot)$ and encode the text prompt $\mathbf{c}_i$ using a CLIP text encoder $\tau_\theta(\cdot)$. Finally, we concatenate the location and text embeddings and feed them to an MLP to obtain a single token embedding $\mathbf{g}_i$ for the instance i: $\mathbf{g}_i = \text{MLP}([\tau_\theta(\mathbf{c}_i), \gamma(\mathbf{p}_i)])$.

We use a different MLP for each location format. Moreover, the per-instance location $\mathbf{l}_i$ can be specified in one or more location formats. Thus, for each instance i, we obtain $\mathbf{g}_i^{\text{mask}}$, $\mathbf{g}_i^{\text{scribble}}$, $\mathbf{g}_i^{\text{box}}$, and $\mathbf{g}_i^{\text{point}}$. If an instance location is specified only using one format, *e.g.*, a single point, we use a learnable null token $\mathbf{e}_i$ for the other location formats:

$$\mathbf{g}_i = \text{MLP}([\tau_\theta(\mathbf{c}_i), s \cdot \gamma(\mathbf{p}_i) + (1 - s) \cdot \mathbf{e}_i]) \tag{7.1}$$

where s is a binary value indicating the presence of a specific location format.

Instance-Masked Attention and Fusion Mechanism. We denote the instance condition tokens, $\mathbf{g}$, per location format for all n instances by $\mathbf{G}$, and the m visual tokens, $\mathbf{v}$, from the backbone as $\mathbf{V}$. We apply masked self-attention (SA) to the instance condition tokens and the backbone features

$$\widetilde{\mathbf{V}} = \text{SA}_{\text{mask}}([\mathbf{V}, \mathbf{G}^{\text{mask}}, \mathbf{G}^{\text{scribble}}, \mathbf{G}^{\text{box}}, \mathbf{G}^{\text{point}}]) \tag{7.2}$$

We consider two design choices, ablated in Table 7.5, for the location inputs in Eq. (7.2): 1) 'Format aware' (default) described above models each location format independently via concatenation. 2) 'Joint format' jointly models all location formats by concatenating embeddings from each format and converting them into a single embedding (via an MLP) to use in the masked self-attention.

We observed that vanilla self-attention, without masking, led to information leakage across instances, *e.g.*, color of one instance bleeding into another. Thus, we construct a mask $\mathbf{M}$ that prevents such leakage across instances:

$$\begin{aligned} &\text{mask for } \mathbf{v}_k \cdot \mathbf{v}_j^T : \mathbf{M}_{k,j} = -\inf \text{ if } I_{\mathbf{v}_k} \neq I_{\mathbf{v}_j} \\ &\text{mask for } \mathbf{v}_k \cdot \mathbf{g}_i^T : \mathbf{M}_{k,m+i} = -\inf \text{ if } I_{\mathbf{v}_k} \neq i \end{aligned} \tag{7.3}$$

where $I_{\mathbf{v}_k} = i$ if the visual token $\mathbf{v}_k$ falls within the region of the instance i defined by either a bounding box or an instance segmentation mask.

Finally, the output of the masked self-attention is added back to the backbone via gated addition

$$\mathbf{V} = \mathbf{V} + \tanh(\omega)\widetilde{\mathbf{V}}[:m] \tag{7.4}$$

where ω is a learnable parameter, initialized to 0, that controls the conditioning contribution of UniFusion.

ScaleU block

In the UNet model, each block merges the main feature map $\mathbf{F}_b$ with the lateral skip-connection features $\mathbf{F}_s$, passing the concatenated feature to the subsequent UNet block. FreeU [286] finds that the main backbone of UNet is critical for denoising, whereas its skip connections primarily contribute high-frequency features to the decoder. Concatenating these two features directly leads to the network neglecting the semantic content of the main features [286]. Therefore, FreeU suggests reducing the low-frequency components of the skip features and enhancing the main features using *channel-independent* and *empirically-tuned* values.

Our findings, however, demonstrate that for instance-conditioned image generation, a notable improvement can be achieved by using *channel-wise* and *learnable* vectors to dynamically recalibrate $\mathbf{F}_b$ and $\mathbf{F}_s$. More specifically, we introduce ScaleU, that has two *learnable*, *channel-wise* scaling vectors: $\setminus_b$, $\setminus_s$ for the main and skip-connected features, respectively. The main features $\mathbf{F}_b$ are scaled by a simple channel-wise multiplication: $\boldsymbol{F}'_b = \mathbf{F}_b \otimes (\tanh(\setminus_b) + 1)$. For the skip-connection features, we select the low-frequency (less than r_{thresh}) components using a frequency mask $\boldsymbol{\alpha}$ and scale them in the Fourier domain: $\mathbf{F}'_s = \text{IFFT}(\text{FFT}(\mathbf{F}_s) \odot \boldsymbol{\alpha})$. Here $\text{FFT}(\cdot)$ and $\text{IFFT}(\cdot)$ denote the Fast-Fourier and Inverse-Fast-Fourier transforms, $\odot$ is element-wise multiplication, and $\boldsymbol{\alpha}(r) = \tanh(\setminus_s) + 1$ if $r < r_{\text{thresh}}$ otherwise $= 1$, where r denotes the radius, and r_{thresh} refers to the threshold frequency. Both $\setminus_b$ and $\setminus_s$ are initially set to zero vectors.

Lightweight in parameters. The ScaleU module is incorporated into each of UNet's decoder blocks. It leads to a negligible ($< 0.01\%$) overall increase in the number of parameters and brings noticeable performance gains.

Multi-instance Sampler

To further minimize the information leakage across multiple instance conditionings, we optionally use Multi-instance Sampler strategy during the model inference which improves the quality and fidelity of the generated image.

Specifically, Multi-instance Sampler (*cf.* Fig. 7.5) involves: 1) For each of the n instances, run a separate denoising operation for M steps (less than 10% of the overall steps) to get the instance latents L_I. Note that, since our model is trained to generate an object within the location token specified for that object, we don't need to explicitly require the model to update the latent representation within the location. 2) Integrate the denoised instance latents $\{L_I^1, \cdots, L_I^n\}$ obtained from step (1) for each of the n objects with the global latent L_G, which is derived from all instance

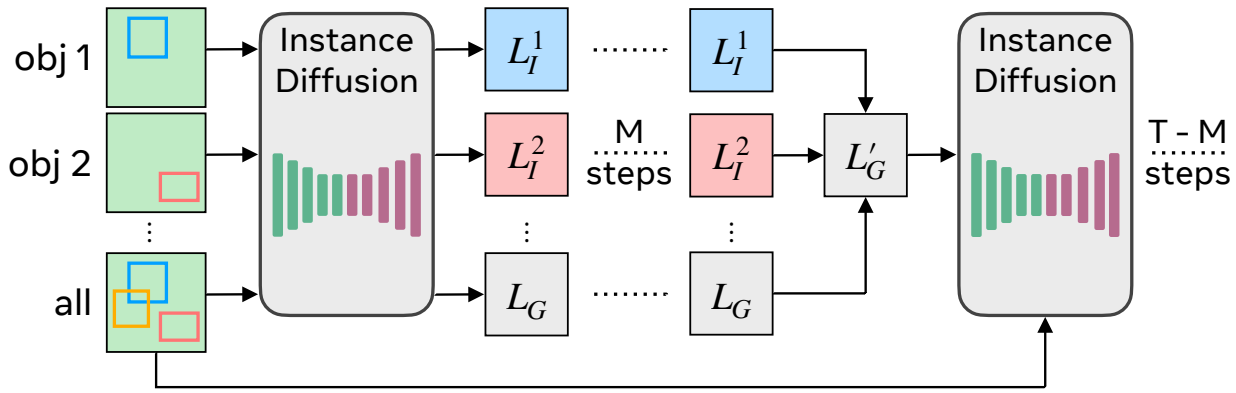


Figure 7.5: Model inference with **Multi-instance Sampler** to minimize information leakage across multiple instance conditionings.

tokens and text prompts, by averaging these latents together. 3) Proceed to denoise the aggregated latent from step (2), utilizing all instance tokens and text prompts.

Data with Instance Captions

Obtaining a large-scale dataset that contains instance conditions is challenging. Standard object detection datasets [24] only contain a sparse category label, rather than a detailed caption, per object location. To capture more detailed information about instances and even instance parts, *e.g.*, attributes, we construct a dataset by using multiple models: ***1) Image-level label generation***: We employ RAM [287], a robust open-vocabulary image tagging model, to generate a list of common image-level tags. ***2) Bounding-box and mask generation***: We then use Grounded-SAM [132], [288] to produce bounding boxes and masks corresponding to these tags. These tags can at the instance-level, *e.g.*, a parrot, or at the part-level, *e.g.*, a bird's beak. ***3) Instance-level text prompt generation***: To generate instance-level text prompts that include descriptions of the instances, we crop the instances using their corresponding bounding boxes and create captions for these cropped instances using a pretrained Vision-Language Model (VLM) BLIP-V2 [289].

Implementation Details

We describe salient implementation details and provide the full details in the supplement.
Model training. We follow the same settings as GLIGEN [253] and initialize our model with a pretrained text-to-image model whose layers are frozen. We train the model with a batch size of 512 for 100K steps using the Adam optimizer [290] with a learning rate that is warmed up to 0.0001 after 5000 steps. More details are in appendix materials.
Training data. We automatically generate instance-level masks, boxes and captions following Sec. 7.3. We obtain scribble by randomly sampling points within the masks. For single-points, we randomly

Location format input →	Boxes				Instance Masks					Points		Scribble	
Method	AP^{box}	AP^{box}_{50}	AR^{box}	FID (↓)	IoU	AP^{mask}	AP^{mask}_{50}	AR^{mask}	FID (↓)	PiM	FID (↓)	PiM	FID (↓)
Upper bound (real images)	50.2	66.7	61.0	-	-	40.8	63.5	58.0	-	-	-	-	-
GLIGEN [253]	19.6	35.0	30.7	27.0	-	-	-	-	-	-	-	-	-
GLIGEN [253]*	19.3	34.6	31.1	-	-	-	-	-	-	-	-	30.2†	32.4†
ControlNet [254]‡	-	-	-	-	-	6.5	13.8	12.9	-	-	-	-	-
DenseDiffusion [265]	-	-	-	-	35.0 / 48.6	-	-	-	-	-	-	-	-
SpaText [264]‡	-	-	-	-	-	5.3	12.1	10.7	-	-	-	-	-
InstanceDiffusion	**39.0**	**55.4**	**52.4**	**23.1**	**61.6 / 71.4**	**23.5**	**46.1**	**34.9**	**25.2**	**79.7**	**27.1**	**67.3**	**26.4**
vs. prev. SoTA	**+19.4**	**+20.4**	**+21.3**	**-3.9**	**+25.4 / +22.8**	**+17.0**	**+32.3**	**+22.0**	-	-	-	**+37.1**	**-6.0**

Table 7.1: Evaluating different location formats as input when generating images. We measure the YOLO recognition performance (AP, AR) for the generated image wrt the location condition provided as inputs, and FID on the COCO `val` set. Most prior methods only support a handful of the location conditions. We observe that InstanceDiffusion, while using the same model parameters, supports various location inputs. In each setting, InstanceDiffusion substantially outperforms prior work on all metrics. *: reproduced using official models and evaluated with YOLOv8. †: GLIGEN's scribble-based results are derived by using the top-right and bottom-left corners as the bounding box for the region encompassed by the scribble. We measure the IoU using [265]'s official evaluation codes (left), and YOLOv8-Seg (right). ‡: ControlNet [254] (and SpaText [264]) only supports *semantic* segmentation mask inputs, and do not differentiate between instances of the same class. We assess ControlNet's AP^{mask} using its official mask conditioned Image2Image generation pipeline.

select a point within a circular region of radius $0.1 \cdot r$, centered at the bounding box's center, where r is the length of the shortest side of the box.

7.4 Experiments

Experimental setup

Training data. Prior work, notably GLIGEN [253], relies on automatic annotations that use open-vocabulary detection models. These do not yield per-instance captions and different location formats such as scribble *etc*(Note: 'mask' conditioning in prior work [253], [264] is per-category and not per-instance). Thus, to support the richer conditioning proposed in our work, we rely on recognition models as described in Sec. 7.3 to generate instance-level annotations include different location formats (masks, boxes, scribbles, single-points) and per-instance captions. To ensure fair comparison to prior work [253], we use approximately the same number of images (5M) from an internal licensed dataset of natural images and paired global text.

Test data. We use standard benchmarks with bounding box and instance masks: 1) COCO [24] `val` with 80 classes; 2) large vocabulary instance segmentation dataset LVIS [61] `val` with over 1200 classes; 3) 250 selected samples ($\sim$2 objects per image) from COCO `val` as in [265]. We do not use the real images from the dataset, and only use the text and location conditions. Notably, we also do not use any information from the `train` splits of the data which makes our evaluations zero-shot.

Figure 7.6: Qualitative comparison of InstanceDiffusion *vs.* GLIGEN conditioned on multiple instance boxes and prompts. Prior work (bottom row) fails to accurately reflect specific instance attributes, *e.g.*, colors for the flower and puppies on the left, and not depicting a waterfall on the right. The generations also do not capture the correct instances, and are prone to information leakage across the instance prompts, *e.g.*, generating two similar instances on the right. InstanceDiffusion effectively mitigates these issues.

Evaluation metrics for alignment to instance locations. We measure how well the objects in the generated image adhere to different location formats in the input.

Bounding box. We follow prior work [253], [265], [291], [292] and use the YOLO score. Specifically, we use a pretrained YOLOv8m-Det [292] detection model. We compare the model's detected bounding boxes on the generated image with the bounding boxes specified in the input using COCO's official evaluation metrics (AP and AR). We report AP_l^{box}, AP_m^{box}, and AP_s^{box}, which evaluate the model's performance based on different object sizes.

Instance mask. We compare YOLOv8m-Seg [292]'s detected instance masks in the generated image to the masks specified in the input using the COCO AP and AR metrics. To compare with [265], we report the IOU score for the mask.

Scribble. Since prior work has not reported on alignment performance using scribble, we introduced a new evaluation metric using YOLOv8m-Seg. We report "Points in Mask" (**PiM**), which measures how many of randomly sampled points in the input scribble lie within the detected mask.

Single-point. Similar to scribble, the instance-level accuracy **PiM** is 1 if the input point is within the detected mask, and 0 otherwise. We then calculate the averaged **PiM** score.

Evaluation metrics for alignment to instance prompts. We measure the alignment of the objects in the generated image to the corresponding text and location conditions from COCO `val` set.

Compositional attribute binding. We measure if the generated instances adhere to the attribute (color and texture) specified in the instance prompts. We use YOLOv8-Det to detect the bounding boxes. We feed the cropped box to the CLIP model to predict its attribute (colors and textures), and measure the accuracy of the prediction with respect to the attribute specified in the instance prompt. We use 8 common colors, *i.e.*, "black", "white", "red", "green", "yellow", "blue", "pink", "purple", and 8 common textures, *i.e.*, "rubber", "fluffy", "metallic", "wooden", "plastic", "fabric", "leather" and "glass".

Instance text-to-image alignment: We report the CLIP-Score on cropped object images (Local

Methods	Color		Texture		Human
	$\text{Acc}^{\text{color}}$	$\text{CLIP}^{\text{local}}$	$\text{Acc}^{\text{texture}}$	$\text{CLIP}^{\text{local}}$	Eval
GLIGEN	19.2	0.206	16.6	0.206	19.7
InstDiff	**54.4**	**0.250**	**25.8**	**0.221**	80.3
Δ	**+35.2**	**+0.044**	**+9.2**	**+0.015**	

Table 7.2: Attribute binding. We measure whether the attributes of the generated instances match the attributes specified in the instance captions. We observe that InstanceDiffusion outperforms prior work on both types of attributes. Human evaluators prefer our generations significantly more than the prior work.

Methods	AP	AP_{50}	AP_s	AP_m	AP_l	AP_r	AP_c	AP_f
Upper bound	44.6	57.7	33.2	55.0	66.1	31.4	44.5	50.5
GLIGEN [253]†	9.9	9.5	1.6	10.5	31.1	7.4	10.0	10.9
InstanceDiffusion	17.9	25.5	5.5	24.2	45.0	12.7	18.7	19.3
vs. prev. SoTA	**+8.0**	**+16.0**	**+3.9**	**13.7**	**+13.9**	**+5.3**	**+8.7**	**+8.4**

Table 7.3: Box inputs on LVIS `val`. We evaluate using a pretrained detector (ViTDet-L [104]) and obtain the upper bound by evaluating the detector on real images resized to 512×512. InstanceDiffusion significantly outperforms prior work across all metrics including object sizes, and class frequencies. †: reproduced results.

CLIP-score [17], [264]), which measures the distance between the instance text prompt's features and the cropped object images.
Global text-to-image alignment: CLIP-Score [17], [171] between the input text prompt and the generated image.
Human evaluation: We evaluate both the fidelity wrt instance-level conditions (locations and text prompts) and the overall aesthetic of the generated images. We prompt users to select results that more closely adhere to the provided layout conditions and the accompanying instance captions. This evaluation is conducted on 250 samples, each accompanied by instance-level captions and bounding boxes.

Comparison with prior work

Single location format at inference. We assess the efficacy of multiple methods in generating images under diverse location formats and report results in Table 7.1. Since our evaluation uses recognition model (YOLO), we establish an upper bound by measuring the recognition performance on the real dataset images corresponding to the text and location conditions. Overall, our results show that InstanceDiffusion outperforms all prior work across various location conditions when measured across all evaluation metrics for object location and image quality. Next, we discuss the results for each location format. Box input: InstanceDiffusion achieves the highest AP^{box} of 39.0 and AR^{box} of 55.4, outperforming the previous state-of-the-art by a significant margin, +19.4 and +20.4 for AP^{box} and AR^{box}, respectively. The reduction in FID score for InstanceDiffusion demonstrates its ability to produce high-quality images while adhering to the prescribed

point	box	mask	PiM	AP^{box}	AP^{box}_{50}	AP^{mask}	AP^{mask}_{50}
✓	✗	✗	79.7	-	-	-	-
✓	✓	✗	86.3	39.0	55.4	-	-
✓	✓	✓	**86.6**	**41.6**	**56.2**	**23.5**	**46.1**

Table 7.4: Multiple location formats at inference improves performance and helps the model to better respect location conditions.

location conditions. Instance mask input imposes stricter constraints on the instance location than box input and is more challenging than the semantic masks studied in prior work [253], [254] that do not distinguish individual instances. Even in this challenging setting, InstanceDiffusion outperforms prior SOTA [265] significantly. Points and Scribble: Given the lack of prior studies that present quantifiable results for these location inputs, we introduce these novel evaluation metrics and benchmarks, establishing a new baseline for future research endeavors. Note that the term 'scribble' in ControlNet [254] refers to object boundary sketches rather than scribbles used in our work which follows [293]–[295].

Attribute binding. In Table 7.2, we measure whether the attributes (color and texture) of the generated instances match the attributes specified in the instance captions. We observe that attribute binding is challenging for the prior SOTA method, GLIGEN while InstanceDiffusion significantly improves on both color and texture binding. Adhering to texture seems to be more challenging than colors, *e.g.*, `wooden dog` *vs.* `red dog`, as reflected by the lower accuracies for all methods on this task. We compare the generations produced by both models using human evaluators and find that humans strongly prefer our generations over prior work (80.3% preference) confirming their high generation quality and controllability.

Challenging box inputs. In Table 7.3, we evaluate zero-shot performance on the challenging LVIS [61] dataset which has $15\times$ more classes than COCO, and many more instances per sample ($\sim$12 objects per images). Even on this challenging dataset, InstanceDiffusion outperforms prior work across all metrics. The gain is particularly strong on medium to large sized objects.

Multiple location formats at inference are analyzed in Table 7.4. We observe that using all formats together provides the best performance and more precise control on the instance location. This confirms the benefit of our design choice to model all location formats.

Qualitative results. Fig. 7.6 provides qualitative comparisons between InstanceDiffusion and the previous SOTA method, GLIGEN [253], when given multiple instance boxes and associated text prompts. We see that GLIGEN often misinterprets specific instance attributes; *e.g.*, it incorrectly renders the colors of flowers and puppies on the left, and fails to produce a waterfall in the right images. GLIGEN also shows 'information leakage' across instance prompts (generating duplicate birds for the two images on the right). In Fig. 7.7, we show more qualitative results using different location conditions for InstanceDiffusion.

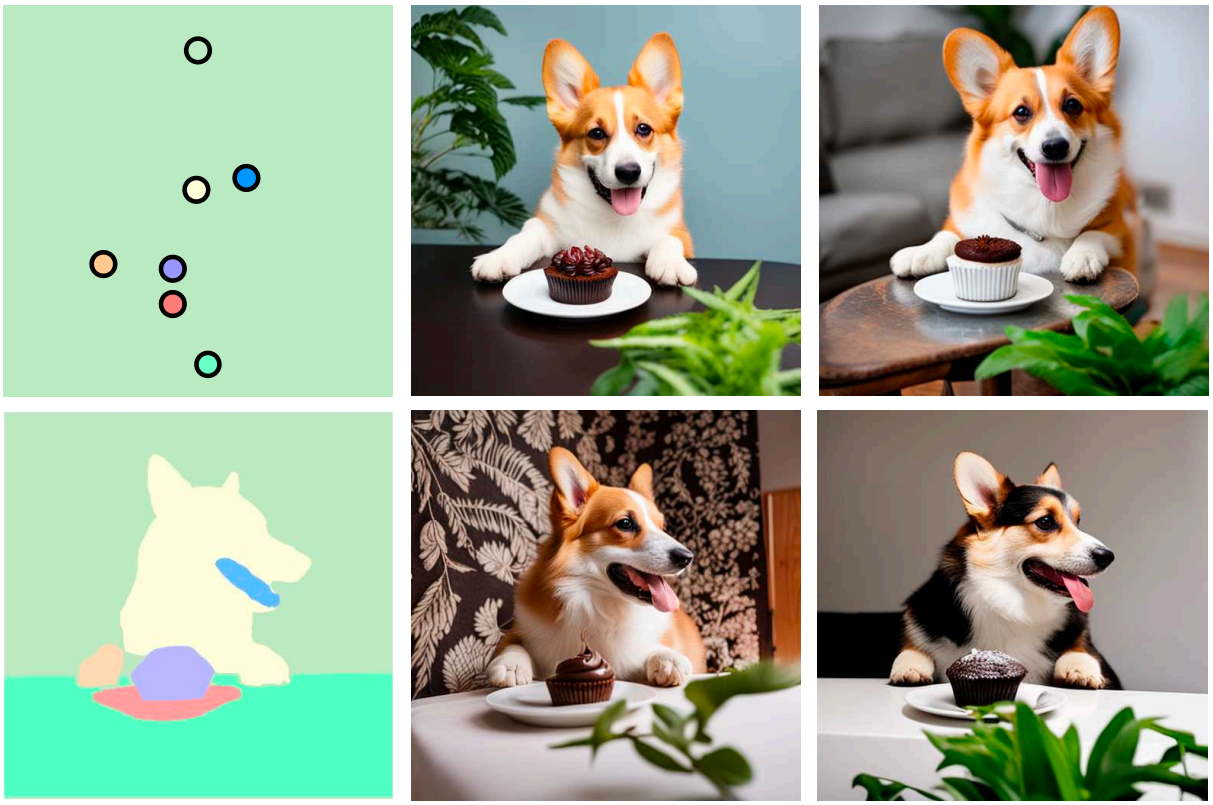


Image Caption: Cute Corgi at table in a living room with plants. A chocolate cake is on the table.
Instance Captions: 1) a Corgi sitting in front of a cupcake 2) Corgi's mouth and tongue 3) a white plate 4) a chocolate cupcake on a white plate 5) a living room with plants and a painting on the wall 6) a white paw 7) a table

Figure 7.7: InstanceDiffusion image generation using various location conditions: points (row 1) and masks (row 2).

Ablation study

We ablate the components in InstanceDiffusion and use the COCO `val` set and provide mask, box and point location formats per-instance as input by default. **Some design choices** used in our method are ablated in in Table 7.6. We compare our proposed ScaleU block with FreeU in Table 7.6a. ScaleU leads to an improved localization AP suggesting that our learnable scaling of the backbone features outperforms the manually tuned FreeU. The impact of the proportion of MIS steps used in inference is explored in Table 7.6b. MIS can effectively improve the quality of the generated images and attribute binding. Lastly, for mask-conditioned input, Tabs. 7.6c and 7.6d show that points derived from both polygons and instance masks and using 128 points per instance mask gives the optimal performance.

Contribution of each component in InstanceDiffusion and its effect on image generation is measured in Table 7.5. We compare using different design choices for the fusion mechanism in UniFusion that fuse the location condition embeddings with the backbone text-to-image features: Format

#	*FA Fusion*	*MaskAttn*	*ScaleU*	*Inst. Cap.*	*MIS*	AP_{50}^{mask}	Acc^{color}	FID (↓)
1	✓	✓	✓	✓	✓	46.1	55.4	23.1
2	✗	✓	✓	✓	✓	40.5(5.6)	49.1(6.3)	23.4(0.3)
3	✓	✗	✓	✓	✓	45.5(0.6)	53.1(2.3)	23.2(0.1)
4	✓	✓	✗	✓	✓	43.7(2.4)	52.2(3.2)	23.2(0.1)
5	✓	✓	✓	✗	✓	44.0(2.1)	38.2(17.2)	23.1(0.0)
6	✓	✓	✓	✓	✗	46.1(0.0)	49.5(5.9)	26.6(3.5)

Table 7.5: Contribution of each component evaluated by removing or adding it and measuring the impact of the generated image in terms of its instance location performance (AP), and instance attribute binding (Acc), and overall image quality (FID). When Format Aware (FA) fusion mechanism is disabled, we use the Joint format fusion mechanism instead. Top row is the default setting for InstanceDiffusion in the paper and we report the drop in performance for each subsequent row in **red**.

Versions →	FreeU [286]	ScaleU
AP_{50}^{box}	52.2	55.4

(a) ScaleU

% of Steps →	0%	5%	10%	15%	20%
FID	26.6	24.2	23.1	23.0	23.0

(b) % of MIS Steps

format →	polygons	polygons+inside-points
AP_{50}^{mask}	43.6	46.1

(c) mask parameterization

# points →	32	64	128	256
AP_{50}^{mask}	40.0	44.2	46.1	46.1

(d) # points per mask

Table 7.6: Ablating design choices where the default settings are indicated in gray. **(a)** Compared to FreeU, our proposed ScaleU block improves the models ability to respect location conditions. **(b)** Multi-instance Sampler (MIS) lowers the FID and improves overall image quality. We use 10% MIS steps for a good tradeoff between inference speed and quality. **(c)** Parameterizing the instance masks using points on their boundaries and inside is beneficial. **(d)** Increasing the number of points used to parameterize masks improves performance.

	GLIGEN [253]	w/ MIS	InstanceDiffusion	w/ MIS
Acc^{color}	19.2	**29.7**	49.5	**55.4**

Table 7.7: Multi-instance Sampler can be adapted for previous location conditioned work, yielding notable performance gains.

Aware fusion (row 1) or the Joint Format fusion (row 2). We find that making the fusion mechanism format-aware significantly improves performance since the location formats specify varying degrees of control on the instance location. Comparing rows 1, 3 shows that using Instance-Masked Attention for fusing the location features helps the model focus on instance-specific regions and thus improves attribute binding (color accuracy). Removing ScaleU (rows 1, 4) causes a significant drop in AP_{50}^{mask} and Acc^{color} scores. This underscores the importance of dynamically adjusting the channel weights of both skip connected and backbone features. In row 5, we observe that our generated instance captions are critical for learning attribute binding, as indicated by the 17% drop in Acc^{color} after removing them. Finally, row 6 shows that Multi-instance Sampler (MIS) improves the overall image quality (lower FID) and attribute binding (color accuracy).

MIS improves prior work. As shown in Table 7.7, we applied Multi-instance Sampler to other

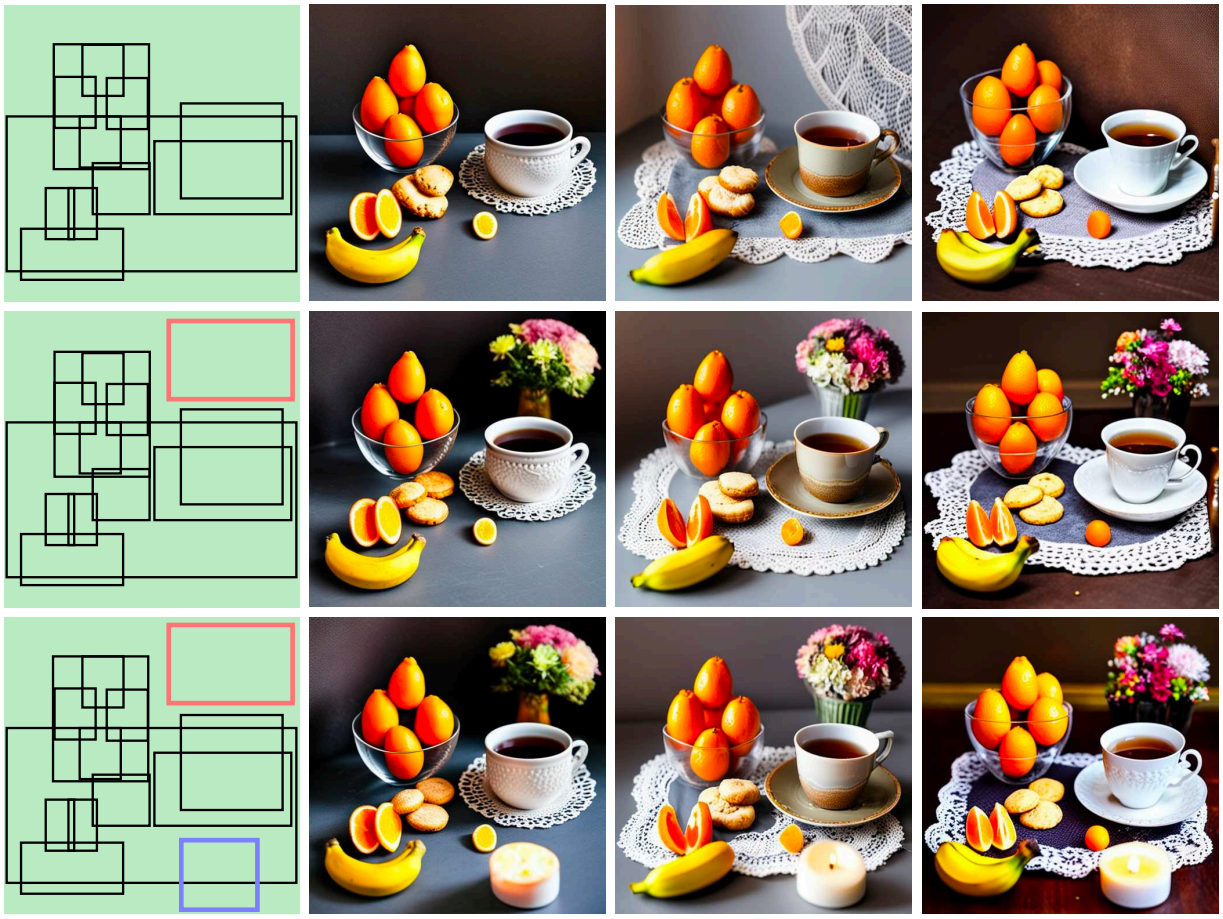


Image Caption: A cup of tea with tangerines, bananas, and cookies on the table. high quality. professional photo.
Instance Captions: 1) a cup of tea on a lace doily 2) a close up of three oranges on a black background 3) oranges in a glass bowl on a table 4) a tray of pastries on a table with oranges 5) a close up of some cookies on a table 6) oranges in a glass bowl 7) oranges in a glass bowl 8) an orange that has been cut in half on a table 9) an orange is cut in half 10) bananas 11) a bouquet of flowers on a table 12) a bouquet of flowers on a table 13) A candle

Figure 7.8: InstanceDiffusion can also support **iterative image generation**. Using the identical initial noise and image caption, InstanceDiffusion can progressively add new instances (like a bouquet of flowers in row two and a candle in row three), while minimally altering the pre-generated instances (row one).

location-conditioned text-to-image models and observed significant gains for the attribute binding ability of GLIGEN. These results confirm that MIS minimizes information leakage and that it can be easily used to improve other location-conditioned models.

Application: Iterative generation. Since InstanceDiffusion allows for precise control over the instances, we show a useful application that benefits from this property in Fig. 7.8. InstanceDiffusion allows users to selectively insert objects into precise locations while preserving the integrity of previously generated objects and the global scene. We hope that the precise control enabled by InstanceDiffusion will lead to many other such useful applications.

Preliminary

Diffusion Models [172], [217], [251] learn the process of text-to-image generation through iterative denoising steps initiated from an initial random noise map, denoted as z_T. Latent diffusion models (LDMs) [266] perform the diffusion process in the latent space of a Variational AutoEncoder [268], for computational efficiency, and encode the textual inputs as feature vectors from pretrained language models [17], [176].

Specifically, starting from a noised latent vector $\mathbf{z}_t$ at the time step t, a denoising autoencoder [219], [266], denoted as ϵ_θ, is trained to predict the noise ϵ that is added to the latent vector $\mathbf{z}$, conditioned on the text prompt $\mathbf{c}$. The training objective is defined as:

$$\mathcal{L} = \mathbb{E}_{\mathbf{z}\sim\mathcal{E}(\mathbf{x}),\epsilon\sim\mathcal{N}(0,1),t}\left[||\epsilon - \epsilon_\theta(\mathbf{z}_t, t, \tau(\mathbf{c}))||_2^2\right], \tag{7.5}$$

where t is uniformly sampled from the set of time steps $\{1, ..., T\}$. τ pre-process the text prompt $\mathbf{c}$ into text tokens $\tau(\mathbf{c})$, utilizing the pretrained CLIP text model [17].

During inference, a latent vector $\mathbf{z}_T$, sampled from a standard normal distribution $\mathcal{N}(0, 1)$ is iterative denoised using DDIM [248] to obtain z_0. Finally, the latent vector $\mathbf{z}_0$ is input into the decoder of VAE to generate an image $\widetilde{\mathbf{x}}$.

Ablation Study

In addition to the ablation study we presented in Sec. 7.4, in this section, we also offer additional ablations focusing on the hyper-parameters of UniFusion modules, design variations for ScaleU, the impact of model inference with hybrid inputs, among other aspects.

Bandwidth →	4	8	16	32
AP^{box}_{50}	50.8	53.9	55.4	55.3

(a) freq. bandwidth

N →	512	2048	3072	4096
AP^{box}_{50}	52.9	53.5	55.4	55.4

(b) MLP dim

Table 7.8: Ablating design choices for UniFusion. Components and default settings are highlighted in gray. **(a)** We vary the frequency bandwidth used in the Fourier embeddings of the point coordinates in the UniFusion block. **(b)** We study the impact of the dimensionality of MLP layers in the UniFusion block.

Design choices for UniFusion. We first analyze the impact of frequency bandwidths when projecting location conditions into a higher-dimensional feature space with Fourier Transform, as depicted in Table 7.8a. The Fourier transform process empowers a multilayer perceptron (MLP) to grasp high-frequency functions in low-dimensional problem domains [285]. We apply the Fourier mapping to the 2D point coordinates associated with each location to convert them into an embedding. The embedding enables MLPs to better learn a high-frequency function for the coordinates. Notably, expanding the frequency bandwidth tends to improve the performance, but a plateau is reached once the bandwidth exceeds 16. The influence of the dimensionality (N) of the MLP layer within UniFusion is assessed in Table 7.8b. We find that a dimension of 3072 emerges as the

optimal balance between model efficacy and its size. Increasing the MLP layers dimensions from 3072 to 4096 does not yield further improvements in performance. Therefore, we select $N = 3072$ by default.

Can we use one single token for all location conditions? Actually, we can still achieve reasonable performance using a unified tokenization function that results in a single token for all forms of location inputs, as demonstrated in Table 7.5. However, having multiple tokens (M tokens) for different input types (M types) leads to optimal performance. This is because these four types of layout conditions necessitate distinct approaches to ensuring that the model respects the layout condition appropriately. Specifically, the model needs to disseminate grounding information to adjacent visual tokens when using point and scribble inputs. In contrast, bounding-box and mask conditions require the model to confine the grounding information injection within the specified box or mask.

Why not employ masks as extra channels, as seen in GLIGEN [253] and ControlNet [254]? In these approaches, the semantic segmentation masks (do not discriminate instances in the same class) are resized to a smaller resolution of 64×64 features. Nonetheless, our observations indicate that when the occlusion ratio between instances is high, particularly in cases where overlapping instances carry similar semantic information, the model's performance is compromised a lot. Additionally, the model encounters difficulties when generating high-quality results for very small objects. Therefore, we convert all masks into point-based inputs. However, it is possible that adding segmentation masks as additional input could further improve our model's performance, we leave it for future research.

Versions →	FreeU [286]	ScaleU	SE-ScaleU
AP^{box}_{50}	52.2	55.4	55.2

Table 7.9: We evaluate the performance of the lightweight ScaleU (Fig. 7.9 b) against the dynamically adaptable SE-ScaleU (Fig. 7.9 c), and further compare our ScaleU with FreeU [286], a previous work that manually tune the scaling vectors.

Design choices for ScaleU are depicted in Fig. 7.9. Beyond the standard ScaleU block described in Sec. 7.3, which re-calibrates both main and skip-connected features before their concatenation in the UNet model, we explored an alternative design, SE-ScaleU (Fig. 7.9c). This variant employs an MLP layer, similar to the Squeeze-and-Excitation module [296], for dynamically generating scaling vectors based on each sample's feature map. However, as demonstrated in Table 7.6a, while SE-ScaleU offers performance on par with the light-weight ScaleU block, it requires additional parameters in the MLP layers. Consequently, we default to using ScaleU.

Design choices for Multi-instance Sampler. There are two design strategies for Multi-instance Sampler: crop-and-paste and instance latents averaging, with the latter being our paper's default approach. The crop-and-paste Multi-instance Sampler involves: 1) Running separate denoising operations for each of the n instances over M steps to obtain instance latents L_I. 2) Cropping instance latents $\{L_I^1, \cdots, L_I^n\}$ as per location conditions and pasting these cropped, denoised latents onto the global latent L_G, derived from all instance tokens and text prompts, at their respective

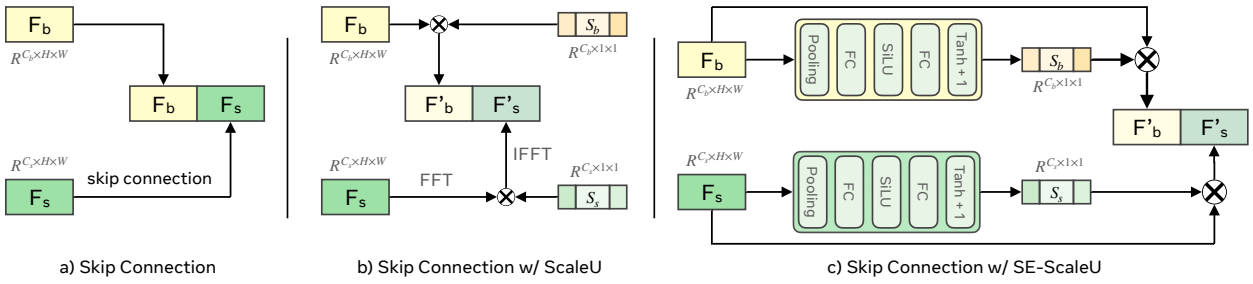


Figure 7.9: Various design choices for the ScaleU block. In the UNet architecture, $\mathbf{F}_b$ represents the main features, while $\mathbf{F}_s$ denotes the skip connected features. Typically, UNet employs skip connections as shown in (a) to pass features from the encoder to the decoder, aiding in recovering spatial information lost in downsampling. We introduce ScaleU (b), which re-calibrates both the main and skip-connected features prior to their concatenation. Additionally, we implement SE-ScaleU (c), which utilizes an MLP layer—akin to the Squeeze-and-Excitation module [296]—to dynamically produce scaling vectors conditioned on each sample's feature map.

	crop-and-paste	latents averaging
FID	23.3	bfseries 23.1
AP_{50}^{mask}	45.5	bfseries 46.1

Table 7.10: Model inference with Multi-instance Sampler using different Multi-instance Sampler design variations.

locations. 3) Continuing the denoising process on the combined latent from step (2) using all instance tokens, instance text prompts, and the global image prompt. This process largely mirrors our default latent averaging Multi-instance Sampler, except for step (2)'s latent merging method.

While crop-and-paste Multi-instance Sampler matches or slightly surpasses the performance of our default averaging approach on some testing cases, it has its limitations: 1) In step (2) of the crop-and-paste Multi-instance Sampler, the model needs to crop instance latents according to the bounding box or mask provided, limiting its application to bounding boxes, and instance masks. For point inputs and scribbles, the model has to conjecture the size/shape of the instance. 2) The presence of overlapping instances presents a challenge. The model can only preserve latents from a single instance in these regions, resulting in blurred and diminished-quality pixels in areas of instance overlap.

Multiple location formats at inference are analyzed in Table 7.11. It is observed that having more location conditions provides the best performance and more precise control on the instance location. This results in significant performance improvements, particularly for instance masks (9.9% AP^{mask}) and scribble (16.3% PiM). Note that many of the other location formats can be automatically derived: For image generation conditioned on instance masks, since both the box and the central point can be inferred from the mask, our model enjoys this performance improvement

Figure 7.10: As the UniFusion module is integrated for an increasing proportion of timesteps (from 5% timesteps to 75% timesteps), the model's adherence to the instance conditions progressively improves. The generation of the sunflower at the top left corner occurs once the UniFusion module is activated for 75% of the total timesteps.

box	point	mask	AP^{box}	AP^{box}_{50}	point	box	mask	PiM
✓	✗	✗	36.1	52.4	✓	✗	✗	79.7
✓	✓	✗	39.0	55.4	✓	✓	✗	86.3
✓	✓	✓	bfseries 41.6	bfseries 56.2	✓	✓	✓	bfseries 86.6
mask	box	point	AP^{mask}	AP^{mask}_{50}	scribble	box	mask	PiM
✓	✗	✗	13.6	27.3	✓	✗	✗	67.3
✓	✓	✗	20.9	40.9	✓	✓	✗	71.8
✓	✓	✓	bfseries 23.5	bfseries 46.1	✓	✓	✓	bfseries 83.6

Table 7.11: Model inference with hybrid location inputs. We found that hybrid inputs can often help the model to better respect the location conditions and lead to performance gains. Default inference setting is colored in gray. *Note: Given a box, one can always determine a point by using its center. Similarly, from a mask, both a box and a central point can be derived without the need for extra user inputs.*

without imposing extra demands on users; Likewise, for boxes, the performance gains achieved by incorporating a point as the instance location condition can be obtained without any additional user inputs. These derived location formats improve location conditioning without additional user inputs.

Impact of UniFusion module. Fig. 7.10 illustrates that as the UniFusion module is applied over a increasing percentage of timesteps (ranging from 10% to 75%), the model's adherence to the instance conditions progressively improves. For instance, the sunflower in the top left corner is generated only when the UniFusion module is active for 75% of the total timesteps. Similarly, the sunflower in the bottom right corner manifests after the module has been active for 25% of the timesteps. Additionally, the model's ability to accurately adhere to the teddy bear's location condition is enhanced as UniFusion is utilized for more extended timesteps.

Model Training

Model training. We follow the same setup as GLIGEN [253] and initialize our model with a pretrained text-to-image model whose layers are kept frozen. We add the learnable parameters for instance conditioning and train the model with a batch size of 512 for 100K steps. We use the Adam optimizer [290] with a learning rate that is warmed up to 0.0001 after 5000 iterations. We learn the model with exponential moving average (EMA) on model parameters with a decay factor of 0.99 and use the EMA model during the inference time. In addition, we have a 10% probability to set all four location inputs as null tokens to support classifier-free guidance, following the approach proposed in [297]. Additionally, for the various location condition tokens, including masks, bounding boxes, points, and scribbles, each has a 10% dropout rate. We use 64 Nvidia A100 GPUs to train the model.

Applications and Qualitative Results

Iterative Image Generation. InstanceDiffusion's capability for precise instance control allows InstanceDiffusion to excel in multi-round image generation, leveraging this feature. InstanceDiffusion enables users to strategically place objects in specific locations while maintaining the consistency of previously generated objects and the overall scene. We outline the process of our iterative image generation in the following three steps:

- 1) Initially, generate images using the global image caption, all instance captions with their respective location conditions, and random noise.
- 2) Users have the option to introduce new instances by supplying additional instance conditions, including text prompts and locations. They can also modify existing instances by altering their descriptions or locations.
- 3) Employ the revised set of instance conditions, the global prompt, and the same random noise as in step 1 to create a new image.

Steps 2 and 3 can be repeated for multiple rounds until the desired outcome is achieved.

In addition to the visuals we have shown in the main paper, we provide more qualitative results on iterative image generation in Fig. 7.11. With minimal changes to pre-generated instances and the overall scene, users can selectively introduce new instances (as seen in row two, where "a bouquet of flowers" and "a donut" are added to the images from row one), substitute one instance for another (in row three, "a donut" is replaced with "a lighted candle"), reposition an instance (in row four, "a lighted candle" is moved to the bottom right corner), or adjust the size of an instance (in row five, the size of "a bouquet of flowers" is increased).

Hierarchical location conditioning in image composition. Our findings, illustrated in Fig. 7.12, reveal that incorporating hierarchical location conditionings - specifically, the locations and sizes of parts and subparts of an instance - as model inputs subtly alters the overall pose of an object (right, left, front). This demonstrates the effective use of spatial hierarchy in visual design. We hope that this capability could inspire more future research and applications in fine-grained control in image generation.

More demo results for InstanceDiffusion's image generation are shown in Figs. 7.13 and 7.14.

7.5 Conclusions, Limitations and Future Work

We presented InstanceDiffusion which enables precise instance-level control for text-to-image generation and significantly outperforms all prior work in terms of complying with instance attributes and accommodates a variety of location formats – masks, boxes, scribbles and points. Our studies indicate that there is a noticeable disparity in the generation quality of small objects compared to larger ones. We also find that texture binding for instances poses a challenge across all methods tested, including InstanceDiffusion. Improving instance conditioning for these cases is an important direction for future research.

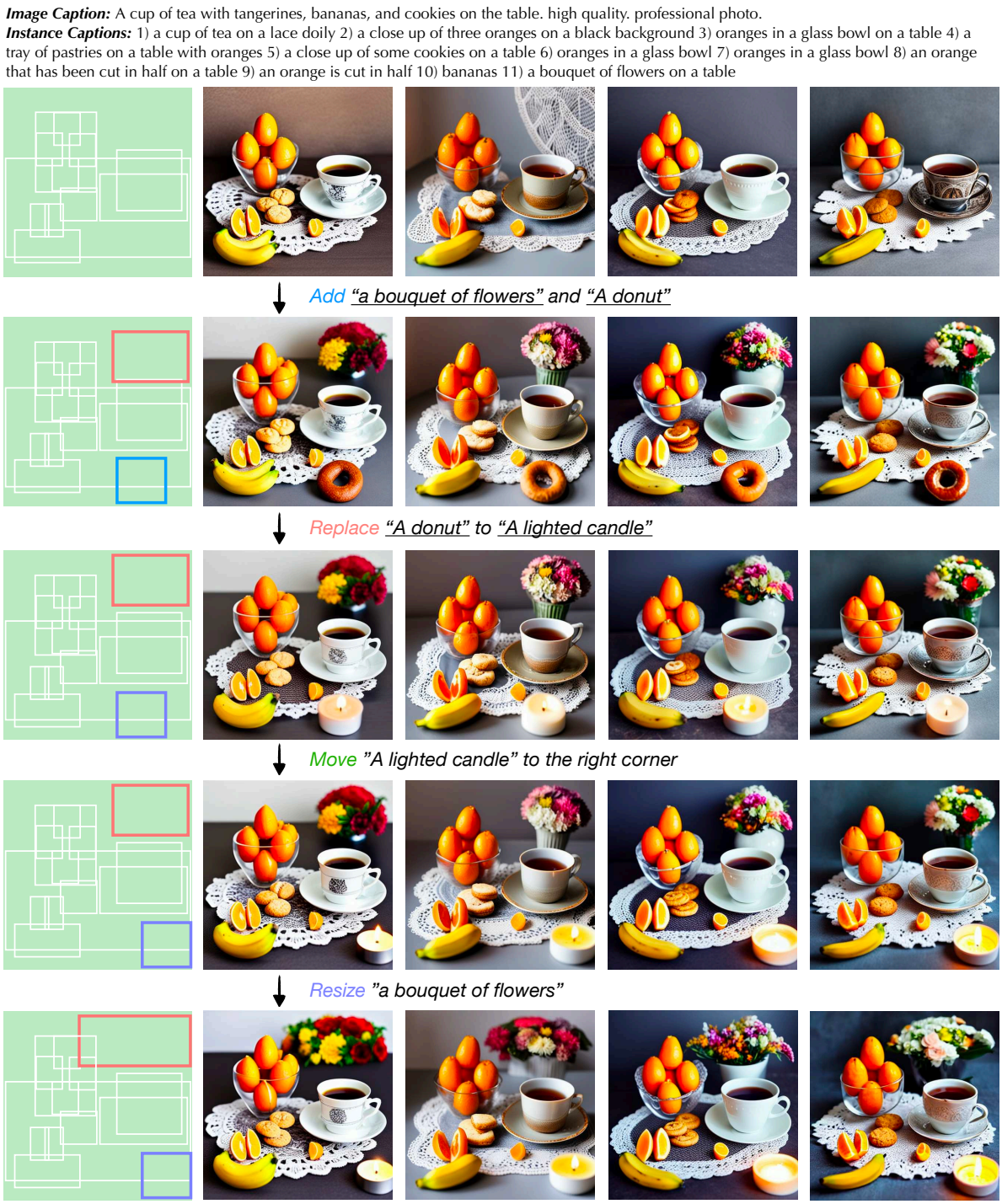


Figure 7.11: Iterative Image Generation. With minimal changes to pre-generated instances and the overall scene, users can selectively introduce new instances (as seen in row two, where "a bouquet of flowers" and "a donut" are added to the images from row one), substitute one instance for another (in row three, "a donut" is replaced with "a lighted candle"), reposition an instance (in row four, "a lighted candle" is moved to the bottom right corner), or adjust the size of an instance (in row five, the size of "a bouquet of flowers" is increased).

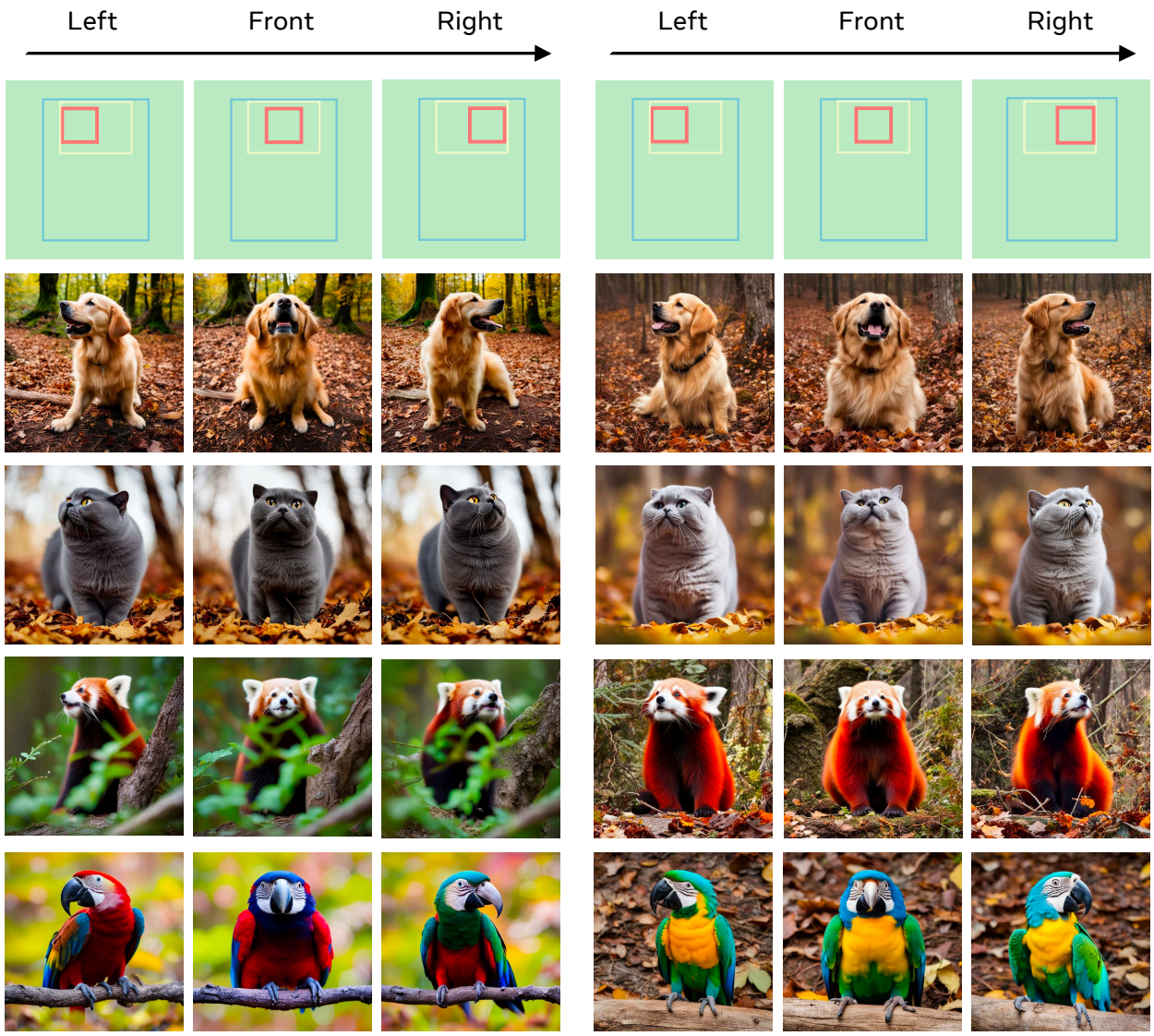


Image Caption: A cute {animal} standing in a forest at autumn, high quality, professional photo.
Instance Captions: 1) a cute {animal} 2) head 3) Golden Retriever / British Shorthair / Red Panda: nose and mouth; Macaw: beak

Figure 7.12: Let's get everybody turning heads! Hierarchical location conditioning in image composition. These results illustrate how the orientation of parts and subparts subtly influences the pose of the whole object (right, left, front), demonstrating the application of spatial hierarchy in visual design. We anticipate that this capability will pave the way for further research and applications in achieving more precise control in image generation.

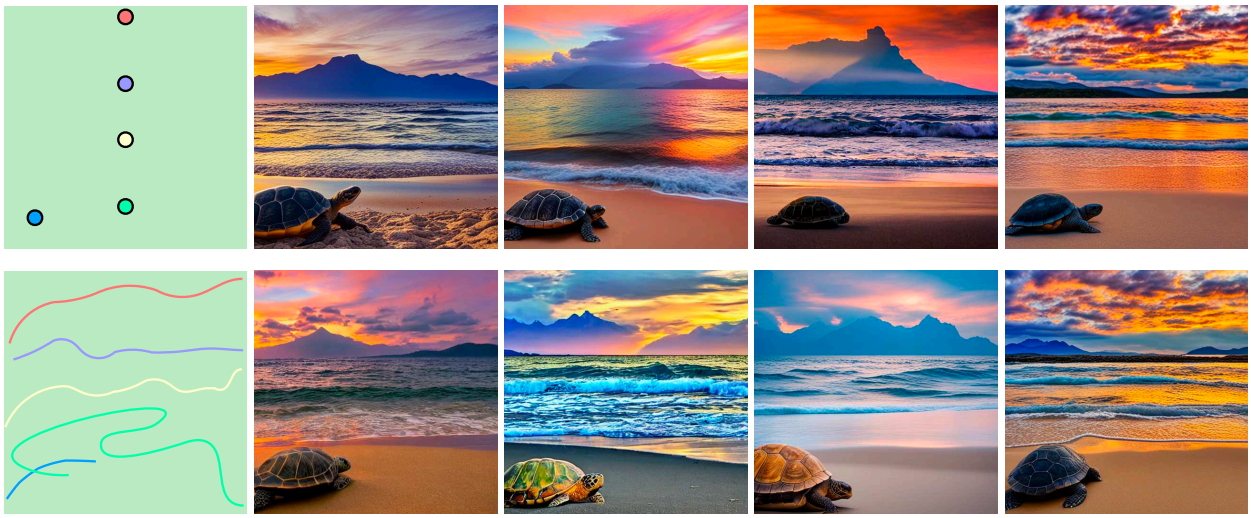


Image Caption: stunning beach scene with at sunset. mountains in the distance. a turtle on the beach. Beautiful summer landscape. Ocean waves on beach at sunset. high quality. professional photo.
Instance Captions: 1) sky at sunset, with blue and purple clouds, beautiful summer landscape 2) mountains at distance 3) ocean waves 4) beach 5) a turtle on the beach

Image Caption: Black Easter eared rabbit sitting in wicker basket with ripe apples on pink wooden background.Thanksgiving day concept with funny cute hare and autumn harvest.
Instance Captions: 1) a black rabbit 2) a wicker basket with a rabbit in it. 3) a close up of a ball of hay on the ground

Figure 7.13: More image generations with point and scribbles as model inputs, which were not supported by previous layout conditioned text-to-image models.

Image Caption: Cathedral of Palma de Mallorca viewed through lush greenery of the island. Vintage painting, background illustration, beautiful picture, travel texture
Instance Captions: 1) a large cathedral with spires and trees in the background; 2) a cathedral with a cloudy sky 3) palm trees 4) palm trees 5) palm trees 6) an ornate building with a spire and a clock tower

Image Caption: Knitted toy animal in flowers chrysanthemums. Floral background. Minsk Botanical Garden
Instance Captions: 1) sunflower; 2) a small crocheted toy sits on top of yellow flowers 3) sunflower

Figure 7.14: More demo images on image generation with point and bounding box as model inputs. The standard Text-to-Image model refers to the pretrained text-to-image model InstanceDiffusion and GLIGEN used. Standard T2I model uses the image caption as the model input to generates these images.

Part IV

Self-supervised Debiased Learning

"Everything we hear is an opinion, not a fact. Everything we see is a perspective, not the truth."

— Marcus Aurelius

Chapter 8

Unsupervised Visual Attention and Invariance for Reinforcement Learning

Vision-based reinforcement learning (RL) is successful, but how to generalize it to unknown test environments remains challenging. Existing methods focus on training an RL policy that is universal to changing visual domains, whereas we focus on extracting visual foreground that is universal, feeding clean invariant vision to the RL policy learner. Our method is completely unsupervised, without manual annotations or access to environment internals.

Given videos of actions in a training environment, we learn how to extract foregrounds with unsupervised keypoint detection, followed by unsupervised visual attention to automatically generate a foreground mask per video frame. We can then introduce artificial distractors and train a model to reconstruct the clean foreground mask from noisy observations. Only this learned model is needed during test to provide distraction-free visual input to the RL policy learner.

Our Visual Attention and Invariance (VAI) method significantly outperforms the state-of-the-art on visual domain generalization, gaining 15$\sim$49% (61$\sim$229%) more cumulative rewards per episode on DeepMind Control (our DrawerWorld Manipulation) benchmarks. Our results demonstrate that it is not only possible to learn domain-invariant vision without any supervision, but freeing RL from visual distractions also makes the policy more focused and thus far better.

8.1 Introduction

Vision-based deep reinforcement learning (RL) has achieved considerable success on robot control and manipulation. Visual inputs provide rich information that are easy and cheap to obtain with cameras [298]–[302]. However, vision-based RL remains challenging: It not only needs to process high-dimensional visual inputs, but it is also required to deal with significant variations in new test scenarios (Fig. 8.1), e.g. color/texture changes or moving distractors [303], [304].

One solution is to learn an ensemble of policies, each handling one type of variations [305]. However, anticipating all possible variations quickly becomes infeasible; domain randomization

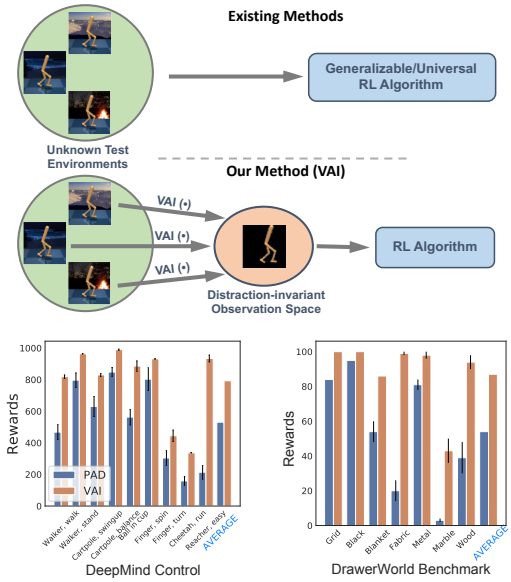


Figure 8.1: Top) Two ways to make vision-based reinforcement learning generalizable to unknown environments at test time: existing methods aim to learn an RL policy universal across domains, while our proposed Visual Attention and Invariance (VAI) extracts domain-invariant visual foregrounds, delivering clean and robust input to the RL agent.

Bottom) VAI significantly outperforms PAD (SOTA), improving cumulative rewards by 49% and 61% in random color tests (DeepMind Control) and random texture tests (DrawWorld), respectively.

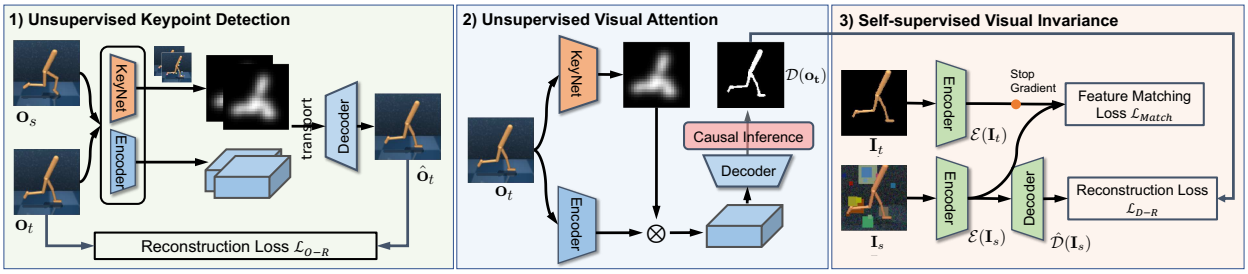


Figure 8.2: Our VAI method has three components. **1) Unsupervised keypoint detection**: Given two adjacent video frames, we learn to predict keypoints and visual features from each image so that foreground features and target keypoints can be used to reconstruct the target frame, without any manual annotations. **2) Unsupervised visual attention**: We apply causal inference to remove the model bias from the foreground mask derived from detected keypoints. **3) Self-supervised visual invariance**: We are then able to add artificial distractors and train a model to reconstruct the clean foreground observations. Keypoint and attention modules are only used during training to extract foregrounds from videos without supervision, whereas only the last encoder/decoder (colored in green) trained for visual invariance is used to remove distractors automatically at the test time.

methods [306]–[310] apply augmentations in a simulated environment and train a *domain-agnostic universal policy* conditioned on estimated discrepancies between testing and training scenarios.

Two caveats limit the appeal of a universal RL policy. **1) Model complexity.** The policy learner must have enough complexity to fit a large variety of environments. While there are universal visual recognition and object detection models that can adapt to multiple domains [311], [312], it would be hard to accomplish the same with a RL policy network often containing only a few convolutional layers. **2) Training instability.** RL training could be brittle, as gradients for (often

non-differentiable) dynamic environments can only be approximated with a high variance through sampling. Adding strong augmentations adds variance and further instability, causing inability to converge. [313] handles instability with weaker augmentations, in turn reducing generalization.

The state-of-the-art (SOTA) approach, PAD [313], performs unsupervised policy adaption with a test-time auxiliary task (e.g. inverse dynamic prediction) to fine-tune the visual encoder of the policy network on the fly. However, there is no guarantee that intermediate representations would fit the control part of the policy network. Drastic environment changes such as background texture change from *grid* to *marble* can cause feature mismatches between adapted layers and frozen layers, resulting in high failure rates.

Instead of pursuing a policy that is universal to changing visual domains, we propose to extract visual foreground that is universal, and then feed clean invariant vision to a standard RL policy learner (Fig. 8.2). As the visual observation varies little between training and testing, the RL policy can be simplified and focused, delivering far better results.

Our technical challenge is to deliver such clean visuals with a completely unsupervised learning approach, without mannual annotations or access to environment internals.

Given videos of actions in a training environment, we first learn how to extract visual foreground with unsupervised keypoint detection followed by unsupervised visual attention to automatically generate a foreground mask per video frame. We can then introduce artificial distractors and train a model to reconstruct the clean foreground mask from noisy observations. Only this learned model, not the keypoint or attention model, is needed during test to provide distraction-free visual input to the RL policy learner.

Our unsupervised Visual Attention and Invariance (VAI) method has several desirable properties.

1. **Unsupervised task-agnostic visual adaption training**. Our foreground extraction only assumes little background change between adjacent video frames, requiring no manual annotations or knowledge of environment internals (e.g. get samples with altered textures). It does not depend on the task, policy learning, or task-specific rewards associated with RL. That is, for different tasks in the same environment, we only need to collect one set of visual observations and train one visual adapter, which gets us a huge saving in real-world robotic applications.
2. **Stable policy training, no test-time adaptation.** By freeing RL from visual distractions, our policy learning is stable and fast without being subject to strong domain augmentations, and our policy deployment is immediate without test-time fine-tuning.
3. **Clear interpretation and modularization.** We extract keypoints from videos to identify foreground, based on which attentional masks can be formed. This unsupervised foreground parsing allows us to anticipate visual distractions and train a model to restore clean foregrounds. Compared to existing methods that work on intermediate features, our method has clear assumptions at each step, which can be visualized, analyzed, and improved.

We conduct experiments on two challenging benchmarks with diverse simulation environments: DeepMind Control suite [313], [314] and our DrawerWorld robotic manipulation tasks with texture distortions and background distractions during deployment. Our VAI significantly outper-

forms the state-of-the-art, gaining 15∼49% (61∼229%) more cumulative rewards per episode on DeepMind Control (our DrawerWorld Manipulation) benchmarks.

To summarize, we make the following contributions.

1. We propose a novel domain generalization approach for vision-based RL: Instead of learning a universal policy for varying visual domains, we decouple vision and action, learning to extract universal visual foreground while keeping the RL policy learning intact.
2. We propose a fully unsupervised, task-agnostic visual adaptation method that removes unseen distractions and restores clean foreground visuals. Without manual annotations, strong domain augmentations, or test-time adaptation, our policy training is stable and fast, and our policy deployment is immediate without any latency.
3. We build unsupervised keypoint detection based on KeyNet [315] and Transporter [316]. We develop a novel unsupervised visual attention module with causal inference for counterfactual removal. We achieve visual invariance by unsupervised distraction adaptation based on foreground extraction. Each step is modularized and has clear interpretations and visualizations.
4. We propose DrawerWorld, a pixel-based robotic manipulation benchmark, to test the adaptation capability of vision-based RL to various realistic textures.
5. Our results demonstrate that it is not only possible to learn domain-invariant vision from videos without supervision, but freeing RL from visual distractions also leads to better policies, setting new SOTA by a large margin.

8.2 Related Works

Unsupervised Learning has made much progress in natural language processing, computer vision, and RL. It aims to learn a feature transferable to downstream tasks [26]–[28], [40], [49], [317]–[320]. In RL, UNREAL [321] proposes unsupervised reinforcement and auxiliary learning to improve learning efficiency of model-free RL algorithms, by maximizing pseudo-reward functions; CPC [322] learns representations for RL in 3D environments by predicting the future in the latent space with autoregressive models; CURL [323] extracts high-level features from raw pixels using contrastive learning and performs off-policy control on extracted features to improve data-efficiency on pixel-based RL.

Domain Adaptation incorporates an adaptation module to align the feature distribution from the source domain and the target domain without paired data [247], [324]–[328]. There are various approaches to this, from using supervised data [325], [326], [329], to assumed correspondences [330], to unsupervised approaches [247], [331], [332].

Multi-domain Learning learns representations for multiple domains known a prior [311], [312], [333], [334]. A combination of shared and domain-specific parameters are adopted. It is also feasible to simply learn multiple visual domains with residual domain adapters [311], [312].

Our work is different from these works, since we do not have prior knowledge of test data distributions and the model needs to generalize to unknown test environments.

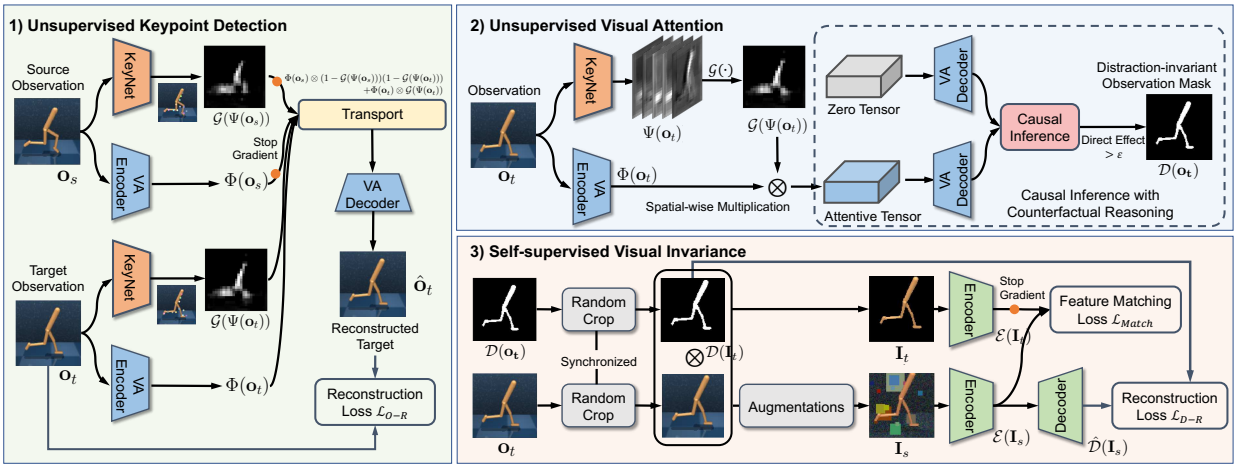


Figure 8.3: Technical implementation of our three components. **1) Unsupervised keypoint detection**: We build unsupervised keypoint detection and visual feature extraction based on KeyNet [315] and Transporter [316]. The goal is to reconstruct the target frame from the target foreground appearance and the source-transported background appearance, capturing a moving foreground on a relatively still background. **2) Unsupervised visual attention**: We remove the model bias in the foreground mask derived from detected keypoints with novel causal inference for counterfactual removal. **3) Self-supervised visual invariance**: We train a model to restore an invariant foreground visual image by adding artificial distractors to extracted foreground and perform self-supervised distraction removal.

Robustness to Distribution Shifts studies the effect of corruptions, perturbations, out-of-distribution examples, and real-world distribution shifts [309], [335]–[339]. Recent deep RL approaches model such uncertainties explicitly.

[340] uses recurrent neural networks for direct adaptive control and determines dynamic model parameters on-the-fly. UP-OSI [341] applies indirect adaptive control for online parameter identification. EPOpt [305] uses simulated source domains and adversarial training to learn policies that are robust and generalizable to a range of possible target domains. PAD [313] uses self-supervision to continue policy training during deployment without any rewards, achieving SOTA in several environments. SODA [342], a concurrent work to ours, alternates strong augmentations associated with self-supervised learning and weak augmentations associated with RL for obtaining both generalizability and stability.

Instead of demanding a universal *policy* that is invariant to distribution shifts or transferable to novel environments, we achieve generalizability by demanding universal *visuals* that can be fed into the subsequent RL policy learner, freeing it from visual distractions and making it more effective.

8.3 Unsupervised Visual Attention & Invariance

Our goal is to extract universal visual foreground and then feed clean invariant vision to an RL policy learner (Fig. 8.2). Our technical challenge is to deliver such clean visuals with a completely unsupervised learning approach, without mannual annotations or access to environment internals.

Our VAI method has three components: Unsupervised keypoint detection, unsupervised visual attention, and self-supervised visual invariance. The first two are only used during training to extract foregrounds from videos without supervision, whereas only the last trained model is deployed to automatically remove distractors from a test video.

Unsupervised Keypoint Detection

We assume that training videos contain moving foregrounds against a relatively still background. Our idea for unsupervised foreground extraction is the following: Given two such source and target frames, we can learn to predict keypoints and visual features from each image so that foreground features and target keypoints can be used to reconstruct the target frame, without requiring manual annotations.

For a particular image pair, the moving foreground may have a still part (*upper body*), or the background may have a moving part (*flickering flames*) . However, when the keypoint predictor and the visual feature extractor have to work consistently across all the videos in the same environment, they would have to focus on the entire moving foreground and disregard the random minor background motion.

Let $\mathbf{o}_s, \mathbf{o}_t \in \mathbb{R}^{C\times H\times W}$ denote the source and target frames sampled from a trajectory, where C, H, and W are the channel dimension, image height and width respectively. Let $\Phi(\cdot)$ denote the visual feature extractor. Let $\Psi(\cdot)$ denote the keypoint network that predicts K keypoints in terms of 2D spatial locations $\{\mu_k\}$. We render each keypoint as a smaller $H' \times W'$ Gaussian heatmap with fixed variance σ^2, and derive a foreground mask by taking the max of all of them:

$$\mathcal{G}(\mu; x) = \max_{k\in\{1,2,\ldots,K\}} \exp\left(-\frac{\|x-\mu_k\|^2}{2\sigma^2}\right). \tag{8.1}$$

We follow KeyNet [315], [316] to reconstruct the target observation $\mathbf{o}_t$ from K landmarks $\Psi(\mathbf{o}_s), \Psi(\mathbf{o}_t)$. We follow [316] to transport the source background appearance to the target frame by putting the source feature at common background areas and the target feature at the target keypoints:

$$\begin{aligned}\hat{\Phi}(\mathbf{o}_t, \mathbf{o}_s) =& \Phi(\mathbf{o}_s) \otimes (1-\mathcal{G}(\Psi(\mathbf{o}_s)))(1-\mathcal{G}(\Psi(\mathbf{o}_t))) \\ &+\Phi(\mathbf{o}_t) \otimes \mathcal{G}(\Psi(\mathbf{o}_t))\end{aligned} \tag{8.2}$$

where $\otimes$ denotes location-wise multiplication applied to each channel. A visual attention (VA) decoder outputs a reconstruction $\hat{\mathbf{o}}_t$ of target frame $\mathbf{o}_t$ from the transported feature $\hat{\Phi}(\mathbf{o}_t, \mathbf{o}_s)$. Minimizing the reconstruction loss below optimizes the KeyNet and the visual attention encoder/decoder end-to-end:

$$\mathcal{L}_{\text{O-R}}(\mathbf{o}_t, \hat{\mathbf{o}}_t) = \|\mathbf{o}_t - \hat{\mathbf{o}}_t\|_2^2. \tag{8.3}$$

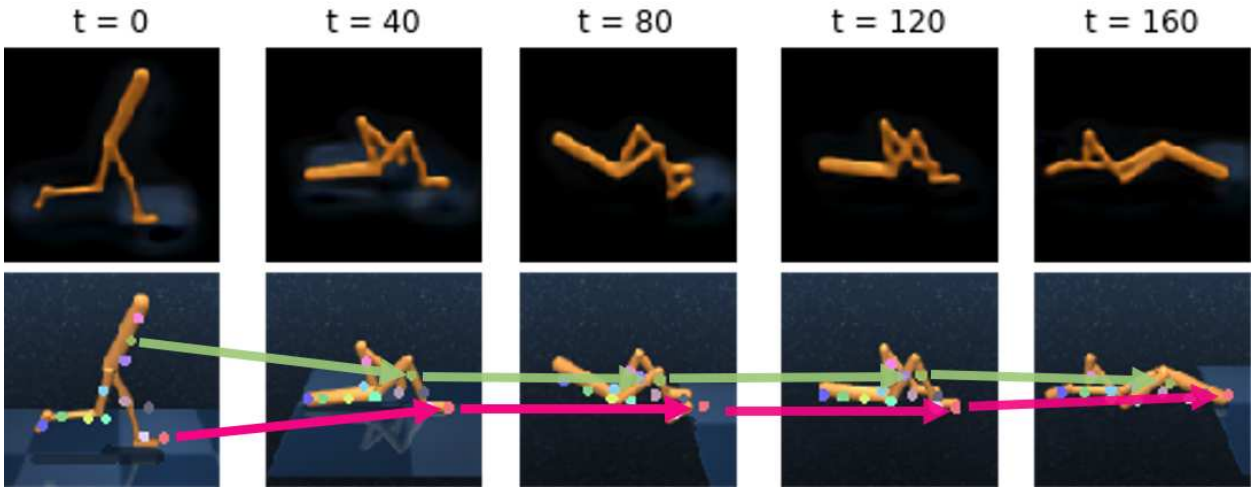


Figure 8.4: Our VAI foreground reconstruction (Row 1) provides clearer and more robust foreground visual information than detecting keypoints across image frames using Transporter (Row 2). Due to occlusion, symmetry, and lacking visual distinctions, it is often impossible to track keypoints consistently across frames. That is, keypoint locations alone are not suitable as an invariant visual representation.

Note that the original Transporter only focuses on changes between frames in the same episode, whereas we also sample frames from different episodes 50% of the time in reacher environment which has a fixed target throughout each episode, so that our keypoints will be able to capture the target and spread over the entire moving foreground.

Unsupervised Spatial Attention

Now we already have an unsupervisedly learned keypoint detector. We first explain why we do not use keypoints for control and instead derive a visual foreground mask. We then describe our novel causal inference formulation for obtaining a foreground mask without model bias.

Transporter [316] successfully makes use of keypoints for RL in Atari ALE [343] and Manipulator [314]. Keypoints are geometrical extraction without visual appearance distractions that they could be potentially used to minimize differences between training and testing environments.

However, there are three major issues with keypoints in practice. **1)** It is often hard to track keypoints consistently across frames; even for humans, whether a keypoint is on the left or right foot is unclear in Fig. 8.4. This implies that using predicted keypoints for control directly would be brittle even in clean images.

2) While keypoints along with image features and LSTM could work on relatively complicated tasks [316], they add substantial model complexity and computational costs. **3)** While keypoints themselves are free of visual distractions, their extractor (*KeyNet*) is only trained for the training environment, with no guarantee for robustness against domain shifts.

We thus propose to generate a foreground mask $\mathcal{G}(\Psi(\mathbf{o}_t))$ from the (un-ordered) collection of predicted keypoints instead. We enhance the visual feature in the foreground: $\mathcal{G}(\Psi(\mathbf{o}_t)) \otimes \Phi(\mathbf{o}_t)$

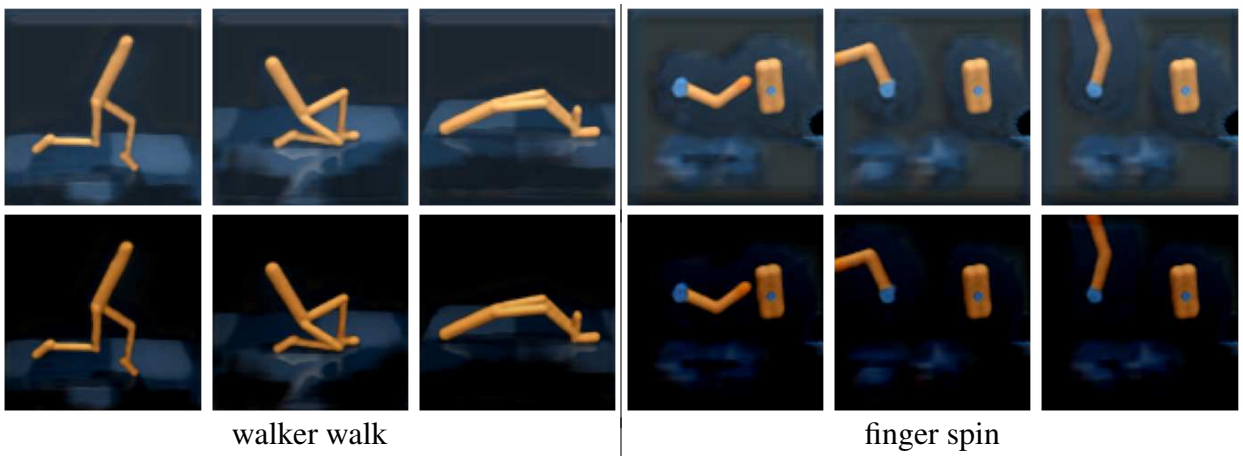


Figure 8.5: Foreground reconstructions with causal inference are cleaner (Row 2) than those without (Row 1).

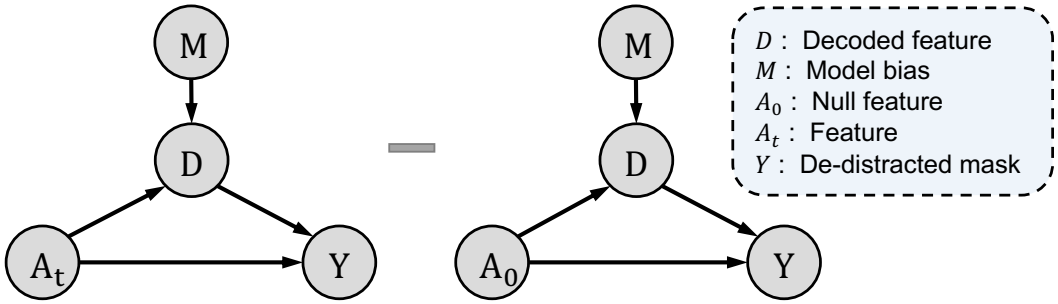


Figure 8.6: Causal graph of casual inference with counterfactual reasoning for our foreground mask extraction. The Controlled Direct Effect (CDE) is measured by the contrast between two outcomes: the counterfactual outcome given the visual feature A_t and that given the null feature A_0.

and pass it to the VA decoder to reconstruct a cleaner image $\hat{\mathbf{o}}_t$ (Fig. 8.5 Row 1). However, it is blurry with common background remnants captured in the bias terms of the decoder. The bias terms are essential for proper reconstruction and cannot be simply set to zero.

We apply causal inference with counterfactual reasoning [344]–[348] to remove the model bias (Fig. 8.6). Intuitively, the predicted foreground mask Y has a direct cause from the visual feature A, and an indirect cause from the model bias M through the decoder D. To pursue the direct causal effect, we perform counterfactual reasoning known as *Controlled Direct Effect* (CDE) [344], [347], which contrasts the counterfactual outcomes (marked by $do(\cdot)$) between visual feature A_t and null visual feature A_0 (set to the zero tensor):

$$\text{CDE}(Y) = [Y|do(A_t), do(M)] - [Y|do(A_0), do(M)]. \tag{8.4}$$

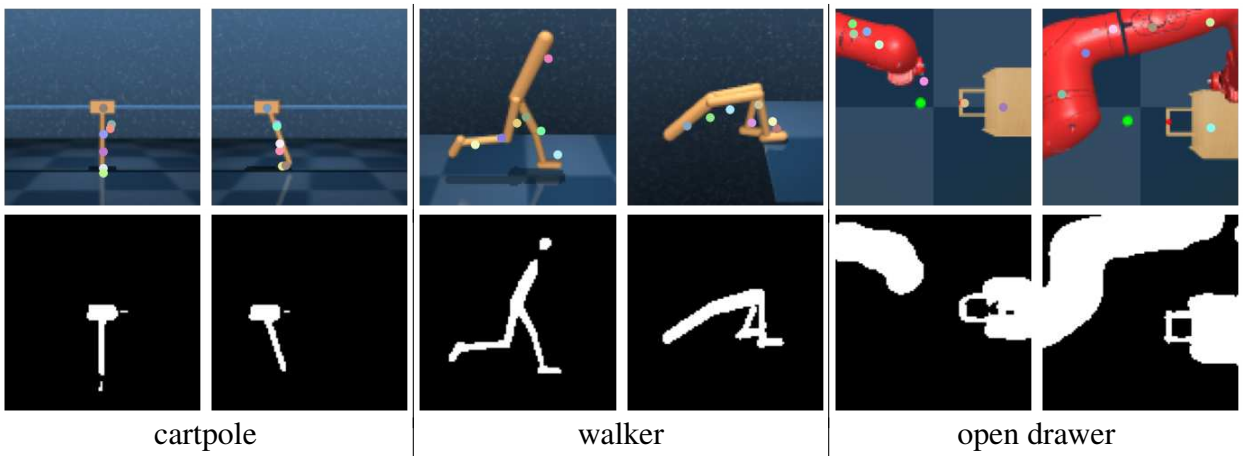


Figure 8.7: Our detected keypoints (Row 1) and generated foreground masks (Row 2) from DeepMind control and DrawerWorld benchmarks. Note that they could cover multiple moving objects in the foreground.

Figure 8.8: Visualization results of testing environments in DeepMind Control benchmark [313], [314]. The testing environment changes include randomized colors, video backgrounds, and background distractions.

We further threshold it to obtain the foreground mask $\mathcal{D}(\mathbf{o}_t)$:

$$\mathcal{D}(\mathbf{o}_t) = \begin{cases} 0, \text{CDE}(Y) < \epsilon \\ 1, \text{CDE}(Y) \geq \epsilon \end{cases} . \tag{8.5}$$

Fig. 8.7 shows our detected keypoints and generated masks: **1)** While the keypoints may be sparse and imprecise, the foreground mask is clean and complete; **2)** Our unsupervisedly learned keypoints do not correspond to semantic joints of articulation, e.g., for the grasper opening a drawer, there are keypoints on both the grasper and the drawer, and our derived foreground mask contains both moving objects.

Self-supervised Visual Invariance

Our spatial attention module outputs a foreground mask, after seeing samples in the training environment. To make it adaptable to unknown test environments, we augment the clean foreground image with artificial distractors and train a model to reconstruct a mask to retrieve clean foreground observation.

Given image $\mathbf{o}_t$, we generate equally cropped clean target image I_t and noisy source image I_s.

$$I_t = \mathcal{T}_c(\mathbf{o}_t) \otimes \mathcal{T}_c(\mathcal{D}(\mathbf{o}_t)) \tag{8.6}$$

$$I_s = \mathcal{T}_f(I_t) + \mathcal{T}_b(\mathcal{T}_c(\mathbf{o}_t) \otimes (1 - \mathcal{D}(I_t))) \tag{8.7}$$

where $\mathcal{D}(I_t) = \mathcal{T}_c(\mathcal{D}(\mathbf{o}_t))$, $\mathcal{T}_c$ denotes synchronized random crop, $\mathcal{T}_f$ adds possible foreground changes such as color jitter and random brightness change, whereas $\mathcal{T}_b$ adds a set of possible background changes such as random colored boxes to the background. We learn a convolutional encoder/decoder pair to reconstruct the clean foreground mask from noisy I_s, so that they could focus more on the foreground and ignore background distractors. We impose a feature matching loss at output of encoder $\mathcal{E}$ and an image reconstruction loss at output of decoder $\hat{\mathcal{D}}$:

$$\mathcal{L}_{\text{total}} = \|\hat{\mathcal{D}}(\mathbf{I}_s) - \mathcal{D}(\mathbf{I}_t)\|_2^2 + \lambda \cdot \|\mathcal{E}(\mathbf{I}_s) - \mathcal{E}(\mathbf{I}_t)\|_2^2 \tag{8.8}$$

where $\mathcal{D}(\mathbf{I}_t)$ is simply the cropped version of $\mathcal{D}(\mathbf{o}_t)$. During RL training and deployment, for any frame I, we feed $I \otimes \hat{\mathcal{D}}(\mathcal{E}(I))$ to the learned RL policy.

What augmentations to use has a big impact on generalization. We propose four additional strong background augmentations on $\mathcal{T}_b$. **1)** The background could randomly assume the training image background, a random color, or the mean foreground color with small perturbations. **2)** Gaussian pixel-wise noise and random boxes are added. MultiColorOut, an extension to Cutout-color [349], adds multiple boxes of random sizes, colors, and positions. **3)** Darkened foreground copies are added to the background areas where the foreground mask values are 0, to simulate distractors that look similar to the foreground. **4)** We follow [342] to randomly select images in the Places dataset [350] as background images for augmentation. For fair comparisons, we list our results with and without this option. With such generic augmentations, our model is able to perform well on realistic textures and unknown testing environments even though it has not encountered them during training.

RL policy training with weak augmentations. Our visual invariance model outputs a clean foreground image with background distractors suppressed. The RL policy learner still needs to handle foreground variations in unknown test environments. We train our RL policy with weak foreground augmentations to make it robust to noise and distortions. We add the usual Gaussian random noise and use only a simple MultiColorOut to simulate the inclusion of backgrounds and missing foreground parts. Empirically we find that such weak augmentations do not affect the RL training stability.

8.4 Experiments

We experiment on two benchmarks, DeepMind and DrawerWorld, and perform ablation studies. The DeepMind Control benchmark contains various background distractions [313], [352] as in Fig. 8.8. We propose a DrawerWorld Robotic Manipulation benchmark, based on MetaWorld [353], in order to test a model's texture adaptability in manipulation tasks.

Random colors	SAC	DR	PAD	SODA+P	VAI	VAI+P	Δ
Walker, walk	414 ±74	594 ±104	468 ±47	692 ±68	819 ±11	**918 ±6**	**+226** (↑ 33%)
Walker, stand	719 ±74	715 ±96	797 ±46	893 ±12	**964 ±2**	**968 ±3**	**+75** (↑ 8%)
Cartpole, swingup	592 ±50	647 ±48	630 ±63	805 ±28	**830 ±10**	819 ±6	**+14** (↑ 2%)
Cartpole, balance	857 ±60	867 ±37	848 ±29	-	**990 ±4**	957 ±9	**+142** (↑ 17%)
Ball in cup, catch	411 ±183	470 ±252	563 ±50	949 ±19	886 ±33	**960 ±8**	**+11** (↑ 1%)
Finger, spin	626 ±163	465 ±314	803 ±72	793 ±128	932 ±3	**968 ±6**	**+165** (↑ 21%)
Finger, turn_easy	270 ±43	167 ±26	304 ±46	-	**445 ±36**	**455 ±48**	**+151** (↑ 50%)
Cheetah, run	154 ±41	145 ±29	159 ±28	-	**337 ±1**	**334 ±2**	**+178** (↑ 112%)
Reacher, easy	163 ±45	105 ±37	214 ±44	-	**934 ±22**	**936 ±19**	**+722** (↑ 337%)
average	*467*	*464*	*531*	-	*793*	***812***	***+281*** *(↑ 53%)*

Table 8.1: VAI outperforms existing methods on DeepMind randomized color tests by a large margin *without* using the external Places dataset; it is even better than SODA+P, which uses Places as a part of the training set. Soft Actor-Critic (SAC) [299], [351] is used as a base algorithm for DR (domain randomization), PAD [313], SODA [342], and our VAI. SODA+P and VAI+P use Places [350] as overlay or adapter augmentation. The results of SAC and DR are copied from PAD [313]. Listed are the mean and std of cumulative rewards across 10 random seeds and 100 random episode initializations per seed. The absolute and relative improvement of VAI over SOTA method are listed in the Δ column.

DeepMind Control Benchmark

Tasks. There are walking, standing, and reaching objects [352], all in 3D simulation. Our agent receives pixel-based inputs instead of state-based inputs from the underlying dynamics unless otherwise stated.

Testing. We follow PAD [313] and test our method under three types of environments: 1) randomized colors; 2) video backgrounds; and 3) distracting objects. For tasks with video background and distracting objects without Places Augmentation, we apply a moving average de-noising trick by subtracting a moving average of the past observations from the current observation and adding back the mean color of moving average. We also introduce a constant factor α multiplied to past moving average to tune its aggressiveness.

Training. For each scenario, we train agents without distractions and evaluate the model across 10 random seeds and 100 random environment initializations. To get observation samples for training, we export 5000 transitions from the replay buffer for the training environment, which are collected with a random policy. We use the same environment settings such as frame skip and data augmentation as in PAD to ensure fair comparisons between VAI, PAD, and others.

Randomized color results. Table 8.1 shows that our VAI outperforms published SOTA on all

Video background	SAC	DR	PAD	SODA	SODA+P	VAI	VAI+P	Δ
Walker, walk	616 ±80	655 ±55	717 ±79	635 ±48	768 ±38	870 ±21	**917 ±8**	**+149** (↑ 19%)
Walker, stand	899 ±53	869 ±60	935 ±20	903 ±56	955 ±13	**966 ±4**	**968 ±2**	**+13** (↑ 1%)
Cartpole, swingup	375 ±90	485 ±67	521 ±76	474 ±143	**758 ±62**	624 ±146	**761 ±127**	**+3** (↑ 0%)
Cartpole, balance	693 ±109	766 ±92	687 ±58	-	-	**869 ±189**	847 ±205	**+182** (↑ 26%)
Ball in cup, catch	393 ±175	271 ±189	436 ±55	539 ±111	**875 ±56**	790 ±249	846 ±229	**-29** (↓ 3%)
Finger, spin	447 ±102	338 ±207	691 ±80	363 ±185	695 ±97	569 ±366	**953 ±28**	**+258** (↑ 37%)
Finger, turn_easy	355 ±108	223 ±91	362 ±101	-	-	419 ±50	**442 ±33**	**+80** (↑ 22%)
Cheetah, run	194 ±30	150 ±34	206 ±34	-	-	**322 ±35**	**325 ±31**	**+119** (↑ 58%)
average	*497*	*470*	*569*	-	-	*678*	***757***	***+188*** *(↑ 33%)*

Table 8.2: VAI+P (VAI) outperforms PAD by more than 33% (19%) on challenging DeepMind video backgrounds. Same settings and conventions as Table 8.1.

Distracting objects	SAC	DR	PAD	VAI	Δ
Cartpole, swingup	815 ±60	809 ±24	771 ±64	**891 ±0**	**+120** (↑ 16%)
Cartpole, balance	969 ±20	938 ±35	960 ±29	**993 ±0**	**+24** (↑ 2%)
Ball in cup, catch	177 ±111	331 ±189	545 ±173	**956 ±4**	**+411** (↑ 75%)
Finger, spin	652 ±184	564 ±288	**867 ±72**	805 ±3	**-62** (↓ 7%)
Finger, turn_easy	302 ±68	165 ±12	347 ±48	**389 ±18**	**+42** (↑ 12%)
average	*583*	*561*	*698*	***806***	***+108*** *(↑ 15%)*

Table 8.3: VAI outperforms current SOTAs by more than 15% on DeepMind Control distracting objects. Although VAI performs worse than PAD on "Finger, spin" task in terms of mean rewards, the reward variance is greatly reduced from 72 to 3 in std. Same settings and conventions as Table 8.1.

the 9 tasks by up-to an astonishing 337% margin in terms of mean cumulative rewards, without seeing samples in the test environment at any time. In contrast, DR is trained with color change to the environment (which requires knowing and changing the internals of the environment), which, to some extent, previews what the test environment would be. Similarly, although PAD does not use any evaluation samples during training, it does use the samples at the test time to tune the encoder. Since VAI does not change model weights, it has no adaptation delay, better stability, and less compute (see more details in supplementary materials). By suppressing distractions and feeding only the foreground image, the RL algorithm ideally sees the same input no matter what

success %	DrawerOpen				DrawerClose			
	SAC	PAD	VAI	Δ	SAC	PAD	VAI	Δ
Grid	98 ±2	84 ±7	**100 ±0**	**+2** (↑ 2%)	**100 ±0**	95 ±3	99 ±1	**-1** (↓ 1%)
Black	95 ±2	95 ±3	**100 ±1**	**+5** (↑ 5%)	75 ±4	64 ±9	**100 ±0**	**+25** (↑ 33%)
Blanket	28 ±8	54 ±6	**86 ±6**	**+32** (↑ 59%)	0 ±0	0 ±0	**85 ±8**	**+85** (↑ ∞%)
Fabric	2 ±1	20 ±6	**99 ±1**	**+79** (↑ 395%)	0 ±0	0 ±0	**74 ±8**	**+74** (↑ ∞%)
Metal	35 ±7	81 ±3	**98 ±2**	**+17** (↑ 21%)	0 ±0	2 ±2	**98 ±3**	**+96** (↑ 4800%)
Marble	3 ±1	3 ±1	**43 ±7**	**+40** (↑ 1333%)	0 ±0	0 ±0	**49 ±13**	**+49** (↑ ∞%)
Wood	18 ±5	39 ±9	**94 ±4**	**+55** (↑ 141%)	0 ±0	12 ±2	**70 ±6**	**+58** (↑ 483%)
average	*40*	*54*	***87***	***+33*** *(↑ 61%)*	*25*	*25*	***82***	***+57*** *(↑ 228%)*

Table 8.4: Our VAI consistently outperforms all the baselines in new texture environments, and on DrawerClose in particular, VAI succeeds 82% vs. SAC/PAD's 25%. Grid is the training environment. Black means a completely dark background without texture. Other textures are shown in Fig. 8.9. DrawerClose is more challenging than DrawerOpen, as the drawer handle is concealed by the effector in DrawerClose, which would require the agent to infer the handle position from the position and the size of the effector. The success rate is the percentage of successful attempts out of 100 attempts to open or close a drawer. The mean/std are collected over 10 seeds.

the environment is and is thus not influenced by background distractions or domain shifts in the test environment.

Video background results. Table 8.2 shows that our VAI outperforms baselines in 7 out of 8 tasks in terms of mean cumulative rewards, often by a large margin.

Distracting object results. Table 8.3 shows that our VAI surpasses baselines on 4 out of 5 tasks in terms of cumulative rewards. It not only obtains nearly full scores on "Cartpole, balance" and "Ball in cup, catch" tasks, but also greatly decreases the variance of results to a negligible level.

DrawerWorld Manipulation Benchmark

A New Texture Benchmark for Manipulation. CNNs are sensitive to textures [354]. We propose to evaluate a model's texture adaptability in manipulation tasks, based on the MetaWorld [353] benchmark for meta RL and multi-task RL.

In the original MetaWorld, the observations include 3D Cartesian positions of the robot, the object, and the goal positions collected with sensors on the object and the robot. Accurate object positions and robot keypoints are hard to get by in real-world applications, we thus propose a variant of MetaWorld, *DrawerWorld*, with visual observations instead. We focus on the variety of *realistic textures* (Fig. 8.9).

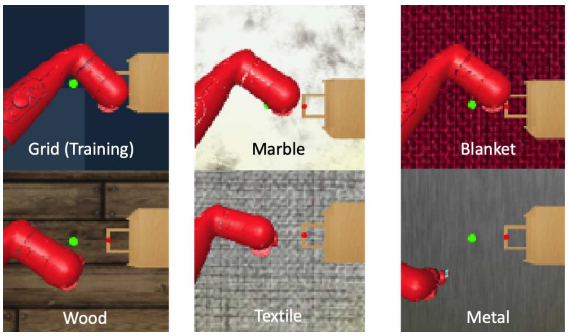


Figure 8.9: DrawerWorld environments. The grid texture is used during training, while the other five evaluation textures are realistic photos—making the task significantly more challenging.

Tasks. There are DrawerOpen and DrawerClose tasks, where a Sawyer arm is manipulated to open and close a drawer. The action space contains the end-effector positions in 3D. We adopt MetaWorld reward functions and success metrics. See supplementary materials for details.
Testing. We test the agent on surfaces of different textures which, unlike the grid texture used for training, come from photos instead of from simulations. These tasks are extremely challenging for two reasons: 1) The agent has never seen any realistic textures during training; 2) Each texture also has a different color, so the agent needs to handle both color change and texture change at the same time.
Texture background results. Table 8.4 shows that our VAI outperforms PAD [313] and SAC [351] significantly in all the test environments. In particular, for 5 out of 6 textures such as blanket, metal, and wood, SAC and PAD have 0% success rate, whereas VAI performs far better at 85%, 98%, and 70% respectively. In the training grid environment, PAD performs worse than SAC, consistent with [313] on the DeepMind Control benchmark, whereas our VAI is on-par or slightly better than SAC, suggesting that we are not gaining texture adaptability at the cost of losing training performance.

CNNs' sensitivity to textures poses a big challenge for visual adaptation. **1)** SAC adapts to unknown test environments with augmentations at the training time. Since textures are not used during training, SAC breaks down during texture testing. **2)** PAD has to change its feature encoder a lot in order to adapt to never-seen textures at the time time, shifting the feature distribution. However, PAD assumes an invariant feature distribution and, therefore, does not fine-tune the control part of the policy network at the test time, which causes the vision-RL pipeline to break down.

Ablation Studies

We evaluate four ablated variants of our methods on the DrawerOpen task:
1. SAC, a base universal policy learning model
2. Method 1 + RL with image augmentations, equivalent to Domain Randomization;
3. Method 2 + Visual invariance module trained without augmentations: $\mathcal{T}_f, \mathcal{T}_b$ are identity functions;
4. Method 3 + We apply the augmentations in Section 8.3 on $\mathcal{T}_f, \mathcal{T}_b$, for greater adaptability.

success rate (%)	Grid	Wood	Metal	Fabric
SAC	98 ±2	18 ±5	35 ±7	2 ±1
+ RL Augmentation	**100 ±1**	18 ±5	41 ±8	24 ±5
+ Foreground Extraction	**100 ±0**	18 ±4	13 ±4	38 ±4
+ Background Augmentation	**100 ±0**	**94 ±4**	**98 ±2**	**99 ±1**

Table 8.5: Ablation studies for augmentation and foreground extraction on DrawOpen task. From top to bottom rows, components are added to the method cumulatively. Each method is trained in the grid environment and tested in new texture enviroments of wood, metal, and fabric. Success rates are collected over 500K steps. Only the last method with all augmentations deliver consistent robustness.

Table 8.5 shows that while all the methods perform well in the training environment, they adapt poorly to realistic textures except the last one. These results suggest that adding visual augmentations during RL or to the entire image as a whole is insufficient; providing a clean observation for RL agents with foreground clues adds significant robustness to vision-based RL.

8.5 Appendix Materials

Additional Environment Descriptions

DeepMind Control

We wrote a short description for each environment in DeepMind Control suite [352] in Table 8.6 to further introduce the environment.

Environment	Descriptions
Walker	A planar walker which encourages an upright torso and minimal torso height in the "stand" task. In "walk" task forward velocity is also encouraged.
Cartpole	A pole tied to a cart at its base, with forces applied to the base. "swingup" task requires the pole to swing up from pointing down while "balance" task requires the pole to balance to be upright.
Ball in cup	A ball attached to a cup, with forces applied to the cup to swing the ball up into the cup in the "catch" task.
Finger	A finger is asked to rotate a rectangular body on a hinge. The top of the body needs to overlap with the object in "turn_easy" task and the body needs to rotate continuously in the "spin" task.
Cheetah	An animal with two feet which is asked to run in the "run" task.
Reacher	A planar reacher with two links connected with a hinge in a plane with a random target location. In the "easy" task, the reacher is asked to reach the object location. The "hard" task is unused in our evaluation since it was not adapted by [313].

Table 8.6: Descriptions for each environment in DeepMind Control suite.

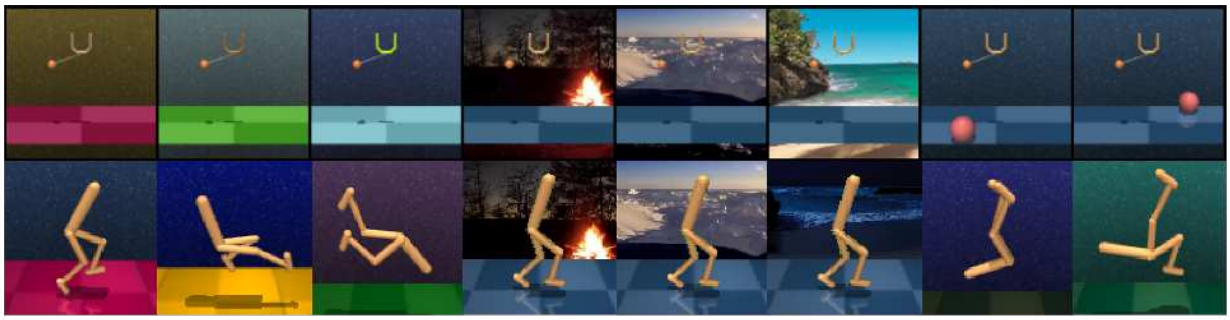


Figure 8.10: Samples in evaluation environments in DeepMind Control. The samples in the first row are from [313].

We also provide samples for the evaluation environments designed by [313] in Fig. 8.10.

DrawerWorld

We propose the DrawerWorld, a benchmark with observations in pixels, based on MetaWorld [353] to enable the agent to work in an environment close to real-life scenarios. There are two tasks in DrawerWorld, which are DrawerOpen and DrawerClose. These tasks ask a Sawyer robot to open and close a drawer, respectively.

The multi-component reward function R is a combination of a reaching reward R_{reach} and a push reward R_{push} as follows:

$$\begin{aligned} R &= R_{\text{reach}} + R_{\text{push}} \\ &= -||h-p||_2 + \mathbb{I}_{||h-p||_2<\epsilon} \cdot c_1 \cdot \exp\{||p-g||_2^2/c_2\} \end{aligned} \tag{8.9}$$

where ϵ is a small distance threshold and is set as 0.08 by default, $p \in \mathbb{R}^3$ be the object position, $h \in \mathbb{R}^3$ be the position of the robot's gripper, and $g \in \mathbb{R}^3$ be goal position. $c_1 = 1000$ and $c_2 = 0.01$ for all tasks in DrawerWorld benchmark.

The goal of distraction-robust RL is to learn a task-conditioned policy $\pi(a|s,z)$, where z indicates an encoding of the task ID, and in this case, different task IDs have different drawer positions. This policy should maximize the average expected return from the task distribution $p(\mathcal{T})$, given by $\mathbb{E}_{\mathcal{T}\sim p(\mathcal{T})}[\mathbb{E}_{\pi}[\sum_{t=0}^{T}\gamma^t R_t(s_t,a_t)]]$. The success metric, which is evaluate the agent in evaluation time, is described by $\mathbb{I}_{||p-g||_2<\epsilon}$, where ϵ is set to 8cm. The difference between training and evaluation time is the texture and color of the table cloth. Image samples of textures that we use are provided in the main text.

De-noise with Past Averages

Since our adapter model works on each frame separately without any assumption on temporal continuity of consecutive frames, our adapter works exactly the same on videos as on fixed backgrounds and is not affected by drastic changes in the background such as flashes of light. However,

in some environments where the assumption of temporal continuity holds, i.e. with a relatively slow-moving background, we may make use of this assumption to better de-noise the observations before passing the them into the adapter.

We exploit the assumption here by keeping a mean of past observations $\mathbf{o}_{\text{mean}} = \frac{1}{t}\sum_{i=1}^{t}\mathbf{o}_i$ and subtract the mean from observation $\mathbf{o}_t$ and compute the observations after de-noise with the formulation: This de-noise step happens before the observation is sent into CNN (adaptor), formulated as:

$$\mathbf{o}_{\text{de-noised}} = \text{filter}(\mathbf{o}_t - \alpha\mathbf{o}_{\text{mean}}, \epsilon) + \alpha\mathbf{o}_{\text{mean\_color}} \tag{8.10}$$

where $\mathbf{o}_{\text{mean\_color}}$ is the mean of $\mathbf{o}_{\text{mean}}$ in spatial dimensions, filter is a function that sets the part with value less than ϵ to 0 to remove some noise, and $\alpha \in [0, 1]$ is the strength in noise removal.

This is completely optional, and since it makes use of an additional assumption, they are only used in the cartpole and ball in cup in experiments with background videos and experiments with distracting objects. In addition, we observe that with Places dataset as augmentation, the model is robust enough without this trick, so we disable it for all models in the training of which Places dataset is used.

Memory Usage and Speed Comparisons

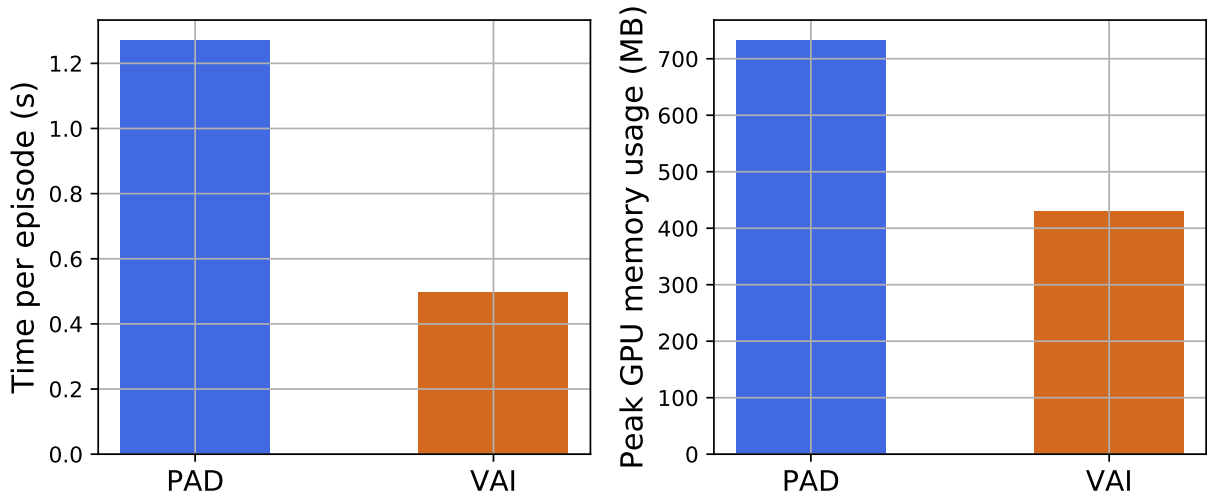


Figure 8.11: Comparisons on the mean time per episode and GPU memory occupancy at evaluation time for *DrawerClose* task in DrawerWorld between current state-of-the-art method PAD [313] and the proposed method VAI. VAI is more than 2 times faster than PAD during testing time and requires ∼40% less GPU memory usage. Both methods are evaluated with exactly the same backbone network. We take the mean of 10 runs for the latency comparison. Memory usage is obtained with `torch.cuda.max_memory_allocated`.

From Fig. 8.11, it seems that VAI is about 3 times as fast as PAD in terms of the evaluation time in each episode and requires substantially less GPU memory than PAD. This is largely due to

the fact that PAD trains the encoder network at evaluation time with back-propagation, which not only requires the intermediate results to be saved in GPU memory but also requires backward computation to update the model parameters, which consumes both time and memory space. Although VAI has an extra adapter module, the computation and memory it takes are much less than the ones required by backward computation and storing intermediate results. According to the requirements of computational resources in terms of speed and memory, our method is more suitable for robots powered by battery and edge inference devices than PAD from this point of view.

RL Observations	Cumulative Reward
Joint Positions, Velocity, Torso Height from the Environment	969±2
Joint Positions from the Environment	935±3
Keypoints Extracted with KeyNet	709±3
VAI on Training Environment	889±3

Table 8.7: Cumulative rewards on *Walker, walk* task with 1) joint positions, velocity, and torso height from the environment as observations; 2) joint positions from the environment as observations; 3) keypoints extracted by KeyNet from images; 4) The proposed method VAI. The first two use the ground truth information, which is not accessible during real-world deployment, and serve as upper bounds. For experiment 2, 3, and 4, we use stack of 3 frames as input for the RL agent to infer the velocity since velocity information is missing. Since walker is a planar environment (the walker will not lean towards to away from the screen), the extracted keypoints should roughly correspond to positions from the significant parts of the walker body. The gap between experiments indicate that a limited number of keypoints from KeyNet on its own is not a sufficiently informative or accurate source for observations for an RL agent, which is in accord with our visualization in the main text about the keypoints' temporal inconsistency.

Further Experiments with Raw Keypoints

To investigate the question of whether raw keypoints extracted from image observations by KeyNet are able to contribute to effective learning of useful behaviors, we set up experiments based on the *Walker, walk* task in DeepMind Control and list the outcomes in Table 8.7.

We first train an agent with the state-based observation provided by the environment, a 24-dimensional tensor, which includes positions of the joint, velocity, and walker's torso height. We do not stack frames for this experiment. This experiment indicates an upper bound that our agent is able to achieve in this environment. However, to make a fair comparison with other experiments, where velocity and torso height information is not directly provided, we also remove these parts from our observation, leaving only the positions of the joint as observation in the second experiment. Thus, in the second experiment, the agent directly reads a 14-dimensional tensor per frame from the environment based on the position of each joint. The third experiment is conducted with the RL agent reading a tensor that contains the (x, y) coordinates of 24 keypoints. The keypoints are from a KeyNet which reads an image input. The KeyNet is pre-trained with transporter. In the last experiment, we run VAI, with the adapter module trained from the same KeyNet that is used

in the experiment above, on the training environment, although it is able to adapt to other environments as well and thus is more general. To infer velocity information, we stack the observations for three frames for experiments 2, 3, and 4. We run both experiments for 500k steps. We compare their efficiency by evaluating the agent in training environment 10 times with 10 seeds.

According to the performance of these RL agents in the training environment, keypoints on its own do not capture all the information needed by the RL agent accurately. This will be even worse if the agent is evaluated in a different environment is has never seen before, since KeyNet itself does not come with the ability to adapt, although the keypoints it generates are not supposed to carry domain-specific or distraction information. Using keypoints information along with image features as well as history observations may help, as illustrated in [316] and described in the main text, but it will add greatly to the complexity of the RL framework. What's more, agents may need information other than what keypoints provide. For example, keypoints do not carry the shape, size, and color information, which may be of paramount importance in certain tasks. Furthermore, since KeyNet allocates an output dimension for each keypoint, the number of parameters as well as computation time scales linearly with the number of keypoints, which prohibits adding a large number of keypoints to compensate the effect of temporal inconsistency or to capture complicated observations. In contrast, since KeyNet is not used in getting adapted observations in our method, the speed and number of parameters of our RL agent, including the adapter, at evaluation time are not affected by the number of keypoints used to generate ground-truth, which allows our method to scale to complicated environments with many moving parts without losing efficiency.

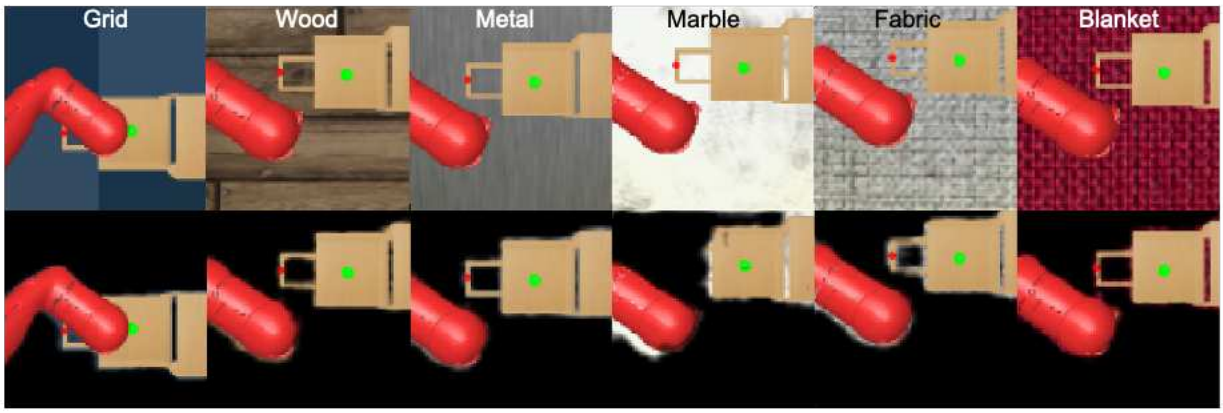


Figure 8.12: Samples from DrawerWorld, *DrawerClose* task with their corresponding observation processed by the adapter module. The Grid task is the training task for the adapter. All the observations use the same adapter for a fair comparison.

Visualizations of the Observation Adapter

How to make sure that RL will adapt to a certain setting that is different from training setting is still an open problem. Our method opts to work on observation-space. In contrast, PAD works on

an intermediate encoder feature space. Our method is much easier to visualize and debug since humans are able to directly understand the quality of adapted observations while it is really difficult to understand what happens in the feature space.

To give examples on how to assess whether an adapter works on a certain environment easily and to illustrate our performance in a visual way in evaluation environments, we gathered 6 pairs of raw samples and samples processed by the adapter in *DrawerClose* task from the same adapter in Fig. 8.12. As can be seen from the examples, the adapter model differentiates most of the evaluation environments well, with the exception of the marble environment, which the adapter confuses parts of the foreground and background such as the handle and the patches around the actuator, probably due to the fact that the reflected light on the actuator has a similar white color to the color of background. This indicates why our model performances worse in marble environment, as illustrated in the experiment section in the main text, and, in real-life applications, means that the adapter needs to be re-trained or fine-tuned with observations from similar environments, or if this is not applicable, with augmentation specially-designed to handle this case. We leave the question of handling adapter fine-tuning and re-training to later research.

This visualization has a large impact on the real-world applications of our method: with only a few observations from an intended deployment environment, one could easily visualize and assess whether our method will adapt to such environment. This does not require any ability to run the policy in the dynamics, nor does it require reward functions or consecutive observations which may be difficult to obtain from deployment environments in real-world applications. We strongly believe that this simple assessment provides a direction for future research in explainable, adaptable, and generalizable reinforcement learning and will present great benefit to potential applications of reinforcement learning.

8.6 Summary

We propose a fully unsupervised method to make vision-based RL more generalizable to unknown test environments. While existing methods focus on learning a universal policy, we focus on learning universal foreground vision.

We learn to extract foregrounds with unsupervised keypoint detection, followed by unsupervised visual attention to remove model bias and generate a foreground mask. We then train a model to reconstruct the clean foreground mask from noise-augmented observations.

We propose an additional challenging DrawerWorld benchmark, which trains manipulation tasks on grid and tests on texture environments. Existing methods fail due to CNN's sensitivity to textures, yet our model with foreground extraction and strong generic augmentation is robust to never-seen textures without sacrificing training performance.

Our method significantly advances the state-of-the-art in vision-based RL, demonstrating that it is not only possible to learn domain-invariant vision without supervision, but freeing RL from visual distractions also improves the policy.

Acknowledgments. This work was supported, in part, by Berkeley Deep Drive.

Chapter 9

Debiased Learning from Naturally Imbalanced Pseudo-Labels

Pseudo-labels are confident predictions made on unlabeled target data by a classifier trained on labeled source data. They are widely used for adapting a model to unlabeled data, e.g., in a semi-supervised learning setting.

Our key insight is that pseudo-labels are naturally imbalanced due to intrinsic data similarity, even when a model is trained on balanced source data and evaluated on balanced target data. If we address this previously unknown imbalanced classification problem arising from pseudo-labels instead of ground-truth training labels, we could remove model biases towards false majorities created by pseudo-labels.

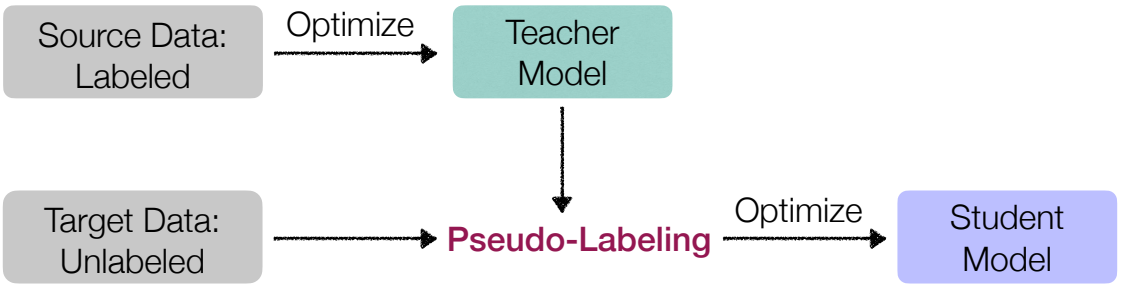
We propose a novel and effective debiased learning method with pseudo-labels, based on counterfactual reasoning and adaptive margins: The former removes the classifier response bias, whereas the latter adjusts the margin of each class according to the imbalance of pseudo-labels. Validated by extensive experimentation, our simple debiased learning delivers significant accuracy gains over the state-of-the-art on ImageNet-1K: 26% for semi-supervised learning with 0.2% annotations and 9% for zero-shot learning.

9.1 Introduction

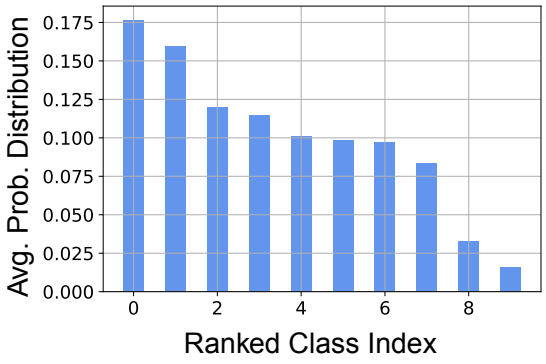
Real-world observations, as well as non-curated datasets, are naturally long-tail distributed [61], [356]. Imbalanced classification [22], [357], [358] tackles such data biases to prevent models from being dominated by head-class instances. Developing visual recognition systems capable of counteracting biases also has significant social impacts [359].

While existing methods focus on debiasing from imbalanced ground-truth labels collected by human annotators, we discover that pseudo-labels produced by machine learning models are naturally imbalanced, creating another source for widespread biased learning!

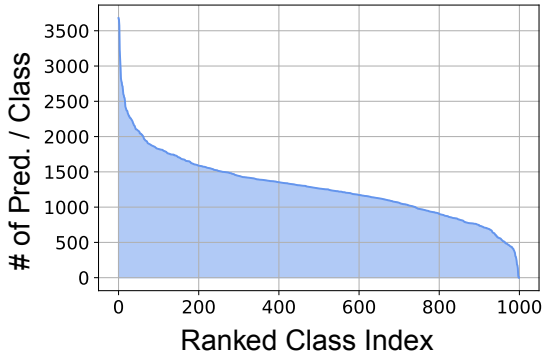
Pseudo-labels are highly confident predictions made by an existing (teacher) model on unlabeled data, which then become part of the training data for supervising the (student) model



(a) Framework of pseudo-labeling



(b) FixMatch on CIFAR10 SSL



(c) CLIP on ImageNet ZSL

Figure 9.1: We study the pseudo-labeling-based Semi-Supervised Learning (SSL) and transductive Zero-Shot Learning (ZSL), where both tasks require transferring semantic information learned from labeled source data to unlabeled target data via pseudo-labeling. Surprisingly, we find that pseudo-labels of target data produced by typical SSL and ZSL methods (i.e., FixMatch [355] and CLIP [17]) are highly biased, even when both source and target data are class-balanced or even sampled from the same domain.

adaptation to unlabeled data (Fig. 9.1a). When the student model is the teacher model itself, the learning process is also known as *self-training* [355], [360]–[363]. Pseudo-labeling is widely used in semi-supervised learning (SSL) [355], [364], domain adaptation [365], [366], and transfer learning [367].

We examine pseudo-label distributions in two common tasks. **1)** In zero-shot transfer learning (ZSL) where the source and target domains are different, a pretrained CLIP model [17] produces highly imbalanced predictions on the curated and balanced ImageNet-1K dataset, although the training set of CLIP is approximately balanced (Fig. 9.1c). More than 3500 instances are predicted as class 0, 3 times the actual number of samples in class 0. **2)** In semi-supervised learning where the source and target domains are the same, FixMatch [355] trained on labeled CIFAR10 images generates highly biased pseudo-labels on unlabeled images, although both the labeled and unlabeled sets are balanced (Fig. 9.1b).

That is, pseudo-labels created by machines are naturally imbalanced, just like ground-truth labels created by humans. If we address this previously unknown imbalanced classification problem arising from pseudo-labels instead of ground-truth training labels, we could improve model

learning based on pseudo-labels and remove the model bias towards false majorities created by pseudo-labels.

We propose a novel and effective debiased learning method with pseudo-labels, without any knowledge about the distribution of actual classification margins that are readily available to debiased learning with ground-truth labels [368]–[370]. It consists of an adaptive debiasing module and an adaptive marginal loss. The former dynamically removes the classifier response bias through counterfactual reasoning, whereas the latter dynamically adjusts the margin of each class according to the imbalance of pseudo-labels.

Validated by our extensive experiments, our simple debiased learning not only improves the state-of-the-art on ImageNet-1K by 26% for SSL with 0.2% annotations and 9% for ZSL, but is also a universal add-on to various pseudo-labeling methods with more robustness to domain shift. The imbalanced pseudo-labeling issue is even more severe when the unlabeled raw data is naturally imbalanced, and the model tends to mislabel tail-class samples as head-class. By applying debiased learning, we improve SSL performance under long-tailed settings by a large margin.

Our work makes four major contributions. **1)** We systematically investigate and discover that pseudo-labels are naturally imbalanced and create biased learning. **2)** We propose a simple debiased learning method with pseudo-labeled instances, requiring no knowledge of their actual classification margins. **3)** We improve the ZSL/SSL state-of-the-art by a large margin and demonstrate that our debiasing is a universal add-on to various pseudo-labeling models. **4)** We establish a new effective ZSL/SSL pipeline for applying vision-and-language pre-trained models such as CLIP.

9.2 Related Work

Semi-Supervised Learning integrates unlabeled data into training a model given limited labeled data. There are four lines of approaches. **1)** Consistency-based regularization methods impose classification invariance loss on unlabeled data upon perturbations [371]–[374]. **2)** Pseudo-labeling expands model training data from labeled data to additional unlabeled but confidently pseudo-labeled data [355], [360]–[363], [375]. **3)** Transfer learning trains the model first on large unlabeled data through self-supervised representation learning, e.g., contrastive learning, and then on small labeled data through supervised classifier learning [376], [377]. **4)** Data-centric SSL assumes that labeled data are not given but can be optimally selected among unlabeled data for labeling [378]. Focusing on this practical issue of labeled data selection turns out to bring substantial gains for SSL.

CReST [379] improves existing SSL methods on class-imbalanced data by leveraging a class-rebalanced sampler, which samples more frequently for the minority class according to the labeled data distribution. CReST does not work when the labeled data is balanced. In contrast, our approach does not assume any prior distribution for the labeled set.

Although previous literature has achieved tremendous success in SSL, the implicitly biased pseudo-labeling issue in SSL is previously unknown and has not been thoroughly analyzed, which, however, has a great impact on the learning efficiency. The focus of this work is on proposing a simple yet effective debiasing module to eliminate this critical issue.

Zero-shot Classification refers to the problem setting where a zero-shot model classifies images from novel classes into correct categories that the model has not seen during training [380]–[382]. Several strategies have been considered from various sets of viewpoints: **1)** hand-engineered attributes [383], [384]; **2)** pretrained embeddings that incorporate prior knowledge in form of semantic descriptions of classes [385], [386]; **3)** modeling relations between seen and unseen classes with knowledge graphs [387], [388]; **4)** learning generic visual concepts with vision-language models, allowing zero-shot transfer of the model to a variety of downstream classification tasks [2], [17].

Long-Tailed Recognition (LTR) aims to learn accurate "few-shot" models for classes with a few instances, without sacrificing the performance on "many-shot" classes, for which many instances are available. **1)** re-balancing/re-weighting method τ-norm [358] tackles LTR problem by giving more importance to tail classes; **2)** margin-based method LDAM [357] proposes a label-distribution-aware margin loss to improve the generalization of minority classes by encouraging larger margins for tail classes; **3)** post-hoc adjustment approach modifies a trained model's predictions according to the prior knowledge of class distribution, such as LA [389], or pursues the direct causal effect by removing the paradoxical effects of the momentum, such as Causal Norm [348]; **4)** ensemble-based approach RIDE [22] optimizes multiple diversified experts and a dynamic expert routing module to reduce model bias and variance on long-tailed data.

In stark contrast to previous works on LTR which either requires the prior knowledge of class distribution or are applied post-hoc to a trained model, the proposed debias module does not require any prior knowledge and focuses on the biased pseudo-labels issue which is previously unknown.

9.3 Pseudo-Labels are Naturally Imbalanced

In contrast to previous work that concentrated on biases caused by trained on *imbalanced* data, our focus is on pseudo-label biases, even when trained on *balanced* data. In this section, we provide an analysis of this previously unknown issue hidden behind the tremendous success of FixMatch [355] on SSL and CLIP [17] on ZSL, both of which require the use of "pseudo-labeling" to transfer knowledge learned in source data to target data.

We first describe the backgrounds for pseudo-labeling approaches and then analyze their bias issue. We attribute the cause of bias to the inter-class correlation problem.

Background

FixMatch for semi-supervised learning. The core technique of FixMatch [355] is pseudo-labeling [360]. It selects unlabeled samples with high confidence as training targets.

Suppose we have a labeled dataset $X_L = \{(x_i, y_i)\}_{i=1}^{L}$ with L labeled instances, and an unlabeled dataset $X_U = \{(x_i)\}_{i=L+1}^{L+U}$ with U instances. x_i is the input instance and $y_i = [y_i^1, ..., y_i^C] \subseteq \{0, 1\}^C$ is a discrete annotated target with C classes. X_U and X_L share the same semantic labels. The optimization objective consists of two terms: $\mathcal{L} = \mathcal{L}_\mathrm{s} + \lambda_u \mathcal{L}_\mathrm{u}$, i.e., the supervised loss $\mathcal{L}_s$ applied to labeled data and an unsupervised loss $\mathcal{L}_u$ applied to unlabeled data, and λ_u is a scalar hyperparameter.

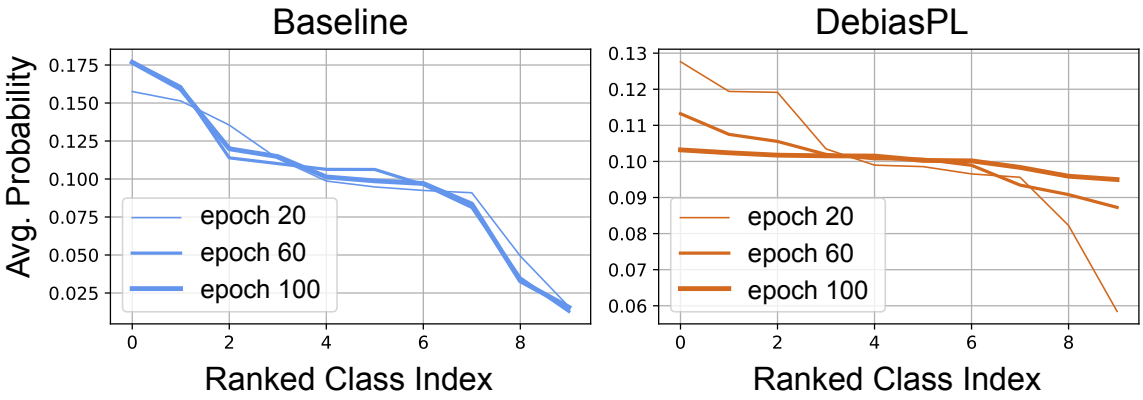


Figure 9.2: FixMatch's pseudo-labels are highly imbalanced across different training stages, even though the unlabeled and labeled data it trains on is class-balanced. In contrast, DebiasPL produces nearly balanced pseudo-labels at late stages. The probability distributions of FixMatch and DebiasPL are averaged over all unlabeled data. The class indices are sorted by average probability. We conduct experiments on CIFAR10 with 4 labeled instances per class.

The supervised loss $\mathcal{L}_s$ is the cross-entropy between the model predictions and the ground truth: $\mathcal{L}_{\mathrm{s}} = \frac{1}{B}\sum_{i=1}^{B} \mathrm{H}(y_i, p(\alpha(x_i)))$, where α is the weak augmentation, and B is the batch size. The pseudo-labels $\hat{y}_i$ for unlabeled instances are generated from the weakly-augmented unlabeled samples, which are used to supervise the model prediction of the strongly-augmented samples. Instances whose largest probability fall under a confidence threshold τ are regarded as unreliable samples and discarded. Formally, the unsupervised loss $\mathcal{L}_u$ can be formulated as:

$$\mathcal{L}_{\mathrm{u}} = \frac{1}{\mu B}\sum_{i=1}^{\mu B} \mathbb{1}[\max(p(\alpha(x_i))) \geq \tau] \cdot \mathrm{H}(\hat{y}_i, p(\beta(x_i))) \tag{9.1}$$

where β is a strong augmentation [390], and μ determines the ratio of labeled and unlabeled samples in the minibatch.

CLIP for zero-shot learning. CLIP [17] is an efficient and scalable way to learn image representations from scratch on a dataset of 400M image-text pairs, which is manually curated to be approximately query-balanced. At pre-training time, an image encoder and a text encoder are optimized by maximizing (minimizing) the similarity between paired (unpaired) captions and visual images.

For producing pseudo-labels of unlabeled data, natural language prompting is used to enable zero-shot transfer to target datasets: CLIP uses the names or descriptions of the target dataset's classes as the set of potential text pairings (e.g. "a photo of a dog") and predicts the most probable class according to the cosine similarity of image-text pairs. Specifically, the feature embedding of the image and the feature embedding of the set of possible texts are first computed by their respective encoders. The cosine similarity of these embeddings is then evaluated, and normalized into a probability distribution via a softmax function.

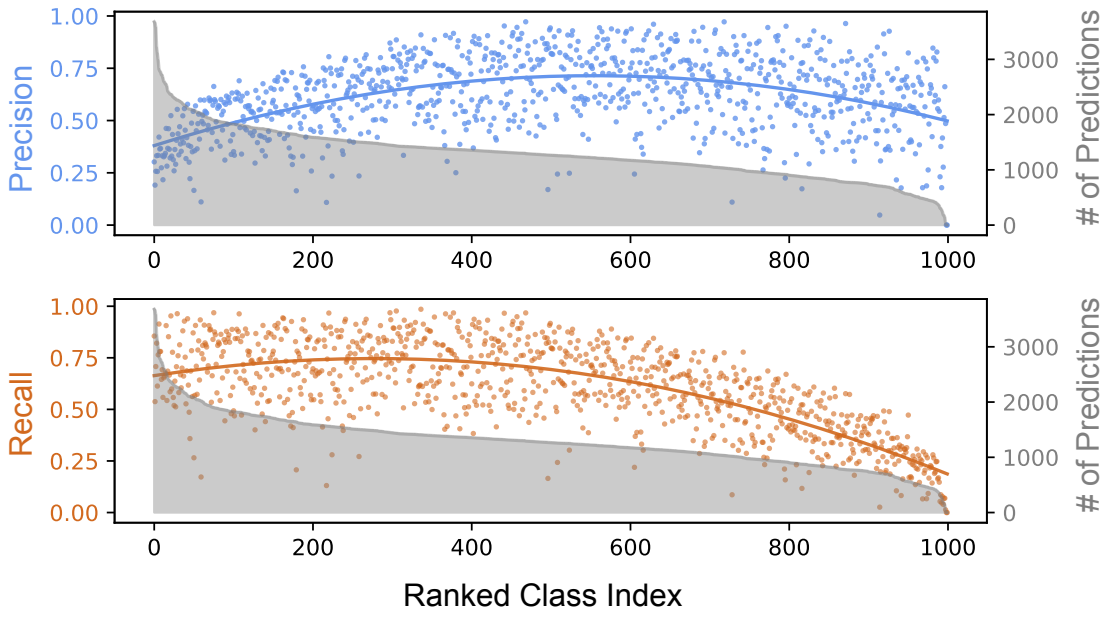


Figure 9.3: Per-class precision and recall of pseudo-label predictions on 1.3M ImageNet instances with a pre-trained CLIP. The majority classes with high recall often have less precise pseudo-labels.

Biases in Semi-supervised Learning

Fig. 9.2 visualizes the FixMatch probability distributions averaged on all unlabeled data at various training epochs. Surprisingly, even when labeled and unlabeled data are both curated (class-balanced), the pseudo-labels are still highly class-imbalanced, most notably at the early training stage. As the training progresses, this situation persists.

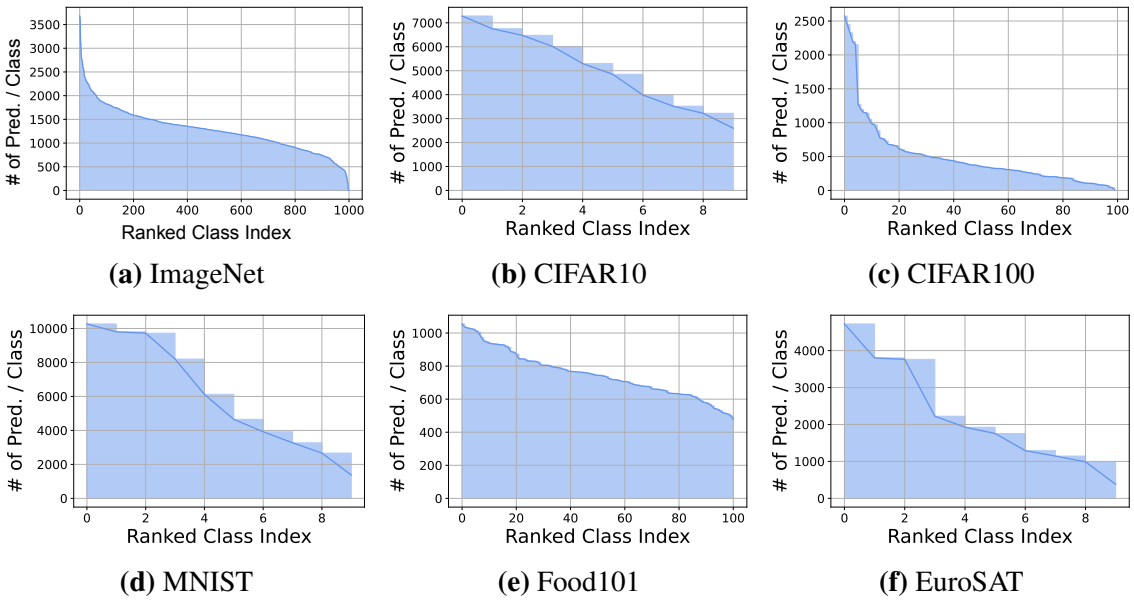
A student model will inherit the implicitly imbalanced pseudo-labels and, in turn, reinforces the teacher model's biases. Once confusing samples are wrongly pseudo-labeled, the mistake is almost impossible to be self-corrected. On the contrary, it may even mislead the model and further amplify existing bias to produce more wrong predictions. Without intervention, the model will get trapped in irreparable biases.

On the contrary, as in Fig. 9.2, although DebiasPL is also troubled by the imbalanced pseudo-labels at the beginning, this situation can be significantly alleviated, and, eventually, we can obtain an almost balanced distribution through dynamically debiasing the model.

Biases in Zero-Shot Learning

CLIP actually generates highly biased predictions on ImageNet, which is hidden behind CLIP's tremendous success in terms of overall zero-shot prediction accuracy.

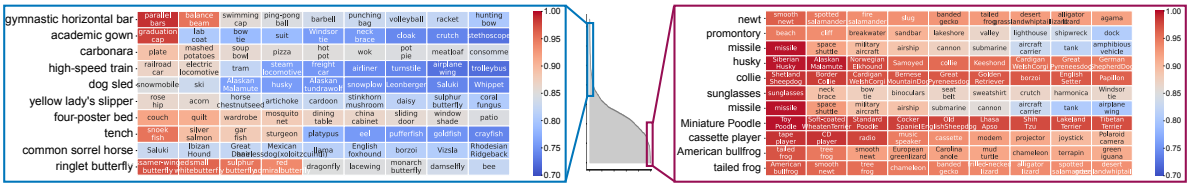
Except for the imbalance problem, the precision and recall of many high-frequency classes are much lower than many medium-/few-shot classes, as illustrated in Fig. 9.3. Thresholding the CLIP



(a) ImageNet **(b)** CIFAR10 **(c)** CIFAR100

(d) MNIST **(e)** Food101 **(f)** EuroSAT

Figure 9.4: CLIP's zero-shot predictions are highly biased for various datasets and benchmarks.



a) top-10 classes b) least-10 classes

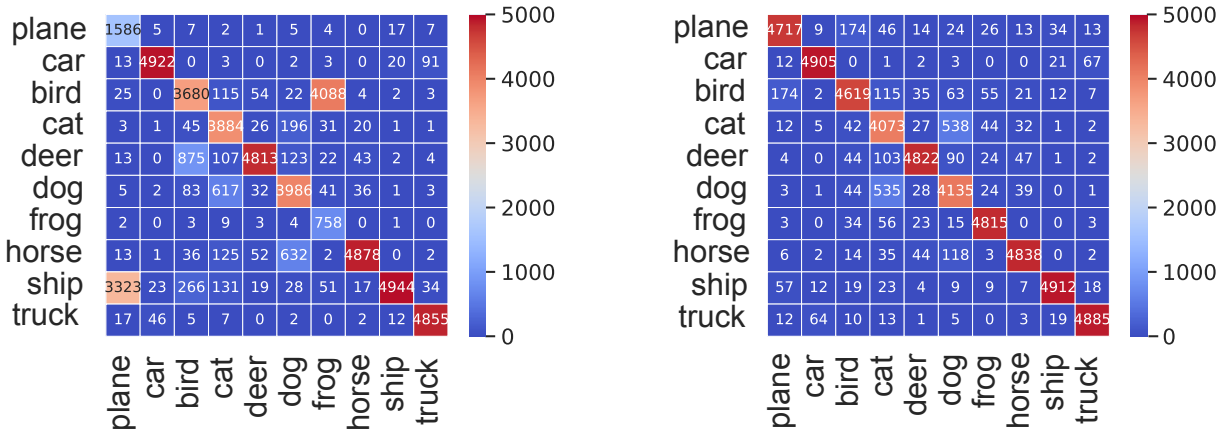
Figure 9.5: The low-frequency classes of ImageNet, with the least-10 number of CLIP predictions per class, usually have strong inter-class correlations, while the high-frequency classes are the opposite. We compare the cosine similarity between each class's image embedding centroid and embedding centroids of its nine closest "negative" classes. (better view zoomed in)

predictions based on the confidence score may help. However, simply setting a higher confidence score threshold could lead to even more imbalanced distributions (more details in appendix). There is a trade-off between imbalance ratio and precision/recall.

Highly biased zero-shot predictions are not unique to ImageNet. They are widely present on many benchmarks, such as EuroSAT [391], MNIST [392], CIFAR10 [393], CIFAR100 [393], and Food101 [394], as shown in Fig. 9.4.

Inter-Class Correlations

To delve into the causes of biased pseudo-labels, we provide an analysis of inter-class correlations. For CLIP, we first compute one image centroid per class by taking the mean of the normalized



(a) FixMatch **(b)** DebiasPL

Figure 9.6: The cause for pseudo-label biases can be partially attributed to inter-class confounding. For example, FixMatch often misclassifies "ship" as "plane". The confusion matrix of FixMatch's and our DebiasPL's pseudo-labels are visualized.

image features, extracted by the image encoder of a pre-trained CLIP model, that belong to this class. The cosine similarity between the image centroid of classes with top-10/least-10 prediction frequency and their closest "confusing" classes are visualized. The prediction confusions indicate image similarities at the class level. Fig. 9.5 shows that the low-frequency classes of ImageNet, with the least-10 number of CLIP predictions per class, usually have strong inter-class confusions.

Fig. 9.6a shows the confusion matrix of FixMatch's pseudo-labels. It is observed that many instances in some categories tend to be misclassified into one or two specific negative classes; for instance, "ship" is often misclassified as "plane".

Based on our analysis of the inter-class correlations, we believe that the blame for the pseudo-label bias can be largely attributed to inter-class confounding, which the proposed DebiasPL can successfully address as in Fig. 9.6b. DebiasPL will be introduced in the next section.

9.4 Debiased Pseudo-Labeling

This section introduces Debiased Pseudo-Labeling (DebiasPL) and methods to integrate it into ZSL and SSL tasks. It is worth noting that the proposed simple yet effective approach is universally applicable to various networks and benchmarks, not limited to the ones introduced here.

Adaptive Debiasing

Our DebiasPL approach aims at dynamically alleviating biased pseudo labels' influence on a student model without leveraging any prior knowledge on marginal class distribution, even when

exposed to source and target data that follow different distributions. An adaptive debiasing module with counterfactual reasoning and an adaptive marginal loss is proposed to fulfill this goal, described next.

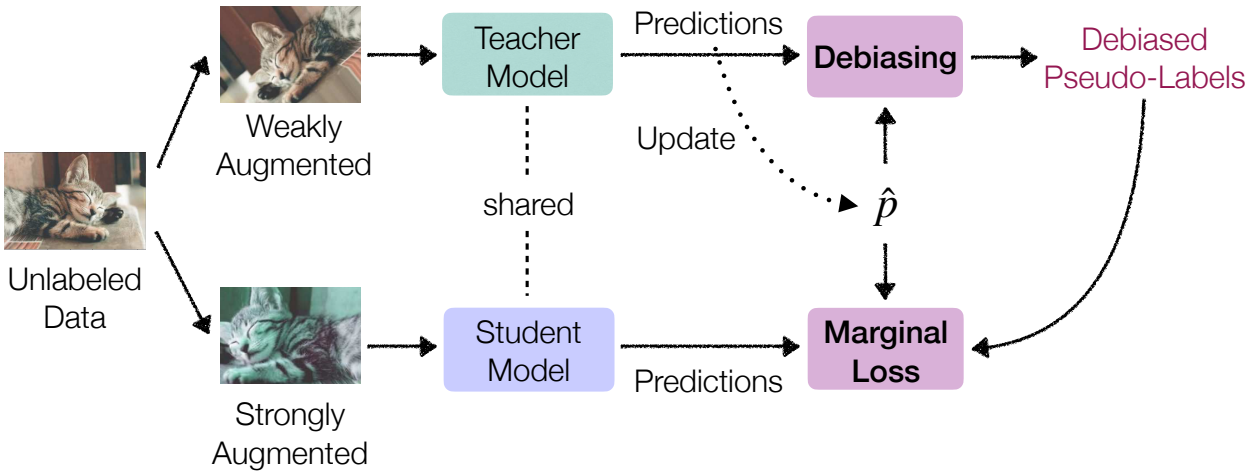


Figure 9.7: Diagram of the proposed Adaptive Debiasing module and Adaptive Marginal Loss, added to the top of FixMatch.

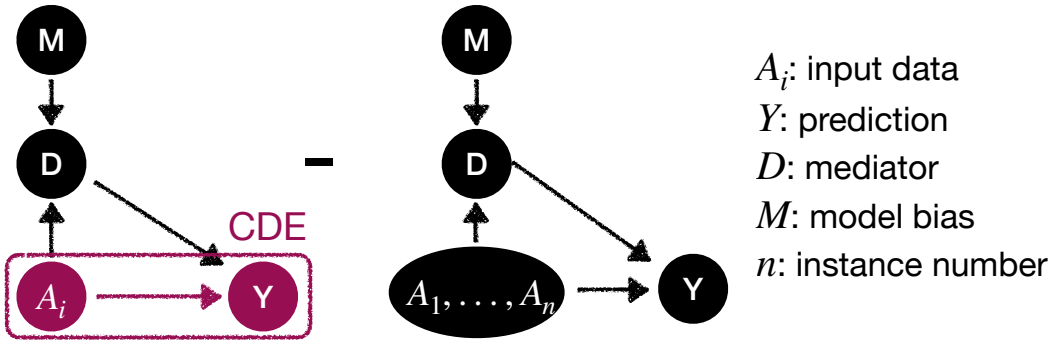


Figure 9.8: Causal graph of debiasing with counterfactual reasoning.

Adaptive Debias w/ Counterfactual Reasoning. Causal Inference is the undertaking of deriving counterfactual conclusions using only factual premises, in which causal graphical models represent the interventions among the variables [344], [347], [395]–[397]. It has been widely studied and applied in various tasks to remove selection bias which is pervasive in almost all empirical studies [398], eliminating the confounding effect using causal intervention [399], disentangling the desired direct effects with counterfactual reasoning [400], etc.

Motivated by this, to dynamically mitigate impacts of unwanted bias (*counterfactual*), we incorporate causality of producing debiased predictions through counterfactual reasoning [344], [346], [395], [401], [402].

Given the proposed causal graph in Fig. 9.8, we can delineate our goal for generating debiased predictions: the pursuit of the direct causal effect along $A_i \rightarrow Y$, defined as Controlled Direct Effect (CDE) [344]–[348]:

$$\text{CDE}(Y_i) = [Y_i|do(A_i), do(D)] - [Y_i|do(\hat{A}), do(D)] \tag{9.2}$$

i.e. the contrast between the counterfactual outcome if the individual were exposed at $A = A_i$ (with $do(A_i)$ notation) and the counterfactual outcome if the same individual were exposed at $A = \hat{A} = \{A_1, ..., A_n\}$, with the mediator set to a fixed level D. CDE [344], [347] disentangles the model bias in a counterfactual world, where the model bias is considered as the Y's indirect effect when $A = \hat{A}$ but D retains the value when $A = A_i$.

However, measuring the counterfactual outcome via visiting all training samples is significantly computational expensive. We use Approximated Controlled Direct Effect (ACDE) instead. ACDE assumes that the model bias is not drastically changed, therefore, the momentum-updated counterfactual outcomes (Eqn. 9.4) can be served as an approximation to the actual $[Y_i|do(\hat{A}), do(D)]$. The debiased logits with counterfactual reasoning, which is later used to perform pseudo-labeling (i.e., replace $p(\alpha(x_i))$ in Eqn. 9.1), can be formulated:

$$\tilde{f}_i = f(\alpha(x_i)) - \lambda \log \hat{p} \tag{9.3}$$

$$\hat{p} \leftarrow m\hat{p} + (1-m)\frac{1}{\mu B}\sum_{k=1}^{\mu B} p_k \tag{9.4}$$

$m \in [0, 1)$ is a momentum coefficient, $f(\alpha(\cdot))$ refers to logits of weakly-augmented unlabeled instance, p_k is the probability distribution for instance $\alpha(x_k)$ obtained via a softmax function. λ denotes the debias factor, which controls the strength of the indirect effect. If the debias factor is too strong, it is hard for a model to fit on the data, while too small a factor can barely eliminate the biases and, ultimately, impairs the generalization ability. Since the scale of logits is unstable, most notably at the early training stage, we use the probability distribution p_k rather than directly using the logit vector in the second term of Eqn. 9.3. A log function is applied to rescale $\hat{p}$ to match the magnitude of logit.

Eqn. 9.3 can be associated with re-weighting and logits adjustment methods in long-tailed recognition, whereas ours is dynamically adaptive.

Adaptive Marginal Loss. As aforementioned in Sec. 9.3, the biases in pseudo-labels may be partially caused by inter-class confusion. Motivated by this, we apply adaptive margin loss to demand a larger margin between hardly biased and highly biased classes, so that scores for dominant classes, towards which the model highly biased, do not overwhelm the other categories. In addition, by enforcing a dynamic class-specific margin, inter-class confusion can be greatly counteracted, which is further empirically evidenced in Fig. 9.6. $\mathcal{L}_{\text{AML}}$ can be formulated as:

$$\mathcal{L}_{\text{AML}} = -\log \frac{e^{(z_{\hat{y}_i} - \Delta_{\hat{y}_i})}}{e^{(z_{\hat{y}_i} - \Delta_{\hat{y}_i})} + \sum_{k \neq \hat{y}_i}^{C} e^{(z_k - \Delta_k)}} \tag{9.5}$$

where $\Delta_j = \lambda \log(\frac{1}{\hat{p}_j})$ for $j \in \{1, ..., C\}$, $z = f(\beta(x_i))$. We use $\mathcal{L}_{\text{AML}}$ to replaced $\text{H}(\hat{y}_i, f(\beta(x_i))$ in Eqn. 9.1. We then get the final unsupervised loss by updating Eqn. 9.1 with Eqn. 9.3 and Eqn. 9.5.
(Optional) All unlabeled instances with low probabilities do not contribute to the final loss. We find it beneficial to apply cross-level instance-group discrimination loss CLD [9] to unlabeled instances to leverage their information fully.

Distinctions and Connections with Alternatives

Please refer to Sec. 9.2 for an introduction to LA, LDAM, and Causal Norm. Another often adopted method in SSL distribution alignment (DA) [362] is also compared. It aims to encourage the actual marginal distribution of the model's predictions to match the *actual* marginal class distribution.

Please refer to Tab. 9.1 to check the distinctions and connections with these alternatives handling distribution mismatch and long-tailed recognition in key properties, and Tab. 9.2 and Tab. 9.3 to compare experimental results.

The use of a momentum updated $\hat{p}$ for debiasing pseudo-labels with counterfactual reasoning and applying adaptive marginal loss is crucial to the success of DebiasPL, which also enables our training objective does not necessarily need to use the true marginal class distribution as prior knowledge. Furthermore, since more training samples per class *do not* necessarily lead to a higher model bias against it, dynamically adjusting the margin rather than measuring margins based on the number of samples per class as in LA and LDAM could better respect the degree of bias against each class. The number of samples alone can not determine the degree of bias. Also, unlike previous works, e.g., LA/LDAM and Causal Norm, that use fixed margins or adjustments, we argue that the degree of bias of each class should never be a fixed value, but is in a process of dynamic change. The cause of bias cannot be attributed to the data alone, but the cause of the interaction between model and data.

For DA, the biggest issue is that it is limited to scenarios where either *true* marginal class distribution is available, or source and target data are collected from the same distribution, which is too ideal in the real world.

Experiments on several benchmarks are made to show the validity and feasibility of DebiasPL. *For imbalanced data*, Tab. 9.1 shows that integrating LA [389] into FixMatch lags far behind FixMatch w/ DebiasPL. *For balanced data*, since the adjustment or re-weighting vector is calculated based on the true class distribution, most existing long-tailed methods that rely on true marginal class distribution are no longer applicable without major changes (balanced class distribution leads to identical treatment for all classes).

DebiasPL for T-ZSL and SSL

For semi-supervised learning, the proposed DebiasPL can be integrated into FixMatch, as in Fig. 9.7, by adopting the adaptive debiasing module and adaptive marginal loss. To further boost the performance of SSL and exploit the power of the vision-language pre-trained model, during the training time, we can also integrate CLIP into FixMatch/DebiasPL by pseudo-labeling the

Desired Properties	LA or LDAM	Causal Norm	DA	Ours
Improve representation learning at training time	✓	✗	✓	✓
No prior knowledge on true marginal class distribution	✗	✓	✗	✓
Adaptive as the training progresses	✗	✗	✓	✓
Applicable to both imbalanced and balanced data	✗	✗	✓	✓
Source and target data can come from varying distributions	✗	✗	✗	✓

Table 9.1: Our method is the only one with all these desired properties. Comparisons with previous works concentrating on resolving training data distribution issues, including LA [389], LDAM [357], DA [362], Causal Norm [348] and our DebiasPL, in key properties. Desired (undesired) properties are in green (red).

Method	CIFAR10-LT: # of labels (percentage)				CIFAR10: # of labels (percentage)		
	γ=100		γ=200		40 (0.08%)	80 (0.16%)	250 (2%)
	1244 (10%)	3726 (30%)	1125 (10%)	3365 (30%)			
UDA [403] §	-	-	-	-	71.0 ±6.0	-	91.2 ±1.1
MixMatch [361] §	60.4 ±2.2	-	54.5 ±1.9	-	51.9 ±11.8	80.8 ±1.3	89.0 ±0.9
CReST w/ DA [379]	75.9 ±0.6	77.6 ±0.9	64.1 ±0.22	67.7 ±0.8	-	-	-
CReST+ w/ DA [379]	78.1 ±0.8	79.2 ±0.2	67.7 ±1.4	70.5 ±0.6	-	-	-
CoMatch w/ SimCLR [28], [375]	-	-	-	-	92.6 ±1.0	94.0 ±0.3	95.1 ±0.3
FixMatch [355] §	67.3 ±1.2	73.1 ±0.6	59.7 ±0.6	67.7 ±0.8	86.1 ±3.5	92.1 ±0.9	94.9 ±0.7
FixMatch w/ DA w/ LA [355], [362], [379], [389] §	70.4 ±2.9	-	62.4 ±1.2	-	-	-	-
FixMatch w/ DA w/ SimCLR [28], [355], [362] §	-	-	-	-	89.7 ±4.6	93.3 ±0.5	94.9 ±0.7
DebiasPL (w/ FixMatch)	**79.2 ±1.0**	**80.6 ±0.5**	**71.4 ±2.0**	**74.1 ±0.6**	**94.6 ±1.3**	**95.2 ±0.1**	**95.4 ±0.1**
gains over the best FixMatch variant	**+8.8**	**+7.5**	**+9.0**	**+6.4**	**+4.9**	**+1.9**	**+0.5**

Table 9.2: Without any prior knowledge of the marginal class distribution of unlabeled/labeled data, the performance of DebiasPL on *both* **CIFAR *and* CIFAR-LT SSL benchmarks** surpasses previous SOTAs, which are *either designed for balanced data or meticulously tuned for long-tailed data*. DibasMatch is experimented with the same set of hyper-parameters across all benchmarks. § states the best-reported results of counterpart methods, copied from [375], [355] or [379]. γ: imbalance ratio. We report results averaged on 5 different folds.

discarded unlabeled instances with CLIP. Because the instances CLIP are not confident on may be noisy, only these unlabeled instances with a CLIP confidence score greater than τ_{clip} are pseudo-labeled by CLIP. We could get CLIP's predictions on all training data and store it in a dictionary without re-predicting per iteration. Therefore, the computational overheads introduced by using the CLIP model are negligible. We only leverage CLIP in large-scale datasets since using CLIP on low-resolution datasets like CIFAR10 can only observe marginal gains, partly due to the lack of scale-based data augmentation in CLIP [17].

For transductive zero-shot learning, to better exploit knowledge learned from the vision-language pre-trained model and alleviate the domain shift problem when transferring the knowledge to downstream ZSL tasks, a new framework to conduct transductive zero-shot learning (T-ZSL) based on FixMatch and CLIP is developed.

Method	B.S.	#epochs	Pre-train	1%		0.2%	
				top-1	top-5	top-1	top-5
FixMatch w/ DA [355], [362]	4096	400	✗	53.4	74.4	-	-
FixMatch w/ DA [355], [362]	4096	400	✓	59.9	79.8	-	-
FixMatch w/ EMAN [355], [404]	384	50	✓	60.9	82.5	43.6*	64.6*
DebiasPL w/ FixMatch	384	50	✓	**63.1** (+2.2)	**83.6** (+1.1)	**47.9** (+3.7)	**69.6** (+5.0)
DebiasPL (multi-views)	768	50	✓	**65.3** (+4.4)	**85.2** (+2.7)	**51.6** (+8.0)	**73.3** (+8.7)
DebiasPL (multi-views)	768	200	✓	**66.5** (+5.6)	**85.6** (+3.1)	**52.3** (+8.7)	**73.5** (+8.9)
DebiasPL (multi-views)	1536	300	✓	**67.1** (+6.2)	**85.8** (+3.3)	-	-
DebiasPL w/ CLIP [17]	384	50	✓	**69.1** (+8.2)	**89.1** (+6.6)	**68.2** (+24.6)	**88.2** (+23.6)
DebiasPL w/ CLIP (multi-views) [17]	768	50	✓	**70.9** (+10.0)	**89.3** (+6.8)	**69.6** (+26.0)	**88.4** (+23.8)
CLIP (few-shot) [17], [405]	256	50	✓	53.4	-	40.0	-
SwAV [46]	4096	50	✓	53.9	78.5	-	-
SimCLRv2 (+ Self-distillation) [376]	4096	400	✓	60.0	79.8	-	-
PAWS (multi-crops) † [377]	4096	50	✓	66.5	-	-	-
CoMatch (multi-views) [375]	1440	400	✓	67.1	87.1	-	-

Table 9.3: DebiasPL delivers state-of-the-arts results on **ImageNet-1K semi-supervised learning** with various fractions of labeling samples, especially for extremely low-shot settings. All results are produced with a backbone of ResNet-50. †: unsupervised pre-trained for 800 epochs, except for PAWS [377], which is pre-trained for 300 epochs with pseudo-labels generated non-parametrically. *: reproduced.

Specifically, we again make use of the *pseudo-labeling* idea by leveraging the one-hot labels (i.e., the $\arg\max$ of the model's output) and retaining pseudo labels whose largest class probability fall above a confidence threshold τ_{clip} ($= 0.95$ by default). These instances, along with their pseudo labels, are considered "labeled data" in SSL.

After this, we could follow the original FixMatch pipeline to optimize "labeled" and "unlabeled" data jointly. To make a fair comparison with previous works and simplify the overall system, all other training recipes and settings are consistent with the original FixMatch+EMAN settings, including the model initialization part. The diagram is in the appendix.

Because CLIP is highly biased, the vanilla FixMatch + CLIP framework under-performs the original CLIP zero-shot learning, confirming our earlier hypothesis that learning from a biased model may further amplify existing bias and produce more wrong predictions. Therefore, we update the unsupervised loss $\mathcal{L}_u$ with our Adaptive Marginal Loss for alleviating the inter-class confusion and Adaptive Debias for producing debiased pseudo-labels as in Sec. 9.4.

9.5 Experiment

In this section, we conduct empirical experiments to show that DebiasPL: 1) delivers state-of-the-art results on both semi-supervised and zero-shot learning benchmarks; 2) works as a universal add-on and brings consistent performance gains to various methods; 3) exhibits stronger robustness to domain shifts; 4) is capable of improving performance on long-tailed, balanced and even hybrid data.

Semi-supervised Learning

Dataset. We perform comprehensive evaluations of DebiasPL on multiple SSL benchmarks, including CIFAR10 [393], long-tailed CIFAR10 (CIFAR10-LT) [393], and ImageNet-1K [406], with varying amounts of labeled data. For the balanced benchmarks, the performance almost saturates when using more than 2% labeled data. We put our focus on the extremely low-shot settings, i.e., 0.08%/0.16%/2% on CIFAR10 and 1%/0.2% on ImageNet-1K. For imbalanced benchmarks, we follow the settings in [379] and test DebiasPL on CIFAR10-LT under various pre-defined imbalance ratios γ, where $\gamma \in [100, 200]$, and percentage of labeled data, including 10% and 30%. More details about datasets are included in the appendix.

Setup. For all experiments on *both* long-tailed CIFAR10 and CIFAR10 datasets, we follow previous works [355], [379] to use the network architecture WRN-28-2 [58], [407]. We also follow the same set of hyper-parameters in FixMatch, except we reduce the total optimization iterations by half.

For experiments on ImageNet-1K, we use ResNet50 as the backbone network and follow the training recipes introduced in FixMatch w/ EMAN [404], which is also the default baseline of all experiments on ImageNet-1K. The model is initialized with MoCo v2 + EMAN as in [404]. For the setting with multiple views, we perform two strong augmentations and two weak augmentations on each unlabeled sample. Each strongly-augmented instance is paired with one weakly-augmented instance, and we jointly optimize the two pairs via pseudo-labeling as in the original setting of Fig. 9.7. Multi-views could increase the convergence speed and stabilize the training process.

DebiasPL is simple yet effective. Tab. 9.2 and Tab. 9.3 show that DebiasPL delivers state-of-the-art performance on all experimented benchmarks, outperforming current approaches by a large margin. Without using CLIP, DebiasPL can outperform CoMatch on CIFAR, and is comparable to CoMatch on ImageNet-1K. DebiasPL wins on its merit of simplicity. Leveraging the power of CLIP could significantly improve the performance of DebiasPL, surpassing CoMatch by about 4% on ImageNet-1K SSL.

Method	Labeled: **LT**; 10% labeled, $\gamma = 200$	
	Unlabeled: **LT**	Unlabeled: **Balanced**
FixMatch [355]	62.3 $\pm$1.6	72.1 $\pm$2.3
DebiasPL	**71.4** $\pm$**2.0** **(+9.1)**	**83.5** $\pm$**2.4** **(+11.4)**

Table 9.4: DebiasPL consistently improves the performance of SSL when the unlabeled data is either the sames as labeled data, i.e., long-tailed distributed, or different with labeled data, i.e., balanced distributed across semantics. We report results averaged on 5 folds.

DebiasPL is agnostic to source/target data distribution. Tab. 9.2 shows that, for both CIFAR and long-tailed CIFAR SSL benchmarks, using a unified framework and the same set of hyper-parameters, DebiasPL can surpass previous state-of-the-art methods, which are either designed for balanced data or meticulously tuned for long-tailed data. Furthermore, Tab. 9.4 shows that when tested in scenarios where labeled and unlabeled data follow different distributions, DebiasPL produces an even greater gain (11.4%) to the baseline.

	FixMatch	MixMatch	UDA
Baseline	89.7 ± 4.6	47.5 ± 11.5	29.1 ± 5.9
+ DebiasPL	$\mathbf{94.6 \pm 1.3}$	$\mathbf{61.7 \pm 6.1}$	$\mathbf{43.2 \pm 5.2}$

Table 9.5: DebiasPL is a universal add-on. Top-1 accuracies of various SSL methods on CIFAR10, averaged on 5 folds, are compared. 4 instances per class are labeled.

Method	#param	Accuracy (%) top-1	top-5
ConSE [408]	-	1.3	3.8
DGP [387]	-	3.0	9.3
ZSL-KG [388]	-	3.0	9.9
Visual N-Grams [409]	-	11.5	-
CLIP (prompt ensemble) [17]	26M	59.6	-
(ours) CLIP + FixMatch	26M	55.7	80.6
(ours) CLIP + DebiasPL	26M	**68.3** (+8.7)	**88.9** (+8.3)
CLIP (few-shot) [17], [405] †	26M	53.4	-
CLIP + CoOp (few-shot) [405] †	26M	60.9	-
CLIP (ViT-B/32) [17]	398M	63.2	-
CLIP (ResNet50x4) [17]	375M	65.8	-

Table 9.6: DebiasPL delivers state-of-the-art results of **zero-shot learning on ImageNet-1K**, outperforming CLIP with bigger models or fine-tuned with labels. †: CoOp and CLIP (few-shot) are fine-tuned with about 1.5% annotated data.

The fewer labeled data, the more significant gains can be observed in Tab. 9.2 and Tab. 9.3, almost eliminating the gap between fully-supervised and semi-supervised learning.
DebiasPL is also a universal add-on as illustrated in Tab. 9.5. Incorporating DebiasPL into various SSL methods can achieve consistent performance improvements.

Tranductive Zero-Shot Learning

Dataset. We evaluate the efficiency of DebiasPL in T-ZSL on ImageNet-1K [406]. EuroSAT [391], MNIST [392], CIFAR10 [393], CIFAR100 [393], and Food101 [394] are also used as evaluation datasets to show the robustness to domain shift.
Setup. T-ZSL assumes that the list of possible class candidates is known for the target data. Following this setting, we do not use any semantic labels for target data. We apply DebiasPL on CLIP in a similar way as we apply DebiasPL on FixMatch, except that the labeled data is "labeled" by CLIP rather than a human annotator. Specifically, all unlabeled instances whose CLIP confidence score greater than τ_{clip} are pseudo-labeled by CLIP and considered as "labeled" data. A backbone of ResNet50 and a threshold τ_{clip} of 0.95 are used for all datasets. The same default hyper-parameters and training recipes as in FixMatch + EMAN are utilized for fair comparisons.

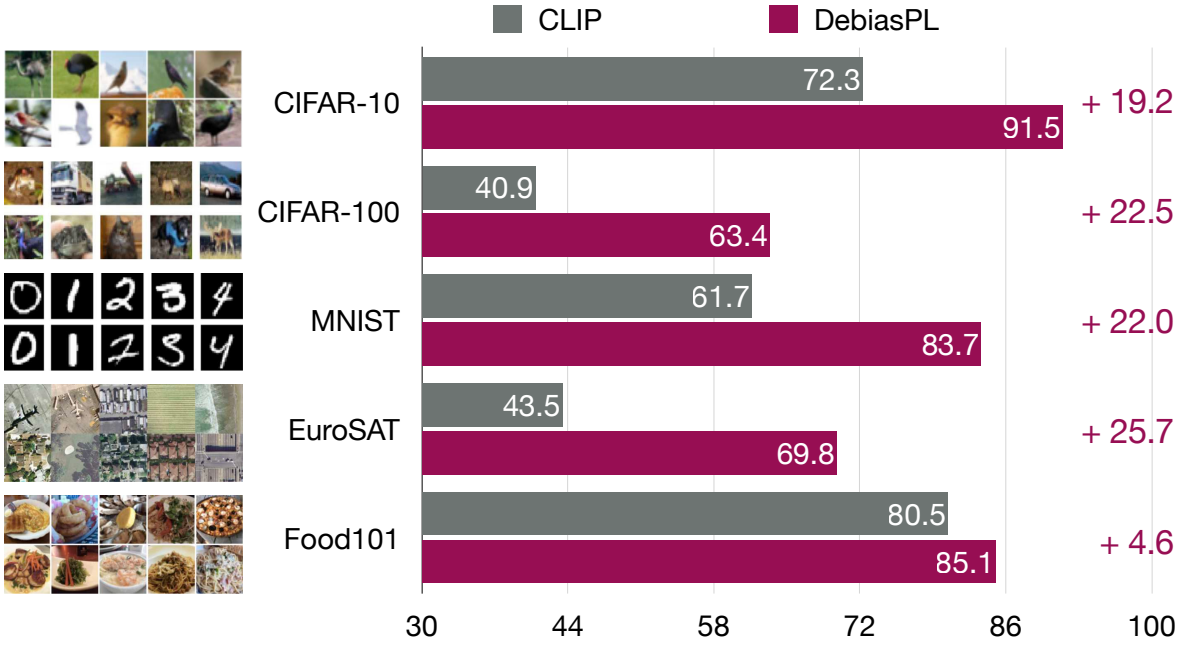


Figure 9.9: DebiasPL exhibits stronger robustness to domain shift when conducting **zero-shot learning on various datasets**. We experiment with ResNet-50 as a backbone network. CLIP results are reproduced with official codes.

More details are in the appendix.

DebiasPL delivers SOTA results on zero-shot learning, even surpassing CLIP [17] and CoOP [405] that are fine-tuned on partial human-labeled data. Moreover, DebiasPL with a backbone of ResNet50 can significantly outperform CLIP with 15× larger backbones, as shown in Tab. 9.6. The time cost of zero-shot training DebiasPL w/ CLIP (without using any human annotations) for 100 epochs is less than 0.01% of CLIP's overall training time.

DebiasPL exhibits stronger robustness to domain shift than zero-shot CLIP without accessing any semantic labels, as depicted in Fig. 9.9. Also, DebiasPL can observe greater gains (more than 20%) on datasets with larger domain shifts, e.g., an astonishing 25.7% gains can be obtained on the satellite image dataset EuroSAT [391].

9.6 Appendix

Details on Datasets and Implementations

The PyTorch-style pseudocode for semi-supervised learning with DebiasPL is available at Algo. 2.

We conduct experiments on several benchmarks to prove the effectiveness and universality of DebiasPL. Here we provide more details on datasets and implementations for each benchmark:

Algorithm 2 PyTorch-style pseudocode for semi-supervised learning with DebiasPL

Initialize $\hat{p}$ with $1/C$, where C is the number of classes
$\hat{p} \leftarrow$ `torch.ones([1, C]) / C`
Load a batch with labeled (x, target) and unlabeled samples u
for $(x, \text{target}), u$ in loader **do**
Augment x (weak) and get two versions of u (strong and weak)
    $x, u_s, u_w \leftarrow \texttt{weak(x)}, \texttt{strong}(u), \texttt{weak}(u)$
Model forward
    $l_x, l_{us}, l_{uw} \leftarrow \texttt{model}(x, u_s, u_w)$
Get debiased pseudo-labels
    $p_{uw} \leftarrow \texttt{softmax}(l_{uw} - \tau \cdot \log(\hat{p}))$
    $\texttt{max\_probs}, \texttt{pseudo\_label} \leftarrow \texttt{torch.max}(p_{uw})$
Get mask for filtering low-confidence instances
    $\texttt{mask} \leftarrow \texttt{max\_probs} \geq \texttt{thresh}$
Update $\hat{p}$
    $\hat{p} \leftarrow \texttt{momentum} \cdot \hat{p} + (1 - \texttt{momentum}) \cdot \texttt{mean}(p_{uw})$
Compute labeled loss
    $\texttt{loss}_x \leftarrow \texttt{cross\_entropy}(l_x, \text{target})$
Compute marginal loss for unlabeled instances
    $l_{us} \leftarrow l_{us} + \lambda \cdot \log(\hat{p})$
    $\texttt{loss}_u \leftarrow \texttt{mean}(\texttt{cross\_entropy}(l_{us}, \texttt{pseudo\_label}) \cdot \texttt{mask})$
Total loss
    $\texttt{loss} \leftarrow \texttt{loss}_x + \lambda_u \cdot \texttt{loss}_u$
Optimization step
    `loss.backward()`
    `optimizer.step()`
end for
Update the EMA model
`model.momentum_update_ema()`

CIFAR10 [393]: The original version of CIFAR10 contains 50,000 images on the training set and 10,000 images on the validation set with 10 categories for CIFAR10. For semi-supervised learning on CIFAR10, we conduct the experiments with a varying number of labeled examples from 40 to 250, following standard practice in previous works [355], [361], [362], [375]. The reported results of each previous method in the paper are directly copied from the best-reported results in MixMatch [361], ReMixMatch [362], FixMatch [355], CoMatch [375], etc.

We keep all hyper-parameters the same as FixMatch, except for the number of training steps. We use WideResNet-28-2 [58], [407] with 1.5M parameters as a backbone network for CIFAR10. The SGD optimizer with a Nesterov momentum of 0.9 is used for optimization. The learning rate is initialized as 0.03 and decayed with a cosine learning rate scheduler [410], which sets the learning rate at training step k as $\cos(\frac{7\pi k}{16K})$ times the initial learning rate, where $K = 2^{19}$ is the total number

of training steps, i.e., about 512 epochs, and is 2 times fewer than the original number of FixMatch training steps. The model is trained with a mini-batch size of 512, which contains 64 labeled samples and 448 unlabeled samples, on one V100 GPU. As in previous works, an exponential moving average of model parameters is used to produce the final performance. The weight decay is set as 0.0005 for CIFAR10. Unless otherwise stated, the only independent hyperparameter of DebiasPL λ is fixed and set to 0.5 in all experiments. Each method is tested under 5 different folds, and we report the mean and the standard deviation of accuracy on the test set.

CIFAR10-LT [379], [393], [411]: The long-tailed version of CIFAR10 follows an exponential decay in sample sizes across different categories. CIFAR10-LT is constructed by sampling a subset of CIFAR10 following the Pareto distribution with the power value $\gamma \in [100, 200]$. Then, we select 10% or 30% of all CIFAR10-LT instances to construct the SSL benchmark labeled dataset, and the others are regarded as the unlabeled datasets. Each algorithm is tested under 5 different folds of labeled data, and we report the mean and the standard deviation of accuracy on the test set. As in previous works, an exponential moving average of model parameters is used to produce the final performance.

To demonstrate the universality of the proposed method DebiasPL and its insensitivity to data distribution, we follow the same hyperparameters and training formulas in CIFAR10. We do not specifically adjust any hyperparameters when conducting experiments in the long-tail SSL benchmarks.

ImageNet-1K [406]: ImageNet-1K is a curated dataset with approximately class-balanced data distribution, containing about 1.3M images for training and 50K images for validation.

For semi-supervised learning, ImageNet-1K with varying amounts of labeled data is experimented with, i.e., 0.2% and 1%. The FixMatch model is trained with a batch size of 64 (320) for labeled (unlabeled) images with an initial learning rate of 0.03. Following [404], we replace batch normalization (BN) layers with exponential moving average normalization (EMAN) layers in the teacher model. EMAN updates its statistics by exponential moving average from the BN statistics of the student model. ResNet-50 is used as the default network and the default hyper-parameters in the corresponding papers [355], [404] are applied. The model is initialized with MoCo v2 + EMAN pre-trained model as in [404]. To make fair comparisons, we report results of FixMatch with EMAN as the baseline model, and all hyper-parameters of FixMatch with EMAN are untouched unless noted otherwise.

For zero-shot learning, no manual annotation is leveraged in the training process. We train CLIP + DebiasPL and CLIP + FixMatch following the same hyperparameters and training recipes as FixMatch with EMAN, except that the labeled data is "labeled" by CLIP rather than a human annotator. Specifically, all unlabeled instances whose CLIP confidence score greater than τ_{clip} are pseudo-labeled by CLIP (with a backbone of ResNet50) and considered as "labeled" data. A backbone of ResNet50 and a threshold τ_{clip} of 0.95 are used. The same default hyper-parameters and training recipes as in FixMatch + EMAN are utilized for fair comparisons.

For experiments on other benchmarks of ZSL, including EuroSAT [391], MNIST [392], DTD [412], GTSRB [413] and Flowers102 [414], we follow the training recipe of ImageNet-1K.

Ablation Study

In this section, we conduct additional ablation studies on the influence of the two components of DebiasPL (Table. 9.7) for SSL, DebiasPL's unique hyperparameter λ (Table. 9.8) for SSL, and CLIP's confidence score threshold τ_{clip} (Table. 9.9) for T-ZSL.

Debiasing	Magirnal Loss	CIFAR10	CIFAR10-LT
		86.1	73.5
✓		93.3	79.6
✓	✓	94.6	80.6

Table 9.7: Ablation study on the **contribution of each component** of DebiasPL. Experimented on CIFAR10 and CIFAR10-LT ($\gamma = 100$) SSL, in which 4 out of 5,000 samples are labeled per class for CIFAR10 and 30% instances are labeled for CIFAR10-LT. Results averaged over 5 different folds are reported.

As shown in Table. 9.7, the two components of DebiasPL lead to significant improvements to *both* CIFAR10 and CIFAR10-LT SSL benchmarks. Compared with the balanced benchmark, the performance improvement obtained by introducing the marginal loss is relatively smaller than the unbalanced benchmark.

λ	0.0	0.25	0.5	0.75	1.0	2.0
DebiasPL	73.5	79.5	80.6	80.5	80.5	77.7

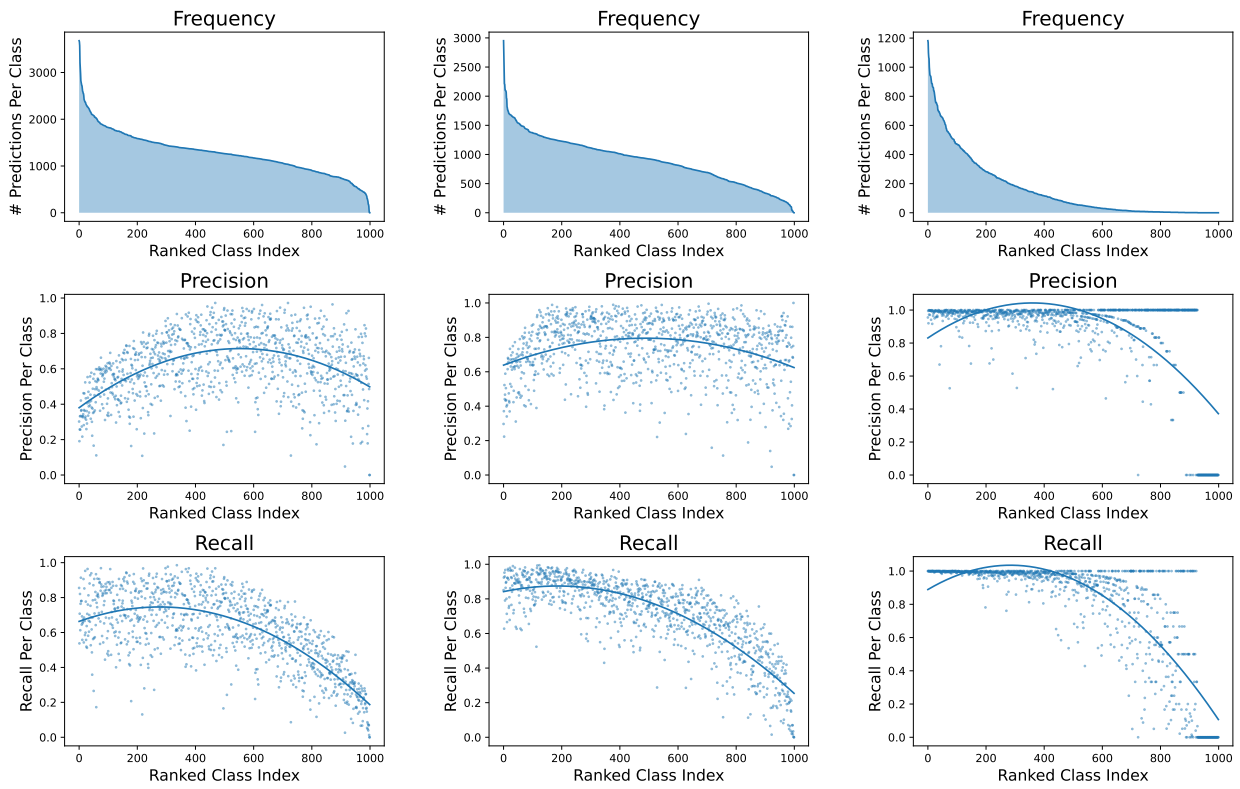
Table 9.8: Ablation study on CIFAR10-LT ($\gamma = 100$) semi-supervised learning with DebiasPL under various **weight** λ of debiasing module and marginal loss. 30% samples are labeled. The model is identical to FixMatch when $\lambda = 0$. Results averaged over 5 different folds are reported.

Table. 9.8 illustrates the influence of debias factor λ. When the value of λ is set to 0, DebiasPL is identical to FixMatch. Adding a debiasing module and marginal loss can improve the performance on CIFAR10-LT by more than 7% when selecting the optimal choice of λ 0.5, which is marginally better than the default value of 1.0. However, there is a trade-off. Suppose the debias factor λ is too strong. In that case, it is hard for a model to fit on the data, while a too-small factor can barely eliminate the biases, ultimately impairs the generalization ability.

τ_{clip}	0.2	0.4	0.6	0.8	0.9	0.95
DebiasPL + CLIP	55.9	63.2	66.2	67.1	67.7	67.7

Table 9.9: Ablation study on ImageNet-1K zero-shot Learning with DebiasPL + CLIP [17] under various **threshold** τ_{clip}.

As illustrated in the main paper, the CLIP predictions are class-imbalanced. Therefore, the natural question is whether we can obtain a more balanced prediction by filtering instances with a



(a) $\tau \geq 0.0$ **(b)** $\tau \geq 0.4$ **(c)** $\tau \geq 0.95$

Figure 9.10: A higher imbalanced ratio is obtained when filtering CLIP's zero-shot predictions with a larger threshold, analyzed on CLIP's zero-shot predictions on 1.3M almost class-balanced ImageNet training samples. Per class number of predictions (row 1), precision (row 2), and recall (row 3) of samples passing various confidence score thresholds τ are visualized. Zero-shot predictions are produced with an ensemble of 80 prompts and a backbone of ResNet50, using official codes.

threshold τ_{clip}? Unfortunately, no, on the contrary, when filtering predictions with a larger threshold, a higher imbalance rate is observed, as in Fig. 9.10. Furthermore, when filtering instances with a threshold of 0.95, more than 60 categories get zero predictions.

The dilemma is that using a smaller threshold τ_{clip} can obtain a smaller imbalanced ratio, which is the desired property. However, it also leads to a lower precision, introducing many outliers and misclassified samples. Therefore, a module to eliminate biases captured by the CLIP model when CLIP is pre-trained on source data is needed to yield a good performance on target data.

Table. 9.9 shows that using a threshold of 0.95 can get the optimal performance on the ImageNet zero-shot learning task, which indicates that the high precision of the labeled data, realized by using a high threshold, is essential for better performance on target data. At the same time, our proposed DebiasPL can greatly alleviate the trouble of a higher imbalance ratio caused by using a larger threshold, eventually obtaining more than 10% performance gains.

9.7 Summary

In this paper, we conduct research on the previously unknown biased pseudo-labeling issue. A simple yet effective method DebiasPL is proposed to dynamically alleviate biased pseudo-labels' influence on a student model, without leveraging any prior knowledge of true data distribution. As a universal add-on, DebiasPL delivers significantly better performance than previous state-of-the-arts on both semi-supervised learning and transductive zero-shot learning tasks and exhibits stronger robustness to domain shifts.

Part V

Looking Ahead: Towards Autonomous Machine Intelligence

Natural intelligence emerges through a lifelong process of perceiving and acting, involving both passive observation and active exploration across behaviors, expressions, and language. These experiences form rich sources of intelligence, enabling agents to associate observed behaviors with environmental cues and learn interactions in a task-independent, unsupervised manner. In contrast, current AI systems are primarily trained on simplified abstractions of human intelligence — labeled datasets for vision models or text-based corpora for language models. While large language models (LLMs) excel at mimicking human language patterns, they operate within a constrained spectrum of intelligence, far removed from the full range of sensory, perceptual, and motor experiences that shape human intelligence [415]–[417].

Despite recent efforts to develop large multimodal models, most approaches remain language-centric, relying on pretrained LLMs with other modalities — *e.g.*, visual, auditory, and sensory inputs — layered on top. However, language is merely an abstraction of the world: a compressed representation of human knowledge that inevitably fails to capture the many nuances and sensory experiences we encounter. *Treating language as the primary representation of intelligence imposes inherent limitations on many tasks, ultimately constraining AI's ability to perceive and interact with the rich, dynamic nature of our world.*

To advance AI toward human-like intelligence, we must break free from these limitations. My future work aims to overcome these constraints by creating AI systems that leverage richer, multimodal sensory inputs to foster a more complete form of intelligence — systems that can operate across the full spectrum of cognitive skills, from perception and reasoning to prediction and strategic action, while possessing the agility to evolve through continual, self-directed interaction with the world, *free from heavy human supervision.* Consider how a child learns: by observing adults, interacting with the environment, and interpreting behaviors and cues. Similarly, while human intelligence has been encoded into text over centuries — forming a framework that LLMs can replicate — true AI requires richer sensory grounding, encompassing physical interaction, spatial navigation, and social dynamics, to genuinely mirror human capabilities.

Interactive, continual learning is another essential component. Beyond passive observation, intelligent agents must actively engage with their environment — forming hypotheses, acting, and adjusting based on feedback. For example, a robot learning to grasp objects cannot master the skill through static observation alone; it must experiment, receive trial-and-error feedback, and iteratively refine its approach until it succeeds. This type of adaptive learning is crucial for building systems that not only comprehend but also intelligently respond to novel situations in real time. Many of the sensory and motor capabilities in humans and animals evolved in this way to enhance survival in dynamic environments. Likewise, developing machine intelligence in a dynamic, interactive context could yield more robust and adaptive AI systems.

In summary, my research vision is to design machine intelligence that approach — and eventually surpass — human intelligence by developing scalable, multimodal models equipped with rich world models and intrinsic drives for exploration. I am committed to *developing the foundations of an AI that is deeply informed by real-world, multimodal experiences, ultimately reflecting and surpassing the complexity, adaptability, and richness of natural human intelligence*. These systems should integrate diverse forms of perception, act to generate their own learning opportunities, and continuously refine their understanding through interaction. Looking forward, the path towards

autonomous machine intelligence will require unifying perception, reasoning, prediction, and action within agents that can learn effectively and efficiently from the world itself — not just from curated datasets — enabling them to adapt, generalize, and thrive in the open-ended complexity of real environments.

Bibliography

[1] J. Achiam, S. Adler, S. Agarwal, *et al.*, "Gpt-4 technical report," *arXiv:2303.08774*, 2023.

[2] T. Brown, B. Mann, N. Ryder, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[3] G. Team, R. Anil, S. Borgeaud, *et al.*, "Gemini: A family of highly capable multimodal models," *arXiv preprint arXiv:2312.11805*, 2023.

[4] H. Touvron, T. Lavril, G. Izacard, *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[5] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International Conference on Machine Learning*, PMLR, 2021, pp. 10 347–10 357.

[6] X. Wang, J. Yang, and T. Darrell, "Segment anything without supervision," *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[7] L. Fu, L. Lian, R. Wang, *et al.*, "Rethinking patch dependence for masked autoencoders," *arXiv preprint arXiv:2401.14391*, 2024.

[8] X. Wang, Z. Liu, and S. X. Yu, "Unsupervised feature learning by cross-level instance-group discrimination," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2021, pp. 12 586–12 595.

[9] X. Wang, Z. Liu, and S. X. Yu, "Unsupervised feature learning by cross-level instance-group discrimination," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 586–12 595.

[10] X. Wang, Z. Wu, L. Lian, and S. X. Yu, "Debiased learning from naturally imbalanced pseudo-labels," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 14 647–14 657.

[11] X. Wang, L. Lian, and S. X. Yu, "Unsupervised selective labeling for more effective semi-supervised learning," in *European Conference on Computer Vision (ECCV)*, Springer, 2022, pp. 427–445.

[12] X. Wang, R. Girdhar, S. X. Yu, and I. Misra, "Cut and learn for unsupervised object detection and instance segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2023, pp. 3124–3134.

[13] D. Niu, X. Wang, X. Han, L. Lian, R. Herzig, and T. Darrell, "Unsupervised universal image segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 22 744–22 754.

[14] X. Wang, I. Misra, Z. Zeng, R. Girdhar, and T. Darrell, "Videocutler: Surprisingly simple unsupervised video instance segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 22 755–22 764.

[15] A. Kirillov, E. Mintun, N. Ravi, *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015–4026.

[16] X. Wang, X. Zhou, A. Fathi, T. Darrell, and C. Schmid, "Visual lexicon: Rich image features in language space," *arXiv preprint arXiv:2412.06774*, 2024.

[17] A. Radford, J. W. Kim, C. Hallacy, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, PMLR, 2021, pp. 8748–8763.

[18] T.-H. Wu, G. Biamby, D. Chan, *et al.*, "See say and segment: Teaching lmms to overcome false premises," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 13 459–13 469.

[19] X. Wang, S. Li, K. Kallidromitis, Y. Kato, K. Kozuka, and T. Darrell, "Hierarchical open-vocabulary universal image segmentation," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, 2023.

[20] X. Wang, S. Zhang, S. Li, *et al.*, "Segllm: Multi-round reasoning segmentation with large language models," *arXiv preprint arXiv:2410.18923*, 2024.

[21] X. Wang, T. Darrell, S. S. Rambhatla, R. Girdhar, and I. Misra, "Instancediffusion: Instance-level control for image generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 6232–6242.

[22] X. Wang, L. Lian, Z. Miao, Z. Liu, and S. Yu, "Long-tailed recognition by routing diverse distribution-aware experts," in *International Conference on Learning Representations (ICLR)*, 2021.

[23] X. Wang, Z. Yu, S. De Mello, *et al.*, "Freesolo: Learning to segment objects without annotations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14 176–14 186.

[24] T.-Y. Lin, M. Maire, S. Belongie, *et al.*, "Microsoft coco: Common objects in context," in *European conference on computer vision*, Springer, 2014, pp. 740–755.

[25] M. Caron, H. Touvron, I. Misra, *et al.*, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 9650–9660.

[26] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9729–9738.

[27] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020.

[28] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*, PMLR, 2020, pp. 1597–1607.

[29] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[30] H. V. Vo, P. Pérez, and J. Ponce, "Toward unsupervised, multi-object discovery in large-scale image collections," in *European Conference on Computer Vision*, Springer, 2020, pp. 779–795.

[31] V. H. Vo, E. Sizikova, C. Schmid, P. Pérez, and J. Ponce, "Large-scale unsupervised object discovery," *Advances in Neural Information Processing Systems*, vol. 34, pp. 16 764–16 778, 2021.

[32] O. Siméoni, G. Puy, H. V. Vo, *et al.*, "Localizing objects with self-supervised transformers and no labels," *arXiv preprint arXiv:2109.14279*, 2021.

[33] Y. Wang, X. Shen, Y. Yuan, *et al.*, "Tokencut: Segmenting objects in images and videos with self-supervised transformer and normalized cut," *arXiv preprint arXiv:2209.00383*, 2022.

[34] A. Bar, X. Wang, V. Kantorov, *et al.*, "Detreg: Unsupervised pretraining with region priors for object detection," in *CVPR*, 2022.

[35] X. Wang, R. Zhang, T. Kong, L. Li, and C. Shen, "Solov2: Dynamic and fast instance segmentation," *Advances in Neural information processing systems*, vol. 33, pp. 17 721–17 732, 2020.

[36] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *ECCV*, 2020.

[37] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," in *International Conference on Learning Representations*, 2020.

[38] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[39] W. Van Gansbeke, S. Vandenhende, and L. Van Gool, "Discovering object masks with transformers for unsupervised semantic segmentation," *arXiv preprint arXiv:2206.06363*, 2022.

[40] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3733–3742.

[41] I. Misra and L. v. d. Maaten, "Self-supervised learning of pretext-invariant representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6707–6717.

[42] J.-B. Grill, F. Strub, F. Altché, *et al.*, "Bootstrap your own latent: A new approach to self-supervised learning," *arXiv preprint arXiv:2006.07733*, 2020.

[43] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 750–15 758.

[44] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *ICML*, 2016.

[45] Y. Asano, C. Rupprecht, and A. Vedaldi, "Self-labelling via simultaneous clustering and representation learning," in *International Conference on Learning Representations*, 2019.

[46] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," in *Thirty-fourth Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[47] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 000–16 009.

[48] H. Bao, L. Dong, S. Piao, and F. Wei, "Beit: Bert pre-training of image transformers," in *International Conference on Learning Representations*, 2021.

[49] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1422–1430.

[50] C. Doersch, A. Gupta, and A. A. Efros, "Context as supervisory signal: Discovering objects with predictable context," in *European Conference on Computer Vision*, Springer, 2014, pp. 362–377.

[51] A. Ziegler and Y. M. Asano, "Self-supervised learning of object parts for semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14 502–14 511.

[52] M. Cho, S. Kwak, C. Schmid, and J. Ponce, "Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1201–1210.

[53] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[54] S. Maji, N. Vishnoi, and J. Malik, "Biased normalized cuts," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 2057–2064.

[55] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," *NeurIPS*, 2011.

[56] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[57] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6154–6162.

[58] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[59] G. Ghiasi, Y. Cui, A. Srinivas, *et al.*, "Simple copy-paste is a strong data augmentation method for instance segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2918–2928.

[60] D. Dwibedi, I. Misra, and M. Hebert, "Cut, paste and learn: Surprisingly easy synthesis for instance detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1301–1310.

[61] A. Gupta, P. Dollar, and R. Girshick, "Lvis: A dataset for large vocabulary instance segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5356–5364.

[62] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[63] S. Gidaris, A. Bursuc, G. Puy, N. Komodakis, M. Cord, and P. Perez, "Obow: Online bag-of-visual-words generation for self-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6830–6840.

[64] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[65] X. Wang, R. Zhang, C. Shen, T. Kong, and L. Li, "Dense contrastive learning for self-supervised visual pre-training," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3024–3033.

[66] W. Wang, M. Feiszli, H. Wang, and D. Tran, "Unidentified video objects: A benchmark for dense, open-world segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 776–10 785.

[67] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[68] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, *Detectron2*, `https://github.com/facebookresearch/detectron2`, 2019.

[69] B. Thomee, D. A. Shamma, G. Friedland, *et al.*, "Yfcc100m: The new data in multimedia research," *Communications of the ACM*, vol. 59, no. 2, pp. 64–73, 2016.

[70] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*, IEEE, 2012, pp. 3354–3361.

[71] N. Inoue, R. Furuta, T. Yamasaki, and K. Aizawa, "Cross-domain weakly-supervised object detection through progressive domain adaptation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5001–5009.

[72] S. Shao, Z. Li, T. Zhang, *et al.*, "Objects365: A large-scale, high-quality dataset for object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 8430–8439.

[73] A. Kuznetsova, H. Rom, N. Alldrin, *et al.*, "The open images dataset v4," *International Journal of Computer Vision*, vol. 128, no. 7, pp. 1956–1981, 2020.

[74] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[75] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[76] R. Girshick, "Fast r-cnn," in *ICCV*, 2015.

[77] J. Xie, W. Xie, and A. Zisserman, "Segmenting moving objects via an object-centric layered representation," in *Advances in Neural Information Processing Systems*, 2022.

[78] C. Yang, H. Lamdouar, E. Lu, A. Zisserman, and W. Xie, "Self-supervised video object segmentation by motion grouping," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 7177–7188.

[79] L. Yang, Y. Fan, and N. Xu, "Video instance segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5188–5197.

[80] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, Springer, 2020, pp. 402–419.

[81] W. Wang, T. Zhou, F. Porikli, D. Crandall, and L. Van Gool, "A survey on deep learning technique for video segmentation," *arXiv preprint arXiv:2107.01153*, 2021.

[82] Y.-T. Hu, J.-B. Huang, and A. G. Schwing, "Unsupervised video object segmentation using motion saliency-guided spatio-temporal propagation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 786–802.

[83] Y. Yang, A. Loquercio, D. Scaramuzza, and S. Soatto, "Unsupervised moving object detection via contextual information separation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 879–888.

[84] 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, pp. 5924–5934.

[85] X. Li, S. Liu, S. De Mello, X. Wang, J. Kautz, and M.-H. Yang, "Joint-task self-supervised learning for temporal correspondence," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[86] A. Jabri, A. Owens, and A. Efros, "Space-time correspondence as a contrastive random walk," *Advances in neural information processing systems*, vol. 33, pp. 19 545–19 560, 2020.

[87] C. Ventura, M. Bellver, A. Girbau, A. Salvador, F. Marques, and X. Giro-i-Nieto, "Rvos: End-to-end recurrent network for video object segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5277–5286.

[88] Z. Yang, Q. Wang, S. Bai, W. Hu, and P. H. Torr, "Video segmentation by detection for the 2019 unsupervised davis challenge," 2019.

[89] T. Zhou, J. Li, S. Wang, R. Tao, and J. Shen, "Matnet: Motion-attentive transition network for zero-shot video object segmentation," *IEEE Transactions on Image Processing*, vol. 29, pp. 8326–8338, 2020.

[90] J. Luiten, I. E. Zulfikar, and B. Leibe, "Unovost: Unsupervised offline video object segmentation and tracking," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2020, pp. 2000–2009.

[91] M. Keuper, B. Andres, and T. Brox, "Motion trajectory segmentation via minimum cost multicuts," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3271–3279.

[92] D. Kondermann, R. Nair, K. Honauer, *et al.*, "The hci benchmark suite: Stereo and flow ground truth with uncertainties for urban autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 19–28.

[93] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1290–1299.

[94] B. Cheng, A. Choudhuri, I. Misra, A. Kirillov, R. Girdhar, and A. G. Schwing, "Mask2former for video instance segmentation," *arXiv preprint arXiv:2112.10764*, 2021.

[95] A. Dosovitskiy, P. Fischer, E. Ilg, *et al.*, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2758–2766.

[96] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *European Conf. on Computer Vision (ECCV)*, A. Fitzgibbon et al. (Eds.), Ed., ser. Part IV, LNCS 7577, Springer-Verlag, Oct. 2012, pp. 611–625.

[97] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung, "A benchmark dataset and evaluation methodology for video object segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 724–732.

[98] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool, "The 2017 davis challenge on video object segmentation," *arXiv preprint arXiv:1704.00675*, 2017.

[99] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[100] M. Hamilton, Z. Zhang, B. Hariharan, N. Snavely, and W. T. Freeman, "Unsupervised semantic segmentation by distilling feature correspondences," *arXiv preprint arXiv:2203.08414*, 2022.

[101] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *ICCV*, 2017.

[102] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6569–6578.

[103] B. Cheng, A. G. Schwing, and A. Kirillov, "Per-pixel classification is not all you need for semantic segmentation," 2021.

[104] Y. Li, H. Mao, R. Girshick, and K. He, "Exploring plain vision transformer backbones for object detection," in *European Conference on Computer Vision*, Springer, 2022, pp. 280–296.

[105] A. Kirillov, K. He, R. B. Girshick, C. Rother, and P. Dollár, "Panoptic segmentation," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9396–9405, 2018. [Online]. Available: `https://api.semanticscholar.org/CorpusID:4853375`.

[106] J. Jain, J. Li, M. Chiu, A. Hassani, N. Orlov, and H. Shi, "OneFormer: One Transformer to Rule Universal Image Segmentation," 2023.

[107] X. Wang, R. Girdhar, S. X. Yu, and I. Misra, "Cut and learn for unsupervised object detection and instance segmentation," *arXiv preprint arXiv:2301.11320*, 2023.

[108] X. Ji, J. F. Henriques, and A. Vedaldi, "Invariant information clustering for unsupervised image classification and segmentation," in *ICCV*, 2019.

[109] J. H. Cho, U. Mall, K. Bala, and B. Hariharan, "Picie: Unsupervised semantic segmentation using invariance and equivariance in clustering," *ArXiv*, vol. abs/2103.17070, 2021.

[110] W. Van Gansbeke, S. Vandenhende, S. Georgoulis, and L. Van Gool, "Unsupervised semantic segmentation by contrasting object mask proposals," *arxiv preprint arxiv:2102.06191*, 2021.

[111] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, "Panoptic segmentation," in *CVPR*, 2019.

[112] Y. Xiong, R. Liao, H. Zhao, *et al.*, "Upsnet: A unified panoptic segmentation network," in *CVPR*, 2019.

[113] A. Kirillov, R. Girshick, K. He, and P. Dollár, "Panoptic feature pyramid networks," in *CVPR*, 2019.

[114] B. Cheng, M. D. Collins, Y. Zhu, *et al.*, "Panoptic-DeepLab: A simple, strong, and fast baseline for bottom-up panoptic segmentation," in *CVPR*, 2020.

[115] Y. Li, H. Zhao, X. Qi, *et al.*, "Fully convolutional networks for panoptic segmentation with point-based supervision," *arXiv preprint arXiv:2108.07682*, 2021.

[116] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *ECCV*, 2020.

[117] H. Wang, Y. Zhu, H. Adam, A. Yuille, and L.-C. Chen, "MaX-DeepLab: End-to-end panoptic segmentation with mask transformers," in *CVPR*, 2021.

[118] W. Van Gansbeke, S. Vandenhende, S. Georgoulis, M. Proesmans, and L. Van Gool, "Scan: Learning to classify images without labels," in *European conference on computer vision*, Springer, 2020, pp. 268–285.

[119] Z. Dang, C. Deng, X. Yang, K. Wei, and H. Huang, "Nearest neighbor matching for deep clustering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 693–13 702.

[120] S. Park, S. Han, S. Kim, *et al.*, "Improving unsupervised image clustering with robust learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 278–12 287.

[121] S. Lloyd, "Least squares quantization in pcm," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[122] E. W. Forgy, "Cluster analysis of multivariate data: Efficiency versus interpretability of classifications," *biometrics*, vol. 21, pp. 768–769, 1965.

[123] H.-P. Kriegel, E. Schubert, and A. Zimek, "The (black) art of runtime evaluation: Are we comparing algorithms or implementations?" *Knowledge and Information Systems*, vol. 52, no. 2, pp. 341–378, 2017.

[124] G. J. McLachlan and T. Krishnan, *The EM algorithm and extensions*. John Wiley & Sons, 2007, vol. 382.

[125] E. H. Adelson, "On seeing stuff: The perception of materials by humans and machines," in *Human Vision and Electronic Imaging*, 2001.

[126] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[127] W. Wang, M. Feiszli, H. Wang, and D. Tran, "Unidentified video objects: A benchmark for dense, open-world segmentation," *CoRR*, vol. abs/2104.04691, 2021. arXiv: 2104.04691. [Online]. Available: https://arxiv.org/abs/2104.04691.

[128] M. Cordts, M. Omran, S. Ramos, *et al.*, "The cityscapes dataset for semantic urban scene understanding," *CoRR*, vol. abs/1604.01685, 2016. arXiv: 1604.01685. [Online]. Available: http://arxiv.org/abs/1604.01685.

[129] A. Kirillov, R. Girshick, K. He, and P. Dollár, "Panoptic feature pyramid networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 6399–6408.

[130] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[131] X. Wang, R. Zhang, C. Shen, T. Kong, and L. Li, "Dense contrastive learning for self-supervised visual pre-training," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2021.

[132] A. Kirillov, E. Mintun, N. Ravi, *et al.*, "Segment anything," *arXiv:2304.02643*, 2023.

[133] A. Young, B. Chen, C. Li, *et al.*, "Yi: Open foundation models by 01. ai," *arXiv preprint arXiv:2403.04652*, 2024.

[134] J. Bai, S. Bai, Y. Chu, *et al.*, "Qwen technical report," *arXiv preprint arXiv:2309.16609*, 2023.

[135] A. Q. Jiang, A. Sablayrolles, A. Mensch, *et al.*, "Mistral 7b," *arXiv preprint arXiv:2310.06825*, 2023.

[136] J. Kaplan, S. McCandlish, T. Henighan, *et al.*, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[137] J. Bill, H. Pailian, S. J. Gershman, and J. Drugowitsch, "Hierarchical structure is employed by humans during visual motion perception," *Proceedings of the National Academy of Sciences*, vol. 117, no. 39, pp. 24 581–24 589, 2020.

[138] S. R. Mitroff, B. J. Scholl, and K. Wynn, "Divide and conquer: How object files adapt when a persisting object splits into two," *Psychological Science*, vol. 15, no. 6, pp. 420–425, 2004.

[139] B. Zhou, H. Zhao, X. Puig, *et al.*, "Semantic understanding of scenes through the ade20k dataset," *International Journal of Computer Vision*, vol. 127, pp. 302–321, 2019.

[140] L. Qi, J. Kuen, Y. Wang, *et al.*, "Open world entity segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[141] J. He, S. Yang, S. Yang, *et al.*, "Partimagenet: A large, high-quality dataset of parts," in *European Conference on Computer Vision*, Springer, 2022, pp. 128–145.

[142] V. Ramanathan, A. Kalia, V. Petrovic, *et al.*, "Paco: Parts and attributes of common objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7141–7151.

[143] D. Niu, X. Wang, X. Han, L. Lian, R. Herzig, and T. Darrell, "Unsupervised universal image segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[144] Y. Wang, X. Shen, Y. Yuan, *et al.*, "Tokencut: Segmenting objects in images and videos with self-supervised transformer and normalized cut," *IEEE transactions on pattern analysis and machine intelligence*, 2023.

[145] S. Cao, J. Gu, J. Kuen, *et al.*, "SOHES: Self-supervised open-world hierarchical entity segmentation," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=PXNrncg2DF.

[146] W. Van Gansbeke, S. Vandenhende, and L. Van Gool, "Discovering object masks with transformers for unsupervised semantic segmentation," *arxiv preprint arxiv:2206.06363*, 2022.

[147] S. Cao, D. Joshi, L. Gui, and Y.-X. Wang, "HASSOD: Hierarchical adaptive self-supervised object detection," in *NeurIPS*, 2023.

[148] H. V. Vo, F. Bach, M. Cho, *et al.*, *Unsupervised image matching and object discovery as optimization*, 2019. arXiv: 1904.03148 [cs.CV].

[149] F. Li, H. Zhang, P. Sun, *et al.*, "Semantic-sam: Segment and recognize anything at any granularity," *arXiv preprint arXiv:2307.04767*, 2023.

[150] X. Zhao, W. Ding, Y. An, *et al.*, *Fast segment anything*, 2023. arXiv: 2306.12156 [cs.CV].

[151] Y. Xiong, B. Varadarajan, L. Wu, *et al.*, *Efficientsam: Leveraged masked image pretraining for efficient segment anything*, 2023. arXiv: 2312.00863 [cs.CV].

[152] J. Cheng, J. Ye, Z. Deng, *et al.*, *Sam-med2d*, 2023. arXiv: 2308.16184 [cs.CV].

[153] T. Lüddecke and A. Ecker, "Image segmentation using text and image prompts," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 7086–7096.

[154] X. Wang, X. Zhang, Y. Cao, W. Wang, C. Shen, and T. Huang, "Seggpt: Segmenting everything in context," *arXiv preprint arXiv:2304.03284*, 2023.

[155] Y. Bai, X. Geng, K. Mangalam, *et al.*, "Sequential modeling enables scalable learning for large vision models," *arXiv preprint arXiv:2312.00785*, 2023.

[156] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 5, pp. 898–916, 2010.

[157] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001.

[158] H. K. Cheng, J. Chung, Y.-W. Tai, and C.-K. Tang, "Cascadepsp: Toward class-agnostic and very high-resolution segmentation via global and local refinement," in *CVPR*, 2020.

[159] Z. Liu, Y. Lin, Y. Cao, *et al.*, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[160] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, *Attention is all you need*, 2023. arXiv: 1706.03762 [cs.CL].

[161] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[162] M. Oquab, T. Darcet, T. Moutakanni, *et al.*, "Dinov2: Learning robust visual features without supervision," *arXiv:2304.07193*, 2023.

[163] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[164] Q. Yu, M. Weber, X. Deng, X. Shen, D. Cremers, and L.-C. Chen, "An image is worth 32 tokens for reconstruction and generation," *NeurIPS*, 2024.

[165] C. Saharia, W. Chan, S. Saxena, *et al.*, "Photorealistic text-to-image diffusion models with deep language understanding," *Advances in neural information processing systems*, vol. 35, pp. 36 479–36 494, 2022.

[166] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, "Hierarchical text-conditional image generation with clip latents," *arXiv preprint arXiv:2204.06125*, 2022.

[167] D. Baranchuk, A. Voynov, I. Rubachev, V. Khrulkov, and A. Babenko, "Label-efficient semantic segmentation with diffusion models," in *ICLR*, 2022.

[168] A. C. Li, M. Prabhudesai, S. Duggal, E. Brown, and D. Pathak, "Your diffusion model is secretly a zero-shot classifier," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 2206–2217.

[169] Y. Gan, S. Park, A. M. Schubert, A. Philippakis, and A. Alaa, "Instructcv: Instruction-tuned text-to-image diffusion models as vision generalists," in *The Twelfth International Conference on Learning Representations*, 2024.

[170] J. Xu, S. Liu, A. Vahdat, W. Byeon, X. Wang, and S. De Mello, "Open-vocabulary panoptic segmentation with text-to-image diffusion models," *arXiv preprint arXiv:2303.04803*, 2023.

[171] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 684–10 695.

[172] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[173] G. E. Hinton and R. Zemel, "Autoencoders, minimum description length and helmholtz free energy," *Advances in neural information processing systems*, vol. 6, 1993.

[174] Y. Bengio, L. Yao, G. Alain, and P. Vincent, "Generalized denoising auto-encoders as generative models," *Advances in neural information processing systems*, vol. 26, 2013.

[175] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 1096–1103.

[176] C. Raffel, N. Shazeer, A. Roberts, *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of machine learning research*, 2020.

[177] J. Betker, G. Goh, L. Jing, *et al.*, "Improving image generation with better captions," *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, vol. 2, no. 3, p. 8, 2023.

[178] C. Wei, C. Liu, S. Qiao, Z. Zhang, A. Yuille, and J. Yu, "De-diffusion makes text a strong cross-modal interface," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 13 492–13 503.

[179] 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 *CVPR*, 2023.

[180] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer, "Sigmoid loss for language image pre-training," in *CVPR*, 2023.

[181] L. Beyer, A. Steiner, A. S. Pinto, *et al.*, "Paligemma: A versatile 3b vlm for transfer," *arXiv:2407.07726*, 2024.

[182] X. Chen, H. Fang, T.-Y. Lin, *et al.*, "Microsoft coco captions: Data collection and evaluation server," *arXiv:1504.00325*, 2015.

[183] J. Xu, T. Mei, T. Yao, and Y. Rui, "Msr-vtt: A large video description dataset for bridging video and language," in *CVPR*, 2016.

[184] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh, "Making the v in vqa matter: Elevating the role of image understanding in visual question answering," in *CVPR*, 2017.

[185] L. Yu, P. Poirson, S. Yang, A. C. Berg, and T. L. Berg, "Modeling context in referring expressions," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, Springer, 2016, pp. 69–85.

[186] R. Girdhar, A. El-Nouby, Z. Liu, *et al.*, "Imagebind: One embedding space to bind them all," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 15 180–15 190.

[187] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," in *ECCV*, 2020.

[188] J. Li, D. Li, C. Xiong, and S. Hoi, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation," in *ICLR*, PMLR, 2022.

[189] O. Patashnik, Z. Wu, E. Shechtman, D. Cohen-Or, and D. Lischinski, "Styleclip: Text-driven manipulation of stylegan imagery," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 2085–2094.

[190] D. Podell, Z. English, K. Lacey, *et al.*, "Sdxl: Improving latent diffusion models for high-resolution image synthesis," in *The Twelfth International Conference on Learning Representations*, 2023.

[191] H. Bao, L. Dong, S. Piao, and F. Wei, "BEit: BERT pre-training of image transformers," in *ICLR*, 2022.

[192] B. Li, Y. Zhang, D. Guo, *et al.*, "Llava-onevision: Easy visual task transfer," *arXiv:2408.03326*, 2024.

[193] M. Chen, A. Radford, R. Child, *et al.*, "Generative pretraining from pixels," in *International conference on machine learning*, PMLR, 2020, pp. 1691–1703.

[194] J. Donahue and K. Simonyan, "Large scale adversarial representation learning," in *NIPS*, 2019.

[195] S. Mahajan, T. Rahman, K. M. Yi, and L. Sigal, "Prompting hard or hardly prompting: Prompt inversion for text-to-image diffusion models," in *CVPR*, 2024.

[196] R. Mokady, A. Hertz, K. Aberman, Y. Pritch, and D. Cohen-Or, "Null-text inversion for editing real images using guided diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 6038–6047.

[197] N. Ruiz, Y. Li, V. Jampani, *et al.*, "Hyperdreambooth: Hypernetworks for fast personalization of text-to-image models," in *CVPR*, 2024.

[198] R. Gal, Y. Alaluf, Y. Atzmon, *et al.*, "An image is worth one word: Personalizing text-to-image generation using textual inversion," *ICLR*, 2023.

[199] X. Chen, Z. Liu, S. Xie, and K. He, "Deconstructing denoising diffusion models for self-supervised learning," *arXiv:2401.14404*, 2024.

[200] X. Yang and X. Wang, "Diffusion model as representation learner," in *ICCV*, 2023.

[201] W. Wang, Q. Sun, F. Zhang, Y. Tang, J. Liu, and X. Wang, "Diffusion feedback helps clip see better," *arXiv:2407.20171*, 2024.

[202] L. Yu, Y. Cheng, Z. Wang, *et al.*, "Spae: Semantic pyramid autoencoder for multimodal generation with frozen llms," *NeurIPS*, 2024.

[203] D. A. Hudson, D. Zoran, M. Malinowski, *et al.*, "Soda: Bottleneck diffusion models for representation learning," in *CVPR*, 2024.

[204] J. Yu, X. Li, J. Y. Koh, *et al.*, "Vector-quantized image modeling with improved vqgan," *ICLR*, 2022.

[205] L. Yu, J. Lezama, N. B. Gundavarapu, *et al.*, "Language model beats diffusion–tokenizer is key to visual generation," *ICLR*, 2024.

[206] F. Mentzer, D. Minnen, E. Agustsson, and M. Tschannen, "Finite scalar quantization: Vq-vae made simple," *ICLR*, 2024.

[207] Y. Zhao, Y. Xiong, and P. Krähenbühl, "Image and video tokenization with binary spherical quantization," *arXiv:2406.07548*, 2024.

[208] K. Tian, Y. Jiang, Z. Yuan, B. Peng, and L. Wang, "Visual autoregressive modeling: Scalable image generation via next-scale prediction," *NeurIPS*, 2024.

[209] X. Li, H. Chen, K. Qiu, *et al.*, "Imagefolder: Autoregressive image generation with folded tokens," *arXiv:2410.01756*, 2024.

[210] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.

[211] J. Yu, Z. Wang, V. Vasudevan, L. Yeung, M. Seyedhosseini, and Y. Wu, "Coca: Contrastive captioners are image-text foundation models," *Transactions on Machine Learning Research*, 2022.

[212] P. Gage, "A new algorithm for data compression," *The C Users Journal*, vol. 12, no. 2, pp. 23–38, 1994.

[213] R. Sennrich, "Neural machine translation of rare words with subword units," *arXiv preprint arXiv:1508.07909*, 2015.

[214] J. Ho and T. Salimans, "Classifier-free diffusion guidance," in *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

[215] E. J. Hu, Y. Shen, P. Wallis, *et al.*, "Lora: Low-rank adaptation of large language models," *arXiv:2106.09685*, 2021.

[216] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, "Soundstream: An end-to-end neural audio codec," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2021.

[217] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *ICML*, PMLR, 2015.

[218] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," *Advances in neural information processing systems*, vol. 32, 2019.

[219] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015.

[220] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *ICML*, PMLR, 2021.

[221] G. Ilharco, M. Wortsman, R. Wightman, *et al.*, *Openclip*, version 0.1, Jul. 2021. DOI: 10.5281/zenodo.5143773. [Online]. Available: https://doi.org/10.5281/zenodo.5143773.

[222] M. Cherti, R. Beaumont, R. Wightman, *et al.*, "Reproducible scaling laws for contrastive language-image learning," in *CVPR*, 2023.

[223] X. Chen, X. Wang, S. Changpinyo, *et al.*, "Pali: A jointly-scaled multilingual language-image model," *arXiv:2209.06794*, 2022.

[224] N. Shazeer and M. Stern, "Adafactor: Adaptive learning rates with sublinear memory cost," in *ICML*, PMLR, 2018.

[225] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.

[226] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[227] G. Team, T. Mesnard, C. Hardin, *et al.*, "Gemma: Open models based on gemini research and technology," *arXiv preprint arXiv:2403.08295*, 2024.

[228] H. Liu, C. Li, Y. Li, and Y. J. Lee, "Improved baselines with visual instruction tuning," in *CVPR*, 2024.

[229] H. Liu, C. Li, Y. Li, *et al.*, *Llava-next: Improved reasoning, ocr, and world knowledge*, 2024.

[230] J. Lin, H. Yin, W. Ping, P. Molchanov, M. Shoeybi, and S. Han, "Vila: On pre-training for visual language models," in *CVPR*, 2024.

[231] T. Salimans and J. Ho, "Progressive distillation for fast sampling of diffusion models," in *ICLR*, 2021.

[232] A. V. Thapliyal, J. Pont-Tuset, X. Chen, and R. Soricut, "Crossmodal-3600: A massively multilingual multimodal evaluation dataset," *arXiv preprint arXiv:2205.12522*, 2022.

[233] O. Sidorov, R. Hu, M. Rohrbach, and A. Singh, "Textcaps: A dataset for image captioning with reading comprehension," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, Springer, 2020, pp. 742–758.

[234] T.-Y. Hsu, C. L. Giles, and T.-H. Huang, "Scicap: Generating captions for scientific figures," *arXiv preprint arXiv:2110.11624*, 2021.

[235] A. Singh, V. Natarajan, M. Shah, *et al.*, "Towards vqa models that can read," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 8317–8326.

[236] K. Marino, M. Rastegari, A. Farhadi, and R. Mottaghi, "Ok-vqa: A visual question answering benchmark requiring external knowledge," in *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, 2019, pp. 3195–3204.

[237] P. Lu, S. Mishra, T. Xia, *et al.*, "Learn to explain: Multimodal reasoning via thought chains for science question answering," *Advances in Neural Information Processing Systems*, vol. 35, pp. 2507–2521, 2022.

[238] D. Gurari, Q. Li, A. J. Stangl, *et al.*, "Vizwiz grand challenge: Answering visual questions from blind people," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3608–3617.

[239] D. A. Hudson and C. D. Manning, "Gqa: A new dataset for real-world visual reasoning and compositional question answering," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 6700–6709.

[240] S. Kazemzadeh, V. Ordonez, M. Matten, and T. Berg, "Referitgame: Referring to objects in photographs of natural scenes," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 787–798.

[241] J. Mao, J. Huang, A. Toshev, O. Camburu, A. L. Yuille, and K. Murphy, "Generation and comprehension of unambiguous object descriptions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 11–20.

[242] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *International conference on machine learning*, PMLR, 2016, pp. 1060–1069.

[243] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[244] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.

[245] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," *arXiv preprint arXiv:1809.11096*, 2018.

[246] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 8110–8119.

[247] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[248] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," *arXiv preprint arXiv:2010.02502*, 2020.

[249] H. Chang, H. Zhang, J. Barber, *et al.*, "Muse: Text-to-image generation via masked generative transformers," *arXiv preprint arXiv:2301.00704*, 2023.

[250] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *Advances in neural information processing systems*, vol. 34, pp. 8780–8794, 2021.

[251] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," *arXiv preprint arXiv:2011.13456*, 2020.

[252] A. Nichol, P. Dhariwal, A. Ramesh, *et al.*, "Glide: Towards photorealistic image generation and editing with text-guided diffusion models," *arXiv preprint arXiv:2112.10741*, 2021.

[253] Y. Li, H. Liu, Q. Wu, *et al.*, "Gligen: Open-set grounded text-to-image generation," in *CVPR*, 2023.

[254] L. Zhang and M. Agrawala, "Adding conditional control to text-to-image diffusion models," *arXiv preprint arXiv:2302.05543*, 2023.

[255] O. Gafni, A. Polyak, O. Ashual, S. Sheynin, D. Parikh, and Y. Taigman, "Make-a-scene: Scene-based text-to-image generation with human priors," in *European Conference on Computer Vision*, Springer, 2022, pp. 89–106.

[256] W. Feng, W. Zhu, T.-j. Fu, *et al.*, "Layoutgpt: Compositional visual planning and generation with large language models," *arXiv preprint arXiv:2305.15393*, 2023.

[257] V. Goel, E. Peruzzo, Y. Jiang, *et al.*, "Pair-diffusion: Object-level image editing with structure-and-appearance paired diffusion models," *arXiv preprint arXiv:2303.17546*, 2023.

[258] D. H. Park, G. Luo, C. Toste, *et al.*, "Shape-guided diffusion with inside-outside attention," *arXiv preprint arXiv:2212.00210*, 2022.

[259] L. Zhang, Q. Chen, B. Hu, and S. Jiang, "Text-guided neural image inpainting," in *Proceedings of the 28th ACM international conference on multimedia*, 2020, pp. 1302–1310.

[260] D. Bau, A. Andonian, A. Cui, *et al.*, "Paint by word," *arXiv preprint arXiv:2103.10951*, 2021.

[261] O. Avrahami, D. Lischinski, and O. Fried, "Blended diffusion for text-driven editing of natural images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 18 208–18 218.

[262] Y. Ge, J. Xu, B. N. Zhao, L. Itti, and V. Vineet, "Dall-e for detection: Language-driven context image synthesis for object detection," *arXiv preprint arXiv:2206.09592*, 2022.

[263] H. Zhao, D. Sheng, J. Bao, *et al.*, "X-paste: Revisiting scalable copy-paste for instance segmentation using clip and stablediffusion," in *International Conference on Machine Learning*, 2023.

[264] O. Avrahami, T. Hayes, O. Gafni, *et al.*, "Spatext: Spatio-textual representation for controllable image generation," in *ICCV*, 2023.

[265] Y. Kim, J. Lee, J.-H. Kim, J.-W. Ha, and J.-Y. Zhu, "Dense text-to-image generation with attention modulation," in *ICCV*, 2023.

[266] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, *High-resolution image synthesis with latent diffusion models*, 2021. arXiv: `2112.10752 [cs.CV]`.

[267] A. Van Den Oord, O. Vinyals, *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.

[268] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2014. [Online]. Available: `http://arxiv.org/abs/1312.6114`.

[269] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," *Advances in neural information processing systems*, vol. 30, 2017.

[270] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[271] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, *et al.*, "Video-to-video synthesis," *arXiv preprint arXiv:1808.06601*, 2018.

[272] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8798–8807.

[273] T. Xu, P. Zhang, Q. Huang, *et al.*, "Attngan: Fine-grained text to image generation with attentional generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1316–1324.

[274] Z. Zhang, J. Ma, C. Zhou, *et al.*, "Ufc-bert: Unifying multi-modal controls for conditional image synthesis," *Advances in Neural Information Processing Systems*, vol. 34, pp. 27 196–27 208, 2021.

[275] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, *et al.*, "Conditional image generation with pixelcnn decoders," *Advances in neural information processing systems*, vol. 29, 2016.

[276] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-Or, "Prompt-to-prompt image editing with cross attention control," *arXiv:2208.01626*, 2022.

[277] D. Stap, M. Bleeker, S. Ibrahimi, and M. Ter Hoeve, "Conditional image generation and manipulation for user-specified content," *arXiv preprint arXiv:2005.04909*, 2020.

[278] O. Bar-Tal, L. Yariv, Y. Lipman, and T. Dekel, "Multidiffusion: Fusing diffusion paths for controlled image generation," 2023.

[279] H. Chefer, Y. Alaluf, Y. Vinker, L. Wolf, and D. Cohen-Or, "Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models," *ACM Transactions on Graphics (TOG)*, vol. 42, no. 4, pp. 1–10, 2023.

[280] Z. Yang, J. Wang, Z. Gan, *et al.*, "Reco: Region-controlled text-to-image generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 14 246–14 255.

[281] W. Feng, X. He, T.-J. Fu, *et al.*, "Training-free structured diffusion guidance for compositional text-to-image synthesis," *arXiv preprint arXiv:2212.05032*, 2022.

[282] M. Chen, I. Laina, and A. Vedaldi, "Training-free layout control with cross-attention guidance," *arXiv preprint arXiv:2304.03373*, 2023.

[283] J. Xie, Y. Li, Y. Huang, *et al.*, "Boxdiff: Text-to-image synthesis with training-free box-constrained diffusion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 7452–7461.

[284] J.-B. Alayrac, J. Donahue, P. Luc, *et al.*, "Flamingo: A visual language model for few-shot learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 23 716–23 736, 2022.

[285] M. Tancik, P. P. Srinivasan, B. Mildenhall, *et al.*, "Fourier features let networks learn high frequency functions in low dimensional domains," *NeurIPS*, 2020.

[286] C. Si, Z. Huang, Y. Jiang, and Z. Liu, "Freeu: Free lunch in diffusion u-net," *arXiv preprint arXiv:2309.11497*, 2023.

[287] Y. Zhang, X. Huang, J. Ma, *et al.*, "Recognize anything: A strong image tagging model," *arXiv preprint arXiv:2306.03514*, 2023.

[288] S. Liu, Z. Zeng, T. Ren, *et al.*, "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," *arXiv preprint arXiv:2303.05499*, 2023.

[289] J. Li, D. Li, S. Savarese, and S. Hoi, "BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models," in *ICML*, 2023.

[290] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[291] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[292] G. Jocher, A. Chaurasia, and J. Qiu, *YOLO by Ultralytics*, version 8.0.0, Jan. 2023. [Online]. Available: `https://github.com/ultralytics/ultralytics`.

[293] D. Lin, J. Dai, J. Jia, K. He, and J. Sun, "Scribblesup: Scribble-supervised convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3159–3167.

[294] J. Bai and X. Wu, "Error-tolerant scribbles based interactive image segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 392–399.

[295] E. Agustsson, J. R. Uijlings, and V. Ferrari, "Interactive full image segmentation by considering all regions jointly," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 622–11 631.

[296] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[297] J. Ho and T. Salimans, "Classifier-free diffusion guidance," *arXiv preprint arXiv:2207.12598*, 2022.

[298] M. T. Mason, *Mechanics of robotic manipulation*. MIT press, 2001.

[299] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[300] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[301] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 2786–2793.

[302] F. Ebert, C. Finn, S. Dasari, A. Xie, A. Lee, and S. Levine, "Visual foresight: Model-based deep reinforcement learning for vision-based robotic control," *arXiv preprint arXiv:1812.00568*, 2018.

[303] A. V. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine, "Visual reinforcement learning with imagined goals," in *Advances in Neural Information Processing Systems*, 2018, pp. 9191–9200.

[304] O. M. Andrychowicz, B. Baker, M. Chociej, *et al.*, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.

[305] A. Rajeswaran, S. Ghotra, B. Ravindran, and S. Levine, "Epopt: Learning robust neural network policies using model ensembles," *arXiv preprint arXiv:1610.01283*, 2016.

[306] F. Sadeghi and S. Levine, "Cad2rl: Real single-image flight without a single real image," *arXiv preprint arXiv:1611.04201*, 2016.

[307] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2017, pp. 23–30.

[308] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2018, pp. 1–8.

[309] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, "Robust adversarial reinforcement learning," *arXiv preprint arXiv:1703.02702*, 2017.

[310] J. Yang, B. Petersen, H. Zha, and D. Faissol, "Single episode policy transfer in reinforcement learning," *arXiv preprint arXiv:1910.07719*, 2019.

[311] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Learning multiple visual domains with residual adapters," in *Advances in Neural Information Processing Systems*, 2017, pp. 506–516.

[312] X. Wang, Z. Cai, D. Gao, and N. Vasconcelos, "Towards universal object detection by domain attention," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 7289–7298.

[313] N. Hansen, Y. Sun, P. Abbeel, A. A. Efros, L. Pinto, and X. Wang, "Self-supervised policy adaptation during deployment," *arXiv preprint arXiv:2007.04309*, 2020.

[314] Y. Tassa, Y. Doron, A. Muldal, *et al.*, "Deepmind control suite," *arXiv preprint arXiv:1801.00690*, 2018.

[315] T. Jakab, A. Gupta, H. Bilen, and A. Vedaldi, "Unsupervised learning of object landmarks through conditional image generation," in *Advances in neural information processing systems*, 2018, pp. 4016–4027.

[316] T. D. Kulkarni, A. Gupta, C. Ionescu, *et al.*, "Unsupervised learning of object keypoints for perception and control," in *Advances in neural information processing systems*, 2019, pp. 10 724–10 734.

[317] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *ECCV*, 2016.

[318] G. Larsson, M. Maire, and G. Shakhnarovich, "Colorization as a proxy task for visual understanding," in *CVPR*, 2017.

[319] 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.

[320] X. Wang, Z. Liu, and S. X. Yu, "Unsupervised feature learning by cross-level discrimination between instances and groups," *arXiv preprint arXiv:2008.03813*, 2020.

[321] M. Jaderberg, V. Mnih, W. M. Czarnecki, *et al.*, "Reinforcement learning with unsupervised auxiliary tasks," *arXiv preprint arXiv:1611.05397*, 2016.

[322] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," in *NeurIPS*, 2018.

[323] A. Srinivas, M. Laskin, and P. Abbeel, "Curl: Contrastive unsupervised representations for reinforcement learning," *arXiv preprint arXiv:2004.04136*, 2020.

[324] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, "Visual domain adaptation: A survey of recent advances," *IEEE signal processing magazine*, vol. 32, no. 3, pp. 53–69, 2015.

[325] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," *arXiv preprint arXiv:1502.02791*, 2015.

[326] A. Rosenfeld and J. K. Tsotsos, "Incremental learning through deep adaptation," *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[327] A. Mallya, D. Davis, and S. Lazebnik, "Piggyback: Adapting a single network to multiple tasks by learning to mask weights," in *ECCV*, 2018, pp. 67–82.

[328] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li, "Deep reconstruction-classification networks for unsupervised domain adaptation," in *European Conference on Computer Vision*, Springer, 2016, pp. 597–613.

[329] F. Zhuang, X. Cheng, P. Luo, S. J. Pan, and Q. He, "Supervised representation learning: Transfer learning with deep autoencoders," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[330] A. Gupta, C. Devin, Y. Liu, P. Abbeel, and S. Levine, "Learning invariant feature spaces to transfer skills with reinforcement learning," *arXiv preprint arXiv:1703.02949*, 2017.

[331] H. B. Ammar, E. Eaton, P. Ruvolo, and M. Taylor, "Unsupervised cross-domain transfer in policy gradient reinforcement learning via manifold alignment," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, 2015.

[332] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7167–7176.

[333] M. Joshi, W. W. Cohen, M. Dredze, and C. P. Rosé, "Multi-domain learning: When do domains matter?" In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Association for Computational Linguistics, 2012, pp. 1302–1312.

[334] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *CVPR*, 2016, pp. 4293–4302.

[335] D. F. Nettleton, A. Orriols-Puig, and A. Fornells, "A study of the effect of different types of noise on the precision of supervised learning techniques," *Artificial intelligence review*, vol. 33, no. 4, pp. 275–306, 2010.

[336] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[337] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, "Using self-supervised learning can improve model robustness and uncertainty," in *Advances in Neural Information Processing Systems*, 2019, pp. 15 663–15 674.

[338] J. Morimoto and K. Doya, "Robust reinforcement learning," *Neural computation*, vol. 17, no. 2, pp. 335–359, 2005.

[339] J. Heinrich and D. Silver, "Deep reinforcement learning from self-play in imperfect-information games," *arXiv preprint arXiv:1603.01121*, 2016.

[340] N. Heess, J. J. Hunt, T. P. Lillicrap, and D. Silver, "Memory-based control with recurrent neural networks," *arXiv preprint arXiv:1512.04455*, 2015.

[341] W. Yu, J. Tan, C. K. Liu, and G. Turk, "Preparing for the unknown: Learning a universal policy with online system identification," *arXiv preprint arXiv:1702.02453*, 2017.

[342] N. Hansen and X. Wang, "Generalization in reinforcement learning by soft data augmentation," *arXiv preprint arXiv:2011.13389*, 2020.

[343] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *Journal of Artificial Intelligence Research*, vol. 47, pp. 253–279, 2013.

[344] J. Pearl, "Direct and indirect effects," *arXiv preprint arXiv:1301.2300*, 2013.

[345] L. Richiardi, R. Bellocco, and D. Zugna, "Mediation analysis in epidemiology: Methods, interpretation and bias," *International journal of epidemiology*, vol. 42, no. 5, pp. 1511–1519, 2013.

[346] J. Pearl and D. Mackenzie, *The book of why: the new science of cause and effect*. Basic Books, 2018.

[347] S. Greenland, J. M. Robins, and J. Pearl, "Confounding and collapsibility in causal inference," *Statistical science*, pp. 29–46, 1999.

[348] K. Tang, J. Huang, and H. Zhang, "Long-tailed classification by keeping the good and removing the bad momentum causal effect," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[349] M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas, "Reinforcement learming with augmented data," arXiv:2004.14990.

[350] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 6, pp. 1452–1464, 2017.

[351] T. Haarnoja, A. Zhou, K. Hartikainen, *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[352] Y. Tassa, S. Tunyasuvunakool, A. Muldal, *et al.*, *Dm_control: Software and tasks for continuous control*, 2020. arXiv: 2006.12983 [cs.RO].

[353] T. Yu, D. Quillen, Z. He, *et al.*, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *Conference on Robot Learning (CoRL)*, 2019. arXiv: 1910.10897 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1910.10897.

[354] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness," *arXiv preprint arXiv:1811.12231*, 2018.

[355] K. Sohn, D. Berthelot, C.-L. Li, *et al.*, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," *arXiv preprint arXiv:2001.07685*, 2020.

[356] G. Van Horn, O. Mac Aodha, Y. Song, *et al.*, "The inaturalist species classification and detection dataset," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8769–8778.

[357] K. Cao, C. Wei, A. Gaidon, N. Arechiga, and T. Ma, "Learning imbalanced datasets with label-distribution-aware margin loss," in *Advances in Neural Information Processing Systems*, 2019, pp. 1567–1578.

[358] B. Kang, S. Xie, M. Rohrbach, *et al.*, "Learning imbalanced datasets with label-distribution-aware margin loss," in *International Conference on Learning Representations*, 2020, pp. 1567–1578.

[359] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, pp. 1–35, 2021.

[360] D.-H. Lee *et al.*, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on challenges in representation learning, ICML*, vol. 3, 2013.

[361] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," *Advances in Neural Information Processing Systems*, vol. 32, pp. 5049–5059, 2019.

[362] D. Berthelot, N. Carlini, E. D. Cubuk, *et al.*, "Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring," in *International Conference on Learning Representations*, 2019.

[363] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-training with noisy student improves imagenet classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 687–10 698.

[364] B. Liu, Z. Wu, H. Hu, and S. Lin, "Deep metric transfer for label propagation with limited annotated data," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[365] J. Na, H. Jung, H. J. Chang, and W. Hwang, "Fixbi: Bridging domain spaces for unsupervised domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1094–1103.

[366] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4893–4902.

[367] A. Arnold, R. Nallapati, and W. W. Cohen, "A comparative study of methods for transductive transfer learning," in *Seventh IEEE international conference on data mining workshops (ICDMW 2007)*, IEEE, 2007, pp. 77–82.

[368] S. M. Kakade, K. Sridharan, and A. Tewari, "On the complexity of linear prediction: Risk bounds, margin bounds, and regularization," *Advances in neural information processing systems*, vol. 21, 2008.

[369] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks.," in *ICML*, vol. 2, 2016, p. 7.

[370] F. Wang, W. Liu, H. Liu, and J. Cheng, "Additive margin softmax for face verification," *arXiv preprint arXiv:1801.05599*, 2018.

[371] M. Sajjadi, M. Javanmardi, and T. Tasdizen, "Regularization with stochastic transformations and perturbations for deep semi-supervised learning," *arXiv preprint arXiv:1606.04586*, 2016.

[372] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *arXiv preprint arXiv:1703.01780*, 2017.

[373] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: A regularization method for supervised and semi-supervised learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018.

[374] Q. Xie, Z. Dai, E. Hovy, T. Luong, and Q. Le, "Unsupervised data augmentation for consistency training," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[375] J. Li, C. Xiong, and S. Hoi, "Comatch: Semi-supervised learning with contrastive graph regularization," *arXiv preprint arXiv:2011.11183*, 2021.

[376] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, "Big self-supervised models are strong semi-supervised learners," *arXiv preprint arXiv:2006.10029*, 2020.

[377] M. Assran, M. Caron, I. Misra, *et al.*, "Semi-supervised learning of visual features by non-parametrically predicting view assignments with support samples," *arXiv preprint arXiv:2104.13963*, 2021.

[378] X. Wang, L. Lian, and S. X. Yu, "Data-centric semi-supervised learning," *arXiv preprint arXiv:2110.03006*, 2021.

[379] C. Wei, K. Sohn, C. Mellina, A. Yuille, and F. Yang, "Crest: A class-rebalancing self-training framework for imbalanced semi-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 857–10 866.

[380] B. Romera-Paredes and P. Torr, "An embarrassingly simple approach to zero-shot learning," in *International conference on machine learning*, PMLR, 2015, pp. 2152–2161.

[381] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[382] W. Wang, V. W. Zheng, H. Yu, and C. Miao, "A survey of zero-shot learning: Settings, methods, and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–37, 2019.

[383] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth, "Describing objects by their attributes," in *2009 IEEE conference on computer vision and pattern recognition*, IEEE, 2009, pp. 1778–1785.

[384] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 3, pp. 453–465, 2013.

[385] A. Frome, G. Corrado, J. Shlens, *et al.*, "Devise: A deep visual-semantic embedding model," 2013.

[386] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng, "Zero-shot learning through cross-modal transfer," in *Advances in Neural Information Processing Systems*, 2013, pp. 935–943.

[387] M. Kampffmeyer, Y. Chen, X. Liang, H. Wang, Y. Zhang, and E. P. Xing, "Rethinking knowledge graph propagation for zero-shot learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 487–11 496.

[388] N. V. Nayak and S. H. Bach, "Zero-shot learning with common sense knowledge graphs," *arXiv preprint arXiv:2006.10713*, 2020.

[389] A. K. Menon, S. Jayasumana, A. S. Rawat, H. Jain, A. Veit, and S. Kumar, "Long-tail learning via logit adjustment," in *International Conference on Learning Representations*, 2021.

[390] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, "Randaugment: Practical automated data augmentation with a reduced search space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 702–703.

[391] P. Helber, B. Bischke, A. Dengel, and D. Borth, "Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 12, no. 7, pp. 2217–2226, 2019.

[392] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.

[393] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[394] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101 – mining discriminative components with random forests," in *European Conference on Computer Vision*, 2014.

[395] J. Pearl *et al.*, "Causal inference in statistics: An overview," *Statistics surveys*, vol. 3, pp. 96–146, 2009.

[396] D. B. Rubin, "Essential concepts of causal inference: A remarkable history and an intriguing future," *Biostatistics & Epidemiology*, vol. 3, no. 1, pp. 140–155, 2019.

[397] D. B. Rubin, "Causal inference using potential outcomes: Design, modeling, decisions," *Journal of the American Statistical Association*, vol. 100, no. 469, pp. 322–331, 2005.

[398] E. Bareinboim and J. Pearl, "Controlling selection bias in causal inference," in *Artificial Intelligence and Statistics*, PMLR, 2012, pp. 100–108.

[399] D. Zhang, H. Zhang, J. Tang, X.-S. Hua, and Q. Sun, "Causal intervention for weakly-supervised semantic segmentation," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[400] M. Besserve, A. Mehrjou, R. Sun, and B. Schölkopf, "Counterfactuals uncover the modular structure of deep generative models," in *International Conference on Learning Representations*, 2019.

[401] P. W. Holland, "Statistics and causal inference," *Journal of the American statistical Association*, vol. 81, no. 396, pp. 945–960, 1986.

[402] J. Pearl, *Causality*. Cambridge university press, 2009.

[403] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, "Unsupervised data augmentation for consistency training," *arXiv preprint arXiv:1904.12848*, 2019.

[404] Z. Cai, A. Ravichandran, S. Maji, C. Fowlkes, Z. Tu, and S. Soatto, "Exponential moving average normalization for self-supervised and semi-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 194–203.

[405] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Learning to prompt for vision-language models," *arXiv preprint arXiv:2109.01134*, 2021.

[406] O. Russakovsky, J. Deng, H. Su, *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015. DOI: 10.1007/s11263-015-0816-y.

[407] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[408] M. Norouzi, T. Mikolov, S. Bengio, *et al.*, "Zero-shot learning by convex combination of semantic embeddings," in *ICLR*, 2014.

[409] A. Li, A. Jabri, A. Joulin, and L. van der Maaten, "Learning visual n-grams from web data," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4183–4192.

[410] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.

[411] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and S. X. Yu, "Large-scale long-tailed recognition in an open world," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2537–2546.

[412] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3606–3613.

[413] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The german traffic sign recognition benchmark: A multi-class classification competition," in *The 2011 international joint conference on neural networks*, IEEE, 2011, pp. 1453–1460.

[414] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, IEEE, 2008, pp. 722–729.

[415] Y. LeCun, "A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27," *Open Review*, vol. 62, no. 1, pp. 1–62, 2022.

[416] S. Levine, "Understanding the world through action," in *Conference on Robot Learning*, PMLR, 2022, pp. 1752–1757.

[417] J. Gottlieb, P.-Y. Oudeyer, M. Lopes, and A. Baranes, "Information-seeking, curiosity, and attention: Computational and neural mechanisms," *Trends in cognitive sciences*, vol. 17, no. 11, pp. 585–593, 2013.