

Warsaw University of Technology



DISCIPLINE OF SCIENCE INFORMATION AND COMMUNICATION TECHNOLOGY
FIELD OF SCIENCE ENGINEERING AND TECHNOLOGY

Ph.D. Thesis

Monika Wysoczańska, M.Sc.

Task Adaptation Strategies for Vision-Language Models

Supervisor
Tomasz Trzcíński

WARSAW 2025

*“One never notices what has been done;
One can only see what remains to be done.”*
- Maria Curie-Skłodowska

Acknowledgements

I would like to express my sincere gratitude to Tomasz, my advisor, for his encouragement and for providing me the opportunity to pursue this PhD. My appreciation also extends to Jacek Komorowski, whose support as my second advisor has been invaluable. Thank you for our collaborative work, especially during the early stages of my PhD, for teaching me how to conduct research with rigor, sharing your technical expertise, and for believing in me.

I wish I could say it was an incredible journey. I wish I could say that the journey was highly inspiring and filled with great moments. It wasn't. Yet, throughout these four years, I have had the privilege of meeting remarkable individuals I might never have encountered otherwise. These relationships are what I am truly grateful for, and I would like to take a moment to acknowledge some of these exceptional people.

I am deeply grateful to my first academic home, CVLab@WUT, particularly to Kamil who has been a continuous source of motivation, for always being willing to talk about the struggles of PhD and simply being a great friend. Special thanks to Kacper, Wojtek, and Mati, my companions in the COVID cohort—it has been a genuine honor to navigate the challenges of our Doctoral School together.

I am thankful to Imagine Lab at ENPC for their hospitality and exemplary academic environment. Thanks to Tom, Mathis, Lucas, Nicolas, Nguyen, Hannah, Georgy, Nermin, and especially Gul, an outstanding role model for women in AI and Computer Vision. Collaborating with Tom was a privilege. Thank you for your help, and motivation, and for sharing your deadline strategies and incredible skills.

I extend my appreciation to the entire Booking.com team for our collaboration during my internship in Amsterdam. Thank you Karen for providing me this opportunity, and Moran for your dedication to my project, motivational guidance, and constructive feedback delivered with empathy and support.

I would like to thank the team at Google DeepMind in Grenoble. Thank you Mathilde for the opportunity and support throughout the internship. You, Thomas, and Lison made my Grenoble stay exceptional. Our Mauna Kea adventure in Hawaii remains among my most extraordinary experiences. Thank you Ahmet for our coffee discussions, mentoring, and politically incorrect humor. Thank you Emanuele for your consistent support during the internship and your kindness. Many thanks to Shyamal and Anurag for their mentorship, and for helping navigate Google's infrastructure and complexities—I truly value our collaboration. Finally, thank you Cordelia for this remarkable opportunity; it was an honor to work alongside you.

Perhaps the most exciting and brightest chapter of this journey was my research stay at Valeo.ai. Special thanks to Patrick for kindly hosting me and letting me benefit

from this amazing research environment he had built. I'm grateful to Matthieu for letting me continue our collaboration and for always being kind. I would like to sincerely thank every single member of the team, from scientists Renaud, Spyros, Gilles, Alexandre, Eloi, Victor to PhD students Amaia, Sophia, Tetiana, Björn, Loick, Victor, Antonin. Thank you for making me finally believe that my research is interesting and that it can be a valuable scientific contribution. Antonin, thank you for showing me that research can also be fun. Working with you was a pleasure, and I appreciate your trust in my ideas and project vision.

Throughout this time, I was extremely fortunate to meet people who became my mentors in research and life. David, thank you for inviting me to Imagine, guiding me, and being deeply involved in my project—a novel experience for me then. Thank you for being available to discuss a broad range of topics from computer vision to communism and climate change. I am also grateful to Andrei for his kindness, support, and confidence in me. Your vast knowledge of related work and exceptional writing skills are impressive. Most importantly, however, thank you for being a great human being. Andrei and David showed me that kindness can exist in this difficult, sometimes cruel research environment, allowing me to be authentic with all my vulnerabilities.

I could write another tens of pages expressing my gratitude to you, Oriane. Thank you for your belief in me and for letting me in Valeo. Dropping you an email was a desperate and probably most random thing I did during this PhD, yet you replied, and the result was nothing but great. Thank you for trusting me, strengthening, inspiring, and motivating me. Thank you for your dynamism, persistence, and dedication to our project, and for sharing your technical and research expertise.

Meeting you, Mic, was undoubtedly the brightest moment of all. This work would never have been possible without you. Thank you for your support and encouragement to complete my PhD. Thank you for being present through all — from coding hacky solutions for quick iterations, through creating impressive figures worthy of popular retweets, to providing comfort and assistance when dealing with unreasonable Reviewers 2. I am completing my PhD with my head held high thanks to you. You are an exceptional partner in research and life. Thank you for being you.

Na koniec chciałabym podziękować mojej rodzinie. Siostro, dziękuję za bezwarunkowe wsparcie i wiarę w moje możliwości. Mamuś, Tatuś, dziękuję za nieustanną akceptację moich, często nietypowych, pomysłów. Dziękuję za wsparcie, za zaszczepienie we mnie chęci ciągłego rozwoju i nauki, za opiekę nad Pajtkiem i zapewnienie poczucia bezpieczeństwa. Mimo częstych zmian miejsca pobytu zawsze wiedziałam, że mam bezpieczną przystań, do której mogę wrócić. Kocham Was bardzo i mam niezwykle szczęście mieć taką rodzinę. Te kolejne strony dedykuję właśnie Wam.

Task Adaptation Strategies for Vision-Language Models

Vision foundation models have emerged as pivotal architectures in artificial intelligence, transforming computational perception by distilling vast visual knowledge into powerful, generalizable representations. This thesis explores the capabilities and adaptability of these models with a particular focus on Vision-Language Models (VLMs), such as CLIP, examining how their image-text-aligned representations can be extended to tasks requiring fine-grained visual understanding. Through a series of five works, we address the central question: To what extent can off-the-shelf visual representations from foundation models be leveraged for various downstream tasks with minimal task-specific supervision?

First, we introduce CLIP-DIY, demonstrating how CLIP’s capabilities can be extended to open-vocabulary semantic segmentation through modified inference strategies rather than model retraining. We further develop CLIP-DINOiser, a method that enhances dense CLIP representations by incorporating priors from self-supervised representations through a lightweight adaptation module. Our third contribution explores the critical role of textual prompts in VLM performance, showing that understanding pre-training data distribution can be an effective strategy for improved downstream performance.

Moving beyond model adaptation, we propose a systematic evaluation framework for visual representations using Visual Question Answering as an exemplary task, providing insights into which foundation models are best suited for this application. Finally, we demonstrate the potential of visual foundation models to address complex real-world challenges, such as the personalization of recommended content. Specifically, we develop a method using a VLM for personalized image collection summarization, leveraging the model’s multimodal capabilities while requiring almost no manual annotations.

Through these contributions, we establish that visual foundation models can be effectively adapted to complex visual understanding tasks with minimal computational and annotation requirements. Our findings advance the field’s understanding of efficient model adaptation strategies while providing practical solutions for exploiting powerful visual systems across diverse downstream applications.

Keywords: downstream task adaptation, image-text alignment, open-world perception, unsupervised/self-supervised learning

Strategie Adaptacji Modeli Wizualno-Językowych do Zadań Docelowych

Niniejsza praca doktorska bada adaptacyjność modeli fundamentalnych do reprezentacji obrazu, do złożonych zadań docelowych, ze szczególnym naciskiem na podejścia wykorzystujące wewnętrzne możliwości modeli fundamentalnych przy minimalizacji kosztów ich modyfikacji, w tym dodatkowego treningu.

Nasze badania koncentrują się głównie na modelach wizualno-językowych (VLM), analizując, jak ich multimodalne reprezentacje obrazu i tekstu mogą być rozszerzone do zadań wymagające szczegółowej reprezentacji obrazu. Poprzez pięć powiązanych ze sobą prac, odpowiadamy na centralne pytanie: W jakim stopniu reprezentacje wizualne z modeli fundamentalnych mogą być wykorzystane do różnych zadań docelowych przy minimalnym nadzorze?

W pierwszej części doktoratu omawiamy metody adaptacji modelu CLIP do zadania semantycznej segmentacji z nieograniczonym słownikiem klas. Nasza pierwsza metoda, CLIP-DIY, demonstruje, możliwości adaptacji CLIP poprzez modyfikacje sposobu inferencji tym samym bez konieczności trenowania modelu. Następnie prezentujemy CLIP-DINOiser, metodę, która wzbogaca reprezentacje CLIP na poziomie pikseli poprzez wykorzystanie komplemetarnych umiejętności lokalizacji obiektów z reprezentacji uczonych z samonadzorem, takich jak DINO, za pomocą prostego modułu adaptacyjnego. W tej części pracy pokazujemy również jak poprawić skuteczność segmentacji poprzez starannie dobrane prompty tekstowe, które pozyskujemy poprzez analizę dużego korpusu danych wykorzystanych do trenowania VLM.

Wykraczając poza adaptację modelu, kolejna część doktoratu prezentuje metodę do ewaluacji reprezentacji wizualnych w złożonym zadaniu Visual Question Answering (VQA). Tak skonstruowany sposób ewaluacji pozwala dogłębnie zrozumieć skuteczność poszczególnych reprezentacji wizualnych do zadań rozumowania na podstawie obrazu. W ostatniej części pracy pokazujemy praktyczne zastosowanie adaptacji VLM do zadania personalizacji wizualnych podsumowań na przykładzie problemu na platformie Booking.com, łącząc analizę reprezentacji wizualnej z analizą tekstowych recenzji użytkowników platformy.

Podsumowując, niniejsza praca pokazuje, że modele fundamentalne do reprezentacji obrazu mogą być efektywnie adaptowane do złożonych zadań wymagających szczegółowego rozumienia obrazu przy minimalnych wymaganiach obliczeniowych i anotacyjnych. Nasze obserwacje pogłębiają wiedzę dotyczącą różnych sposobów reprezentacji obrazu jednocześnie dostarczając praktycznych rozwiązań do adaptacji modeli fundamentalnych w różnorodnych zastosowaniach.

Słowa kluczowe: metody adaptacji do zadań złożonych, reprezentacje wizualno-

tekstowe, segmentacja semantyczna z nieograniczonym słownikiem, uczenie nienadzorowane/ samonadzorowane

Contents

Acknowledgements	5
1. Introduction	16
1.1. Motivation & Challenges	16
1.2. Thesis Objective & Outline	18
1.2.1. Downstream task adaptation at minimal cost	19
1.2.2. Task adaptation for model selection and analysis	21
1.2.3. Task adaptation to real-world scenarios	22
1.3. List of contributions	23
1.4. Works not Included in the Dissertation	25
2. Background	26
2.1. Visual Foundation Models	26
2.1.1. Label supervision	26
2.1.2. Image-only self-supervision	27
2.1.3. Language supervision	28
2.1.4. Hybrid approaches	30
2.2. Downstream task adaptation strategies	30
2.2.1. Dataset adaptation	30
2.2.2. Parameter-Efficient Fine-tuning	30
2.2.3. Feature Extraction and Linear Probing	31
2.2.4. Zero-shot and Few-shot Adaptation	31
2.2.5. Adaptation for Dense Prediction Tasks	31
2.2.6. Multimodal Adaptation	32
3. Task Adaptation through Modified Inference	34
3.1. Introduction	35
3.2. Related work	36
3.2.1. Zero-shot open-vocabulary semantic segmentation	37
3.2.2. Unsupervised object localization	37
3.2.3. Combining self-supervised features & CLIP	39

3.3. CLIP-DIY	39
3.3.1. Dense inference with CLIP	39
3.3.2. Guided segmentation	41
3.4. Experiments	42
3.4.1. Experimental setup	42
3.4.2. Results	43
3.4.3. Ablations	44
3.4.4. Failure cases	46
3.4.5. Our method in the wild	49
3.5. Conclusions	49
3.6. Additional material	49
3.6.1. Foreground-background segmenters	49
3.6.2. More qualitative results	50
4. Leveraging Complementary Visual Foundation Model	55
4.1. Introduction	56
4.2. Related Work	58
4.3. Method	59
4.3.1. Problem statement	60
4.3.2. Preliminaries on MaskCLIP	60
4.3.3. DINOising open-vocabulary features	61
4.3.4. Teaching CLIP a first DINO trick: object correlations	62
4.3.5. Teaching CLIP a second DINO trick: background filtering	63
4.4. Experiments	65
4.4.1. Experimental setup	65
4.4.2. Open-vocabulary semantic segmentation	66
4.4.3. Ablation study	69
4.5. Conclusions	70
4.6. Additional material	70
4.6.1. The impact of the training dataset	70
4.6.2. Self-supervised features discussion	71
4.6.3. Background evaluation with FOUND	72
4.6.4. Details on the methods compared	72
4.6.5. More qualitative results	72
4.6.6. Failure modes	74
5. Leveraging Statistics from Pre-training Dataset	76
5.1. Introduction	77

5.2. Related work	79
5.3. Open-world open-vocabulary segmentation with test-time contrastive concepts	81
5.3.1. Introducing test-time contrastive concepts	81
5.3.2. Contrasting with “background” (\mathcal{CC}^{BG})	83
5.3.3. Automatic contrastive concepts (\mathcal{CC}) generation	84
5.4. Evaluation	86
5.4.1. Evaluating open-world segmentation	86
5.4.2. Evaluated methods	87
5.4.3. Contrastive concepts generation results	88
5.4.4. Ablation studies	90
5.4.5. Qualitative results	90
5.5. Conclusion	91
5.6. Additional material	92
5.6.1. Details on the evaluation	92
5.6.2. About the IoU-single metric	93
5.6.3. More quantitative results	93
5.6.4. Failure case analysis	94
5.6.5. More qualitative results	95
5.6.6. Hyperparameter selection	95
5.6.7. Average number of contrastive concepts vs performance	98
5.6.8. On separability of CLIP patch-features	99
5.6.9. Replacing \mathcal{CC} with sigmoid operation	99
5.6.10. Ontology-based filtering with WordNet	101
5.6.11. Prompting the LLM	101
6. Task Adaptation for Foundation Model Selection and Analysis	105
6.1. Introduction	106
6.2. Related work	107
6.2.1. Visual question answering (VQA)	108
6.2.2. Generic representation learning	109
6.3. Evaluation framework	110
6.3.1. Motivation	111
6.3.2. Pipeline overview	111
6.3.3. Evaluation constraints	112
6.4. Experiments	114
6.4.1. Datasets	114
6.4.2. Implementation details	115

6.4.3. Quantitative results	118
6.4.4. Low-shot evaluation study	120
6.5. Conclusions	121
6.6. Appendix	122
6.6.1. Visual encoders extraction and adaptation details	122
6.6.2. Memory adaptation details	123
6.6.3. Training details	123
6.6.4. DINO ViT architecture comparison	124
6.6.5. CLEVR-Math experiments	124
7. Task adaptation in Real-world Scenarios	125
7.1. Introduction	126
7.2. Method	128
7.2.1. Task Definition	129
7.2.2. Image Embeddings & Filtering	129
7.2.3. Clustering	130
7.2.4. Text2topic: Topics Detection Model	130
7.2.5. Matching Images to Topics	130
7.2.6. Evaluation Metrics	131
7.3. Experiments	133
7.3.1. Experimental Setup	133
7.3.2. Offline Evaluation	134
7.3.3. User Studies	136
7.4. Application	137
7.4.1. Deployment & Maintenance	137
7.4.2. Application example	138
7.5. Conclusions & Limitations	139
7.6. Additional material	139
7.6.1. Definition of relevant image classes	139
8. Final Remarks	141
8.1. Summary of contributions	141
8.2. Open problems	141
Bibliography	144
List of Figures	172
List of Tables	175

1. Introduction

1.1. Motivation & Challenges

Vision is a fundamental sensory channel through which humans and other living beings perceive and understand their environment. A central goal in artificial intelligence (AI) has been to develop computer vision systems that can match this remarkable ability to process and interpret visual information [1]. Visual perception underlies many aspects of intelligent behavior, from recognizing objects and actions in complex scenes to understanding spatial relationships and temporal dynamics. Therefore, the goal of emulating human visual capabilities has driven innovation in computer vision for decades, motivated by its wide potential across society, from AI systems supporting medical diagnosis [2] to autonomous vehicles enhancing transportation safety [3] and advanced tools revolutionizing creative industries [4].

From specialized solutions to adaptation of generalist foundation models.

The emergence of foundation models has fundamentally transformed the landscape of AI. In the context of computer vision, *visual foundation models* (VFM) *translate raw perceptual information from diverse sources and sensors* [5]. This approach represents the natural evolution of computer vision over the past decade. With the introduction of ImageNet [6] and supervised pre-training, the field shifted significantly from traditional task-specific feature engineering [7, 8] towards more versatile models. These models, trained on large-scale datasets, can be adapted for various tasks, from image recognition to segmentation [9, 10, 11]. The fundamental principle of *learn-once, adapt-many* continues to drive the development of foundation models.

What advantages do foundation models offer over specialized models for different tasks? Foundation models offer several compelling advantages over maintaining multiple specialized models. From an efficiency standpoint, a single foundation model that can be adapted to various tasks requires significantly fewer computational resources than separate models for each task. This approach also enables knowledge transfer across domains, where concepts learned from one task can benefit the performance on others [12]. Furthermore, foundation models demon-

strate superior robustness and generalization capabilities since they learn from diverse data and tasks [13]. Finally, and perhaps most importantly, foundation models can leverage their broad understanding to tackle novel tasks with minimal additional training, providing the flexibility that individual specialized models cannot match [14].

Current landscape of visual foundation models. The current landscape of visual foundation models is broad, from models for visual understanding [9, 13, 15] to generative approaches [16, 17]. This thesis focuses on the first group – visual perception models, where we primarily distinguish three representation paradigms based on their supervision. One paradigm relies on label supervision, predominantly through image classification tasks. This methodology has been refined through seminal datasets such as ImageNet [18], with industry research extending to proprietary large-scale labeled collections [19]. The second paradigm employs image-only self-supervision, where the model learns representations directly from the inherent structure of visual data. This approach encompasses various methodologies, from contrastive [20, 21] and non-contrastive learning frameworks [15, 22] to masked image modeling techniques [23], which paved the most promising path towards unsupervised visual representation learning. The third paradigm benefits from language supervision, also called Vision-Language Models (VLM), utilizing image-text pairs widely available on the Internet. Models trained through this approach, such as CLIP [13] and ALIGN [24], have demonstrated remarkable capabilities in zero-shot settings. In this thesis, we focus mainly on the last group, as image-text-aligned representations opened new avenues for research. We believe that VLMs’ adaptation strategies and an in-depth understanding of their cross-modal space remain largely unexplored.

Challenges in the adaptation of foundation models to downstream tasks.

Visual foundation models exhibit remarkable generalization capabilities, yet their adaptation to downstream tasks poses some challenges. The conventional approach of task-specific fine-tuning with full supervision proves effective but presents significant limitations. First, fine-tuning large foundation models end-to-end is expensive, as large models typically require substantial computational resources for training. Some research efforts address this limitation with parameter-efficient methodologies [25]. However, these achievements may not be as effective for complex downstream tasks [26]. Dense-level task annotations are particularly expensive to obtain, and certain domains face inherent constraints in collecting adequate labeled datasets. For example, in medical imaging, annotations can only be provided by expert radiologists or pathologists for tumor segmentation [27]. This annotation challenge is further

compounded when models require adaptation to evolving operational conditions, as each iteration requires traversing the complete cycle from annotation acquisition to model retraining. It is therefore appealing to consider solutions that reduce the need for human annotations and task-specific supervision.

Alternatively, one could exploit visual foundation models directly with no modifications. While for image-level tasks such as classification and retrieval, these models can typically be adapted with minimal modifications, tasks requiring more fine-grained understanding expect more tailored solutions. For example, CLIP, trained with image-level objective, performs poorly on semantic segmentation when used off-the-shelf [28]. The discrepancy stems from the fundamental mismatch between the global image-text alignment objective used during pre-training and the dense prediction requirements of pixel-level understanding tasks.

In this thesis, we focus on the latter approach and investigate how visual foundation models and VLMs, in particular, can be adapted to downstream tasks with minimal computational effort.

1.2. Thesis Objective & Outline

The goal of this thesis is to explore the adaptability of visual foundation models for specific downstream applications, with particular emphasis on tasks that go beyond image classification, requiring visual understanding at a fine-grained level. Rather than extensively modifying these models, we explore strategies that preserve their learned representations while extending their capabilities to new contexts. By using "off-the-shelf" models, we aim to maximally leverage the inherent flexibility of these visual understanding systems while maintaining their original design. This approach leads to our central research question: *To what extent can off-the-shelf visual representations from foundation models be leveraged for various downstream tasks with minimal task-specific supervision?* We structure our study around several specific research questions to systematically address this broader question.

In Chapter 2, we start with an overview of visual foundation models, analyzing their unique capabilities and limitations. We give particular attention to Vision-Language Models. Although these models demonstrate remarkable zero-shot performance, their training focuses primarily on image-level tasks, creating challenges when applied to tasks that require fine-grained visual understanding. This limitation is the primary motivation for the first section of this work, in which we explore novel approaches to overcome these constraints.

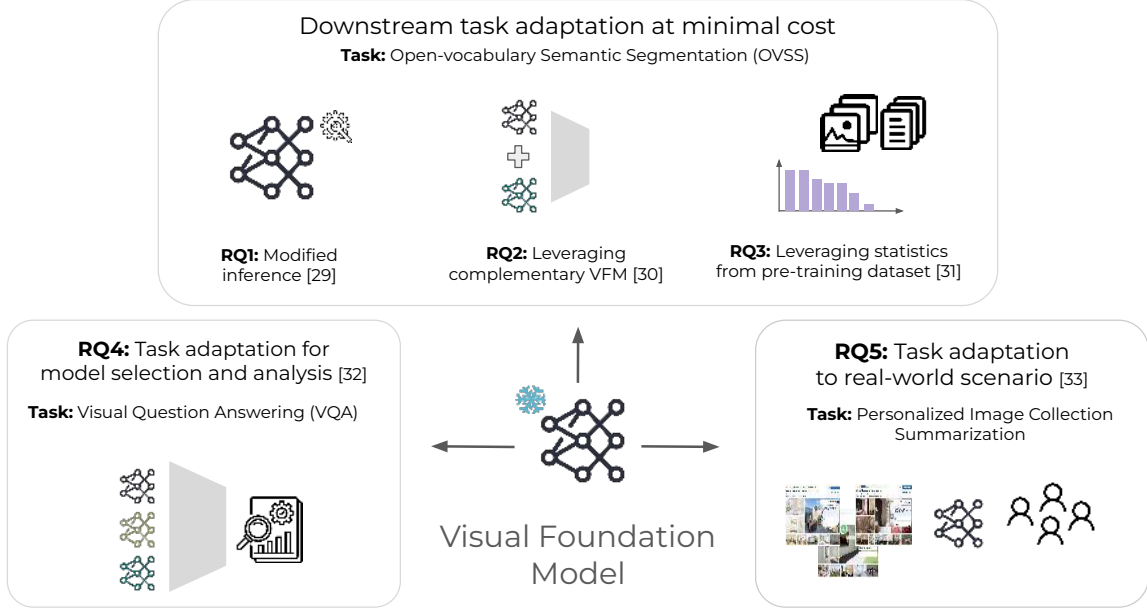


Figure 1.2.1. The outline of the thesis. We examine how visual foundation models adapt to downstream applications beyond image classification. First, we address computational costs when adapting VLMs for Open-Vocabulary Semantic Segmentation. Next, we explore the evaluation and selection of optimal VFMs for specific tasks, using Visual Question Answering as an example. Finally, we investigate VLMs’ capabilities in navigating complex real-world scenarios like content personalization.

1.2.1. Downstream task adaptation at minimal cost

The first part of this thesis addresses the resource-intensive nature of modifying large foundation models by investigating adaptation strategies that require minimal computational training and eliminating the need for manual data annotation.

Research Question 1: *Can we adapt visual foundation models for fine-grained localization tasks without training?*

To address the first question, we focus our investigation on open-vocabulary semantic segmentation (OVSS). Chapter 3 introduces our first contribution, demonstrating how CLIP’s capabilities can be extended to OVSS tasks through modified inference strategies rather than model retraining. Our approach, CLIP-DIY, introduces a multi-scale architecture that uses CLIP’s classification abilities across different spatial resolutions. We enhance segmentation accuracy by incorporating foreground/background separation scores derived from unsupervised object localization techniques. Notably, CLIP-DIY achieves this without requiring additional training or manual annotations, instead building upon existing unsupervised lo-

calization methods. Our method yields competing results on classical semantic segmentation benchmarks compared to methods requiring training.

Chapter 3 is based on the publication *"CLIP-DIY: CLIP Dense Inference Yields open-vocabulary semantic segmentation for free"* by Monika Wysoczańska, Michael Ramamonjisoa, Tomasz Trzcinski and Oriane Simeoni presented at IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) 2024.

Our CLIP-DIY approach combines CLIP with an unsupervised saliency detection method, which is built upon DINO, a self-supervised learning (SSL) foundation model demonstrating strong object localization capabilities without requiring fine-grained supervision. The successful integration in CLIP-DIY of these complementary approaches—combining vision-language modeling (CLIP) with self-supervised learning—inspired us to further investigate the potential synergies between these two distinct paradigms.

Research Question 2: *Can we leverage the complementarity of different visual representations for improved downstream adaptation?*

In our next contribution, motivated by the observation that image-text-aligned and self-supervised visual representations exhibit complementary capabilities, we propose CLIP-DINOiser, a method which takes *the best of both worlds*. Our approach enhances dense CLIP representations by incorporating DINO’s self-supervised localization capabilities, eliminating the need for task-specific annotations. Crucially, we demonstrate that these localization priors from DINO can be directly integrated into CLIP’s representation space, avoiding the computational overhead of running two large models. This is achieved through a lightweight adaptation module trained with DINO’s supervision while preserving CLIP’s original representations. CLIP-DINOiser achieves state-of-the-art results on challenging and fine-grained benchmarks while maintaining minimal computational requirements: it needs only a single CLIP forward pass and two lightweight modules during inference, with no additional supervision or memory overhead.

Chapter 4 is based on the publication *"CLIP-DINOiser: Teaching CLIP a few DINO tricks for open-vocabulary semantic segmentation"* by Monika Wysoczanska, Oriane Simeoni, Michael Ramamonjisoa, Andrei Bursuc, Tomasz Trzcinski and Patrick Perez presented at the European Conference on Computer Vision (ECCV) 2024.

Our work on CLIP-DINOiser revealed that text prompt engineering plays a crucial role in open-vocabulary semantic segmentation and open-vocabulary tasks

in general. We noticed that a good set of textual prompts can drastically improve a concept’s segmentation performance, which leads us to our next research question.

Research Question 3: *How can we leverage models’ pre-training data statistics for improved downstream performance?*

To address this question, we conduct a deeper investigation into the nuances of open-vocabulary semantic segmentation. We discover that the performance of VLM-based semantic segmentation is highly sensitive to the choice of textual prompts at test time. Specifically, we observe that segmentation accuracy for a given concept can be substantially improved by introducing carefully selected contrasting concepts—a phenomenon that aligns with CLIP’s contrastive learning foundation. In our next contribution (Chapter 5), we explore this dynamic by analyzing the frequency distribution of target dataset concepts within CLIP’s pre-training data. Based on these insights, we develop two automated approaches for generating effective contrasting concepts at test time: one utilizing an external model and another leveraging statistical patterns from the VLM’s pre-training dataset. In addition, we identify limitations in current OVSS evaluation benchmarks, leading us to propose a new evaluation framework that better reflects real-world challenges.

Chapter 5 is based on a work *“Test-time Contrastive Concepts for Open-world Semantic Segmentation with Vision-Language Models”* by Monika Wysoczanska, Antonin Vobecky, Amaia Cardiel, Tomasz Trzcinski, Renaud Marlet, Andrei Bursuc, Oriane Simeoni. The work is currently under review for Transactions on Machine Learning Research.

1.2.2. Task adaptation for model selection and analysis

In the next part of the thesis, we will examine task adaptation from a different perspective. As we explore visual representations throughout this thesis, a natural question arises.

Research Question 4: *How to determine which visual foundation model is the best for specific downstream tasks?*

In our next contribution, detailed in Chapter 6, we explore an alternative perspective on task adaptation. When visual representations are used off-the-shelf without modification, the process of task adaptation can serve as a framework for evaluation. By utilizing unmodified visual representations, one can isolate and assess their quality, as any performance gaps directly reflect the representation’s encoded properties. This approach establishes a standardized testing environment where different visual models can be compared under identical downstream conditions, effectively revealing which aspects are naturally embedded within each

representation. Building on this insight, we propose using task adaptation as a systematic protocol to assess different visual representations. To thoroughly test this approach, we focus on visual reasoning—a complex task that requires a detailed and high-level understanding of a visual scene. Specifically, we introduce a protocol to evaluate off-the-shelf visual representations for Visual Question Answering (VQA). To facilitate the comparison and integration of various visual representations, we design a universal reasoning module with a flexible modification of input representation sizes to allow for fair studies. This evaluation framework allows us to study the differences between various visual representations and determine which is most suitable for the selected task. Taking VQA as an exemplary task, we make several findings that can further foster the development of stronger visual representations. **Chapter 6** is based on a work *"Towards unsupervised visual reasoning: Do off-the-shelf features know how to reason?"* by Monika Wysoczańska, Tom Monnier, Tomasz Trzciński, David Picard presented first at NeurIPS 2022 Workshop: Self-Supervised Learning - Theory and Practice & accepted in a full format for IEEE Access (2024).

1.2.3. Task adaptation to real-world scenarios

In our investigations so far, we have demonstrated the adaptability of visual foundation models across different downstream tasks, reducing both human annotation requirements and computational overhead. We now turn our attention to a more ambitious challenge: the real world. The final part of this thesis explores whether visual foundation models can successfully navigate complex real-world scenarios.

Research Question 5: *Can we adapt VLM's representation to real-world challenges, such as content personalization?*

Content personalization is one of the representative challenges in real-world applications, particularly for digital platforms. Contemporary web services must deliver content tailored to individual user preferences, a capability that extends beyond academic benchmarks to directly influence user experience. This requirement for customized content presentation has emerged as a fundamental determinant of user engagement and satisfaction across digital ecosystems. As users increasingly expect interfaces and recommendations that reflect their specific interests and behavioral patterns, the ability to adapt content delivery becomes essential. Chapter 7 discusses some practical aspects of adapting a VLM to a personalization problem of summarizing large image collections. While traditional image collection summarization task aims to create concise visual summaries through carefully selected subsets, web platforms face an additional challenge: different users have different preferences and priorities when viewing the same content. This challenge is

particularly relevant for Booking.com, where it effectively presents property previews that match users’ specific interests and directly impact their decision-making process. We address this by developing CrossSummarizer, a method that personalizes hotel visual summaries by analyzing textual reviews from previous travelers’ experiences to identify key aspects that matter to particular groups of users. In our approach, we employ a VLM to establish connections between visual content and textual user feedback. As a result, our cross-modal strategy creates more relevant and personalized visual summaries without requiring additional manual annotations. Through comprehensive evaluation, including human perceptual studies, we demonstrate that CrossSummarizer significantly outperforms both non-personalized approaches and baseline clustering-based methods.

Chapter 7 is based on a work *“Tell me what is good about this property – leveraging reviews for segment-personalized image collection summarization”* by Monika Wysoczańska, Moran Beladev, Karen Lastmann Assaraf, Fengjun Wang, Ofri Kleinfeld, Gil Amsalem, Hadas Harush Boker presented at AAAI 2024.

Chapter 8 concludes this thesis by summarizing contributions and giving an outlook of open problems in task adaptation of visual foundation models.

1.3. List of contributions

This thesis comprises five first-authored works, four of which were accepted to peer-reviewed conferences or journals. The following sections describe each work and delineate the PhD candidate’s contributions.

[29] *CLIP-DIY: CLIP Dense Inference Yields open-vocabulary semantic segmentation for free.* Monika Wysoczańska, Michael Ramamonjisoa, Tomasz Trzcinski and Oriane Simeoni. Accepted at IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) 2024. (140 MEiN points)

As the first author, the PhD candidate led all aspects of this work, from the design of CLIP-DIY to manuscript preparation. Michael Ramamonjisoa and Oriane Simeoni provided technical guidance and contributed to manuscript preparation. Tomasz Trzcinski provided high-level supervision.

[30] *CLIP-DINOiser: Teaching CLIP a few DINO tricks for open-vocabulary semantic segmentation.* Monika Wysoczańska, Oriane Simeoni, Michael Ramamonjisoa, Andrei Bursuc, Tomasz Trzcinski and Patrick Perez. Accepted at European Conference on Computer Vision (ECCV) 2024. (200 MEiN points)

As the first author, the PhD candidate led all technical aspects of the work, including design, implementation of CLIP-DINOiser and experimental validation, ablation studies, and manuscript writing. Oriane Simeoni participated in the initial experiments, provided supervision throughout the project, and coordinated the writing of the paper. Michael Ramamonjisoa and Andrei Bursuc provided technical guidance and contributed to manuscript preparation. Tomasz Trzciński and Patrick Perez provided high-level supervision.

[31] *Test-time Contrastive Concepts for Open-world Semantic Segmentation with Vision-Language Models.* Monika Wysoczańska, Antonin Vobecky, Amaia Cardiel, Tomasz Trzciński, Renaud Marlet, Andrei Bursuc, Oriane Simeoni. Under review for Transactions on Machine Learning Research (TMLR).

The PhD candidate served as the primary investigator, leading all aspects of the research, including direction setting, problem formulation, methodology development, implementation of the two proposed solutions, experimental validation, and paper writing. Antonin Vobecky supported the experimental validation phase and contributed to the preparation of the manuscript. Amaia Cardiel provided technical expertise for one of the two proposed solutions. Tomasz Trzciński provided high-level supervision. Renaud Marlet provided research supervision and significantly contributed to the manuscript preparation process. Andrei Bursuc provided research supervision and participated in the manuscript preparation. Oriane Simeoni made significant contributions to manuscript writing and the design of the technical solution while providing ongoing project supervision.

[32] *Towards unsupervised visual reasoning: Do off-the-shelf features know how to reason?* Monika Wysoczańska, Tom Monnier, Tomasz Trzciński, David Picard. Accepted at NeurIPS 2022 Workshop: Self-Supervised Learning - Theory and Practice & IEEE Access (2024). (100 MEiN points)

The PhD candidate led the project's core development, including conceptualization, methodology design, implementation, and experimental validation, as well as preparation of the initial draft. Tom Monnier contributed to the project's conceptual framework and methodology and played a key role in manuscript preparation and revision. Tomasz Trzciński provided resources and a manuscript review. David Picard provided comprehensive supervision and guidance, contributing to the project's conceptualization and methodology, while also providing resources and manuscript writing.

[33] *Tell me what is good about this property – leveraging reviews for segment-personalized image collection summarization.* Monika Wysoczańska, Moran Beladev, Karen Lastmann Assaraf, Fengjun Wang, Ofri Kleinfeld, Gil Amsalem, Hadas Harush Boker. 39th Annual AAAI Conference on Artificial Intelligence (AAAI-IAAI) 2024. (200 MEiN points)

The project was developed during PhD candidate’s research internship at Booking.com. The PhD candidate led the project’s core development, including conceptualization, design, and implementation of the CrossSummarizer approach and its experimental validation, as well as preparation of the entire manuscript and revision. Moran Beladev provided technical advice and general supervision throughout the project, as well as contributed to the manuscript preparation and review. The rest of the team provided technical support and strategic oversight.

1.4. Works not Included in the Dissertation

PhD Candidate, through other collaborations, has co-authored additional works that are not part of this thesis, including:

1. *OVFact: Measuring and Improving Open-Vocabulary Factuality for Long Caption Models.* **Monika Wysoczańska**, Shyamal Buch, Anurag Arnab, Cordelia Schmid. Under review for the 63rd Annual Meeting of the Association for Computational Linguistics (ACL) 2025.
2. **[34] *EgoNN: Egocentric Neural Network for Point Cloud Based 6DoF Relocalization at the City Scale.*** Jacek Komorowski, **Monika Wysoczańska**, Tomasz Trzciński. Robotics and Automation Letters (RA-L) 2022. (200 MEiN points)
3. **[35] *MinkLoc++: Lidar and Monocular Image Fusion for Place Recognition.*** Jacek Komorowski, **Monika Wysoczańska**, Tomasz Trzciński. International Joint Conference on Neural Networks (IJCNN) 2021 (140 MEiN points)

2. Background

In this chapter, we provide background on visual foundation models necessary for further discussion. We begin in Sec. 2.1 with a non-exhaustive overview of various visual foundation model architectures, with particular emphasis on two predominant learning paradigms: self-supervised learning and image-text alignment. Following, Sec. 2.2 examines selected methodologies for adapting these foundation models to downstream tasks, highlighting key strategies that maximize their utility across various application domains.

2.1. Visual Foundation Models

Pre-training robust visual backbones is fundamental to downstream computer vision tasks. The broad literature on visual representation learning can be categorized into three main groups, differentiated by their supervision paradigms, which we discuss in detail in the following.

2.1.1. Label supervision

Pre-training on large-scale human-labeled datasets such as ImageNet [6] and ImageNet21K [18] has become a standard approach for developing transferable visual representations. The core idea is to map images to discrete labels representing visual concepts. The availability of such large-scale human-labeled datasets has catalyzed significant architectural innovations, including AlexNet [9], ResNet [36], Vision Transformer [37], and Swin Transformer [38], and thus label supervision serves as the experimental foundation for contemporary vision backbones. These supervised representations have enabled advances across diverse computer vision tasks, from classification and object detection/segmentation to visual question answering, image captioning, and video action recognition, which adopted *ImageNet pretraining* and pre-trained backbones in their pipelines. However, the effectiveness of such representations remains bounded by the amount of human-annotated data, which is expensive to obtain, and annotations themselves can introduce unnecessary biases.

2.1.2. Image-only self-supervision

Motivated by the aforementioned limitations of label supervision, a substantial body of work explores image-only self-supervised learning methods for visual representation learning. These techniques derive supervision signals intrinsically from the images themselves. We can divide them into masked image modeling [23, 39], contrastive learning-based approaches [40, 20], and non-contrastive learning frameworks [22, 41, 15, 42].

Masked Image Modeling (MIM) refers to a distinctive paradigm in self-supervised visual representation learning characterized by its reconstruction-based approach which was directly inspired by achievements in Natural Language Processing (NLP), i.e. BERT [43]. MIM methods explicitly mask portions of input images and train models to predict the masked content. The specific masking strategy significantly impacts the representation power, with techniques such as block-wise masking and attention-guided masking designed to create more challenging objectives that encourage richer feature learning for dense prediction tasks. The Masked Autoencoder (MAE) [23] is one of the pioneering works in adopting the MIM approach in the visual domain. MAE is an encoder-decoder architecture that processes partially masked images to reconstruct the original, unmasked content. This reconstruction task compels the model to develop representations capturing the underlying image structure. Alternatively, BEiT [39] and iBOT [44] predict discrete visual tokens corresponding to masked patches rather than pixel values.

Contrastive methods develop representations by creating positive image pairs through augmentations and employing a loss function that maximizes their similarity while minimizing similarity to other samples. The implementation of this approach is a softmax-based classifier distinguishing the positive pair among multiple negative pairs. To learn general features, contrastive implementation requires comparing features from a large number of images simultaneously. To achieve this goal, memory bank approaches [45] store features from previous epochs but face scalability limitations. Momentum contrast (MoCo) [20] maintains consistency through an auxiliary network that computes an exponential moving average of the main network’s parameters. Later, MoCov2 [40] utilize examples from the current mini-batch as positive and negative pairs.

Non-contrastive methods share the fundamental goal of aligning representations from different transformations of the same input with contrastive methods. However, they distinctively eliminate the need for negative examples from the learning process.

This approach requires specific strategies to prevent representation collapse, where the model outputs identical representations regardless of input.

BYOL [22] addresses collapse through exponential moving average (EMA) for one backbone and asymmetric projection heads. SimSiam [46] later demonstrated that, while EMA provides minimal benefits, projection head asymmetry is critical to performance.

Perhaps the most well-known non-contrastive method, DINO [15] frames the representation learning problem as knowledge distillation [47] between two image encoders, a teacher and a student. The framework processes two distinct random transformations of a source image through parallel student and teacher networks. Although these networks share an identical architecture, they maintain separate parameters. The training process is as follows. First, student and teacher networks take different views of the same image and produce K -dimensional feature vectors. The output of the teacher network is first centered using a batch-computed mean value. Then outputs of both networks are normalized via temperature-controlled softmax operations across the feature dimension. Next, cross-entropy loss quantifies the similarity between these normalized representations, and the gradients are only propagated through the student network while the teacher network is updated with an *exponential moving average* (EMA) of the student’s network parameters. A follow-up DINOv2 [48] further improves DINO by integrating MIM in embedding space (iBOT [44]) and scaling the training dataset. A particularly important property of DINO and its successors [48, 49] is that the proposed training methodology paired with transformer architecture results in patch-level visual representations that contain explicit information about the semantic layout of a scene without explicit supervision.

In Chapter 3 and Chapter 4, we discuss how emergent localization properties of DINO can be leveraged for other tasks.

2.1.3. Language supervision

Natural language provides a more nuanced supervisory signal than traditional closed-set class labels. That is why contrastive language-image pre-training (CLIP) directly utilizes alt-text for learning transferable visual representations [13] collected at scale from the Internet. Models trained through this approach — including ALIGN [24] — demonstrate strong zero-shot image classification transfer and cross-modal retrieval capabilities by projecting images and text into a unified embedding space.

CLIP model consists of two parallel encoders (presented in Fig. 2.1.1): an image encoder (typically a ViT [37] or ResNet [36]) and a text encoder (typically a Transformer

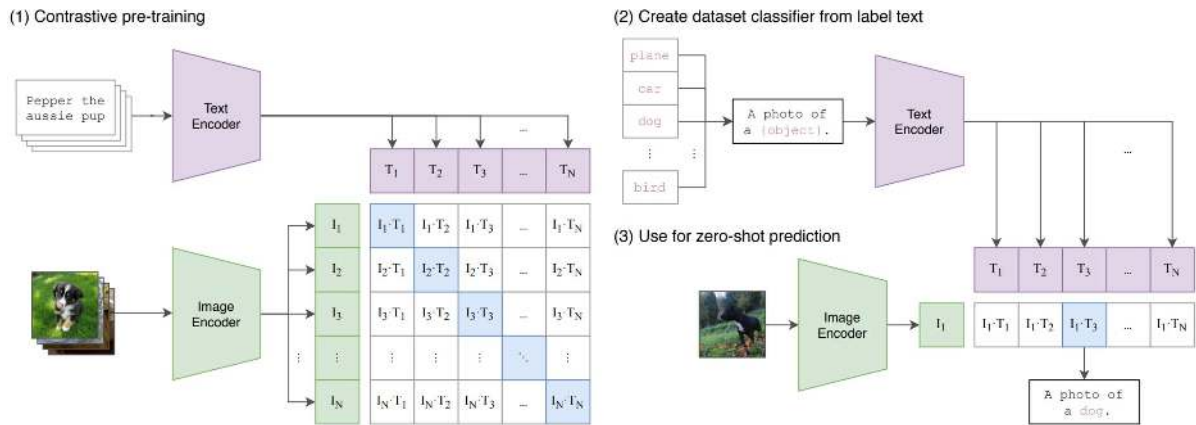


Figure 2.1.1. Image-text alignment via contrastive learning. Source [13].

model [50]). During training, a batch of pairs of images and their corresponding text descriptions are passed through their respective encoders to obtain embeddings. The embeddings are then normalized to the unit sphere to calculate cosine similarities between all image-text pairs within a batch. Finally, the symmetric cross-entropy loss encourages matching image-text pairs to have high similarity while pushing non-matching pairs apart in both directions (image-to-text and text-to-image).

CLIP enables zero-shot image classification by reformulating classification as a retrieval problem in the image-text-aligned space, thus inherently supporting zero-shot image-text retrieval. Beyond these applications, the aligned multimodal embedding space facilitates open-vocabulary extension for traditional vision tasks, inspiring numerous developments in open-vocabulary object detection and segmentation [51, 52, 53].

Data constitutes an essential part of the CLIP training recipe. Original CLIP [13] utilized 400M image-text pairs sourced from web mining, whereas ALIGN employed a proprietary corpus comprising 1.8B image-text pairs. While these extensive datasets remain inaccessible to the public, and the computational demands for training such models are substantial, the research community tries to reproduce and improve on the initial efforts [54]. For example, MetaCLIP [55] extracts a balanced subset from a raw data collection by leveraging metadata (derived from CLIP’s concepts). DFN [56] learns a data filtering network to induce the correct subset of large-scale noisy datasets.

Other improvements over the original CLIP include a modified objective function, specifically SigLIP [57] uses a simple pairwise sigmoid loss for image-text pretraining, which operates on image-text pairs only and does not require a global view of the pairwise similarities for normalization, yielding the training more efficient.

2.1.4. Hybrid approaches

Recent work also sought to produce image-text aligned representations that leverage SSL pretraining methods to make CLIP-like training more efficient [58] or to produce better patch features [59, 60, 61, 62]. In later sections, we will study in depth the benefits of combining multiple approaches to tackle fine-grained tasks that these methods were not necessarily trained for.

2.2. Downstream task adaptation strategies

The emergence of visual foundation models has led to significant interest in the development of efficient strategies to adapt these models to downstream tasks. In this section, we discuss some of the adaptation strategies, from full fine-tuning to more parameter-efficient methods and zero-shot strategies.

2.2.1. Dataset adaptation

The conventional approach for adapting foundation models involves fine-tuning all model parameters on task-specific data. This technique has demonstrated remarkable success across various vision tasks [13, 63]. However, as foundation models continue to grow in size, full fine-tuning becomes increasingly computationally expensive and requires substantial task-specific annotations [5]. This approach also faces challenges with catastrophic forgetting, where the adaptation process can degrade the model’s original capabilities [64].

2.2.2. Parameter-Efficient Fine-tuning

Parameter-efficient adaptation methods have gained prominence in addressing the computational limitations of full fine-tuning. These approaches modify only a small subset of parameters while keeping the majority of the foundation model frozen. [65] introduced adapter modules — small bottleneck layers inserted between pre-trained transformer blocks. Similar approaches have been applied in vision models [66, 67], demonstrating competitive performance with full fine-tuning while updating only 1-5% of parameters. LoRA [25] decomposes weight updates into low-rank matrices, significantly reducing the number of trainable parameters. This approach has proven effective for visual foundation models by targeting specific weight matrices within the architecture. Visual prompt tuning [68, 69] prepends trainable tokens to the input sequence. These learnable prompts guide the model toward task-specific behavior without modifying its internal parameters.

2.2.3. Feature Extraction and Linear Probing

A simpler adaptation strategy treats a foundation model as a fixed feature extractor, using its representations as input to task-specific heads [70, 71]. Linear probing—training a linear classifier on frozen features—has become a standard evaluation protocol for visual representations [40, 13] due to its simplicity and flexibility to apply to various representations. Linear probing always requires downstream task annotations.

2.2.4. Zero-shot and Few-shot Adaptation

Vision-Language Models, as discussed in Sec. 2.1.3, have enabled zero-shot adaptation through their aligned image-text representations. CLIP [13] pioneered this approach by framing downstream tasks as text-conditioned classification problems. Subsequent work has extended zero-shot capabilities to object detection [52, 72], segmentation [28, 51], and other structured prediction tasks. Few-shot adaptation strategies further refine zero-shot predictions using minimal labeled examples [73]. These methods leverage the foundation model’s generalization capabilities while incorporating task-specific information from a small set of examples.

2.2.5. Adaptation for Dense Prediction Tasks

Dense prediction tasks such as semantic segmentation and object detection present unique challenges for foundation model adaptation due to their requirement for spatially aware features. Linear probing – extended to patch-level probing – can be considered one of the strategies to adapt visual features to some of the dense tasks by classifying patch-level representations within a model. This typically means probing features from layers just before pooling layers for convolution-based backbones, such as ResNet [36], or patch features from the last transformer blocks within ViT [37]. Linear probing, however, lacks task-specific mechanisms that can be important for downstream performance, thus not fully exploiting the abilities of visual representations to address dense tasks. Additionally, the linear probing strategy can only be used with full supervision; therefore, it always requires fully annotated datasets.

Decoder-based approaches attach task-specific decoders to the foundation model encoder, often incorporating multi-scale features [51, 52, 53, 74]. While effective, they typically require substantial task-specific training.

For segmentation tasks, mask-based strategies [75, 28] leverage foundation models to classify mask proposals, enabling open-vocabulary segmentation without direct dense supervision. We give a more detailed overview of the related work for

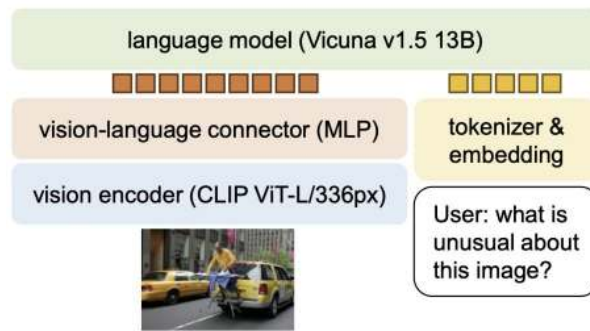


Figure 2.2.1. LLaVa 1.5 architecture - a generic framework for MLLMs. Source [82].

adapting CLIP to open-vocabulary semantic segmentation in the later sections of this thesis (Sec. 3.2.1).

2.2.6. Multimodal Adaptation

Finally, visual foundation models have become major building blocks for recent large Multimodal Large Language Models (MLLMs). Flamingo [76], BLIP2 [77], PaLI [78] and LLaVa [79] were among the first architectures to integrate visual understanding with language capabilities for complex reasoning tasks, while being prototypes of currently strongest reasoning models, including Gemini [80] or GPT4v [81].

LLaVA introduces a minimalist architectural framework, as illustrated in Fig. 2.2.1, comprising a frozen CLIP image encoder for visual representation extraction coupled with a streamlined adaptation mechanism implemented as a Multi-Layer Perceptron (MLP). This adaptation layer facilitates integration with the pre-trained Large Language Model, specifically Vicuna 1.5 [83]. A distinguishing architectural element across MLLMs is the connector module design—LLaVA employs a relatively simple MLP projection, whereas alternative approaches such as BLIP2 [77] implement more sophisticated structures like the Querying Transformer.

The methodological training protocol follows a two-phase procedure: (1) connector layer pre-training utilizing image-text pairs to establish cross-modal alignment, followed by (2) instruction tuning—a specialized fine-tuning regimen conducted on a corpus of image-instruction-response triplets. This latter phase enables the model to interpret natural language directives and generate contextually appropriate responses. Instruction tuning constitutes a critical transformation mechanism, converting foundation models from generalized predictive systems into interactive assistive frameworks with user-oriented interface capabilities. Through conversational prompting methodologies, these models demonstrate proficiency across diverse

visual reasoning tasks, including Visual Question Answering and Image Captioning functionalities.

Recently, [84] presents a comprehensive analysis of design and training considerations for MLLMs. Their work proposes utilizing MLLMs as an evaluation framework for comparing visual representations, offering an alternative to conventional benchmarks. In Chapter 6, we similarly advocate for Visual Question Answering as an assessment protocol for visual representations, albeit employing a significantly less complex adaptation module, as our initial studies date before the rise of open-source MLLMs.

3. Task Adaptation through Modified Inference

In the previous chapter, we examined the capabilities of Vision- Language Models. These models, particularly through text alignment, demonstrate remarkable zero-shot image recognition capabilities. CLIP’s emergence has pioneered new research directions in open-vocabulary tasks, including detection and semantic segmentation. The term *open-vocabulary* marks a separation from traditional closed-set solutions, which are constrained to finite class sets. Instead, through CLIP’s zero-shot recognition of novel concepts, we can now develop methods that operate on unrestricted class vocabularies.

CLIP-like models, however, face a significant limitation: their patch-level representations exhibit weak localization properties [28]. This constraint makes VLMs unsuitable for dense prediction tasks, such as semantic segmentation, without substantial architectural modifications.

This chapter presents our first contribution, demonstrating that CLIP can be adapted for pixel-level tasks with minimal adaptation cost of training and task-specific annotations through a simple modification of its inference mechanism. We explore this approach in the context of open-vocabulary semantic segmentation — an extension of semantic segmentation that addresses a fundamental computer vision challenge in an open-world setting. We introduce CLIP-DIY, a method that achieves this adaptation without requiring additional training or annotations, instead leveraging existing unsupervised object localization techniques. CLIP-DIY employs a multi-scale approach that directly utilizes CLIP’s classification capabilities on varying patch sizes, aggregating these decisions into a unified map. The segmentation is further refined using foreground/background scores derived from an unsupervised object localization method. Notably, our approach achieves competitive results on standard semantic segmentation datasets when compared to methods that require explicit training.

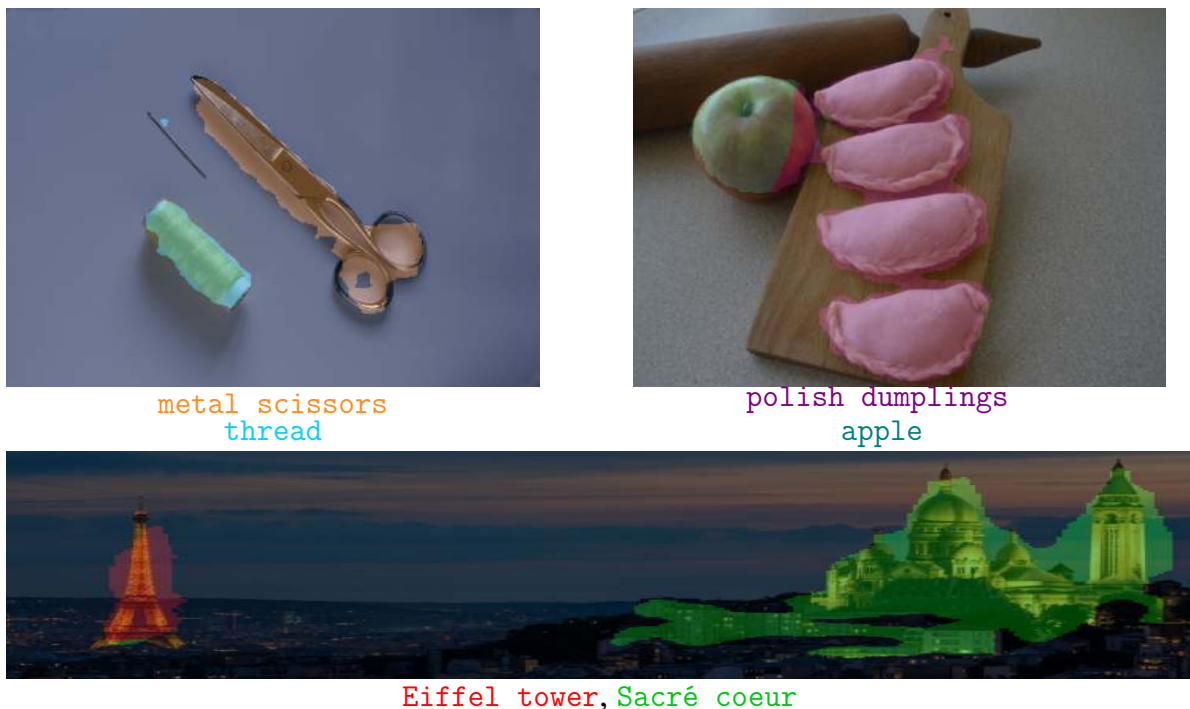


Figure 3.1.1. Simple and accurate semantic segmentation for-free using CLIP-DIY. Our method processes regions of an image *separately, in parallel* to produce per-patch-per-class scores, given a set of prompts that can take *any length*. CLIP-DIY does not require re-training and can, therefore, immediately adapt to any new vocabulary.

3.1. Introduction

The task of *semantic segmentation*, which aims at predicting the class of every pixel in an image, has been widely tackled using fully-supervised approaches [85, 86, 87], which require tedious and, therefore, expensive per-pixel annotations. Moreover, semantic segmentation has typically been performed with a *finite set of classes* [88, 89, 90] describing the types of objects that should be discovered in images. However, using a fixed number of classes is limiting for real-world applications as interesting object classes may vary in time and per application – having to perform annotation and re-training on new classes is expensive and sub-optimal. In this context, recent advances in Visual Language Models [54, 76, 24] have paved the way to *open-vocabulary* perception. Indeed, VLMs, trained with cheap and widely available image-text pairs, e.g. captions, offer new possibilities to describe images with a *large and open vocabulary*. Using such models can help alleviate both the problem of supervision and finite vocabulary.

In particular, the popular VLM CLIP [54] has been exploited to perform *open-vocabulary perception*. While it has high performance on image classification, applying CLIP on dense tasks is more challenging [28]. In order to improve CLIP segmentation abilities, different methods have been proposed to modify the architecture [28, 91],

to add new modules [92, 93, 75] or to train new specifically designed models [91, 94] from scratch. Instead, we propose CLIP-DIY, a new *zero-shot open-vocabulary semantic segmentation* approach which makes direct use of the high-performance image classification properties of CLIP, *does not* need architecture changes or additional training. In particular, our method applies CLIP to a multi-scale grid of patches and aggregates the information into a single prediction map.

Moreover, to further improve the quality of the localization of our predicted maps, we propose with CLIP-DIY to leverage the recent efforts in *unsupervised object localization*. This task aims at discovering every object depicted in images in a class-agnostic fashion and thus *without manual annotation*. Recent methods [95, 96, 97, 98, 99, 100] achieve impressive localization results by leveraging self-supervised features [15, 40]. While some methods discover one object per image [95, 96], more recent ones try to highlight all objects in an image [98, 97, 100]. We therefore propose to leverage those capabilities in CLIP-DIY, by guiding CLIP predictions with a very lightweight unsupervised foreground/background strategy which greatly improves the predictions’ quality.

To summarize, our novel approach CLIP-DIY best leverages the open-world classification capabilities of CLIP and the high-quality of unsupervised object localization approaches yielding the following contributions:

- We introduce CLIP-DIY, a novel, simple technique for open-vocabulary semantic segmentation which *does not require additional training* or any *pixel-level annotation* but instead leverages strong self-supervised features with good localization properties combined with CLIP.
- Our multi-scale approach, which uses simply CLIP as it was designed –for image classification– enables CLIP-DIY to produce well-localized predictions.
- We demonstrate that unsupervised foreground/background methods can be effectively used to provide spatial guidance to CLIP predictions.
- We achieve a new state-of-the-art zero-shot open-vocabulary semantic segmentation on PASCAL VOC dataset and perform on par with the best methods on COCO.
- We perform an extensive validation of the design of our method and show that it is robust as it can be directly applied to in-the-wild open-world segmentation.

3.2. Related work

In this section, we discuss previous work related to ours, starting in Sec. 3.2.1 with zero-shot open-vocabulary semantic methods, then following with unsupervised object localization methods in Sec. 3.2.2. Finally, in Sec. 3.2.3, we focus more specifi-

cally on works that leverage a combination of self-supervised learned features and CLIP to perform open-world segmentation.

3.2.1. Zero-shot open-vocabulary semantic segmentation

With the aim to build generalizable models, *zero-shot* methods for semantic segmentation [101, 102, 103, 104, 101, 105, 106, 107, 108] propose to extend models trained in a fully supervised fashion on a set of *seen* classes to new *unseen* classes. Many leverage relationships encoded in pre-trained word embeddings [109, 110] to discover new unseen concepts. Such methods require annotation for the seen classes while we aim to use no pixel-level annotation.

Alternatively, *open-vocabulary* approaches [111] exploit image-text alignment without needing to pre-define vocabulary. Several [28, 112, 92, 113, 112] build on top of the popular CLIP [13] model which showed impressive *global* text-image alignment properties but lacks localization quality [114]. Using class-agnostic object masks, it is possible to learn to align the embeddings of selected pixels with text [113, 51, 115], but at the cost of pixel-level annotations. Without extra supervision, MaskCLIP [28] alters the last pooling layer of CLIP to produce dense predictions and use them as pseudo-labels to train a segmentation model, forming MaskCLIP+. Using only image captions—cheap to acquire and widely available—as supervision, [94, 112, 93, 116] learn local alignment between image *regions* and paired text with contrastive objectives. Regions are formed using a learnt hierarchical mechanism [94], cross-attention based clustering [112], using a clustering head trained with diverse view [93] or slot-attention [116]. PACL [92] adds an embedder module that learns affinity between patches and the global text token, TCL [75] builds a new local contrastive objective which directly aligns captions with pre-selected patches and ViewCO [117] proposes a multi-view consistent learning approach. CLIPpy [91] proposes to fully re-train CLIP with a few well-designed modifications to directly obtained denser features. Alternatively, ReCO [118] builds prototypes of the desired vocabulary (using CLIP-based retrieval), which are then used for co-segmentation.

Rather than modifying the architecture of CLIP or training a new module specifically designed to densify its outputs, we propose to directly use *as is* the good classification ability of the model. Indeed, we perform a dense multi-scale patch classification. By doing so, our method can easily be adapted to any new dataset or vocabulary.

3.2.2. Unsupervised object localization

Interestingly, recent works have shown that ViT features trained in a self-supervised fashion [41, 15, 40] on images—with no human-made annotations—have

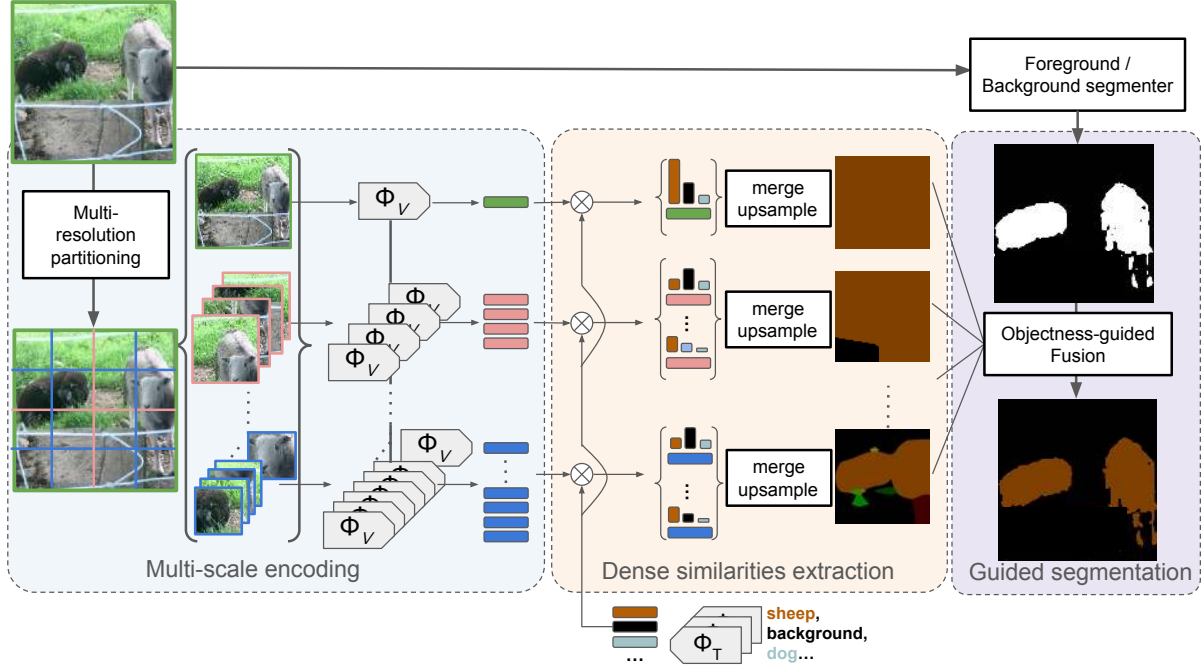


Figure 3.2.1. Overview of CLIP-DIY, our two-step pipeline for segmentation map extraction. We note Φ_V and Φ_T the CLIP encoders for image and text, respectively. (1) An input image is partitioned into smaller patches, and each of them is fed *independently* to the image encoder, yielding a vector of per-class similarity to an *arbitrary-length* vocabulary of classes. (2) Patches are then aggregated back before upsampling to produce dense similarity maps. (3) An objectness score obtained by an off-the-shelf foreground-background segmentation method such as FOUNDED [100] is used to guide the prediction of the final segmentation.

good localization properties [95, 96]. Such properties have been exploited to tackle the problem of *unsupervised object localization* [119, 120], which requires to localize objects—any object—depicted in images and without any cue. A set of methods [95, 96, 121, 98, 97] exploit the good correlation properties of the feature and find an object as the set of patches which highly differs to the other patches. Alternatively [122], exploits the attention mechanisms with different queries and produces maps that are ranked and filtered and [100] proposes to look for the background instead of the objects in order to avoid single object discovery and to need priors about objects. The coarse object localization results obtained using those methods can be used as pseudo-labels to train large instance or segmentation models in a class-agnostic fashion [95, 96, 122, 123, 97]. Recent FOUNDED [100] is a very light model—a single conv1x1—trained to produce foreground/background segmentation of good quality. When self-trained FOUNDED achieves even better results and discovers more objects per image [95, 97]. In this work, we propose to leverage the good object localization properties of unsupervised object localization models, which make no hypotheses about object classes and remain, therefore *open*. In particular, we take advantage of the efficiency of FOUNDED [100] to guide our zero-shot segmentation.

3.2.3. Combining self-supervised features & CLIP

Combining self-supervised learning [15, 20, 40, 23] with VLMs has been previously explored by different open-vocabulary segmentation methods [118, 91, 124]. Correlation qualities are used to perform co-segmentation [118] when pre-training properties are directly leveraged to initialize the visual encoder backbone [91, 116]. Related to our work, ZGS [124] builds potential masks using clustering strategies on self-supervised features and assigns them a class. Contrary to all previous approaches, it explores the task without predefined text prompts. Although ZGS currently obtains lower results than other baselines on the task, it opens an interesting direction for future work.

In this work, we do not perform co-segmentation (which expects to know classes of interest) nor retrain a model from scratch. Instead, we propose to guide CLIP prediction with an unsupervised foreground/background segmentation method, which, to the best of our knowledge, has not yet been explored.

3.3. CLIP-DIY

We tackle the problem of open-vocabulary semantic segmentation with no supervision. Let us consider a set of queries $t_j \in T$ formulated in natural language. Our goal is to localize each query if present in the image, yielding one mask per query, i.e. a segmentation map. Our approach, summarized in Fig. 3.2.1, consists of two stages. In Sec. 3.3.1, we describe our first step, where we run our proposed multi-scale dense inference to obtain coarse semantic maps by running CLIP on image patches at different scales. Our second step, which we cover in Sec. 3.3.2, consists of refining the initial segmentation using an off-the-shelf foreground-background extractor.

3.3.1. Dense inference with CLIP

Our method leverages CLIP [13] as a backbone. Contrary to most CLIP-based approaches for zero-shot semantic segmentation (as discussed in Sec. 3.2.1), we do not rely on patch tokens within the image encoder. Instead, we leverage CLIP’s zero-shot capabilities by running the model densely on image partitions. Thus, we calculate the alignment between each of the multi-scale patches and the considered textual queries.

Encoding prompts. Given a set of textual queries \mathcal{T} , we encode a prompt $t_j \in \mathcal{T}$ with $j \in [0, |\mathcal{T}|]$ using CLIP text encoder $\Phi_T(t_j) \in \mathbb{R}^d$, where $d = 512$ in CLIP. Following previous works [75], we formulate a prompt as: a photo of a $\{t_j\}$. In what follows

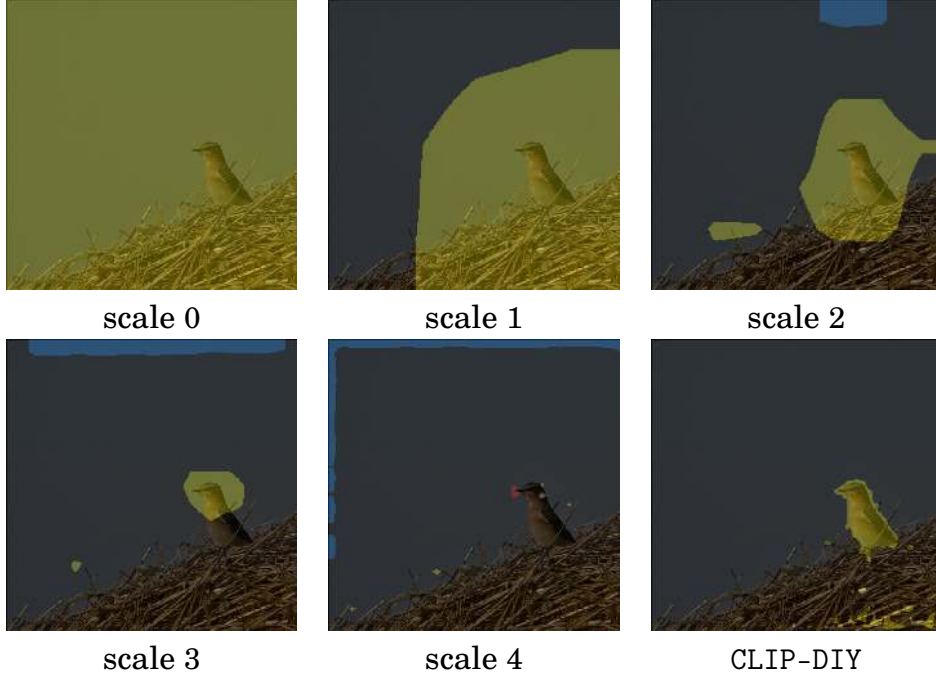


Figure 3.3.1. Multi-scale maps ρ_s of CLIP-DIY. Our method produces multi-scale predictions of dense logits that capture information at different levels of granularity. While the coarsest ($s = 0$) scale pools global information from the image, finer scales induce more localized and potentially discover different object classes. While our method can incorporate as many scales as needed, very fine resolutions such as scale 4 and above do not generate meaningful information.

we use the notation $\Phi_T(\mathcal{T}) \in \mathbb{R}^{d \times |\mathcal{T}|}$ as the set of encodings of all text queries \mathcal{T} . Note that we always consider a background class, thus we always assume $t_0 = \text{background}$.

Multi-scale image partitioning. Given an input image $x \in \mathbb{R}^{H \times W \times 3}$, we first reshape it into a sequence of 2D patches $\mathcal{X} = \{x_i \in \mathbb{R}^{P \times P \times 3}\}_{i=1..N}$, where $N = \lceil \frac{H}{P} \rceil \cdot \lceil \frac{W}{P} \rceil$ and $P \times P$ is the patch size. We then extract visual embeddings $\Phi_V(x_i) \in \mathbb{R}^d$ for each patch. In practice, since we use a ViT [125] as our visual encoder Φ_V , and take the [CLS] output as our visual embedding $\Phi_V(x_i)$ ¹. We can then run the same partitioning for different scales \mathcal{S} using different patch sizes, yielding a set of partitions $\{\mathcal{X}_s\}_{s \in \mathcal{S}}$.

Dense similarities extraction. Given our multi-scale partitions, and a text query $t \in \mathcal{T}$, we build a dense similarity map ρ_s^t for each scale $s \in \mathcal{S}$, such that:

$$\rho_s^t = \text{Upsample}_{H,W} \left(\bigcup_{x \in \mathcal{X}_s} [\Phi_V(x) \otimes \Phi_T(t)] \right), \quad (3.1)$$

¹ For convolutional network backbones such as Resnet [36], we could use the output of AvgPool.

where \cup is a merging operator that puts patches back onto a 2D grid, \otimes denotes the inner product computed between the visual and text embeddings and $\text{Upsample}_{H,W}$ is a bilinear up-sampling operator which upsamples its input to the resolution $H \times W$.

The resulting map $\rho_s^t \in \mathbb{R}^{H \times W}$ yields an estimate of the similarity between each pixel in the input image and a text query $t \in \mathcal{T}$. In Fig. 3.3.1 we show aggregated similarity maps $\rho_s = \{\rho_s^t\}_{t=1..|\mathcal{T}|} \in \mathbb{R}^{H \times W \times |\mathcal{T}|}$ obtained at different scales. We can see that while the coarsest scale $s = 0$ is responsible for pooling global information on the objects to segment, finer scales $s > 1$ result in more localized maps.

Having obtained similarity maps ρ_s^t for each text query $t \in \mathcal{T}$ and each scale $s \in \mathcal{S}$ we aggregate the multi-scale predictions into a map M_{CLIP}^t such that:

$$M_{CLIP}^t = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \rho_s^t. \quad (3.2)$$

3.3.2. Guided segmentation

Finally, we propose to refine the multi-scale segmentation maps M_{CLIP}^t of Eq. 3.2 using an objectness map produced by an off-the-shelf *unsupervised* foreground-background segmentation method [98, 100], noted Θ . As discussed in related work, such methods exploit self-supervised features, e.g. [15], to discover the pixels likely depicting objects.

When fed with an image $x \in \mathbb{R}^{H \times W}$, the foreground-background segmentation method produces an output $\Theta(x) \in [0, 1]^{H \times W}$ with a per-pixel confidence score close to 1 for a pixel in a foreground object. We use this proxy for objectness estimation to refine our segmentation masks M_{CLIP}^t . In particular, we refine M_{CLIP}^t using the output of Θ for each text query $t \in \mathcal{T}$ except for $t_0 = \text{background}$, where we take the complement of $\Theta(x)$ following:

$$\hat{M}^t = \begin{cases} 1 - \Theta(x) & \text{if } t = \text{background}, \\ \Theta(x) & \text{otherwise,} \end{cases} \quad (3.3)$$

such that similarities with the background class are down-weighted for pixels deemed as *salient* by Θ .

Finally, we compute the output of CLIP-DIY as:

$$M = \text{SoftMax}_{t \in \mathcal{T}} (M_{CLIP}^t \odot \hat{M}^t) \in \mathbb{R}^{H \times W \times |\mathcal{T}|}, \quad (3.4)$$

where \odot denotes the Hadamard product and SoftMax is the softmax operator computed over text queries. In Fig. 3.3.1 we show how the aggregation of all scales paired with the guidance of Θ results in accurate object segmentation.

3.4. Experiments

In this section, we present the experiments conducted to evaluate our method and justify particular design choices. First, in Sec 3.4.1 we give details about our experimental setup. In Sec. 3.4.2, we discuss how our method compares against other open-vocabulary semantic segmentation approaches, both quantitatively and qualitatively. We then give more insight into our method with a series of ablations (Sec. 3.4.3), failure mode analysis (Sec. 3.4.4) and finally real open-world evaluation (Sec. 3.4.5).

3.4.1. Experimental setup

Datasets & metric. We evaluate our method on two common semantic segmentation benchmarks: PASCAL VOC 2012 [88] and COCO [90], comprising of 20 and 80 foreground classes respectively. PASCAL VOC has an additional background class, and we adopt a unified protocol [94, 75] considering a background class in all datasets. We evaluate results with the mean Intersection-over-Union (mIoU) metric. For evaluation, we resize input images to have the shorter side of length 448 following [75].

Implementation details. If not otherwise specified, we use the CLIP ViT-B/32 model OpenCLIP version [54] trained with LAION [126]. The input images are resized to 224×224 , and the patch size is 32×32 . We empirically find that running our model on 3 different scales, i.e. $|S| = 3$ with patch sizes of $P_0 = 256, P_1 = 128, P_2 = 64$, gives the best results for both evaluated datasets. We discuss this later in Sec. 3.4.3.

Baselines. We compare our method with the existing state-of-the-art zero-shot open-vocabulary methods. In particular, we evaluate against methods, including self-trained MaskCLIP+ [28], learning grouping strategies: GroupViT [94], SegCLIP [112], ViL-Seg [93], OVSegmentor [116], ViewCo [117], using class prototypes: ReCo[†] [118], text-grounding strategy: TCL [75] and with CLIPpy [91] which uses a T-5 backbone and improves dense abilities of CLIP. We detail in Tab. 3.4.1 the VLM backbones used per method, and the if additional training data was. Every method (except for MaskCLIP) requires training a specific module/model used to get denser predictions; instead, we use the vanilla CLIP model.

A note on fair comparison. Following TCL [75], we use a unified evaluation protocol corresponding to an open-world scenario where prior access to the target data before evaluation is not allowed. In particular, we do not consider query expansion, e.g.

Method	extra training	Backbones Visual	Text	PASCAL VOC	COCO Object
ReCo [†] [118]	✓	ViT-L/14*	CLIP-ViT-L/14*	25.1	15.7
ViL-Seg [93]	✓	ViT-B/16		37.3	-
MaskCLIP+ [†] [28]	✓	ResNet101 [36]		38.8	20.6
CLIPpy [91]	✓	ViT-B/16	T-5 [127]	52.2	32.0
GroupViT [94]	✓	ViT-S/16	12T	52.3	-
ViewCo [117]	✓	ViT-S/16	12T	52.4	23.5
SegCLIP [112]	✓	ViT-B/16	CLIP-ViT-B/16	52.6	26.5
OVSegmentor [116]	✓	ViT-B/16	BERT-ViT-B/16	53.8	25.1
TCL [75] + PAMR [128]	✓	ViT-B/16	CLIP-ViT-B/16	<u>55.0</u>	<u>31.6</u>
CLIP-DIY (ours)		ViT-B/16	CLIP-ViT-B/16	59.0	30.4
CLIP-DIY (ours)		ViT-B/32	CLIP-ViT-B/32	59.9	31.0

Table 3.4.1. Zero-shot open-vocabulary segmentation. Comparison of our approach to the state of the art (under the mIoU metric). While our method does not need any additional training it performs significantly better than the current SOTA on PASCAL VOC (**+4.9**) and performs on par with its competitors on COCO object, ranking 3rd on COCO. We mark with [†] results from [75]. All methods are evaluated considering that background is a class of the dataset. We note with * when more than one backbone was used, we refer here to CLIP-like backbones. GroupViT and ViewCo use a 12 Transformer layers backbone following [13], noted 12T.

class name expansion or rephrasing. As discussed in [75], exploring language biases can greatly improve the overall segmentation performance. However, we only use the original class names from the compared datasets. We also report best-reported scores for all methods. It is to be noted that TCL uses a post-processing technique, namely PAMR [128], while other methods do not.

3.4.2. Results

In this section, we compare our method to previous work both quantitatively and qualitatively.

Quantitative results. We summarize in Tab. 3.4.1, the comparison of our CLIP-DIY to baselines. We report results of CLIP-DIY with both CLIP ViT-B/32 and CLIP ViT-B/16, given that most methods use the latter.

First, comparing our results with the two different backbones, we remark that better scores are obtained with CLIP ViT-B/32, in which patch inputs are larger and are, therefore, less expensive at inference time. We believe this could be explained by an existing upper bound on CLIP accuracy w.r.t the level of granularity; patches too small might induce noisy classification—as we observed in Fig. 3.3.1.

Method	PASCAL VOC	COCO
CLIP-DIY	59.9	31.0
w/o multi-scale	56.0	25.9
w/o objectness	24.1	15.5

Table 3.4.2. Ablation studies. We find that while all components are improving the performance of our method, objectness is particularly critical. Numbers are reported using a ViT-B/32 backbone.

Compared to baselines, we observe that our method achieves the best mIoU result on PASCAL VOC dataset and outperforms all previous works by more than 4 mIoU pts, and such without post-processing. This result is particularly interesting given that our method does not require dedicated training to improve CLIP segmentation abilities but instead leverages the unsupervised object localization method.

Moreover, we obtain 31.0 mIoU on COCO when the best-performing method CLIPpy achieves 32.0, so just 1 mIoU pt better than our approach. We can observe in Fig.3.4.1 that CLIPpy discovers more queries per image than us, even though its segmentation outputs appear to be noisier. Such results might suit a better COCO benchmark and less PASCAL VOC.

Qualitative results. We qualitatively compare here our method with best performing TCL [75] and CLIPpy [91] in Fig. 3.4.1.

Our method consistently produces better masks with better object boundaries, which we attribute to the high-quality saliency maps. Moreover, we observe that CLIP-DIY produces correct semantic results on the foreground objects, with fewer artifacts than CLIPpy. Our method seems also less sensitive to biases, for instance, both TCL and CLIPpy hallucinate the sheep class on the grass (on the very left image) and CLIPpy also predicts aeroplane in the sky (in most left image) and zebra next to the elephants (middle image in COCO dataset).

3.4.3. Ablations

In this section, we perform ablation studies to validate the individual choices in the design of CLIP-DIY. First, we study in Tab. 3.4.2 the impact of the different elements of our method. In particular, we investigate the impact of using a multi-scale mechanism and leveraging the objectness produced by the foreground/background segmenter. We notice that by removing our multi-scale scheme, we drop results by 3.9 and 5.1 mIoU pts on Pascal and COCO, respectively, showing the benefit of considering patches of different sizes. Additionally, the largest drop is observed when removing the foreground/background saliency guidance, showing the effectiveness

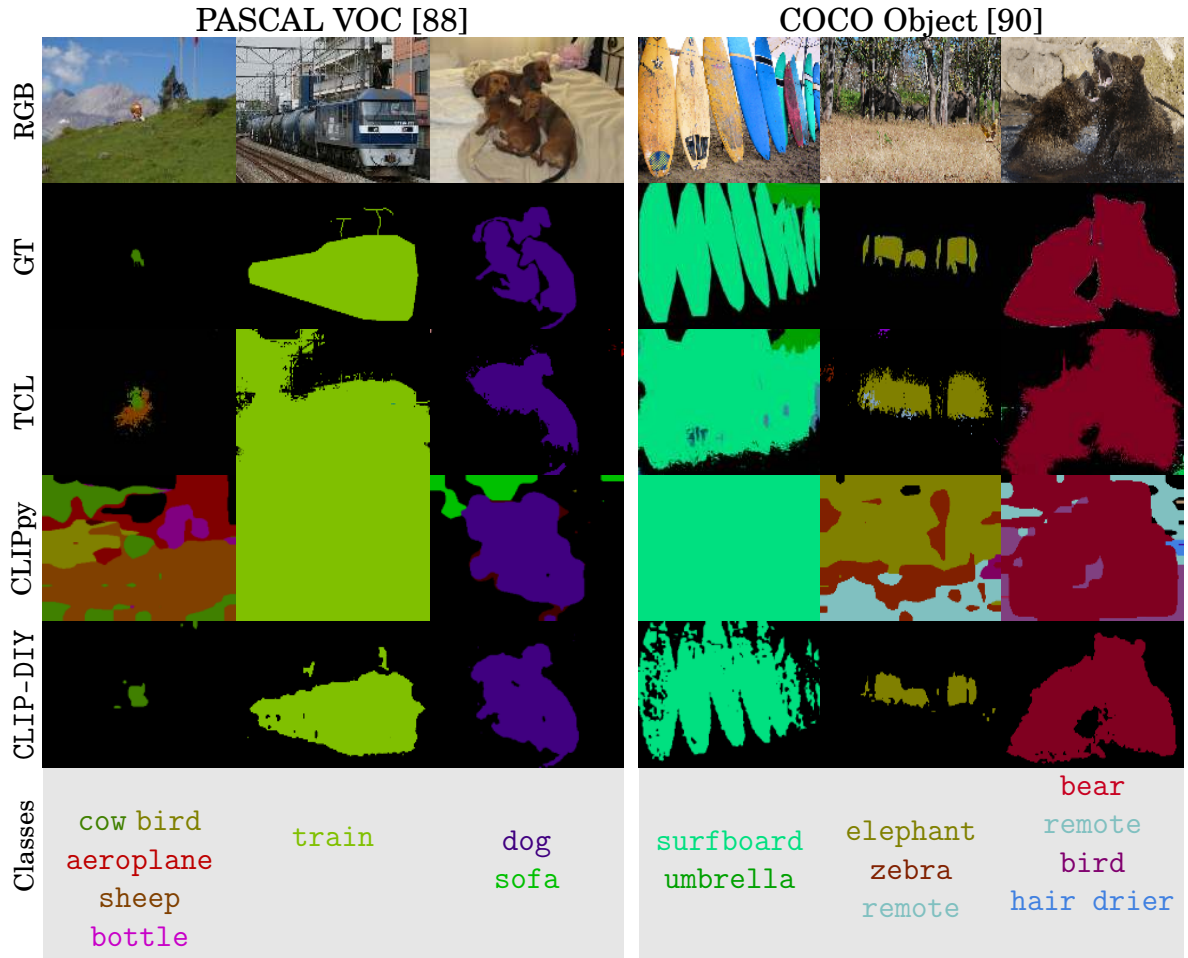


Figure 3.4.1. Qualitative open-vocabulary segmentation results. We compare our method against CLIPpy [91] and TCL (with PAMR post-processing) [75]. Our method consistently outperforms two other methods by producing accurate segmentation masks. TCL and CLIPpy also both suffer from hallucinating classes based on context, such as the **aeroplane** and **sheep** in 1st column, or **zebra** in 5th column. All pixels annotated in black are from the background class.

of combining CLIP with the current lightest unsupervised object localization model, FOUND.

Multi-scale approach. We conduct an ablation study on the scales used in our multi-scale scheme to generate the predictions. We progressively add inner scales in Eq. 3.2 and report the resulting accuracy on all datasets. The results, detailed in Tab. 3.4.3, show an optimum when using three fine scales. Adding more does not appear to improve results or downgrade them, showing the stability of our method. Interestingly, we also see that removing information from the global scale greatly reduces performance on all datasets (-5.7/-3.5 mIoU pts on PASCAL VOC/COCO, respectively). In Fig. 3.4.2, we visualize the individual contribution of each scale to our final prediction. As in Fig. 3.3.1, we observe that coarse scales capture the

Scales used					PASCAL	COCO
0	1	2	3	4	VOC	Object
✓					56.0	25.9
✓	✓				58.8	28.7
✓	✓	✓			59.9	31.0
✓	✓	✓	✓		59.5	29.9
✓	✓	✓	✓	✓	59.3	29.4
	✓	✓	✓		53.8	26.4

Table 3.4.3. Ablation of our multi-scale approach. To validate our multi-scale design, we report the results of our method with different sets of scales, by progressively adding more fine-grained scales. We find empirically that after scale 3, adding more scales gives similar or worse results. We also report results without the first scale $s = 0$ (global scale), which leads to worse results. Numbers are reported using a ViT-B/32 backbone.

Saliency Method	w. training	PASCAL VOC
FOUND-bkg [100]	✗	48.4
FOUND [100]	✓	59.9
CutLER saliency [97]	✓	55.4
CutLER mask [97]	✓	50.8

Table 3.4.4. Comparison of different objectness methods used for objectness-guided fusion. We find that the version of FOUND that was trained *in the original paper* performs the best, and therefore keep this method as our objectness guide.

global context, while finer scales capture more local one, such that objects can be separated.

Foreground segmenters. We compare different foreground-background segmenters and the overall performance of our method using each one of them. The results, summarized in Tab. 3.4.4, show that our method performs the best when using FOUND, more specifically, the FOUND model that has been re-trained with self-training in the original work. We also experiment with CutLER [97], which performs unsupervised instance segmentation. We use the predicted instance masks or compute a saliency. In both cases, we obtain slightly worse results. We give more details in Sec. 3.2.2.

3.4.4. Failure cases

Failure cases We qualitatively analyze failure cases of our method by showing a couple of examples in Fig. 3.4.3. We observe that some of the failures are due to inaccurate annotations: in (a) bear is only partially annotated, and in (b) a mask for an elephant is annotated too coarsely. Our method, benefiting from FOUND’s

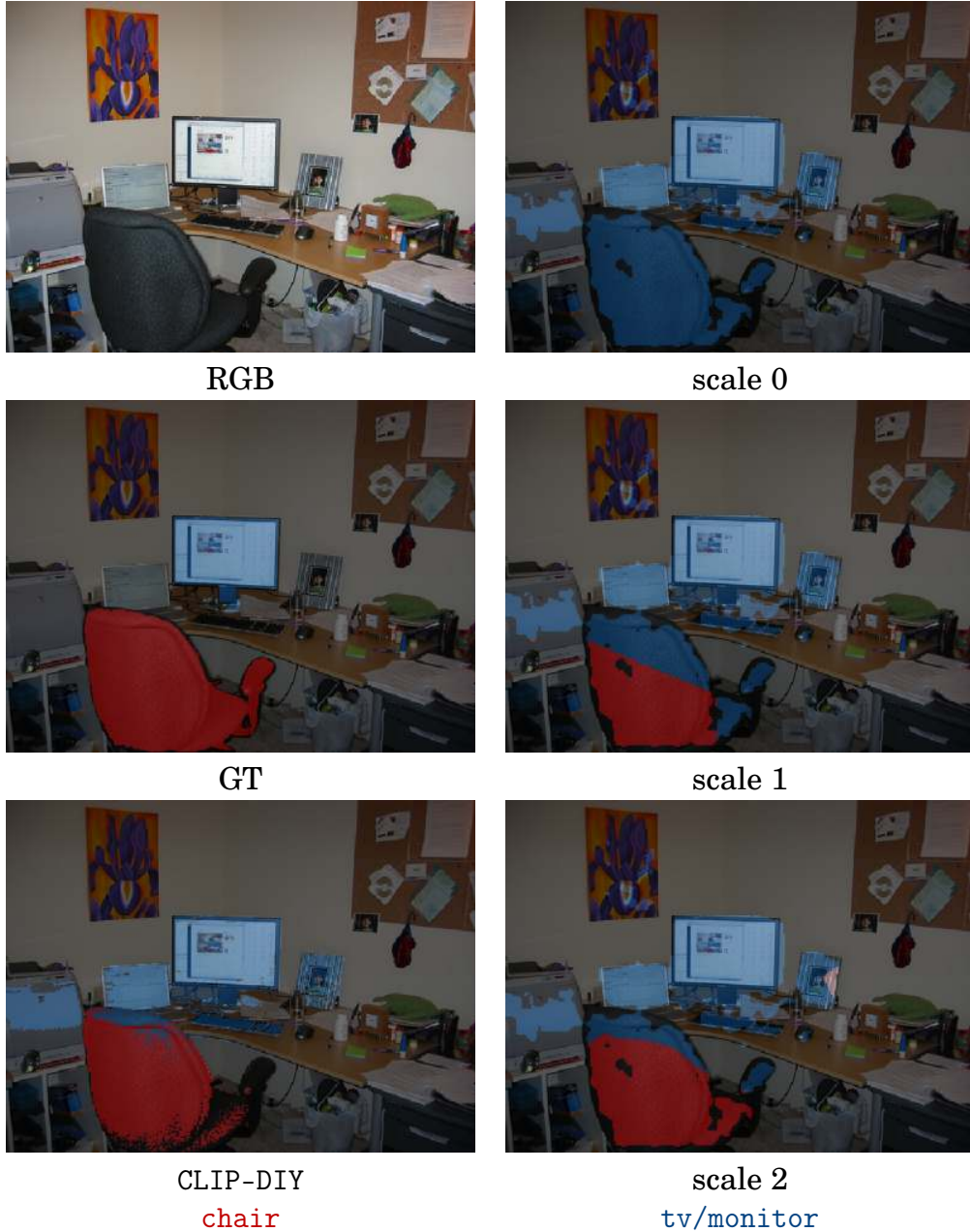


Figure 3.4.2. Qualitative ablation study. In the right column we show the results when progressively adding more scales in the multi-scale approach. Running with 3 scales enables segmenting most of the chair.

accurate saliency predictions is able to produce better segmentation masks. We also observe that our method is limited by the quality of the saliency (c, d), which we comment more on below.

Ambiguities. In the examples (c) and (d) of Fig. 3.4.3, we observe that our method can fail to segment objects with significant overlap, resulting in ambiguity regarding the foreground class, which is especially harmful when annotations are too coarse or incomplete. In (c), only the bench is annotated, while our method segments only the orange, which would result in a low IoU score. In (d), the opposite behavior occurs,

where CLIP-DIY discovers more classes than the annotated ones. Finally, similarly to most of the open-vocabulary methods based on CLIP, our method suffers from sensitivity to text ambiguities. We show more failure cases in Sec. 3.6.2.

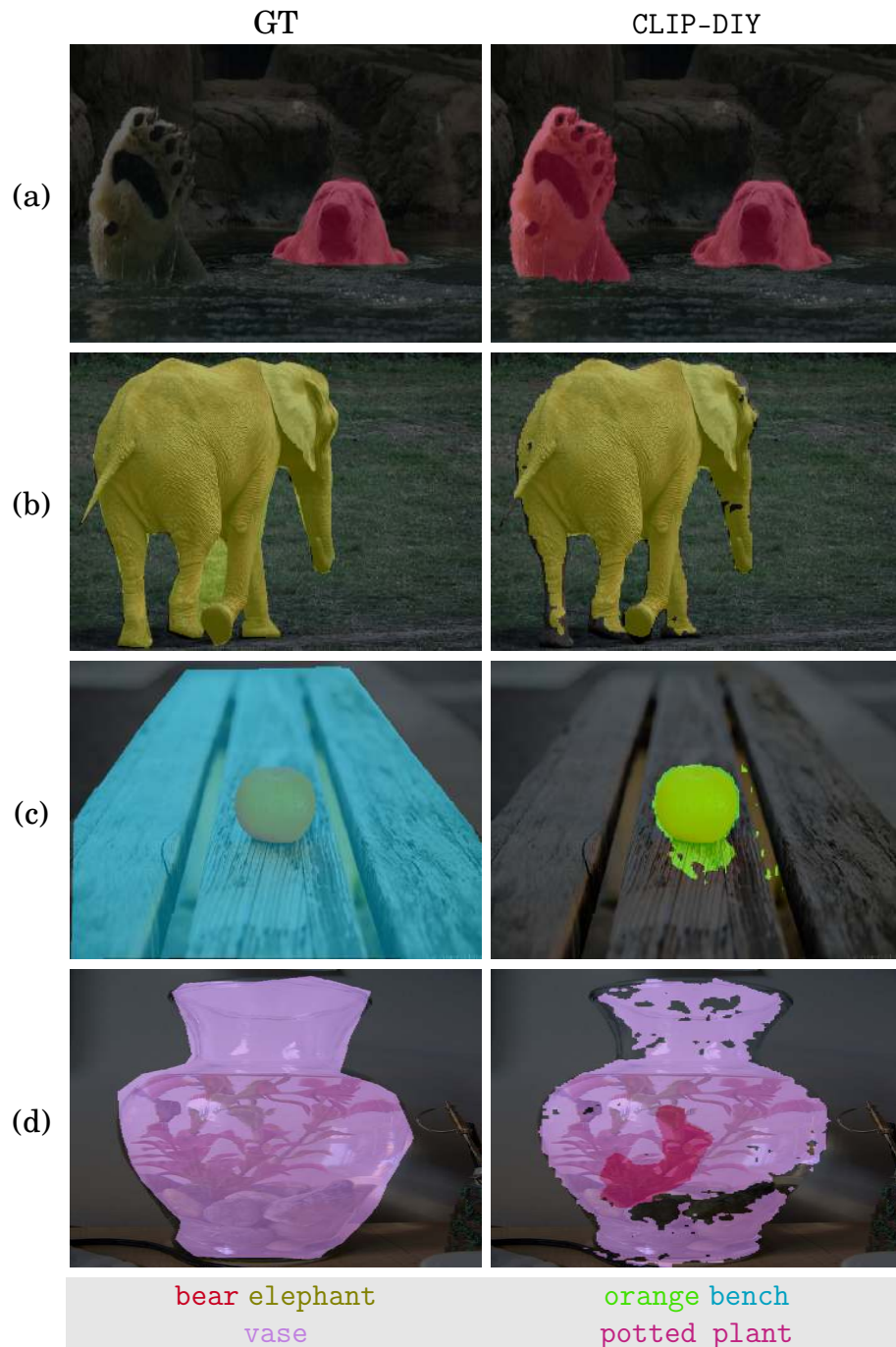


Figure 3.4.3. Failure cases. Our method is not robust in a few cases, such as (a) incomplete or (b) coarse labeling. Another failure mode of our method is when there is an ambiguity in the foreground class (c-d).

3.4.5. Our method in the wild

We also test our method in the wild. We randomly download a set of images and provide textual queries we find most suitable. We show in Fig. 3.1.1 that our method exhibits off-the-shelf open-world segmentation capabilities, being able to produce masks for specific prompts. More results, including comparisons against other methods of in-the-wild open-world segmentation, are presented in Sec. 3.6.2.

3.5. Conclusions

We introduce a new method for open-vocabulary semantic segmentation, namely CLIP-DIY, which exploits CLIP’s open-vocabulary classification abilities. As opposed to recent approaches we run CLIP densely at multiple scales to obtain coarse semantic mask proposals. When further guided by the quick fully unsupervised object localization method FOUND, which estimates foreground saliency, our model obtains state-of-the-art results on PASCAL VOC and performs on par with baselines on COCO dataset. Since our method does not require any specific training, it can be used as an off-the-shelf method for open-world segmentation, and could therefore serve as a tool to help dataset annotators. While CLIP-DIY already yields competitive results, we believe future work could make it more diverse and efficient.

3.6. Additional material

3.6.1. Foreground-background segmenters

In this section, we provide more details on the foreground-background segmenters we use in our work. We consider the two variants of FOUND [100]: we first use the coarse saliency maps produced without self-training, noted FOUND-bkg in the paper, which corresponds to the set of similar pixels to the least salient *background seed* pixel in the self-supervised feature space. We also use the simple conv1x1 model, named FOUND, in the main paper, which is self-trained on only 10,553 images [129] and produces more pixel-aligned results. We refer the reader to the original paper for more details.

We also experiment with the state-of-the-art unsupervised panoptic segmentation method CutLER [97]. It produces a single mask per discovered object in a scene. We test CutLER in two different setups. First, since our method was designed to take one saliency map for the whole image we adapt the output of CutLER to obtain a saliency map as follows. We run CutLER per image obtaining the binary instance

masks $\{\zeta_n \in [0, 1]^{H \times W}\}_{n=1..N}$, with N the total number of output binary masks. We also extract the confidence scores corresponding to the masks $\{\sigma_n \in \mathbb{R}\}_{n=1..N}$. Note that the output masks are of the size of the input image. We then filter the masks and discard those with a confidence score $\sigma_n < 0.3$ similar to the value on the official CutLER repository ². We then aggregate the remaining masks into a saliency map M_{CUT} with:

$$M_{CUT} = \frac{1}{\mathcal{Z}} \sum_{n \in N} \sigma_n \zeta_n, \quad \text{where} \quad (3.5)$$

$$\mathcal{Z} = \max_{\text{px}} \left(\sum_{n \in N} \sigma_n \zeta_n \right) \in \mathbb{R},$$

such that $M_{CUT} \in \mathbb{R}^{H \times W}$ is a normalized 2D-mask with its maximum value being 1.

Second, for a fair comparison, we also use CutLER off-the-shelf as a mask extractor. We use previously described masks $\zeta_n \in [0, 1]^{H \times W}$, and with each one of them, we create an image I_n mask where the background is masked out. Each masked image I_n is then fed separately to CLIP to obtain a CLIP prediction. We denote this approach as *CutLER mask* in Tab. 3.4.4.

Overall, CLIP-DIY achieves the best performances with the light self-trained FOUND, as discussed in the main paper.

3.6.2. More qualitative results

We provide in this section more qualitative examples produced with CLIP-DIY. We first compare our method against other state-of-the-art approaches in Fig. 3.6.1 for PASCAL VOC and Fig. 3.6.2 for COCO.

In Fig. 3.6.3, we then present more in-the-wild examples and conclude this section by discussing failure cases and limitations of our method in detail.

Comparisons

We first present comparisons with other methods on the two segmentation datasets used in this work, namely PASCAL VOC [88] and COCO Object [90].

PASCAL VOC Fig. 3.6.1 shows randomly sampled images from PASCAL VOC dataset and the results of CLIP-DIY and our baselines. Our method produces accurate masks for all of the images, and the result of CLIP-DIY is the closest to the ground truth compared to other methods. We observe that the two other methods, TCL [75] and CLIPpy [91], produce masks that are too coarse, with the latter frequently even assigning most of the image to one segment.

² <https://github.com/facebookresearch/CutLER/>

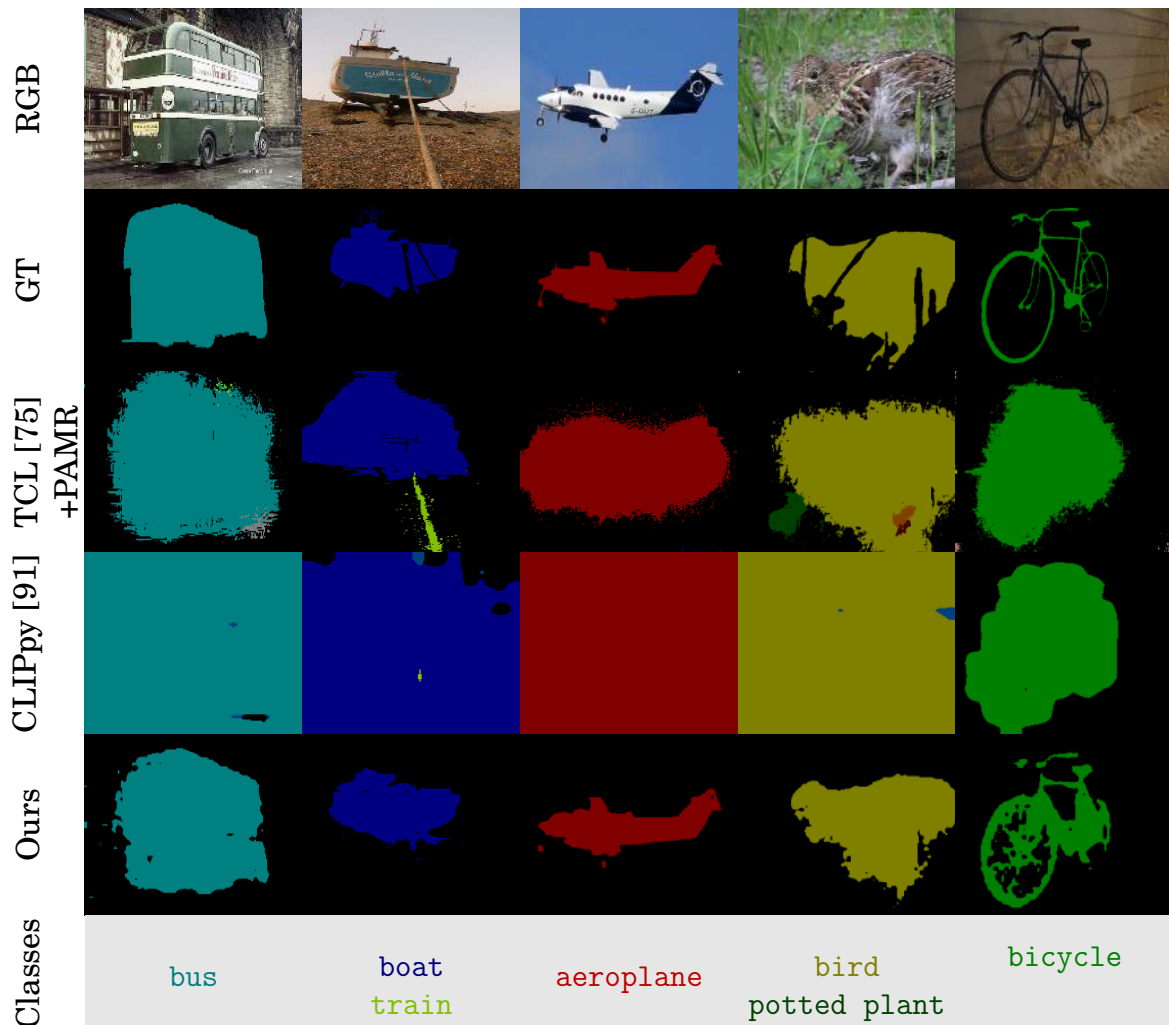


Figure 3.6.1. Qualitative segmentation results on PASCAL VOC. We compare our method against CLIPpy [91] and TCL (with PAMR post-processing) [75]. Our method consistently outperforms two other methods by producing accurate segmentation masks.

COCO Object Fig. 3.6.2 shows the examples from COCO dataset. While generating masks with mostly the correct category, TCL produces very noisy boundaries compared to CLIP-DIY. CLIPpy not only generates noisy masks but also produces a lot of clutter, assigning wrong labels to background pixels.

In-the-wild examples

We provide more in-the-wild examples to showcase the open-vocabulary abilities of our method. In Fig. 3.6.3, we present a couple of randomly mined images from the Web in comparison with TCL [75]. Both of the methods correctly assign queries to proper segments, even very specific types of objects, such as traditional dishes e.g. **polish dumplings** and **pasteis de nata**; monuments **Eiffel tower** and **Sacré coeur**. Moreover, thanks to the CLIP backbone, both methods can distinguish

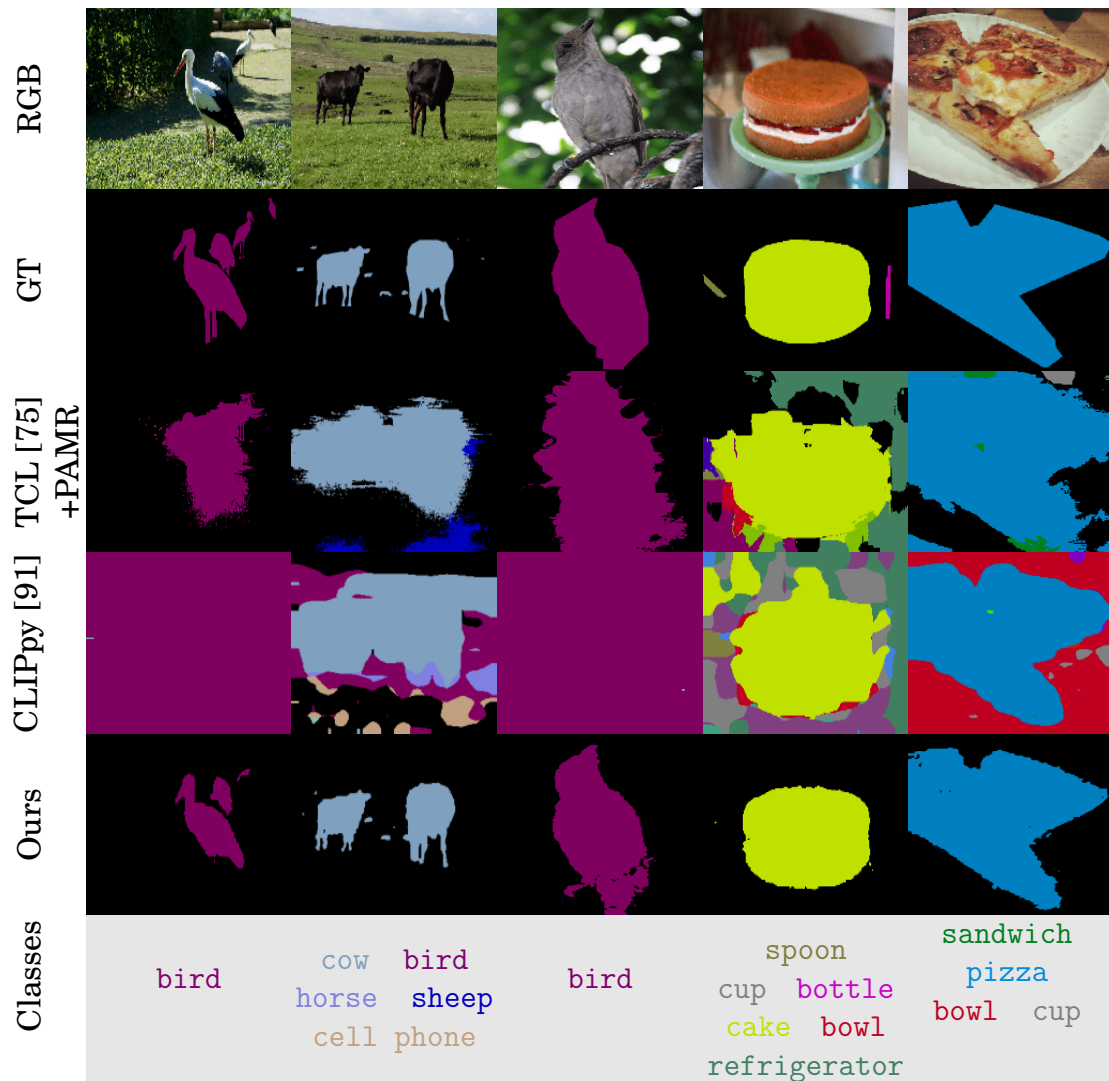


Figure 3.6.2. Qualitative segmentation results on COCO. We compare our method against CLIPpy [91] and TCL [75] (with PAMR [128] post-processing) [75]. Our method consistently outperforms two other methods by producing accurate segmentation masks. TCL and CLIPpy also both suffer from hallucinating or producing noisy masks.

between different colours, e.g. grey elephant against pink elephant. However, we observe that the quality of masks produced by TCL again is not as detailed as ours. Note that TCL uses the PAMR post-processing technique; thus, we would expect the generated masks to be more precise.

Failure cases

We analyze failure cases of our method in Fig. 3.6.4. We can see that CLIP-DIY suffers from producing incomplete masks column (a) and missing objects (c). This happens due to the saliency produced by the foreground-background segmenter, which, in the case of complex, multi-object scenes, focuses on certain aspects of a



Figure 3.6.3. More examples of in-the-wild open-world segmentation. We compare the segmentation produced by our method with the results of TCL [75] (with PAMR [128] post-processing). While both methods are able to detect and locate each class, including distinguishing between pink elephant and grey elephant, TCL largely over-segments objects.

scene. Moreover, our method has limited performance in the case of overlapping objects, such as dog and chair in column (d). Finally, we find more failures due to inaccurate annotations, such as the one in column (b), where bowl is misclassified by what is inside, i.e. carrot.

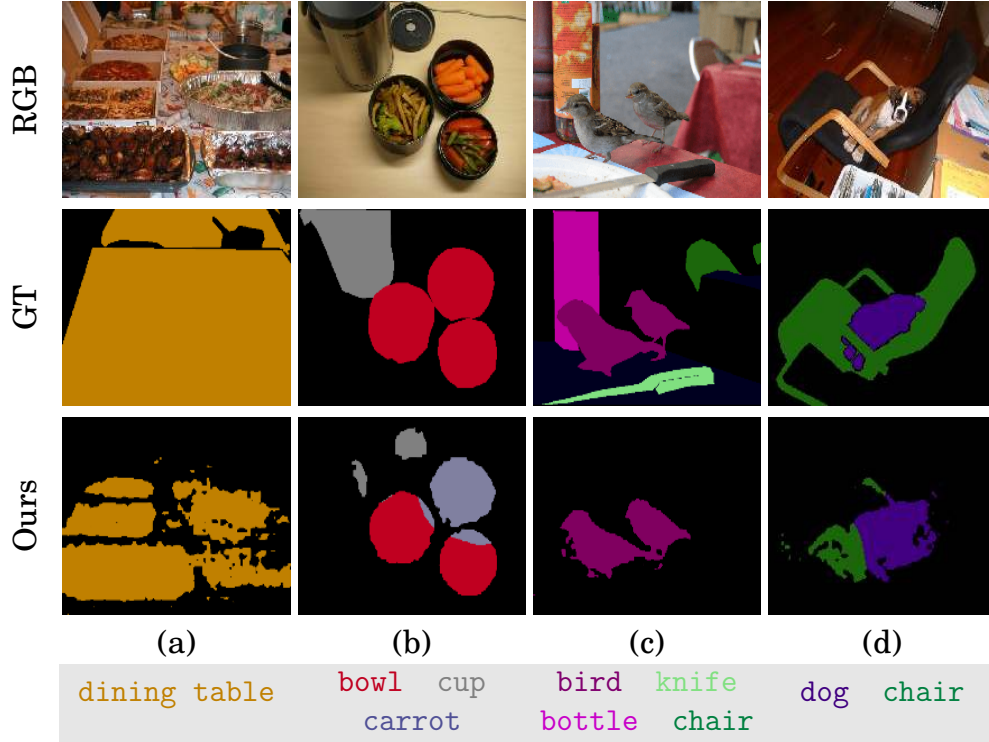






















Figure 3.6.4. Failure cases. We show examples from both datasets. Our method at times produces incomplete masks when saliency focuses only on parts of the scene such as in (a) and (c), ambiguous classification in (b), as well confusion when classes overlap (d).

Table 3.6.1. CLIP-DIY zero-shot performance (IoU) on the 21 classes from Pascal VOC. The background class is denoted as \square .

																				
87.6	78.1	33.8	77.5	62.6	65.4	71.4	66.0	81.6	16.1	75.2	20.0	78.5	69.8	63.8	56.6	37.3	79.9	25.0	68.3	37.5

Detailed quantitative results

We present detailed quantitative results on PASCAL VOC in Tab. 3.6.1 for each class. We observe that the worst performance is obtained on classes which are typically only partially visible in images, such as furniture (chair, sofa and table). This is mostly due to the decreased performance of the saliency detector in those classes, which is biased towards object-centric images. The high performance (87.6 IoU) for the background class confirms the efficacy of our saliency detector.

4. Leveraging Complementary Visual Foundation Model

In the previous chapter, we demonstrated a task adaptation strategy through modified inference, exemplified by CLIP-DIY. While this approach eliminates the need for training and human annotations, its effectiveness is limited by two key constraints: the fixed partitioning mechanism and the method’s dependency on foreground/background segmentation for localization. These limitations make CLIP-DIY suboptimal for complex scenes containing multiple objects and challenging backgrounds.

Our use of the unsupervised saliency detector FOUND [100] led us to reconsider how external models might enhance CLIP’s patch-level representations. FOUND, built upon the self-supervised foundation model DINO (discussed in Sec. 2.1.2), demonstrates robust object localization capabilities without requiring fine-grained supervision. The successful integration of these complementary approaches – vision-language modeling (CLIP) with self-supervised learning (FOUND and DINO) – suggested promising opportunities for hybrid solutions.

This chapter introduces a novel approach to unsupervised downstream task adaptation through the employment of complementary visual foundation models. Our method, CLIP-DINOiser, leverages the complementary strengths of image-text-aligned and self-supervised visual representations to enhance performance in open- vocabulary semantic segmentation. A key innovation lies in our ability to directly integrate DINO’s localization priors into CLIP’s representation space, thus avoiding the computational burden of running two large models in parallel. This integration is achieved through a lightweight adaptation module trained with DINO’s supervision, while preserving CLIP’s original representations. CLIP-DINOiser achieves state-of-the-art performance on challenging and fine-grained benchmarks while maintaining minimal computational requirements. The method requires only a single CLIP forward pass and two lightweight modules during inference, operating without additional supervision or memory overhead.

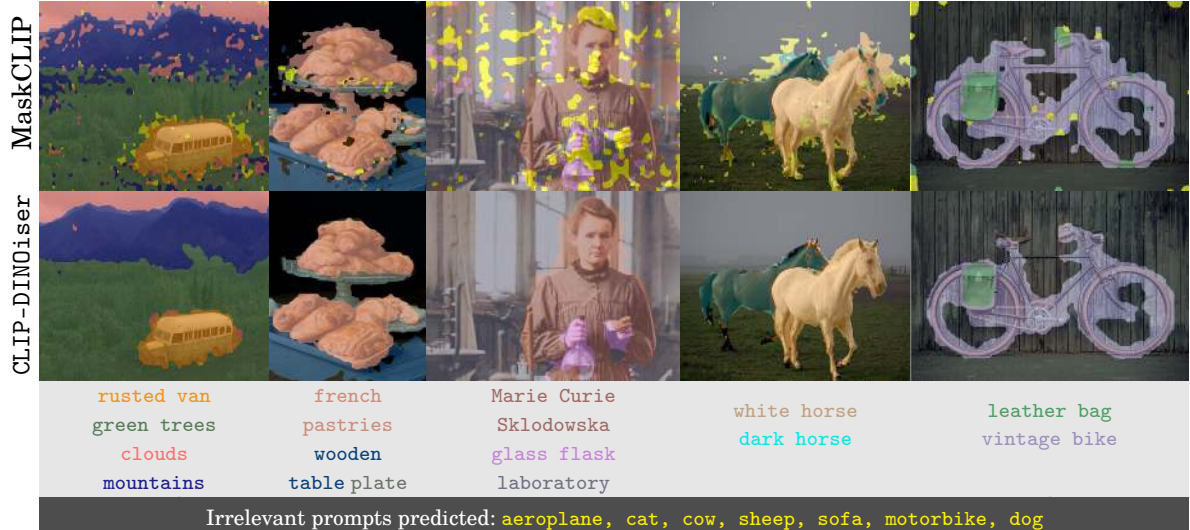


Figure 4.1.1. Examples of open-vocabulary semantic segmentation results obtained with our method CLIP-DINOiser on ‘in-the-wild’ images vs. those of MaskCLIP [28]. Our method improves MaskCLIP features with a smart pooling strategy, which does *not alter the original* open-vocabulary properties. We use self-supervised DINO [15] as a guide to *teach CLIP* [54] to produce DINO-like localization features through two light convolutional layers. Our method, which achieves state-of-the-art results, only requires a *single forward* pass through CLIP model and our two layers. In addition to the correct prompts (light grey row), we list the irrelevant prompts predicted (in yellow) that we query in all images shown here.

4.1. Introduction

Semantic segmentation is a key visual perception task for many real-world systems, e.g., self-driving cars and industrial robots. Typically tackled in a dataset-oriented manner, best methods require a training dataset which is manually annotated for a *specific and finite* set of classes. The advent of powerful Vision-Language Models [13, 24, 57] is stimulating a shift from a closed-vocabulary paradigm to an *open-world* one. Such models are trained with a simple but scalable objective: to align pairs of image and coarse text captions that can be obtained in large amounts with limited manual supervision. VLMs excel at associating *global* image content with arbitrary text inputs with remarkable generalization capabilities [130, 131], but struggle to provide dense *open-vocabulary features* [28, 51]. Obtaining such an alignment between pixels and language can lead to open-vocabulary extensions for multiple other modalities, such as point clouds [132, 133, 134, 135], 3D scenes [136], 3D shapes [137], radiance fields [138], inter-modality alignment [139, 132], with multiple potential applications for which the construction of training datasets is even more challenging and where CLIP-derived models show promising results.

Different strategies have been recently proposed towards improving CLIP’s patch-level feature extraction abilities by modifying the original CLIP architecture

for dense pooling and retraining [94, 75, 91, 116, 92] or finetuning on an annotated segmentation dataset with pre-defined classes [93, 28]. The former requires long training and/or large collections of annotated data, while the latter leads to an alteration of the vision-language associations of the CLIP features. An alternative line of approaches freezes the CLIP encoder and directly densifies its features with different heuristics, often with multiple forward passes [138, 137, 132, 29, 118, 140], but are less practical due to the extensive computational overhead. MaskCLIP [28] arises as a computationally efficient dense CLIP extractor. It converts CLIP’s global self-attention layer into a convolutional one to produce patch features with original vision-language qualities. If such features are local, they appear to be too noisy for high-quality segmentation mask extraction (see Fig. 4.3.1b middle column).

Meanwhile, recent self-supervised learning (SSL) approaches [15, 44, 40, 41] produce strong visual representations displaying object localization properties and such without requiring any manual annotation. DINO [15] stands out with its semantically meaningful features, which have been exploited for unsupervised object discovery [95, 96, 100, 97]. DINO features prove useful also for zero-shot semantic segmentation [29, 141, 138], but require expensive sliding window sampling [29, 138] or building concept-specific prototypes and ensemble strategies [141].

In this work, we aim for unaltered patch-level CLIP features with minimal run-time overhead. To this end, we re-examine the localization properties of MaskCLIP features and observe that it is possible to easily refine them with guidance from SSL models. In detail, we train a simple convolutional layer on unlabeled data to produce pooling weights to perform correlation-guided dense feature pooling from CLIP without distorting the vision-language alignment. This layer is optimized to mimic the patch correlations of DINO [15] that indicate likely layouts of visual concepts in the images. Furthermore, we show that the unsupervised objectness information given by FOUND [100] from DINO features can also be directly learned from CLIP features again in a fully unsupervised fashion with a single convolutional layer and helps improve the segmentation of the ill-defined ‘background’ prompt. With CLIP-DINOiser, we obtain high-quality masks in *a single forward pass* on CLIP (see Fig. 4.1.1). CLIP-DINOiser is amenable to producing dense semantic maps.

To summarize, our contributions are:

- We propose a light pooling mechanism to refine MaskCLIP features by leveraging guidance from SSL features without degrading its original open-vocabulary properties. CLIP-DINOiser does not require any annotations nor retraining CLIP from scratch, but only a single CLIP forward pass.
- We show that CLIP *already contains good localization properties* which can be exploited. We leverage simple convolutional layers to emphasize visual concept

layouts from dense CLIP features. We train them without any annotation on only 1k of raw images randomly sampled in ImageNet [6]. We believe that this finding could be further exploited in different contexts.

- Our method achieves state-of-the-art results on complex semantic segmentation datasets such as COCO [142], Pascal Context [88], Cityscapes [143] and ADE20K [144].

4.2. Related Work

Zero-shot semantic segmentation. This task has been typically approached by methods which aim at generalizing from *seen* classes to *unseen* ones [102, 101, 103, 104, 105, 106, 107, 108]. Such strategies train models with full supervision on the set of seen classes and propose different solutions to extend them to unseen ones without new images (labeled or unlabeled), e.g., by exploiting class information and relationships encapsulated in popular word embeddings [109, 110]. While they produce fine segmentations without computational overhead, these methods require pixel-level annotations for the seen classes.

From CLIP to open-vocabulary segmentation. The surge of VLMs with aligned image-language representations [13, 24, 54] brought back into the spotlight the zero-shot classification task. However, the extension to zero-shot segmentation is not obvious as the CLIP architecture is not equipped to yield dense vision-language features [28, 51]. To produce dense CLIP features, several approaches fine-tune or train from scratch pixel-aligned CLIP-like models with additional modules, mechanisms or supervision objectives [94, 75, 91, 116, 92] on datasets with annotations of varying granularity and quality: dense annotations [145, 146], class-agnostic object masks [113, 51, 115], coarse captions [51, 91, 147, 112, 146, 94, 93, 116, 92, 75] or pseudo-labels [28]. Recent works leverage image-level captions to align text to regions (obtained without supervision): PACL [92] trains an embedder module to learn patch-to-text affinity, TCL [75] proposes a local contrastive objective to align well-selected patches to the text and ViewCO [117] leverages multi-view consistency. On the downside, such models require long training on millions of images or specific types of very costly annotations. Also, fine-tuning CLIP with a defined vocabulary is more computationally appealing [28, 145, 146], but alters the open-vocabulary properties of the features [132].

Most related to us is a line of works that investigate how to directly densify CLIP features [28, 29, 132, 137, 138] to obtain per-patch CLIP features. Such densification

can be performed by aggregating features from multiple views [137, 138] or from sliding windows [29, 132] at the extra-cost of multiple forward passes. MaskCLIP [28] drops the global pooling layer of CLIP and matches the projected features directly to text via a 1×1 convolution layer. By doing so they achieve dense predictions, however noisy.

With a concept-driven perspective, some methods [118, 140, 141] build codebooks of visual prototypes per concept, including negative prototypes [141], and then perform co-segmentation [118]. While such an approach yields good results, it is, however, at the cost of building expensive *class-specific prototypes*, therefore diverging from open-vocabulary scenarios. Instead, we aim to remain *open* to avoid retraining a model or building new expensive prototypes whenever a new concept is considered. To that end, we devise a dense CLIP-feature extraction method that preserves the open-vocabulary quality.

Leveraging self-supervised models & CLIP. Recent self-supervised ViTs [15, 44, 20, 41, 49] have demonstrated features with good localization properties [95, 96, 97, 100]. Such features have also been exploited in the context of open-vocabulary segmentation methods, e.g. for pre-training for the visual backbone [91, 116, 148], co-segmentation [118], clustering patches into masks [124], representing object prototypes [141]. Related to us is the recent CLIP-DIY [29], which computes patch-level representations from CLIP features from different image crops with guidance from an unsupervised saliency segmenter [100] FOUND. While we also leverage the latter, in contrast with CLIP-DIY, which runs multiple forward passes to build their dense CLIP features, our method requires only a *single forward pass* of CLIP. Furthermore, our method mitigates the limits of FOUND in cluttered scenarios by integrating an uncertainty constraint. Finally, we leverage the informative patch correlation properties of DINO [15] and show that it is possible to *teach CLIP* to produce DINO-like features through light convolutional layers.

4.3. Method

We present in this section CLIP-DINOiser, a simple and efficient strategy to improve MaskCLIP using localization information extracted from CLIP –with a lightweight model trained to mimic some of DINO’s properties. We first set the goal in Sec. 4.3.1 and present MaskCLIP [28] in Sec. 4.3.2. We then introduce our strategy which leverages self-supervised features localization information to consolidate MaskCLIP features in Sec. 4.3.3 and discuss how such localization information can directly be

learnt from CLIP in Sec. 4.3.4 (we visualize both steps in Fig. 4.3.2). We also propose a way to improve the ‘background’ filtering in Sec. 4.3.5.

4.3.1. Problem statement

In this work, we aim to produce open-vocabulary¹ semantic segmentation of an image. We consider an image $X \in \mathbb{R}^{H \times W \times 3}$ which we split into a sequence of N patches of dimensions $P \times P \times 3$ with $P \times P$ the patch size and $N = \lceil \frac{H}{P} \rceil \cdot \lceil \frac{W}{P} \rceil$. A class token noted CLS, is added to the input sequence, and we feed the $N + 1$ patches to a ViT [125] model. We aim at producing dense visual features $F \in \mathbb{R}^{N \times d}$, with d the feature dimension, that can later be matched to *any* set of text inputs embedded in the same space. In particular, the goal is to produce a segmentation map per textual query.

4.3.2. Preliminaries on MaskCLIP

Extracting dense open-vocabulary features. The popular CLIP [54] model pre-trained on image/caption pairs produces good *global* image features but was not trained to generate high-quality 2D feature maps. To extract such dense feature maps relevant to semantic segmentation, Zhou et al. [28] revisit the global attention pooling layer of the last attention layer of the model. The authors discard the *query* and *key* embeddings of the layer and transform both the *value* projection and the last linear layer into a conv 1×1 layer. With this new model, named MaskCLIP and denoted $\phi(\cdot)$, we extract d -dimensional features $\phi^L(X) \in \mathbb{R}^{N \times d}$ from the last layer L which retains most of the open-vocabulary properties of CLIP [28].

Semantic segmentation given textual queries. We also extract CLIP textual features $\phi_T(t_j)$ for each text query $t_j \in \mathcal{T}$ with $j \in \{1, \dots, |\mathcal{T}|\}$. Segmentation maps are then generated by computing the cosine similarity between each of the visual patch features and of the textual prompts after L2-normalization. The most similar prompt is assigned to each patch. Note that a query ‘background’ can be added in order to obtain *negative* patches. Using MaskCLIP allows us to produce dense segmentation maps with a single forward pass of the classic CLIP model, but its outputs are noisy, as visible in Fig. 4.3.1b (middle column).

¹ We adopt the taxonomy defined in the recent survey [14] and define our method as ‘open-vocabulary’, with capabilities to generalize to unseen datasets.

4.3.3. DINOising open-vocabulary features

In this work, we aim to improve MaskCLIP’s open-vocabulary features described above. To do so, we propose to leverage the known good localization properties of self-supervised features [15, 48, 95, 96, 100, 149] .

Extracting self-supervised correlation information. Recent works [95, 96] have shown that the patch correlation information of the embeddings from the last attention layer of the self-supervised model, DINO [15] can help highlight objects in images. We use here the *value* embeddings, which we observe have finer correlation than those of key and query (more discussion in Sec. 4.6). We extract such self-supervised features $\xi(X) \in \mathbb{R}^{N \times d_\xi}$ and discard the CLS token. We then compute the per-patch cosine-similarity and produce the affinity map $A^\xi \in [-1, 1]^{N \times N}$. We compare in Fig. 4.3.3 the patch-similarities obtained for a patch *seed* with MaskCLIP and DINO features and observe that the self-supervised features are more densely and accurately correlated than those of CLIP.

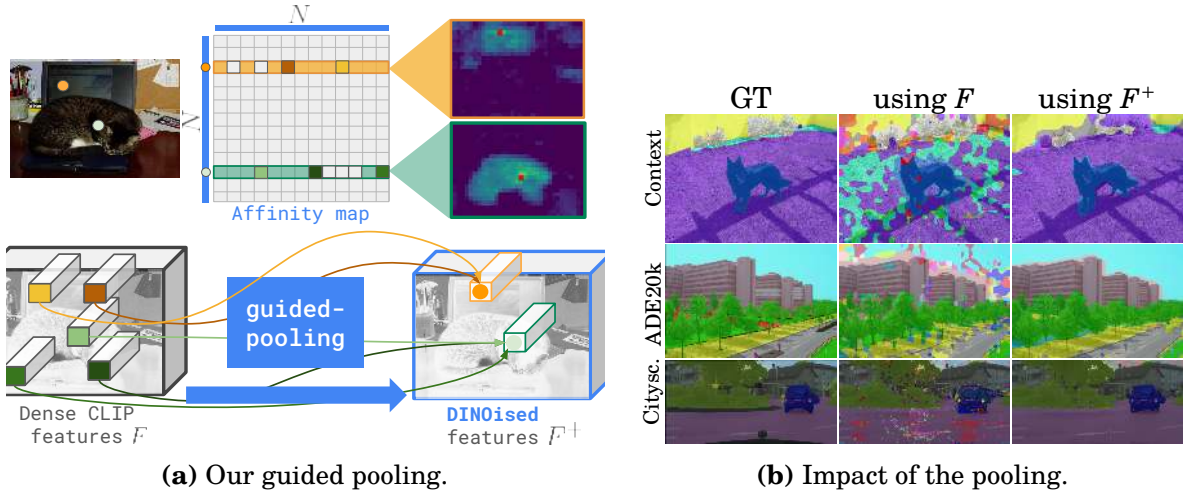


Figure 4.3.1. We present in (a) is our *guided pooling* strategy defined in Eq. (4.1). The $N \times N$ affinity matrix is computed from patch features and is used to refine MaskCLIP features (bottom left). In (b) we compare our results with F^+ (right) versus those obtained with MaskCLIP features (middle).

Strengthening features with guided pooling. In order to locally consolidate MaskCLIP features $\phi^L(X)$, now noted F , we propose to perform a *concept-aware* linear combination of the features per patch with guidance from the patch affinity A^ξ . The feature combination strategy can be seen as a form of voting mechanism that enforces similar patches to have similar CLIP features (and prediction) while attenuating noisy features. Specifically, we compute the new features $F^+ \in \mathbb{R}^{N \times d}$ as an average of MaskCLIP features F weighted by A^ξ , presented in 4.3.1a. We zero-out

A^ξ correlations below a threshold γ , following [95, 96], and compute the new features for patch $p \in \{1, \dots, N\}$:

$$F_p^+ = \frac{1}{\sum_{q=1}^N A_{p,q}^\xi} \sum_{q=1}^N A_{p,q}^\xi \cdot F_q. \quad (4.1)$$

We then produce the segmentation maps $S \in [-1, 1]^{N \times |T|}$, by comparing the new features F^+ to each textual queries in \mathcal{T} . As shown in 4.3.1b, when using such consolidated features, we obtain more accurate outputs and the high-frequency predictions observed in MaskCLIP are smoothed out, showing the benefit of the pooling.

4.3.4. Teaching CLIP a first DINO trick: object correlations

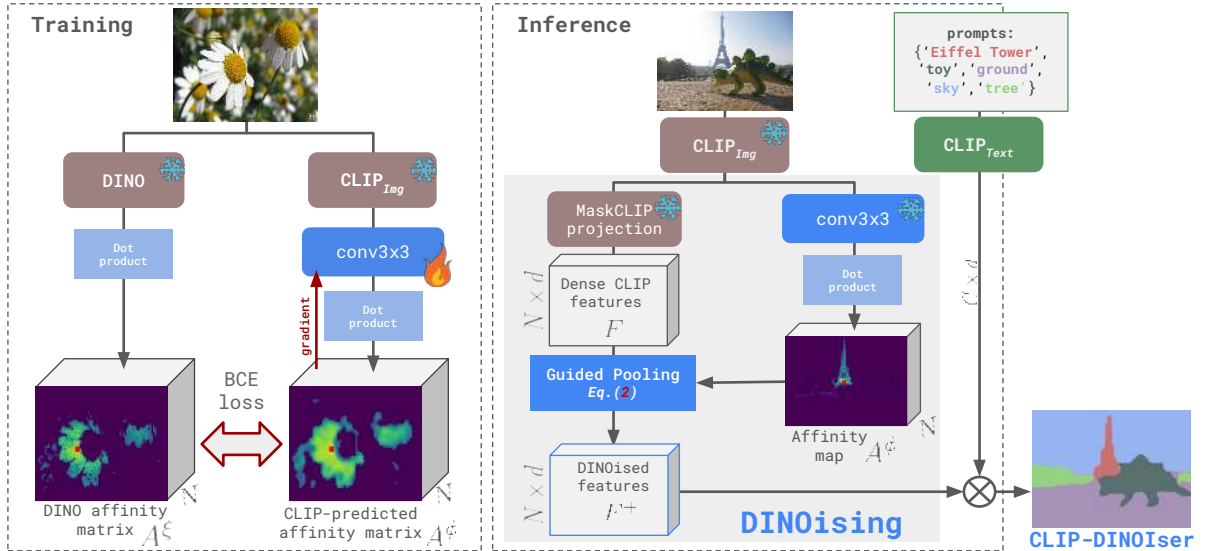


Figure 4.3.2. Overview of CLIP-DINOiser which leverages the quality of self-supervised features to improve the notoriously noisy MaskCLIP feature maps. We use DINO as a teacher which ‘teaches’ CLIP how to extract localization information. We train (left) a $\text{conv}3 \times 3$ layer to reproduce the patch correlations obtained with DINO. At inference (right), an input image is forwarded through the frozen CLIP image backbone and MaskCLIP projection. The produced features are then improved with our *pooling* strategy which is guided by correlations predicted with the trained convolutional layer applied on CLIP. With this light ‘DINOising’ process, we obtain ‘DINOised’ features which are matched against the prompts features to produce CLIP-DINOiser outputs.

We have shown in the previous section that self-supervised correlation information can successfully be used to improve the dense quality of open-vocabulary features. If the difficulty of densifying CLIP is well-known, we show here that CLIP features already contain *good localization information*, which can be extracted with

a light model. We indeed predict DINO correlations A^ξ from CLIP with a single convolutional layer.

In order to predict the DINO affinity map A^ξ from CLIP features, we train a *single* 3×3 convolutional layer $g(\cdot): \mathbb{R}^d \rightarrow \mathbb{R}^{d_g}$ which projects intermediate features $\phi^l(X)$ —extracted from layer l —into a smaller space of dimension $d_g < d$. We enforce the patch correlations of the generated features $A^\phi \in [-1, 1]^{N \times N}$:

$$A^\phi = \frac{g(\phi^l(X))}{\|g(\phi^l(X))\|} \otimes \left(\frac{g(\phi^l(X))}{\|g(\phi^l(X))\|} \right)^\top, \quad (4.2)$$

with \otimes denoting the outer product, to be close to the binarized correlations $D = A^\xi > \gamma$ (we use here the same γ as defined above), using the binary cross-entropy loss \mathcal{L}^c :

$$\mathcal{L}^c = \sum_{p=1}^N \left[D_p \log A_p^\phi + (1 - D_p) \log(1 - A_p^\phi) \right]. \quad (4.3)$$

We present our layer training in Fig. 4.3.2 (left part) and observe the quality of CLIP-predicted affinity matrix A^ϕ . We also show in Fig. 4.3.3 another example of obtained A^ϕ and observe their similarity to DINO-based correlations. We use the CLIP-produced correlations A^ϕ to replace A^ξ in Eq. (4.1) to weight the pooling and observe a similar boost over MaskCLIP, thus showing that good patch correlations can indeed be extracted directly from CLIP. We can now discard DINO, and we name CLIP-DINOiser the guided-pooling strategy, which uses CLIP-based correlation. As shown in Fig. 4.3.2 (*inference* step), our method runs with a single forward pass of CLIP model and a small extra layer.

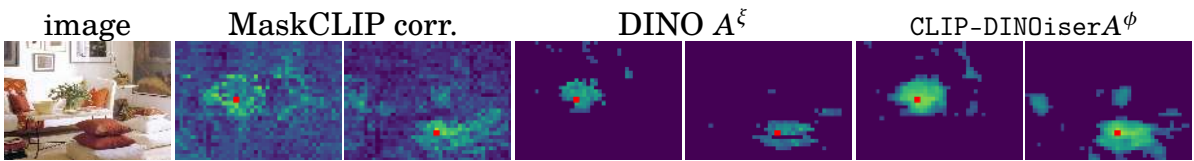


Figure 4.3.3. Comparison of the affinity maps between a *seed* (one on the ‘plant’ and the other on a ‘pillow’) and the other patch features when using features of MaskCLIP, DINO and ours after training.

4.3.5. Teaching CLIP a second DINO trick: background filtering

Moreover, as discussed earlier, a ‘background’ query may be added to the set of textual queries \mathcal{T} in order to help filter out patches falling in the *background* and not corresponding to any objects. We do not assume here any prior knowledge about classes of interest and focus rather on the foreground/background paradigm [100]. We argue that relying solely on the textual prompt ‘background’ to catch

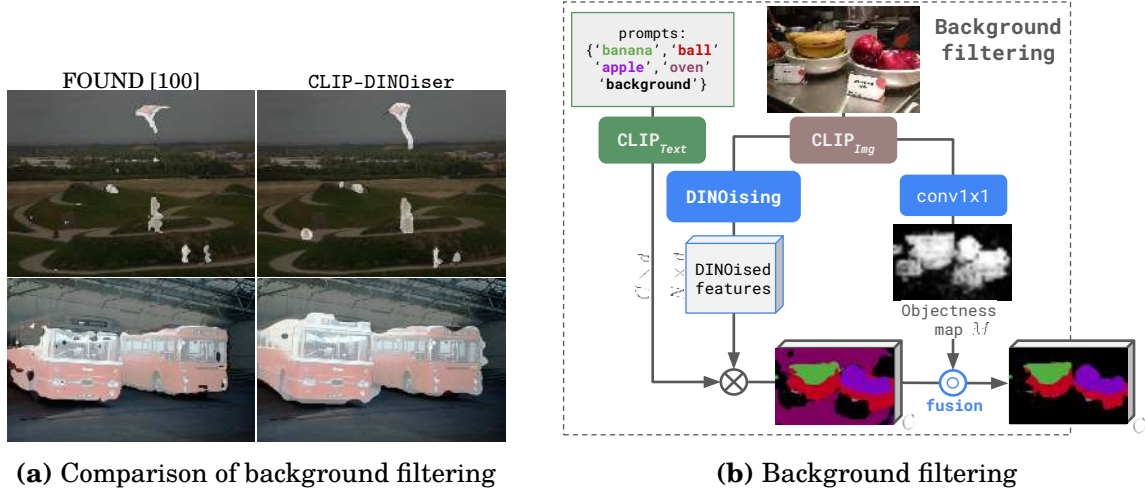


Figure 4.3.4. We present in (a) a comparison of *objectness* mask generated by FOUND [100] and with our layer using CLIP features. We carefully define the fusion operation and the simple training strategy of the $\text{conv}1 \times 1$ again using DINO as a teacher in Sec. 4.3.5. In (b) is an overview of our *background filtering* which is applied when a ‘background’ prompt is provided and helps reduce hallucinations.

all non-salient patches is underperforming and, similarly to [29], we propose to use a very light-weight *unsupervised* foreground/background segmentation method, namely FOUND [100] which also relies on DINO self-supervised features. We run FOUND on the entire image and extract a prediction mask $M \in \{0, 1\}^N$ in which a patch is assigned the value 1 if falling into the foreground and 0 otherwise. We also observe that saliencies produced by FOUND can be too restrictive, and objects that are partially visible or in clutter should be discarded. To mitigate this behavior, we propose to relax the background selection by integrating an additional uncertainty constraint. To this end, we fuse the background information from both modalities by assigning the ‘background’ prompt to patches p which are both *uncertain*, e.g. have low confidence score $\sigma(S)_p < \delta$, with $\sigma(\cdot)$ the softmax operation, *and* which fall in the background in M .

Learning FOUND objectness. Moreover, we are also able to learn the predictions of FOUND [100] directly from CLIP features. To do so, we train a *single* 1×1 convolutional layer $h(\cdot): \mathbb{R}^d \rightarrow \mathbb{R}$ which predicts from the features $\phi^l(X)$ an objectness map $M^\phi = h(\phi^l(X)) \in \mathbb{R}^N$. We train the model to predict the FOUND binary mask M with the binary cross-entropy loss \mathcal{L}^m :

$$\mathcal{L}^m = \sum_{p=1}^N \left[M_p \log(M_p^\phi) + (1 - M_p) \log(1 - M_p^\phi) \right]. \quad (4.4)$$

We show examples of predicted CLIP-based objectness in Fig. 4.3.4a and observe their very high similarity to those produced with DINO. Moreover, we can now replace M defined above with the binarized CLIP-based scores $\zeta(M^\phi) > 0.5$, with $\zeta(\cdot)$ the sigmoid operation, and observe a minimal drop in performances. We show an example of background filtering with trained objectness in Fig. 4.3.4b.

4.4. Experiments

We detail in Sec. 4.4.1 the experimental setup used in our evaluation. We produce state-of-the-art results on the task of open-vocabulary semantic segmentation in Sec. 4.4.2 and ablation studies in Sec. 4.4.3.

4.4.1. Experimental setup

Technical details. We use in all experiments a *frozen* CLIP ViT-B/16 pre-trained following OpenCLIP [54]. Our method CLIP-DINOiser uses two convolutional layers to extract DINO-like information from CLIP layer $l = 10$ (the 3rd before the last which was shown to provide the best results [150]). The first layer $g(\cdot)$ has a kernel 3×3 and output dimension $d_g = 256$ and $h(\cdot)$ a kernel 1×1 with $d_h = 1$. The first is trained to match the correlation information extracted from the *value* embeddings of the last layer of a ViT-B/16 model trained following DINO [15]. The second layer is trained to replicate the unsupervised object localization predictions of FOUND [100]—which also uses DINO model. We train both layers with a binary cross-entropy loss on *only 1k raw images* randomly sampled from ImageNet [6] dataset *without any annotation*. We report average scores over 3 runs with different sampling seeds and provide standard deviations in Sec. 4.6. We follow [96] and binarize the correlations with $\gamma = 0.2$. In the background filtering step, we use a high confidence score, i.e., $\delta = 0.99$. We train our model for 6k iterations with a batch size of 16 images using Adam optimizer [151], which takes approximately 3 hours on a single NVIDIA RTX A5000 GPU. We decrease the learning rate for both heads by a factor of 0.1 after 5k iterations. We apply data augmentations during training (random scale and cropping, flipping and photometric distortions).

Datasets and metric. We evaluate our method on eight benchmarks typically used for zero-shot semantic segmentation [75]. Following [75], we split them into two groups. The first consists in datasets with a ‘background’ query: PASCAL VOC [88] (noted ‘VOC’), PASCAL Context [152] (noted ‘Context’), and COCO Object [90] (noted ‘Object’) and the second without: PASCAL VOC20 [88] (noted ‘VOC20’), PASCAL

Context59 [152] (noted ‘C59’), COCO-Stuff [142] (noted ‘Stuff’), Cityscapes [143] (noted ‘City’), and ADE20K [144] (noted ‘ADE’). We evaluate results with the standard mIoU metric. We also follow the evaluation protocol of [75], use the implementations provided by MMSegmentation [153], employ a sliding window strategy, resize the input image to have a shorter side of 448. We also do not perform text expansions of the class names and use only the standard ImageNet prompts following [54, 94, 28].

Baselines. We compare our method against state-of-the-art methods on OVSS. For a fair comparison between methods, we report results without any post-processing step. In our evaluations, we follow the taxonomy presented in [14] and compare our model with the methods relying on language-image pretraining, also called open-vocabulary. We split the compared baselines into four categories: (1) *dataset specific* which employ pseudo-labeling and supervised training of a segmentation model on target dataset: NamedMask [140], MaskCLIP+ [28]); (2) *construct prototypes*: ReCO [118], OVDiff [141]; (3) *train with text supervision* including GroupViT [94], ZeroSeg [124], SegCLIP [112], TCL [75], CLIPpy [91], OVSegmentor [116], which all require access to additional datasets of millions of image/caption pairs (we note in the table the exact datasets used for the training); and finally *use frozen CLIP* i.e. CLIP-DIY [29] and MaskCLIP [28], which use pre-trained CLIP. Our method falls into the last category as we do not modify CLIP, and do not need access to additional caption annotations as we use only 1k unannotated images.

4.4.2. Open-vocabulary semantic segmentation

In this section, we discuss state-of-the-art results on the task of open-vocabulary semantic segmentation.

Evaluation with no ‘background’ class. We first compare in Tab. 4.4.1 (‘No background prompt’ column) the results on datasets which aim at the segmentation of most of the pixels in an image and do not consider a ‘background’ class. We observe that our method CLIP-DINOiser achieves the best results on four datasets yielding +2.2, +5.0, +6.7 and +5.1 mIoU over the second best performing method. Interestingly, we outperform methods which build expensive prototypes per visual concept on fine-grained datasets, showing the benefit of our lightweight and generalizable method. The only drop (-0.8 mIoU) is seen on VOC20 with respect to OVDiff; we believe it is due to the benefit of generating per-concept negative prototypes, which likely benefits this object-centric dataset. An adaptive granularity of feature correlation could help mitigate this drop, which we leave for future work.



Methods	Concept spec.		Extra data	Inference backbone	No background prompt					W/ bkg prompt		
					VOC20	C59	Stuff	City	ADE	Cont.	Object	VOC
Dataset specific												
MaskCLIP+	✓	✗	I	DLv2	-	31.1	18.0	-	-	-	-	-
NamedMask	✓	✗	I	DLv3+	-	-	-	-	-	-	27.7	59.2
Build prototypes per visual concept												
ReCo	✓	✓	I	CLIP	57.8	22.3	14.8	21.1	11.2	19.9	15.7	25.1
OVDiff	✓	✓	✗	CLIP+ DINO+SD	81.7	<u>33.7</u>	-	-	<u>14.9</u>	30.1	34.8	67.1
Text/image alignment training with captions												
GroupViT	✗	✗	IT	CLIP	79.7	23.4	15.3	11.1	9.2	18.7	27.5	50.4
ZeroSeg	✗	✗	IT	CLIP	-	-	-	-	-	21.8	22.1	42.9
SegCLIP	✗	✗	IT	CLIP	-	-	-	11.0	8.7	24.7	26.5	52.6
TCL	✗	✗	IT	CLIP	77.5	30.3	<u>19.6</u>	23.1	<u>14.9</u>	24.3	30.4	51.2
CLIPpy	✗	✗	IT	CLIP	-	-	-	-	13.5	-	32.0	52.2
OVSegmentor	✗	✗	IT	CLIP	-	-	-	-	5.6	20.4	25.1	53.8
Frozen CLIP												
CLIP-DIY	✗	✓	✗	CLIP+ DINO	79.7	19.8	13.3	11.6	9.9	19.7	31.0	59.9
MaskCLIP	✗	✓	✗	CLIP	53.7	23.3	14.7	21.6	10.8	21.1	15.5	29.3
MaskCLIP*	✗	✓	✗	CLIP	61.8	25.6	17.6	<u>25.0</u>	14.3	22.9	16.4	32.9
MaskCLIP* †	✗	✓	✗	CLIP	71.9	27.4	18.6	23.0	<u>14.9</u>	24.0	21.6	41.3
CLIP-DINOiser	✗	✓	I(1k)	CLIP	<u>80.9</u>	35.9	24.6	31.7	20.0	32.4	34.8	<u>62.1</u>

Table 4.4.1. Open-vocabulary semantic segmentation quantitative comparison using the mIoU metric. We separate in two columns the evaluation datasets: those without a ‘background’ prompt and those with (noted ‘W/ bkg prompt’), as discussed in Sec. 4.4.1. We report all methods without post-processing. We note with * methods for which we computed scores; we obtained MaskCLIP* scores with OpenCLIP [54] and mark with † the use of MaskCLIP refinement. The first and second best methods are, respectively, **bold** and underlined. We specify if a method assumes prior access to names of concepts (‘Concept spec.’) and if it employs a frozen backbone (). We specify what additional data is used at training (‘Extra data’) (‘I’ stands for images and ‘IT’ for image/text aligned data). Our CLIP-DINOiser only needs 1k images from ImageNet to be trained. ‘SD’ stands for Stable Diffusion [154]. We refer to Sec. 4.4.1 for more details on baselines and we detail the datasets used for training by each method in the supplementary material.

Evaluation with ‘background’ class. We now compare our method on datasets which include a ‘background’ query in Tab. 4.4.1 (‘W/ bkg prompt’ column). In this setup, we also apply our background detection mechanism (detailed in Sec. 4.3.5) on VOC and Object in order to improve the stuff-like background detection. We observe that CLIP-DINOiser significantly outperforms all methods which do not construct prototypes. Moreover, we surpass OVDiff (which uses an ensemble of three models) on Context dataset by +2.3 mIoU and are on par on Object. It is to be noted that with a single feature extractor, the performance of OVDiff drops by -10 mIoU,

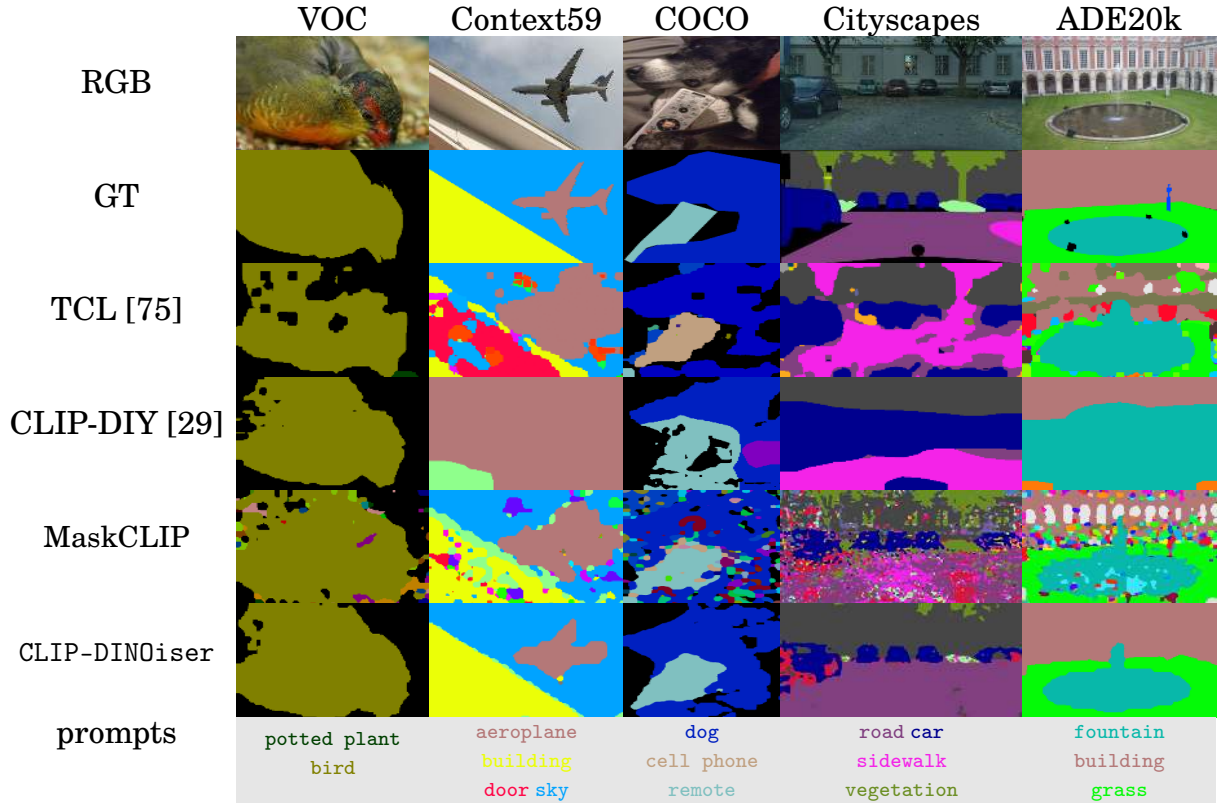


Figure 4.4.1. Qualitative open-vocabulary segmentation results. We compare ours against CLIP-DIY [29], TCL [75] and MaskCLIP [28]. For a fair comparison, we do not apply post-processing. All pixels annotated in black are from the background class. We observe that our method achieves more accurate results both in terms of localization and class assignment.

and the method requires the construction of a ‘background’ prototype *per concept*, otherwise losing another -10 mIoU on VOC. On the other hand, CLIP-DINOiser produces segmentation masks in a *single* pass of CLIP with the light addition of two convolutional layers while remaining fully open-vocabulary as it does not require *any* concept-specific constructs.

Qualitative results. We qualitatively compare in Fig. 4.4.1 CLIP-DINOiser with high-performing TCL [75], CLIP-DIY [29] (two recent methods which provide code) and our baseline method MaskCLIP [28] on images taken from the datasets considered in the evaluation. We observe that our method generates predictions accurate both in terms of localization and assignment. Indeed, we obtain fined-grained results on the challenging datasets, e.g. in the Cityscapes example, the text query ‘car’ and in the ADE20k example ‘fountain’ are accurately located when CLIP-DIY and TCL produce coarser results. Versus MaskCLIP, we can see the denoising capabilities of CLIP-DINOiser as MaskCLIP hallucinations grow with the number of text queries prompted at evaluation. Finally, in Fig. 4.1.1, we present ‘in the wild’ examples

beyond the evaluation benchmarks and show that CLIP-DINOiser produces accurate segmentation masks for arbitrary and very specific prompts, such as ‘wooden table’ or ‘leather bag’.

4.4.3. Ablation study

We now conduct an ablation study of the different components of CLIP-DINOiser and investigate the impact of both our feature pooling strategy and background detection.

The impact of the pooling mechanism. We propose with CLIP-DINOiser to combine MaskCLIP *features* with a well-defined linear combination and compare different solutions in Tab. 4.4.2a. In [28], the authors proposed to refine the *predictions* with a combination weighted by CLIP *key* embeddings (noted ‘CLIP keys (preds.)’ in the table) and boost MaskCLIP results by more than +8 mIoU on VOC and VOC20, +1.8 and +1.0 and +0.6 mIoU on the other datasets. However, we show that working directly at the feature level allows us to achieve better results; we obtain consistent improvements ranging from +6 to +19 mIoU on all datasets when using DINO-based weight A^ξ and further improve when using trained CLIP-based weights A^ϕ .

Pooling strategy	VOC	VOC20	C59	Stuff	ADE	Pooling	Bkg det.	Object	VOC
MaskCLIP [28] - <i>baseline</i>	32.9	61.8	25.6	17.6	14.3	MaskCLIP [28] - <i>baseline</i>		16.4	32.9
CLIP keys (preds.) [28]	41.3	71.9	27.4	18.6	14.9	ours w. DINO A^ξ		29.9	53.7
ours w. CLIP keys	39.2	73.2	23.0	12.6	7.7	ours w. DINO A^ξ	FOUND	32.1	60.1
ours w. DINO A^ξ	53.7	79.1	35.5	24.7	20.4	ours w. DINO A^ξ	ours w. M	34.1	62.1
ours w. trained A^ϕ	54.0	80.9	35.9	24.6	20.0	ours w. DINO A^ξ	ours w. M^ϕ	34.2	61.9
						ours w. trained A^ϕ	ours w. M^ϕ	34.8	62.1

(a) Pooling strategy

(b) Background detection

Table 4.4.2. Impact of the pooling strategy (a) and background detection (b) on diverse datasets reported with the mIoU metric.

The impact of the background detection. We now discuss the improvement provided by our background refinement strategy, which is applied when *stuff*-like background patches need to be detected. We report such results in Tab. 4.4.2b when employing our pooling strategy (either using DINO features, noted ‘w. DINO A^ξ ’ or those extracted from CLIP, noted ‘w. trained A^ϕ ’). When using solely ‘FOUND’ for background detection, as in [29], we improve by +6.4 mIoU on VOC (achieving 60.1 mIoU), but when relaxing FOUND (see Sec. 4.3.5) with an uncertainty condition, we boost scores up to 62.1 on VOC, showing the limitation of using FOUND alone. We also achieve similar results when using CLIP-based predictions M^ϕ both with DINO-based A^ξ and trained CLIP-based A^ϕ correlations, although we observe that best results are overall obtained with trained A^ϕ . We visualize CLIP-based mask M^ϕ

in Fig. 4.3.4a and see high similarity to DINO-based predictions, therefore showing the localization quality of CLIP.

4.5. Conclusions

In this work, we propose to make the most out of CLIP features and show that the features already contain useful *localization information*. Indeed, with light convolutional layers, we are able to learn both good patch correlation and objectness information by using the DINO self-supervised model as a guide. With such information, our method CLIP-DINOiser performs zero-shot open-vocabulary semantic segmentation in a single pass of CLIP model and with two light extra convolutional layers. CLIP-DINOiser reaches state-of-the-art results on complex semantic segmentation datasets.

Limitations Despite yielding strong results on open-vocabulary semantic segmentation, CLIP-DINOiser is still bounded by the capability of the CLIP model to separate classes, as it inherits its granularity. We believe that better prompt engineering paired with better image-text models could further boost the performance of CLIP-DINOiser.

4.6. Additional material

4.6.1. The impact of the training dataset

Training stability. We report the average results of three different randomly sampled subsets of ImageNet used for the training. In the first row of Tab. 4.6.1, we report the corresponding standard deviation. We observe that in all cases, the standard deviation equals 0.1 mIoU or lower, therefore showing the stability of our training.

Training with different datasets. Our method CLIP-DINOiser does not require any labels to be trained. We investigate here the impact of training on the datasets used to train self-supervised DINO [15], and FOUND [100], namely ImageNet and DUTS-TR [155]. We report scores in Tab. 4.6.1. We also provide results when increasing the dataset size to 10k on ImageNet. In all cases, we observe no significant difference when using one dataset or another, and the size of the dataset does not seem to impact results positively.

Training dataset	C59	V20	Stuff	City	ADE
IN-1k	35.9 \pm 0.1	80.9 \pm 0.0	24.6 \pm 0.1	31.7 \pm 0.1	20.0 \pm 0.0
IN-10k	35.9 \pm 0.0	80.3 \pm 0.1	24.7 \pm 0.0	31.9 \pm 0.1	20.1 \pm 0.0
DUTS-TR [155]	35.9	80.5	24.6	31.3	19.9

(a) Benchmark **without** ‘background’ prompt

Training dataset	VOC	Con.	Obj
IN-1k	62.1 \pm 0.0	32.4 \pm 0.1	34.8 \pm 0.1
IN-10k	61.9 \pm 0.0	32.4 \pm 0.0	34.6 \pm 0.1
DUTS-TR [155]	62.0	32.4	34.8

(b) Benchmark **with** ‘background’ prompt

Table 4.6.1. Performance with different training datasets. When using random splits extracted from ImageNet (noted ‘IN’), we report the average score and **standard deviation** computed over training with three random splits (of 1k or 10k) extracted in ImageNet. In (a), we report the scores on the datasets without ‘background’ class, and in (b) with.

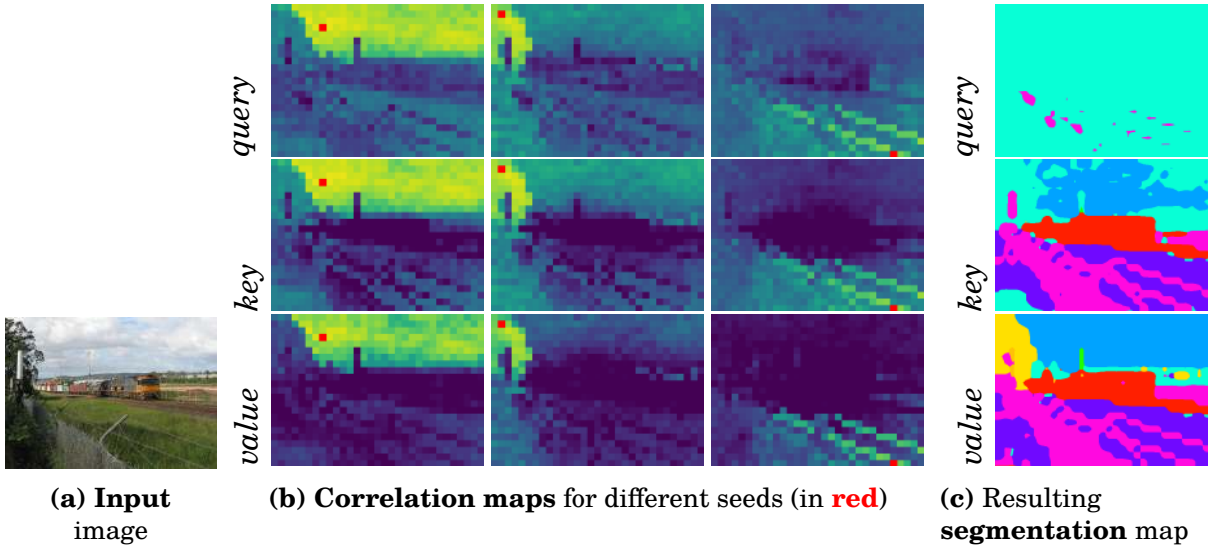


Figure 4.6.1. Visualization of correlation and segmentation obtained with different embeddings of DINO: *query*, *key* and *value*. The predicted prompts are: *sky*, *tree*, *train*, *ground*, *fence*, *grass*.

4.6.2. Self-supervised features discussion

We present in Fig. 4.6.1 visualizations of correlation obtained using different DINO embeddings extracted from DINO’s last attention layer, namely ‘query’, ‘key’ and ‘value’. Most unsupervised localization methods [95, 96, 100, 97] use the ‘key’ embeddings which allow the easy separation of *foreground* from *background*. However, we observed in this work that using instead the *value* features allows us to

Method	Single object discovery			Unsupervised saliency detection		
	VOC7	VOC12	C20k	DUT-O.	DUTS-T.	ECSSD
FOUND [100]	72.5	76.1	62.9	60.8	65.4	80.5
ours	73.1	75.9	64.4	60.6	66.6	81.3

Table 4.6.2. Results of single object discovery and unsupervised saliency detection obtained when following FOUND [100] protocol. We compute the single object discovery scores on classic VOC benchmarks [88] and 20k images of COCO (noted ‘C20k’) following [100] and use the CorLoc metric. We report the mIoU metric for unsupervised saliency detection and provide all results with the post-processing bilateral solver. We note ‘DUT-O.’ DUT-OMRON [156] and ‘DUTS-T.’ stands for DUTS-TEST [155].

separate better elements in the background, as visible in the figure. Patches in the background correlate to fewer background patches, and regions are, therefore, better separated.

We also depict the final segmentation when using each type of feature and observe the best result with ‘value’. We observe that more objects in the background are well-segmented and labelled, e.g., ‘tree’ and ‘sky’.

4.6.3. Background evaluation with FOUND

We now evaluate the quality of our background filtering using the class-agnostic foreground/background protocol defined in [100]. We report in Tab. 4.6.2 the scores on the task of unsupervised object discovery (on VOC07 [157], VOC12 [88] and COCO20k [90] datasets with CorLoc metric) and unsupervised saliency detection in the ‘multi’ setup of [100] (all results are provided when using post-processing bilateral solver on the classic DUT-OMRON [156], DUTS-TEST [155] and ECSSD [158] datasets, with the mIoU metric). For more details on the evaluation setup, we refer to [100]. On both tasks, we observe on par or even better results than [100], therefore showing the quality of our foreground predictions learnt from CLIP.

4.6.4. Details on the methods compared

We detail in Tab. 4.6.3 the additional datasets used by baseline methods to train their models if they use any. CLIP-DINOiser requires a significantly smaller amount of data for training than other methods; it needs only 1k images randomly sampled from ImageNet.

4.6.5. More qualitative results

In this section, we illustrate the benefits of our method through additional comparative qualitative results.

Methods	Extra data
MaskCLIP+ [28]	Target dataset
NamedMask [140]	ImageNet(1.2M)+Target dataset
ReCo [118]	ImageNet(1.2M)
GroupViT [94]	CC12M [159]+RedCaps [160]
ZeroSeg [148]	ImageNet(1.2M)+CC12M [159]
SegCLIP [112]	CC3M [161]+COCO(400k)
TCL [75]	CC12M [159]+CC3M [161]
CLIPpy [91]	HQITP-134M [91]
OVSegmentor [116]	CC4M [116]
CLIP-DINOiser	ImageNet (random 1k images)

Table 4.6.3. Details on additional data required for training.

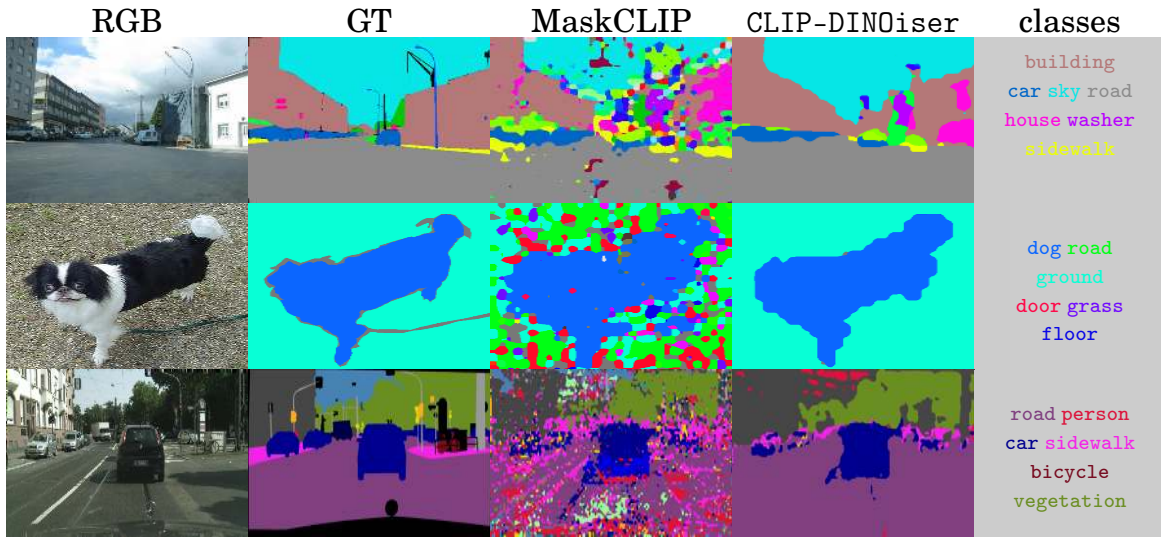


Figure 4.6.2. Visual ablations of the impact of our pooling method. Examples from ADE20K (top), PASCAL Context (middle), and Cityscapes (bottom) datasets.

Visual ablation of our spatial pooling. We show more examples of the application of our method CLIP-DINOiser and compare it to MaskCLIP results in Fig. 4.6.2. We observe that in all cases, our pooling reduces the noise in the predictions and helps produce good-quality segmentation.

Visual ablation of our background filtering. By visualizing more results with and without the background refinement step in Fig. 4.6.3, we observe that the background refinement step helps remove uncertain segmentation such as the snow area (which was classified as ‘snowboard’) in the top image, or on the cabinet, which is not annotated in VOC (right image).

In-the-wild examples We show more in-the-wild examples in Fig. 4.6.4, where we compare CLIP-DINOiser against MaskCLIP. MaskCLIP produces very noisy masks,

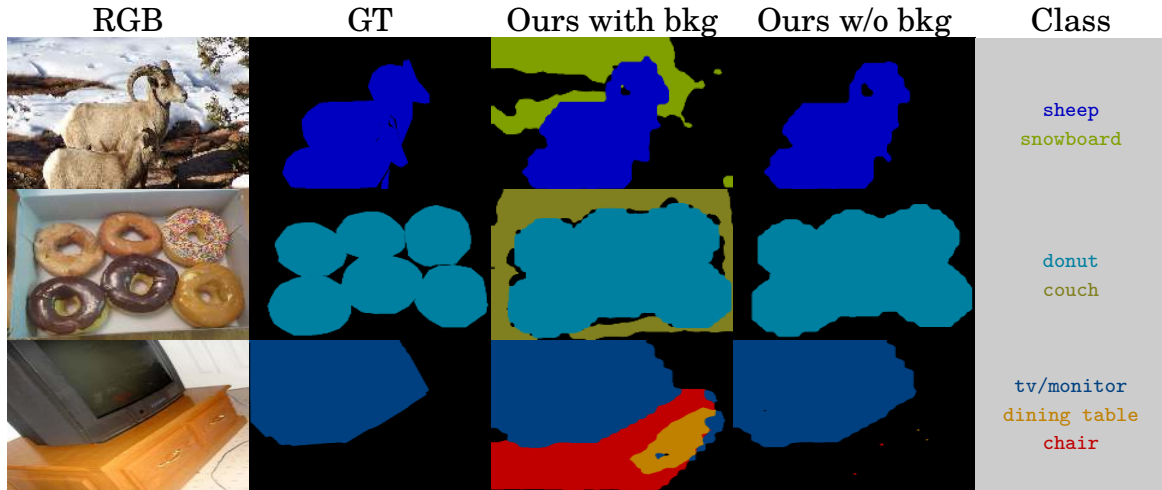


Figure 4.6.3. Visual ablations of the impact of background detection. We show examples from COCO Object (top, middle) and PASCAL VOC (bottom). We note with ‘bkg’ our background refinement.

especially when multiple *false positive* text queries are considered (we define such false positive queries as prompt queries that appear in the final segmentation but are not depicted in the image). Instead, CLIP-DINOiser eliminates such false positive predictions and produces less noisy segmentation.

4.6.6. Failure modes

We discuss here the known failure modes of our method CLIP-DINOiser and visualize some in Fig. 4.6.5. We first observe some of the biases of CLIP, which, for instance, produces similar features for ‘train’ and ‘train tracks’ (left image), likely due to their frequent co-occurrence across images. We have observed other instances of this bias, e.g., for ‘boat’ and ‘sea’ queries. Second, although CLIP-DINOiser can produce rather fine-grained segmentation (in terms of object sizes and classes), it can miss small or far-away objects as in Cityscapes (middle image). Finally, as with other open- vocabulary semantic segmentation methods, CLIP-DINOiser is not robust to the ambiguities of the text queries. The example from ADE20K (right image) is such a case, where ‘house’ is mistaken for ‘building’. In our experiments, we observed multiple segmentation ambiguities, and we believe that the redefinition of evaluation metrics could help address the issue. We stress that the current evaluation setup, which is taken directly from fully supervised settings, might be limiting in an open-vocabulary paradigm.



Figure 4.6.4. In-the-wild comparative examples between MaskCLIP (top) and CLIP-DINOiser (bottom). While MaskCLIP generates noisy masks when prompted with *false positive* classes our method is robust and produces cleaner masks.

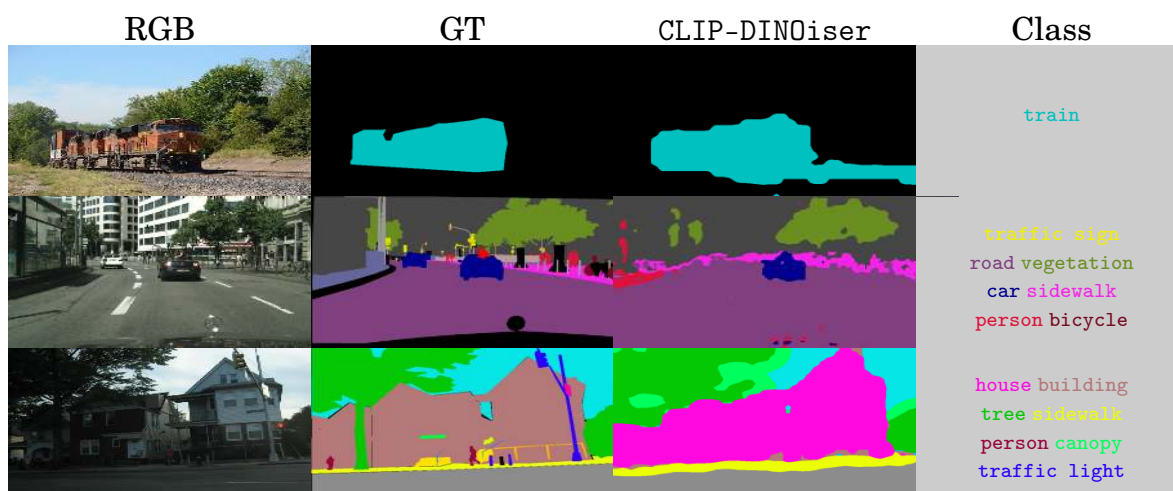


Figure 4.6.5. Failure cases of our method. From left to right: input RGB image, ground truth (GT) masks, masks predicted by CLIP-DINOiser, predicted text prompts. We discuss these failure cases in Sec. 4.6.6.

5. Leveraging Statistics from Pre-training Dataset

In previous chapters, we demonstrated how to adapt visual foundation models, in particular CLIP, to open-vocabulary semantic segmentation without requiring pixel-level annotations. In this chapter, we dive deeper into some of the intricacies of OVSS and open-vocabulary tasks in general. Our experiments with CLIP-DINOiser revealed the critical influence of text prompting on downstream performance (see Fig. 4.1.1). Our investigations showed that segmentation accuracy for a given concept can be substantially improved through the strategic selection of contrasting concepts — a finding that aligns with CLIP’s contrastive learning objective.

This chapter presents our contribution to investigating the role of contextual text prompts in OVSS. We begin by systematically analyzing the distribution of concepts within CLIP’s pre-training data. Building on these findings, we develop two automated approaches for generating effective contrasting concepts at inference time: one utilizing a large language model (LLM) and another exploiting statistical patterns from the VLM’s pre-training dataset. Moreover, our analysis reveals limitations in current OVSS evaluation benchmarks, leading us to propose an evaluation framework that better reflects real-world challenges. We find that understanding concept distribution in the pre-training dataset can enhance the performance of various CLIP-based semantic segmentation methods, from fully supervised to training-free approaches.

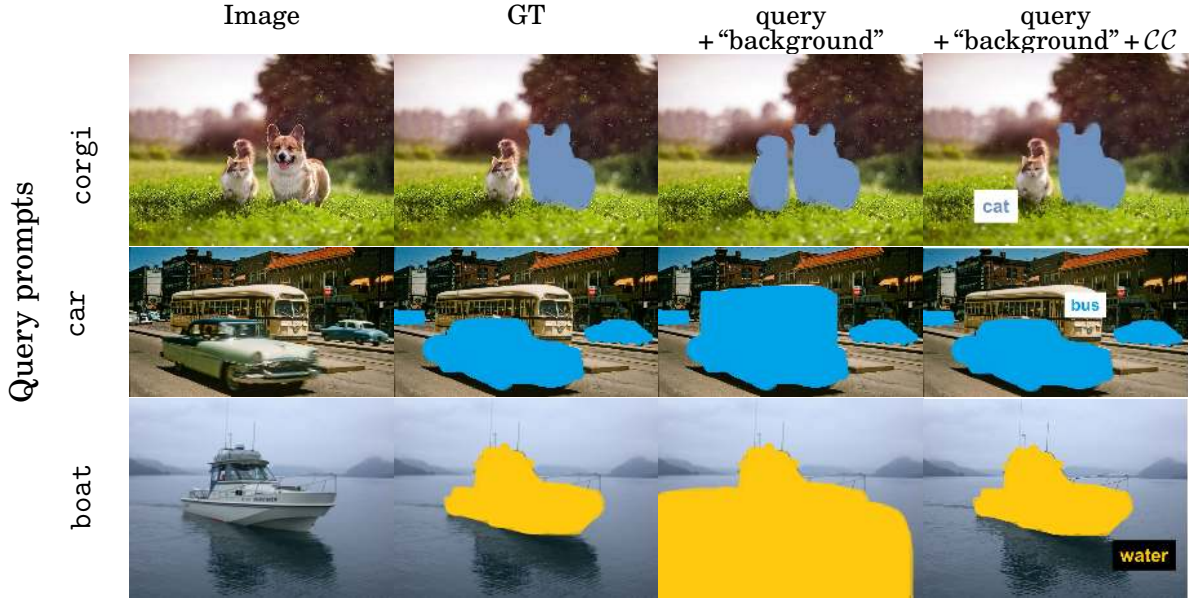


Figure 5.1.1. Illustration of our proposed open-world scenario and benefits of contrastive concepts (CC). We investigate open-world segmentation, where only one (or a few) visual concepts are to be segmented (2nd column), while all concepts that can occur in an image are unknown. Contrasting the query with “background” allows us to obtain a coarse segmentation [91, 30] (3rd column), but is not enough to catch all pixels *not* corresponding to the query when they are related or co-occur frequently in the VLM training set. Our automatically-generated *contrastive concepts* (CC) (4th column) help to separate and disentangle pixels of the query (right column, generated CC in text boxes), therefore achieving better segmentation.

5.1. Introduction

Vision-language models such as CLIP [13] are trained to align text and global image representations. Recently, VLMs have been proposed for denser tasks [28, 51, 145]. This includes the challenging pixel-level task of open-vocabulary semantic segmentation, which consists of segmenting arbitrary *visual concepts* in images, i.e., visual entities such as objects, stuff (e.g., grass), or visual phenomena (e.g., sky). To that end, several methods exploit a frozen CLIP model with additional operations [28, 162, 30, 29], or fine-tune the model with specific losses [94, 91, 75, 112, 92].

Most OVSS methods label each pixel with the most probable prompt (or query) among a finite set of prompts provided as input, contrasting concepts with each other. This works well for benchmarks that provide a large and nearly exhaustive list of things that can be found in the dataset images, such as ADE20K [144] or COCO-Stuff [142]. However, when given a limited list of queries, these methods are bound to occasionally suffer from hallucinations [30, 163]. In particular, common setups do not handle the case where only a single concept is queried [75, 94], which results in classifying all pixels using the same concept.

To catch such hallucinations, a common strategy consists of using an extra class labeled ‘background’, intended to capture pixels that do not correspond to any visual concept being queried. This extra class is already present in object-centric datasets, such as Pascal VOC [88]. It provides an easy, generic concept to be used as a negative query, i.e., to be used to contrast with actual (positive) queries but to be discarded from the final segmentation. However, the notion of background is not well defined as it is context-dependent, therefore providing suboptimal contrasts. This strategy also fails when a queried concept (e.g., “tree”) falls in the learned background (which commonly encompasses trees).

In this work, we consider a practical and realistic OVSS task in which only one or a few arbitrary concepts are to be segmented, leaving out the remaining pixels without any prior knowledge of other concepts that may occur in an image. We name this setup *open-world*¹ Given a query, instead of assuming access to a dataset-specific set of classes (a closed-world setup), we propose to automatically suggest contrastive concepts that are useful to better localize the queried concept, although they can later be ignored. In particular, we focus on predicting concepts likely to co-occur with the queried concept, e.g., “water” for the query “boat” (as visible in Fig. 5.1.1), thus leading to better segment boundaries when prompted together.

Moreover, we argue that this scenario needs to be evaluated to better understand the limitations of open-vocabulary segmentation methods. We therefore propose a new metric to measure such an ability, namely IoU-single, which considers one query prompt at a time and thus does not rely on the knowledge of potential domain classes.

To summarize, our contributions are as follows:

- We introduce the notion of test-time contrastive concepts and discuss the importance of contrastive concepts in open-vocabulary semantic segmentation.
- We analyze the usage of “background” as a test-time contrastive concept, which has been accepted but not discussed so far.
- We propose a new single-query evaluation setup for open-world semantic segmentation that does not rely on domain knowledge. We also propose a new metric to evaluate the grounding of visual concepts.
- We propose two different methods to generate test-time contrastive concepts automatically and show that our approaches consistently improve the results of 7 different popular OVSS methods or backbones.

¹ We distinguish our setup from open-world/ open-set setting known from literature [14], where a segmentation model identifies novel classes and marks them as “unknown”. Here, we consider the task of *open-world open-vocabulary* segmentation, thus considering only OVSS models, where the goal is to segment queried concepts unknown at test time and leave the remaining pixels in an image with no class. For the full name of our setup, we thus consider *open-world open-vocabulary segmentation* but keep *open-world* throughout the rest of the work for brevity.

5.2. Related work

Open-vocabulary semantic segmentation. VLMs trained on web-collected data to produce aligned image-text representations [13, 24, 57] had a major impact on open-vocabulary perception tasks and opened up new avenues for research and practical applications. While CLIP can be used *off-the-shelf* for image classification in different settings, it does not produce dense pixel-level features and predictions, due to its final global attentive-pooling [28, 132]. To mitigate this and produce dense image-text features, several methods finetune CLIP with dense supervision. Other approaches devise new CLIP-like models trained from scratch using a pooling compatible with segmentation. Their supervision comes from large datasets annotated with coarse captions [51, 91, 146, 94, 93, 116, 92, 75], object masks [113, 51, 115] or pixel labels [145, 146]. However, when models are finetuned, they face feature degradation [132], or require long training cycles on huge amounts of images when trained from scratch.

CLIP densification methods have emerged as a low-cost alternative to produce pixel-level image-text features while keeping CLIP frozen [28, 29, 132, 137, 30, 162]. The seminal MaskCLIP [28] mimics the global pooling layer of CLIP with a 1×1 conv layer. The aggregation of features from multiple views and crops [137, 138, 29, 132] also leads to dense features, yet with the additional cost of multiple forward passes. Some methods [118, 140, 141] rely on codebooks of visual prototypes per concept, including per-dataset negative prototypes [141], or leverage self-self attention to create groups of similar tokens [162]. The recent CLIP-DINOiser [30] improves MaskCLIP features with limited computational overhead thanks to a guided pooling strategy that leverages the correlation information from DINO features [15].

Prompt augmentation. Prompt engineering is a common practice for adapting Large Language Models (LLMs) to different language tasks [164] without updating parameters. This strategy of carefully selecting task-specific prompts also improves the performance of VLMs. For instance, in the original CLIP work [13], dataset-specific prompt templates, e.g., “a photo of the nice {...}” were devised towards improving zero-shot prediction performance. Although effective, manual prompting can be a laborious task, as templates must be adapted per dataset and sufficiently general to apply to all classes. Afterwards, different automated strategies were subsequently explored, e.g., scoring and ensembling predictions from multiple prompts [165]. Prompts can also be augmented by exploiting semantic relations between concepts defined in WordNet [166] to generate new coarse/fine-grained [167] or synonym [168] prompts. LLMs can be used as a knowledge base to produce

rich visual descriptions adapted for each class starting from simple class names [169, 170]. Prompt features can be learned by considering visual co-occurrences [171], a connection between training and test distributions [172], mining important features for the VLM [173] or by test-time tuning on a sample [174]. Most of these strategies have been designed and evaluated for the image classification task, and their generalization and scalability for semantic segmentation are not always trivial. Here, we aim to obtain better prompts for semantic segmentation to separate queried object pixels from their background. We do this automatically without supervision and without changing the parameters of either the text encoder or the image encoder, leveraging statistics from VLM training data or LLM-based knowledge.

Dealing with contrastive concepts in OVSS. Our contrastive concept discovery is tightly related to *background handling* in the context of open-vocabulary semantic segmentation, since the standard benchmark datasets for this task, originally designed for supervised learning, use *background* to describe unlabeled pixels, for example, to cover concepts outside of the dataset vocabulary. There are three main types of approaches to address this problem. The first one is to threshold uncertain predictions [75, 162, 94] with a given probability value [94, 162] or clip similarities [75]. The second group of methods leverages the object-centric nature of certain datasets by defining background through visual saliency [29, 30]. Finally, a significant body of work addresses the same issue by defining dataset-level concepts either by adding handcrafted names of concepts to the background definition [175, 176, 91, 53] or by extracting visual *negative prototypes* with a large diffusion model [141]. In contrast, in this work, we aim for automatic discovery of contrastive concepts without prior access to the vocabulary used to annotate the dataset.

Visual grounding is the task of localizing within images specific objects from text descriptions. The major instances of visual grounding tasks are *referring segmentation* that produce pixel-level predictions for one [177, 178, 179] or multiple target objects [180] given a text description, and *referring expression comprehension* [181, 182, 183, 184] that detects objects. Similarly to referring segmentation we aim to segment specific user-defined objects. In contrast, we do not use supervision to align textual descriptions with object masks and do not focus on text-described relations between objects and mine contrastive concepts to disentangle target objects from the background.

5.3. Open-world open-vocabulary segmentation with test-time contrastive concepts

We consider the following segmentation task: given an image and a set of textual queries characterizing different visual concepts, the goal is to label all pixels in the image corresponding to each concept, leaving out unrelated pixels, if any. Moreover, we want to do so without any prior knowledge of the concepts that could be prompted at the test time. We do not only want to be *open-vocabulary* in terms of the choice of words for querying, but we also want to be *open-world*, not specialized in a given domain or set of categories. For evaluation purposes, segmenting a specific dataset thus shall not assume anything about the dataset, such as knowledge of represented classes.

5.3.1. Introducing test-time contrastive concepts

Closed-world vs open-world open-vocabulary semantic segmentation. Even when it is open-vocabulary, traditional semantic segmentation is *closed-world* in the following sense. Given an RGB image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$ and a set of textual queries $q \in Q$, semantic segmentation produces a map $\mathbf{S}_{\text{closew}} : \{1 \dots H\} \times \{1 \dots W\} \mapsto Q$, where each image pixel has to be assigned one of the queries as a label. In contrast, *open-world* segmentation considers an additional dummy label ‘ \perp ’ to represent any visual concept that is different from the queries. The segmentation map, in this case, is then $\mathbf{S}_{\text{openw}} : \{1 \dots H\} \times \{1 \dots W\} \mapsto Q \cup \{\perp\}$. For instance, to label a boat, it is enough to ask for the “boat” segment; other pixels (sky, sea, sand, rocks, trees, swimmers, etc.) are expected to be labeled \perp and thus ignored.

In the following, we show how to use any open-vocabulary segmenter in an open-world fashion. We only assume that the segmenter uses a CLIP-like architecture with a text encoder, noted $\phi_T(\cdot)$, used to extract textual features $\phi_T(q) \in \mathbb{R}^d$ for any query q , where d is the feature dimension. Patch-level features $\phi_V(\mathbf{I}) \in \mathbb{R}^{h \times w \times d}$ are generated using the visual encoder, noted $\phi_V(\cdot)$, where $h = H/P$, $w = W/P$, and P is the patch size. The cosine similarities between each query feature and a patch feature are then used as logits when upsampling to obtain pixel-level predictions. It yields a closed-world segmentation, given our definition above.

From such segmentation, open-world segmentation could be derived by assigning a pixel (or patch) to a query if the cosine similarity between the visual and query embedding is above a given threshold. However, in practice, it has been commonly observed that the CLIP space is not easily separable [163], thus making the definition

of such a threshold difficult without overfitting the query or datasets [162, 75]. We further discuss the separability of CLIP patch features in Sec. 5.6.8.

Train-time contrastive concepts. Cues to separate visual concepts without supervision primarily come from data where these concepts occur separately and are described in their captions. If some concepts always co-occur, they are harder to be told apart. This applies in particular to OVSS models trained only from captioned images rather than from dense information. Sharing a caption pushes their embedding to align on a common textual feature, which in turn tends to bring the visual embeddings closer together. Still, such frequently co-occurring visual concepts can often be separated in a closed-world setting: pixels (or patches) are then just mapped to the query with which they align the most. However, a problem arises if a visual concept of a query q can be mistaken for another visual concept present in the image but not queried (e.g., querying “boat” but not “water” as in Fig. 5.1.1).

Test-time contrastive concepts. To address this problem, we propose to use one or more additional textual queries of visual concepts that are likely to contrast well with q . For example, when querying “boat”, we want to add the query “water”. We name such queries *test-time contrastive concepts* and note them \mathcal{CC}_q . We further propose different solutions to automatically generate \mathcal{CC}_q , and such without assuming prior access to the image domain. Given prompt queries $\{q\} \cup \mathcal{CC}_q$, we perform closed-world segmentation and assign to the dummy label \perp any patches that are labeled \mathcal{CC}_q .

Multi-query segmentation. This principle can be generalized to several simultaneous queries Q , with $|Q| > 1$, considering the union of their contrastive concepts $\mathcal{CC}_Q = \bigcup_{q \in Q} \mathcal{CC}_q$. Open-world multi-query segmentation consists in segmenting $Q \cup \mathcal{CC}_Q$, and ignoring pixels not assigned to the queries in Q , as in the single-query case. However, some queries in Q may already contrast with each other, which puts them in competition with the set of contrastive concepts \mathcal{CC}_Q and could lead to their elimination when pixels labeled in \mathcal{CC}_Q are discarded. To prevent it, we propose to exclude contrastive concepts \mathcal{CC}_Q that are too similar to queries Q , e.g., with a cosine similarity of text features above some threshold β : $\mathcal{CC}_Q = \bigcup_{q \in Q} \{q' \in \mathcal{CC}_q \mid \phi_T(q') \cdot \phi_T(q) \leq \beta\}$. In the following, for simplicity, we only consider the single-query scenario, where $|Q| = 1$.

Moreover, to our knowledge, none of the evaluation benchmarks currently used for OVSS allows us to measure the effectiveness of such \mathcal{CC} . We, therefore, propose a variant of the traditional evaluation metric for semantic segmentation and discuss it in detail in Sec. 5.4.1.

5.3.2. Contrasting with “background” (\mathcal{CC}^{BG})

In recent work [91, 29, 30], the word “background” has been used to try to capture a generic visual concept to help segment foreground objects, separating them from their background. In our framework, it amounts to defining “background” as a test-time contrastive concept to any query q . In other words, it defines $\mathcal{CC}_q^{BG} = \{\text{“background”}\}$.

However, if the word “background” feels natural to us, it is not obvious why it should also make sense in the CLIP space. This formulation is not contextual, meaning that the contrastive concept is not specific to the query, which might be suboptimal. Worse, the “background” samples from which CLIP learned could accidentally include the visual concept of the query, making the query representation close to the background representation and defeating the contrast mechanism.

We investigate the occurrence of “background” in VLM training data to sort it out. First, we use the metadata provided by [185], which describes the representation of four thousand common concepts in LAION-400M [186], which is a subset of the web-crawled LAION-2B dataset [126] used to train CLIP. In Fig. 5.3.1a, we plot the frequency of occurrence of “background” among other VOC class names. We observe that “background” is significantly more frequent than all other words, hinting that it is widely available in CLIP training data and in general web-crawled data.



Figure 5.3.1. Statistics about “background” in metadata of web-crawled datasets. (a) Frequency of some of the concepts from VOC dataset in LAION-400M caption samples. Examples of images in web-crawled data with a caption including the words “background” (b) or “in the background” (c).

Fig. 5.3.1b shows images sampled from the LAION dataset with a caption containing “background”. We observe that they display a high diversity in colours and textures. Images captioned with “in the background” (Fig. 5.3.1c) appear more photo-oriented. We believe that the combination of a high frequency of the “back-

ground” word in the dataset and the diversity of associated images make it a good generic contrastive concept and hence make CC^{BG} a baseline. However, superior results have been obtained by applying well-designed tricks to handle the background [29, 30, 75, 162], emphasizing the necessity of applying something more than simply “background”.

An option is to define a generic background class list, as done by CLIPpy [91] or CAT-Seg [53], which adds to the concept “background” a fixed list of concepts potentially appearing in the background, e.g. “sky”, “forest”, “building”, to be discarded. First, since these visual concepts are intended to be discarded, it would not be possible to query them. Second, such a list is defined at the dataset level, making it domain-specific. As it is impossible to exhaustively describe all visual concepts appearing in any “background” (without prior knowledge of the domain or dataset), we propose generating such complements specifically per query, as discussed below.

5.3.3. Automatic contrastive concepts (CC) generation

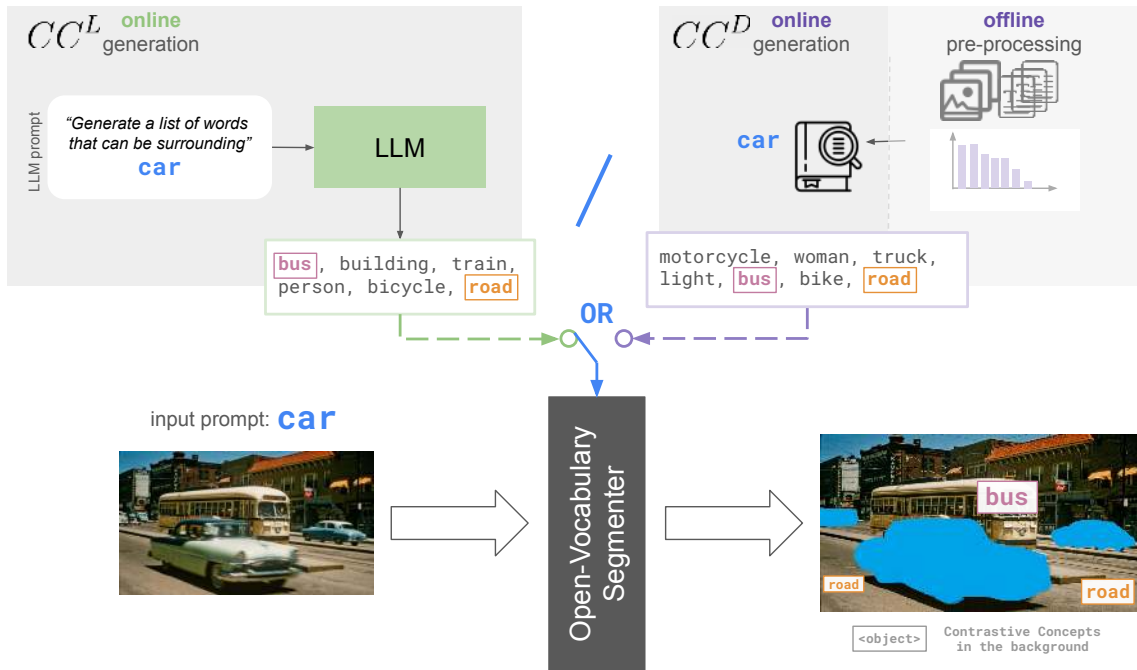


Figure 5.3.2. Overview of our method. We propose two solutions to generate CC automatically, the first one (*top-left*) based on LLM prompting (CC^L) and the second one, CC^D that relies on the distribution of co-occurring concepts in a pre-training dataset of a VLM (*top-right*). Both methods can be effectively integrated into various open-vocabulary segmentation methods.

To generate contrastive concepts that are query-specific but also domain-agnostic, the only data we can then leverage are (i) the VLM’s training data, or (ii) unspecific

external data. As we focus on text-based contrasts, we can (i) exploit the large vocabulary of concepts used for VLM training or (ii) generate prompts via an LLM. Finally, as we want good contrasts, we must find hard negatives. These are concepts that surround queries in images. To gather them, we can (i) look for word co-occurrences in training data or (ii) ask an LLM to list such concepts. Sec. 5.3.3 investigates option (i), and Sec. 5.3.3, option (ii) and Fig. 5.3.2 presents a high-level overview of both approaches.

Mining co-occurrence-based contrastive concepts (\mathcal{CC}^D)

As discussed above, ambiguity in segmentation for unsupervised approaches arises from co-occurrences in training data. Yet, OVSS does better when prompted to create segments simultaneously for co-occurring concepts. To list contrastive concepts specific to a given query q , we propose thus to use the information of *co-occurrence* in the VLM training captions. For efficiency, we construct *offline* a *co-occurrence dictionary*, built for a large lexicon of textual concepts extracted from the captions. We note \mathcal{CC}_q^D the co-occurrence-based contrastive concepts we extract for a query q based on this lexicon.

Co-occurrence extraction. We consider as lexicon a set of textual concepts \mathcal{T} extracted from captions of the VLM training dataset and construct the co-occurrence matrix $X \in \mathbb{N}^{|\mathcal{T}| \times |\mathcal{T}|}$. Concretely, two concepts $\{i, j\} \subset \mathcal{T}$ co-occur if they appear simultaneously in the caption of an image. $X_{i,j}$ counts the number of times concepts $\{i, j\}$ co-occur in some images. Next, we normalize the symmetric matrix X row-wise by the number of occurrences of concept i in the dataset, producing the frequency matrix \hat{X} . We then consider only concepts with frequent co-occurrences: for each $i \in \mathcal{T}$, we select concepts $\mathcal{T}_i = \{j \in \mathcal{T} \mid \hat{X}_{i,j} > \gamma\}$, for some frequency threshold γ . Selecting only a few contrastive concepts in this way is also consistent with the fact that we target online segmentation: we need to be mindful of computational costs.

Concept filtering. To improve the quality of selected contrastive concepts \mathcal{T}_i , we design a simple filtering pipeline. For each target concept $i \in \mathcal{T}$ (which can be considered a future query), we remove from \mathcal{T}_i any concept that might interfere with i and induce false negatives. First, we discard uninformative words in captions: {"image", "photo", "picture", "view"}. Then, we remove *abstract* concepts, such as "liberty". To do so, we ask an LLM whether a given word can be visible or not in an image (more details in Sec. 5.6.11). We also filter out concepts that are too semantically similar to target concept i , e.g., such that their cosine similarity with $\phi_T(i)$ is more than a threshold δ . We also consider an alternative approach to filtering, which uses the structured ontology WordNet [166] to remove the \mathcal{CC} s that possibly interfere

with q . However, our experiments, which are discussed in Sec. 5.6.10, show that our proposed filtering mechanisms based on dataset statistics are more effective.

Generalization to arbitrary concepts. So far, we discussed how to select contrastive concepts \mathcal{CC}_i^D for a target concept $i \in \mathcal{T}$. Now, when we are given an arbitrary textual query q , to make the generation of contrastive concepts truly open-vocabulary, we first find in the CLIP space the nearest neighbour i of q in \mathcal{T} and then use for q the contrastive concepts of i : $\mathcal{CC}_q^D = \mathcal{CC}_i^D$.

Prompting an LLM to generate contrastive concepts (\mathcal{CC}^L)

Instead of extracting contrastive concepts from the VLM training set, here we investigate another strategy, generating them using an LLM. For a given text query q , we ask an LLM to directly generate contrastive concepts \mathcal{CC}_q^L , without the need for subsequent filtering. To that end, we design a prompt that excludes potential synonyms, meronyms (e.g., “wing” for “plane”), or possible contents (e.g., “wine” for “bottle”). We present a shorter version of the prompt in Fig. 5.3.3 and include the complete version in Sec. 5.6.11.

You are a helpful AI assistant with visual abilities. Given an input object \mathbf{O} , I want you to generate a list of words related to objects that can be surrounding input object \mathbf{O} in an image to help me perform semantic segmentation.

Figure 5.3.3. An abbreviated version of the prompt we use to generate \mathcal{CC}^L .

Using an LLM has the benefit of producing specific contrastive concepts \mathcal{CC}_q for any target query q , without returning to a fixed and practically limited lexicon.

5.4. Evaluation

5.4.1. Evaluating open-world segmentation

We discuss here our evaluation protocols and present our new metric IoU-single specifically designed to evaluate open-world segmentation.

Evaluation datasets. We conduct our experiments on six datasets widely used for the task of zero-shot semantic segmentation [75], fully-annotated COCO-Stuff [142], Cityscapes [143] and ADE20K [144] and object-centric VOC [88], COCO-Object [90] and Context [152], when considering “background” pixels. We treat the input images following the protocol of [75], which we detail in Sec. 5.6.1.

Our IoU-single metric. To better evaluate the ability of a method to localize a visual concept when given *no other information*, we propose the IoU-single metric.

It modifies the classic IoU by considering each concept independently and then averaging. Concretely, we individually segment each class annotated in the dataset for the considered image, thus with $|Q|=1$. The IoU-single is then the average of each IoU with the corresponding ground-truth class segment. We illustrate this metric in Fig. 5.4.1, and provide its pseudo-code in Sec. 5.6.2. If a dataset contains a *background* class, we do not consider it in the mIoU calculation.

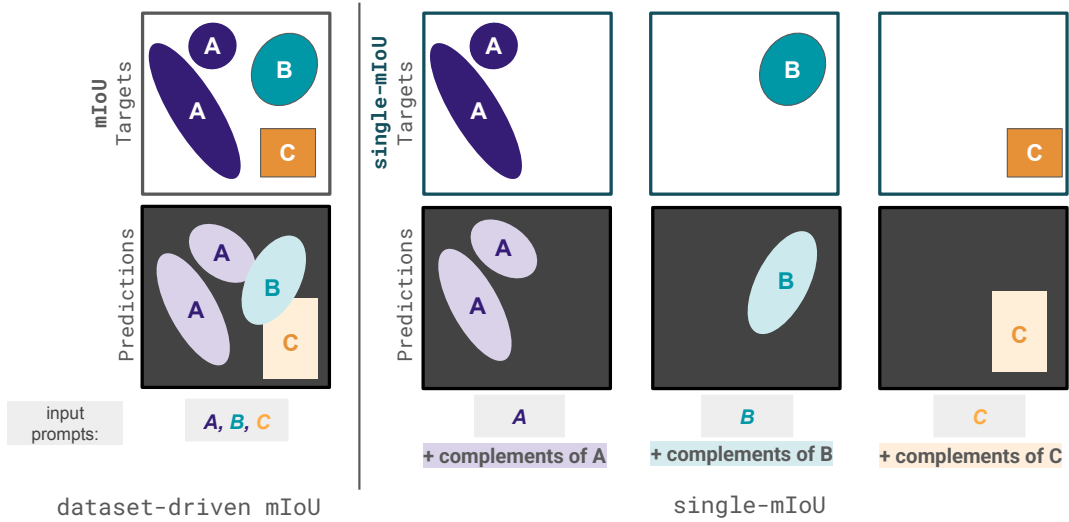


Figure 5.4.1. Illustration of IoU-single metric. We show the difference with the standard mIoU metric (dataset-driven mIoU), where all the concepts present on an image are considered at once. On the contrary, our IoU-single considers each of the present concepts separately to measure the single-class segmentation ability of open-vocabulary semantic segmenters.

Classic mIoU evaluation. We also evaluate the impact of using our \mathcal{CC} in the classic mIoU scenario on the datasets that consider “background” as a class, i.e., VOC and COCO-Object. We prompt at once all dataset classes together with their \mathcal{CC} s, using our multiple-query strategy discussed in 5.3.1. We then assign pixels that fall into any of the \mathcal{CC} s to “background”, ensuring that none of the concepts competes with the dataset queries. It allows us to verify if our \mathcal{CC} s can act as background without hurting the performance on foreground classes.

5.4.2. Evaluated methods

Test-time contrastive concepts. For \mathcal{CC}^D generation, we use the statistics gathered by [185] for four thousand common concepts in the LAION-400M dataset, which is a subset of LAION-2B [126] and which is used to train CLIP [54]. We filter contrastive concepts using a low co-occurrence threshold $\gamma = 0.01$ and a high CLIP

similarity threshold $\delta = 0.8$. In the classic mIoU scenario, we use a threshold $\beta = 0.9$ to account for possible similarities between one query and contrastive concepts close to the other queries. We discuss the selection of these values in Sec. 5.6.6. To generate \mathcal{CC}^L , we use the recent Mixtral-8x7B-Instruct model [187]. More details about the setup can be found in Sec. 5.6.11 alongside our designed prompts in Sec. 5.6.11. In our experiments, unless stated otherwise, we include “background” in all \mathcal{CC} ’s: $\mathcal{CC}^D \leftarrow \{\text{“background”}\} \cup \mathcal{CC}^D$ and $\mathcal{CC}^L \leftarrow \{\text{“background”}\} \cup \mathcal{CC}^L$.

Baselines. To evaluate the impact of using contrastive concepts, we experiment on 6 popular or state-of-the-art methods, one of which (MaskCLIP) uses 3 different backbones, thus resulting in 7 different segmenters, which we believe represent the current OVSS landscape. Concretely, we study two training-free methods that directly exploit the CLIP backbone, namely MaskCLIP [28] and GEM [162], where MaskCLIP may exploit different CLIP backbones [54] pre-trained either on LAION [126], MetaCLIP [55], or by default on the original OpenAI training data [13]. We also include TCL [75], CLIP-DINOiser [30] and supervised methods: CAT-Seg [53] and SAN [188]. Details on the evaluation protocol, including background handling strategies, can be found in Sec. 5.6.1. All compared methods use CLIP ViT-B/16.

5.4.3. Contrastive concepts generation results

We first present in Tab. 5.4.1 results obtained with our IoU-single metric on 3 datasets, namely ADE20K, Cityscapes and VOC. We compare results when using different \mathcal{CC} ’s proposed in this work. We also include results when having access to privileged information (\mathcal{CC}^{PI}), i.e., the list of concepts present in images as given by the evaluation dataset. More results can be found in Tab. 5.6.2.

Method	CLIP training data	VOC			Cityscapes				ADE20k			
		\mathcal{CC}^{BG}	\mathcal{CC}^L	\mathcal{CC}^D	\mathcal{CC}^{BG}	\mathcal{CC}^L	\mathcal{CC}^D	\mathcal{CC}^{PI}	\mathcal{CC}^{BG}	\mathcal{CC}^L	\mathcal{CC}^D	\mathcal{CC}^{PI}
MaskCLIP	OpenAI	44.2	52.2	53.4	15.0	22.5	22.0	30.6	20.2	23.5	25.2	29.8
DINOiser	LAION-2B	59.3	63.1	64.7	23.2	30.6	27.3	36.0	28.9	29.7	31.6	35.5
TCL	TCL’s	52.9*	52.6*	53.6*	9.8	26.3	22.0	29.7	14.9*	25.9	26.5	32.6
GEM	MetaCLIP	48.6*	61.3*	64.6*	14.5*	21.5	14.6	20.6	21.5*	26.3	29.1	33.0
SAN	OpenAI	50.2	73.4	69.5	19.9	37.6	32.0	44.2	24.5	35.2	35.1	42.8
CAT-Seg	OpenAI	52.8	69.5	67.7	—	—	—	—	25.7	38.4	39.7	46.8

Table 5.4.1. Benefits of \mathcal{CC} measured in IoU-single. “*” indicates that the method’s original background handling is applied, if any and provided it gives the best results. Note that CAT-Seg input resolution is 640x640, whereas it is 448x448 for all the other methods. We note \mathcal{CC}^{PI} the unrealistic setup where we have access to all of the dataset classes and use them as systematic contrastive concepts (except for VOC, as its annotations do not cover all pixels). Please note that \mathcal{CC}^{BG} is our baseline.

“Background” is not enough. We start by analyzing the overall impact of our proposed CC s. In all cases, we observe a significant improvement when using contrastive concepts CC^D and CC^L compared to the CC^{BG} . Even for object-centric VOC where CC^{BG} already provides a strong baseline, our proposed CC generation methods bring significant gains ranging from 0.7 to 16.7 points. Interestingly, test-time CC s also work well for supervised CAT-Seg, showing that our method is beneficial for open-vocabulary segmenters with all levels of supervision.

CC^L generalize better to domain-specific datasets. For both VOC and ADE20K, the co-occurrence-based CC^D outperforms most of the time the LLM-based CC^L , with a margin ranging from 0.6 to 2.8 points. However, this trend does not hold for Cityscapes, where CC^L gives the best results for all methods. In particular, Cityscapes is a dataset of urban driving scenes that contains images depicting a few recurring concepts. This may suggest that LLMs can produce better results than CC^D for such domain-specific tasks. We also note that CC^L generally produces fewer CC s, but we do not observe a correlation between segmentation performance and $|CC|$, as shown in Sec. 5.6.7.

Test-time concepts are different from train-time concepts. We also observe that CC^{PI} results overall do not exceed 50% mIoU. The segmentation quality might thus be limited by the VLM capacity or by a mismatch between the dataset classes and the training data. Well-designed prompt engineering could help address this issue [189] and improve segmentation results.

Classic mIoU evaluation. Additionally, in Tab. 5.4.2, we present results with the standard mIoU for MaskCLIP (with LAION-2B backbone) and GEM. We report results with various contrastive concepts (CC) and the original background handling strategy when applicable. We observe that in all cases, the results with CC^D and CC^L are better than baseline CC^{BG} . We also notice that for GEM the results are better than when applying the background handling strategy originally proposed in [162]. This shows that integrating our contrastive concepts does not hurt or can even improve performance in the classic mIoU setup. We provide more results in Tab. 5.6.1.

Method	Bkg.	Object VOC	
MaskCLIP	CC^{BG}	17.8	35.1
	CC^L	25.9	46.2
	CC^D	25.1	46.4
GEM	threshold	27.4	46.6
	CC^L	35.7	60.0
	CC^D	35.5	60.5

Table 5.4.2. Results w/ mIoU.

<i>co-occ.</i>	<i>no abs.</i>	<i>sem. sim.</i>	Mask CLIP	TCL	DINO iser	Method	Cityscapes		ADE20k		MaskCLIP w/ CLIP training set	VOC		
							w/o	w/	w/o	w/		$\mathcal{C}\mathcal{C}^{BG}$	$\mathcal{C}\mathcal{C}^L$	$\mathcal{C}\mathcal{C}^D$
✓			20.2	22.4	23.9	MaskCLIP	22.3	22.5	22.5	23.5				
✓	✓		20.9	23.2	25.5	DINOiser	30.3	30.6	27.5	29.7				
	✓	✓	18.4	20.0	26.3	TCL	26.0	26.2	25.4	26.3	LAION-2B	47.9	51.8	53.8
✓	✓	✓	25.2	26.0	31.6	GEM	21.3	21.4	25.7	26.1	OpenAI	44.2	52.2	53.4
											MetaCLIP	46.8	50.6	50.0

(a) **Impact of filtering in $\mathcal{C}\mathcal{C}^D$** on ADE20K (%IoU-single).

(b) **Adding “background” or not** to our LLM-based $\mathcal{C}\mathcal{C}^L$.

(c) **Impact of pre-training dataset** on VOC (%IoU-single).

Table 5.4.3. Ablation studies. (a) The impact of filtering steps: ‘co-occ.’ is the co-occurrence-based filtering; ‘no abs.’ is the removal of abstract concepts; ‘sem. sim.’ is the semantic-similarity filtering. (b) Relevance of adding “background” to $\mathcal{C}\mathcal{C}^L$. (c) Varying the pre-training dataset.

5.4.4. Ablation studies

$\mathcal{C}\mathcal{C}^D$ concept filtering. In Tab. 5.4.3a, we analyze the impact of the different filtering steps discussed in Sec. 5.3.3 on the challenging ADE20K dataset. We observe that each step boosts results by removing noisy or detrimental concepts. The largest gain is obtained when filtering highly similar (‘sem. sim.’) concepts. We also note that the improvement is consistent for all methods. We report the performance without the co-occurrence thresholding (w/o ‘co-occ.’) and observe a significant degradation. More experiments in Sec. 5.6.10 suggest that ontology-based filtering (e.g., using WordNet) does not help and can even be harmful.

Adding “background” to $\mathcal{C}\mathcal{C}^L$. In Tab. 5.4.3b, we study the influence of adding the word “background” to the set of contrastive concepts $\mathcal{C}\mathcal{C}^L$ generated with the LLM. We observe that it is always beneficial, in most cases with little gain, except on ADE20k, where the gain is up to 2.2 IoU-single pts.

Impact of the pre-training dataset. Tab. 5.4.3c shows the results of MaskCLIP with different datasets used to train CLIP. We observe that using $\mathcal{C}\mathcal{C}^D$ always gives a boost over using “background” alone ($\mathcal{C}\mathcal{C}^{BG}$) across all pre-training datasets, including on the highly-curated MetaCLIP. However, we notice that for MetaCLIP, $\mathcal{C}\mathcal{C}^L$ gives even better results, suggesting that leveraging LLMs can also be more profitable with backbones pre-trained on carefully curated datasets.

5.4.5. Qualitative results

In Fig. 5.4.2, we present qualitative examples when using different contrastive concepts proposed in this work. We compare $\mathcal{C}\mathcal{C}^L$ and $\mathcal{C}\mathcal{C}^D$ with ground truth (GT) and baseline $\mathcal{C}\mathcal{C}^{BG}$. For both $\mathcal{C}\mathcal{C}^L$ and $\mathcal{C}\mathcal{C}^D$, we present the output segmentation mask

for the queried concept together with its contrastive concepts (noted *all*) as well as the single queried concept (noted *single*), where $\mathcal{C}\mathcal{C}$ s are discarded. We observe that the output masks produced by our methods are more accurate, removing the noise from related concepts, e.g. “tree” for the bird or “sofa” for the “bed”.

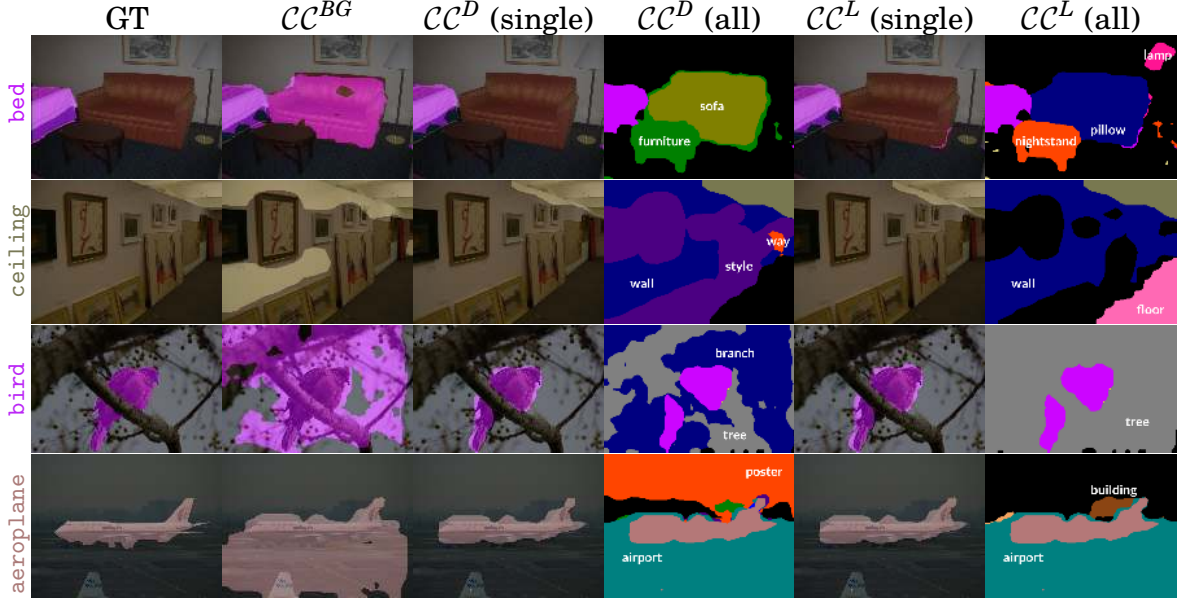


Figure 5.4.2. Qualitative results. We show segmentation examples from ADE20K (1st and 2nd row) and Context (3rd and 4th row), with segments produced by CLIP-DINOiser. For $\mathcal{C}\mathcal{C}^D$ and $\mathcal{C}\mathcal{C}^L$, we additionally show the joint segmentation of all contrastive classes (all).

Generalization to arbitrary concepts. Fig. 5.4.3 presents results when prompting queries that are not included in the subset of concepts \mathcal{T} extracted from the VLM training dataset, such as “muffin” or “cavalier” (a dog breed). We show the closest neighbour for the query q below each example and visualize masks for both MaskCLIP and CLIP-DINOiser. We observe that the $\mathcal{C}\mathcal{C}^D$ generation method leveraging statistics from pre-training datasets is also robust to examples outside of the co-occurrence dictionary by accurately mapping q to its closest concept in \mathcal{T} , e.g., mapping “cavalier” to “dog”.

5.5. Conclusion

In this work, we identify limitations of the current evaluation setup for open-vocabulary semantic segmentation tasks, which are inherited from closed-world evaluation benchmarks. To bridge the gap between closed- and open-world setups, we propose the single-class segmentation scenario. We study the limitations of current state-of-the-art models when we assume no prior access to in-domain classes and propose to automatically discover contrastive concepts $\mathcal{C}\mathcal{C}$ that are useful to better

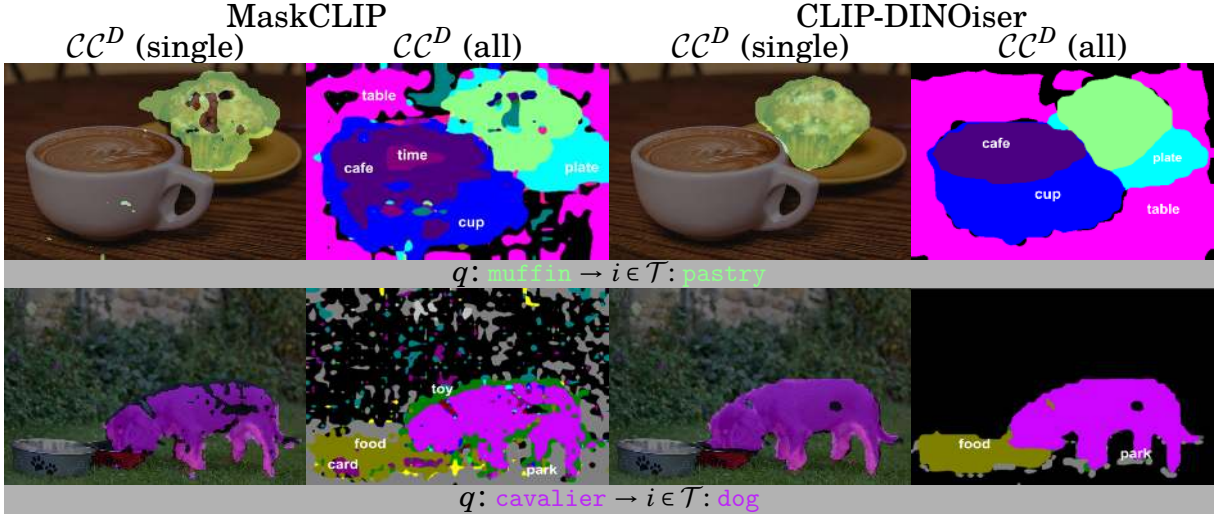


Figure 5.4.3. In the wild examples. We visualize results for MaskCLIP and CLIP-DINOiser for query concepts beyond \mathcal{T} . The closest neighbour to a query is presented below each example (grey row).

localize any queried concept. To do so, we propose two methods leveraging either the distribution of co-occurrences in the VLM’s training set or an LLM to generate such CC . Our results show the generalizability of our proposed method across several setups.

5.6. Additional material

5.6.1. Details on the evaluation

Evaluation protocol. Our experiments follow the evaluation protocol of [75]. We use MMSegmentation implementation [153] with a sliding window strategy and resize input images to have a shorter side of 448. In the case of CAT-Seg, we retain the original model framework and integrate IoU-single into Detectron [190]. We also use its evaluation protocol, meaning that the input images differ from other evaluated methods, i.e., with an input image size of 640x640. Regarding the text prompts, we keep the native prompting of each method to stay as close as possible to the methods.

Background handling of baselines. We detail here the different strategies employed in the methods that we evaluate to handle the background.

TCL [75] applies thresholding and considers pixels with maximal logit ≤ 0.5 to be in the background, where the logits are the cosine similarities of the visual embedding with the embedding of queries.

GEM [162] applies a background handling strategy only for Pascal VOC. It only predicts the foreground classes. The background is obtained by thresholding the softmax-normalized similarity between the patch tokens and the text embedding of each class name. The threshold is fixed (set to 0.85). In our experiments with VOC, we explore the performance of GEM both with and without background handling and report each time a better score. For other datasets than VOC, we apply only our methods.

MaskCLIP [28] does not use any dedicated mechanism for background. Therefore, we do not report the original setup for it.

CLIP-DINOiser [30] leverages a foreground/background saliency strategy which focuses on foreground pixels. In that case, the foreground/background is defined following FOUND [100], which focuses on objectness and mainly discards pixels corresponding to stuff-like classes, which might also be of interest.

CAT-Seg [53] does not apply any background handling strategy. Instead, for VOC they create a list of potential background classes and use them as "dummy" classes. This approach is closest to what we propose. In practice, for the VOC dataset, the authors use class names from the Context dataset, an extension of VOC with +40 class names.

SAN [188] does not design any background handling strategy and does not evaluate datasets with "background" class.

5.6.2. About the IoU-single metric

We present a pseudo-code of our metric in Algorithm 1.

5.6.3. More quantitative results

State-of-the-art results under classic mIoU. In Tab. 5.6.1, we report the results under the classic mIoU metric for selected state-of-the-art methods on open-vocabulary semantic segmentation. For each of the methods, we detail the specific background handling techniques (if any), the CLIP backbone used as well as additional datasets used for training.

Extending the dataset vocabulary with our generated contrastive concepts does not hurt the overall performance under a normal setup when all dataset labels are considered prompts. For GEM and MaskCLIP we observe significant improvements over their original setups on VOC. This holds for both contrastive concept generation methods \mathcal{CC}^D and \mathcal{CC}^L . Looking at the results of CLIP-DINOiser, we observe that saliency is still more effective in the object-centric scenario.

Algorithm 1: IoU-single

Results: mean IoU-single– a mIoU score for a single-query scenario for a given image

Inputs : I – input image: $I \in \mathbb{R}^{H \times W \times 3}$

Y – ground-truth annotations of I : $gt \in \mathbb{N}^{H \times W \times 1}$

T – ground-truth text labels

CC – a dictionary of contrastive concepts per query

model – segmenter producing pixel-level predictions given text queries

procedure IoUsingle(I, Y):

 // Get unique classes from Y

$gt_{cls} \leftarrow \text{unique}(Y)$

 scores $\leftarrow \emptyset$

for $i \in gt_{cls}$ **do**

$q \leftarrow T_i$

 // Text prompts include query q and contrastive concepts of q

$t_q \leftarrow q \cup CC_q$

 // Get model predictions for given prompt set

$\hat{y} \leftarrow \text{model}(I, t_q)$

 // Get binarized version of predicted mask

$\hat{y} \leftarrow \text{binarize}(\hat{y}, i)$

 // Get ground-truth binary mask for gt class i

$y \leftarrow \text{binarize}(Y, i)$

 // Record corresponding IoU

 scores $\leftarrow \text{scores} \cup \text{IoU}(\hat{y}, y)$

end for

return mean(scores)

More open-world evaluation results. Tab. 5.6.2 extends Tab. 5.4.1 and completes the results obtained with the IoU-single on all the datasets that we considered.

5.6.4. Failure case analysis

We present some failure cases of our approach in Fig. 5.6.1. Precisely, we show examples of CLIP-DINOiser when one of the generation methods fails. The first example (first row), CC^L suggests “blanket” for “bed”, which typically covers the query concept. One of the potential improvements would be to instruct an LLM to ignore potentially occluding objects. In the second row, both methods fail to provide “floor” to contrast with “rug”. We notice that CC^L tend to be more oriented towards objects, as opposed to stuff-like classes. We also observe that in the example, a small part of a painting on the wall is segmented as “rug”. This suggests that CC s might not give a complete set of. Finally, in the third example, both methods fail to generate “person” to contrast with “bedclothes”. However, CC^L includes “pyjamas”, which results in a better segmentation overall. Image-conditioned generation (e.g., with VLMs) could be a candidate solution to this problem, but we leave it for future work.

Methods	Background handling	Type of \mathcal{CC}	CLIP backbone	Training dataset	Dataset		
					Context	Object	VOC
GroupViT	threshold	\emptyset	scratch	CC12M+RedCaps	18.7	27.5	50.4
CLIP-DIY	saliency	\emptyset	LAION-2B	-	19.7	31.0	59.9
TCL	threshold	\emptyset	OpenAI	CC12M+CC3M	24.3	30.4	51.2
MaskCLIP [†]	\emptyset	\emptyset	OpenAI	-	21.1	15.5	29.3
MaskCLIP*	\emptyset	\emptyset	LAION-2B	-	22.9	16.4	32.9
MaskCLIP* (+keys)	\emptyset	\emptyset	LAION-2B	-	24.0	21.6	41.3
CLIP-DINOiser	\emptyset	\emptyset	LAION-2B	ImageNet (1k im.)	32.4	29.9	53.7
GEM	\emptyset	\emptyset	MetaCLIP	-	-	-	46.8
CLIP-DINOiser	saliency	\emptyset	LAION-2B	ImageNet (1k im.)	—	34.8	62.1
	\mathcal{CC}	\mathcal{CC}^{BG}	LAION-2B	ImageNet (1k im.)	32.4	29.5	54.0
	\mathcal{CC}	\mathcal{CC}^L	LAION-2B	ImageNet (1k im.)	31.3	35.0	60.8
	\mathcal{CC}	\mathcal{CC}^D	LAION-2B	ImageNet (1k im.)	31.8	33.3	60.4
MaskCLIP	\mathcal{CC}	\mathcal{CC}^{BG}	LAION-2B	-	23.6	17.8	35.1
	\mathcal{CC}	\mathcal{CC}^L	LAION-2B	-	22.5	25.9	46.2
	\mathcal{CC}	\mathcal{CC}^D	LAION-2B	-	23.2	25.1	46.4
GEM	threshold	\emptyset	MetaCLIP	-	33.4*	27.4*	46.6*
GEM	\mathcal{CC}	\mathcal{CC}^L	MetaCLIP	-	31.6	35.7	60.0
GEM	\mathcal{CC}	\mathcal{CC}^D	MetaCLIP	-	32.1	35.5	60.5

Table 5.6.1. Results with standard mIoU metric when employing different contrastive concept generation strategies. ‘*’ denotes our implementation, ‘†’ denotes results from TCL [75], and ‘MaskCLIP (+keys)’ denotes keys refinement proposed in the original paper [28]. Training datasets include CC12M [159], RedCaps [160], ImageNet [6], CC3M [161].

5.6.5. More qualitative results

More qualitative results are provided in Fig. 5.6.2, comparing \mathcal{CC}^D to \mathcal{CC}^L .

5.6.6. Hyperparameter selection

In this section, we discuss the selection of hyperparameters for our \mathcal{CC} generation. For the frequency threshold γ and the cosine similarity threshold δ , we randomly select 100 images from the training set of the ADE20K dataset and report IoU-single on this subset — which we observed was enough to select the values. We report in Tab. 5.6.3 a parameter study of both hyperparameters and mark in grey selected values, i.e., $\gamma = 0.01$ and $\delta = 0.8$. For γ , we observe that values $\gamma < 0.005$ are too low, most likely introducing too much noise in selected contrastive concepts.

Tab. 5.6.4 presents a parameter study of the cosine similarity of text queries β in multi-query segmentation. Here, we randomly select 100 images from the VOC training set and report classic mIoU for different β values. We select $\beta = 0.9$ because it gives the best result for most methods. We also note that controlling the similarity between query concepts and contrastive concepts in the multiple-query

Method	CLIP dataset	Original	CC^{PI}	CC^{BG}	CC^L	CC^D
VOC						
MaskCLIP	LAION-2B	—	49.9	47.9	51.8	53.6
	OpenAI	—	47.1	44.2	52.2	53.4
	MetaCLIP	—	47.9	46.6	50.6	50.1
CLIP-DINOiser	LAION-2B	63.8*	61.0	59.3	63.1	64.7
TCL	TCL's	52.9*	53.0*	52.9*	52.6*	53.6*
GEM	MetaCLIP	—	—	48.6*	61.3*	64.6*
CAT-Seg	OpenAI	—	—	52.8	69.5	67.7
Cityscapes						
MaskCLIP	LAION-2B	—	32.2	16.2	27.2	24.0
	OpenAI	—	30.6	15.0	22.5	22.0
	MetaCLIP	—	30.0	13.6	24.6	23.3
CLIP-DINOiser	LAION-2B	20.8	36.0	23.2	30.6	27.3
TCL	TCL's	18.6*	29.7	9.8	26.3	22.0
GEM	MetaCLIP	—	20.6	14.5*	21.5	14.6
COCO-Stuff						
MaskCLIP	LAION-2B	—	34.1	26.4	28.8	29.5
	OpenAI	—	33.6	24.1	28.4	28.8
	MetaCLIP	—	34.0	25.8	28.1	28.1
CLIP-DINOiser	LAION-2B	28.0*	35.3	32.4	33.9	34.4
TCL	TCL's	25.0*	34.7	17.4	29.5	30.6
GEM	MetaCLIP	—	38.3	22.9*	32.2	33.6
ADE20k						
MaskCLIP	LAION-2B	—	33.2	22.7	26.8	27.8
	OpenAI	—	29.8	20.2	23.5	25.2
	MetaCLIP	—	32.1	21.5	24.7	26.0
CLIP-DINOiser	LAION-2B	28.8*	35.3	28.9	29.7	31.6
TCL	TCL's	14.8*	32.6	14.9*	25.9	26.5
GEM	MetaCLIP	—	33.0	21.5*	26.3	29.1
CAT-Seg	OpenAI	—	46.8	25.7	38.4	39.7
COCO-Object						
MaskCLIP	LAION-2B	—	32.1	27.7	33.7	32.9
	OpenAI	—	31.3	24.3	34.5	33.3
	MetaCLIP	—	30.9	27.4	32.2	31.1
CLIP-DINOiser	LAION-2B	38.8*	38.9	35.5	41.6	39.9
TCL	TCL's	37.1*	38.1	37.2*	38.1*	37.2*
GEM	MetaCLIP	—	—	31.4	39.7	40.1
Pascal Context						
MaskCLIP	LAION-2B	—	40.5	34.4	35.2	37.4
	OpenAI	—	41.1	32.9	34.7	36.8
	MetaCLIP	—	41.1	32.6	34.2	35.8
CLIP-DINOiser	LAION-2B	33.9*	45.8	41.5	41.6	44.2
TCL	TCL's	29.7*	41.7	29.7*	36.8	38.2
GEM	MetaCLIP	—	—	26.9	40.1	42.1

Table 5.6.2. Results on all datasets with our IoU-single. “*” denotes the result when the original background handling gives the best results.

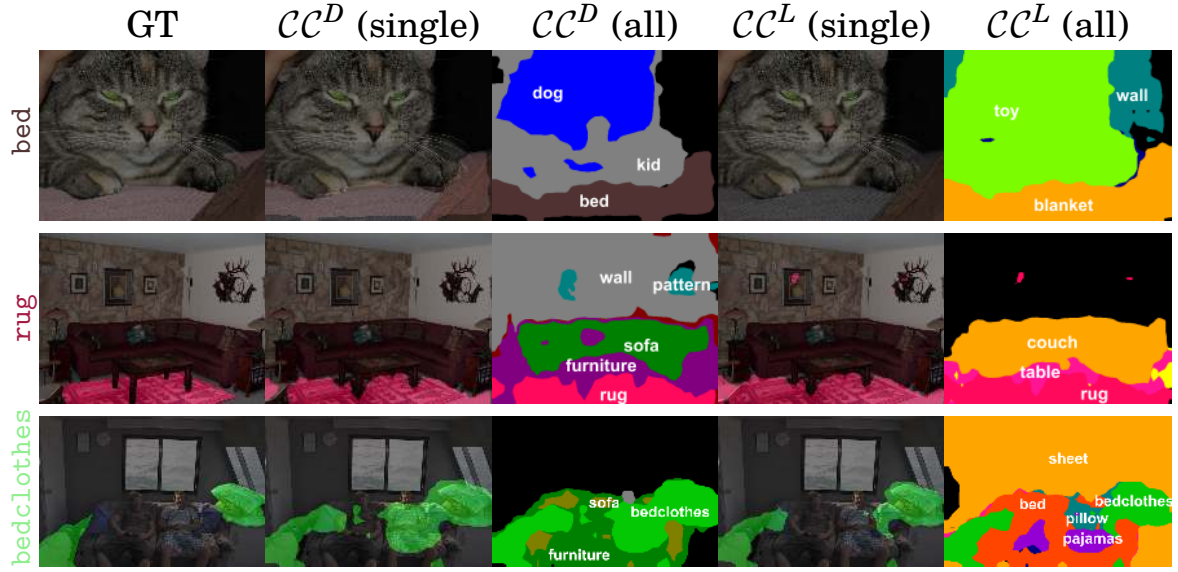


Figure 5.6.1. Failure cases of our method. We show examples of CLIP-DINOiser when one of the methods fails to generate accurate CC . In the first example CC^L suggests “blanket” for “bed” which typically covers the query concept. In the second row, both methods fail to provide “floor” to contrast with “rug”. Finally, in the third example, both methods fail to generate “person” to contrast with “bedclothes”, however, CC^L suggest “pyjamas”, which results in a better segmentation.

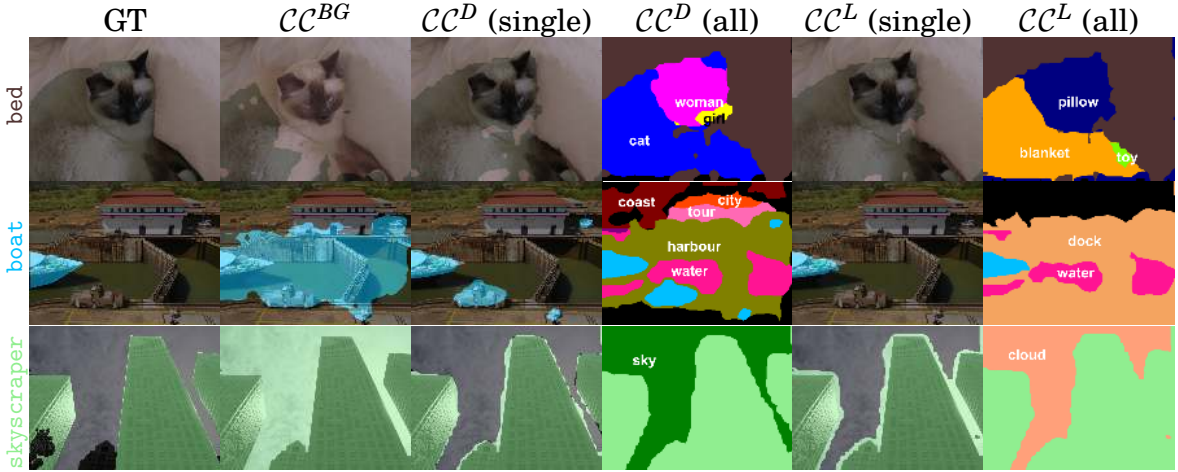


Figure 5.6.2. More qualitative results of CLIP-DINOiser with different CC . Here we focus on cases where CC^D and CC^L give different results. For “boat” (2nd row), CC^L gives a better result providing a good CC (“dock”). On the other hand, for “skyscraper” (3rd row), CC^D yields slightly better results suggesting “sky” and not “cloud”. Note that in this last example, CC^{BG} completely fails, possibly due to a difficult (uncommon) angle of view.

scenario is necessary. Not including this step (see results for $\beta = 1.0$) greatly degrades performance.

Method	CLIP tr. data	values of γ					values of δ				
		0.001	0.005	0.01	0.015	0.02	0.95	0.9	0.85	0.8	0.75
MaskCLIP	OpenAI	24.4	26.0	24.8	24.4	23.2	19.9	21.0	23.0	24.4	22.8
	Laion2B	25.8	27.8	27.4	26.0	25.4	23.0	24.1	26.4	27.4	24.6
	MetaCLIP	22.0	24.1	24.4	23.8	23.4	22.7	23.7	25.9	27.2	23.7
DINOiser	Laion2B	24.4	27.2	27.9	27.9	27.7	23.5	24.6	26.4	27.9	26.9

Table 5.6.3. Parameter study of γ and δ . Selection (marked in grey) of the hyperparameters γ and δ with IoU-single on 100 randomly-selected images in ADE20k training dataset.

Method	CLIP training data	1.0	0.95	0.9	0.85	0.8
MaskCLIP	OpenAI	26.0	40.4	41.1	39.1	32.1
	Laion2B	35.3	43.7	44.0	44.6	42.2
	MetaCLIP	24.4	39.1	40.3	34.3	30.6
DINOiser	Laion2B	51.3	57.8	58.6	58.8	55.2
TCL	TCL’s	37.2	47.6	47.7	47.1	47.7

Table 5.6.4. Selection of β with classic mIoU on 100 randomly-selected images in the VOC training dataset. Results are reported for CC^L .

5.6.7. Average number of contrastive concepts vs performance

We present in Fig. 5.6.3 a plot of performance vs the number of contrastive concepts when considering CC^D (Fig. 5.6.3(a)) and CC^L (Fig. 5.6.3(b)). The points correspond to the IoU-single scores per class obtained with CLIP-DINOiser on all the datasets we evaluate. We do not observe a strong correlation between the number of contrastive concepts and performance, although there is a small mode of around 20 concepts when using CC^D . We also observe that, on average, $|CC^D| > |CC^L|$.

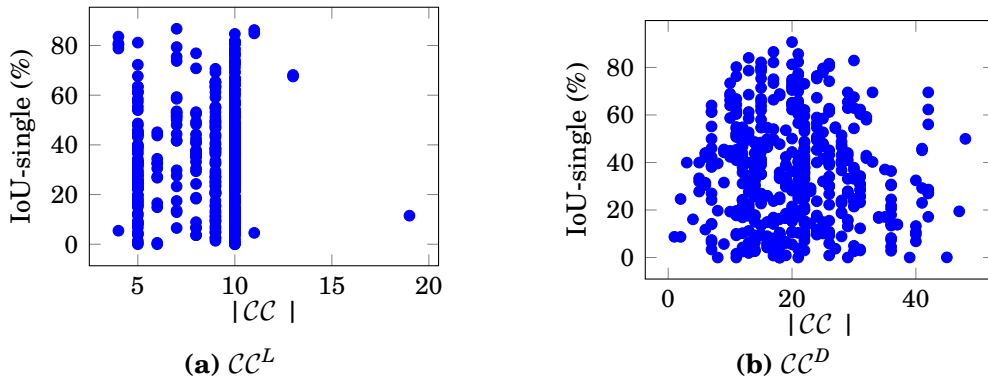


Figure 5.6.3. Number of CC vs performance. We compare the number of CC against the performance of CLIP-DINOiser for each class used in our evaluations (considering all datasets). Performance is reported with per class IoU-single %.

5.6.8. On separability of CLIP patch-features

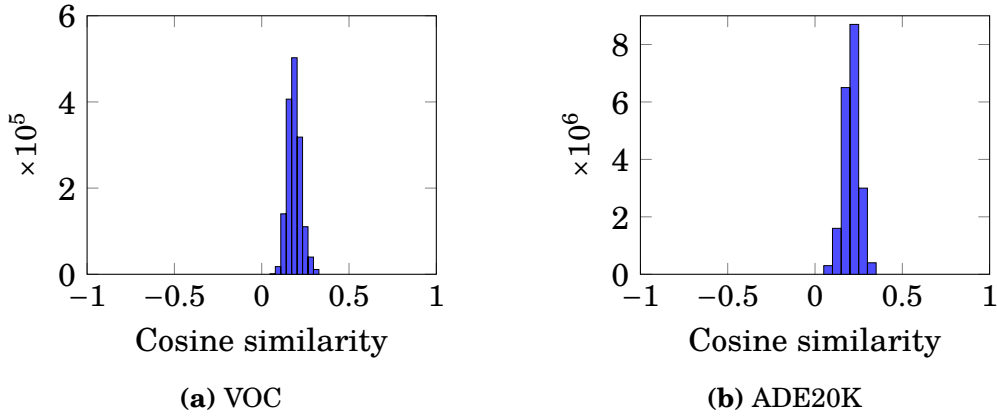


Figure 5.6.4. Distribution of maximum patch similarities with text prompts. We plot histograms for 100 images of VOC (a) and ADE20K (b) of patch similarities in MaskCLIP.

In Fig. 5.6.4, we present an analysis of the patch-level CLIP space using MaskCLIP features. The figure shows histograms of patch-level maximum text similarities (in cosine similarity) across 100 randomly sampled images from VOC (a) and ADE20k (b). We notice an overall concentration of cosine similarity scores in $[0.1, 0.3]$, suggesting that the feature space is not easily separable.

To illustrate how our approach overcomes this issue, we present in Fig. 5.6.5 a t-SNE analysis over patch features from an image of the VOC dataset for the $q = \text{"bird"}$. We plot the result of classification for each separate set of CC s. We highlight in orange the patches that belong to the ground truth mask of class "bird". We observe that CC^{BG} already helps to separate the space of background concepts from "bird" patches. However, we notice that only with CC^L or CC^D we can separate one visible cluster left, possibly belonging to the patch features of a branch in the image, by providing "branch" in the case of CC^D or "tree" in CC^L . Both of our proposed methods improve the final segmentation result.

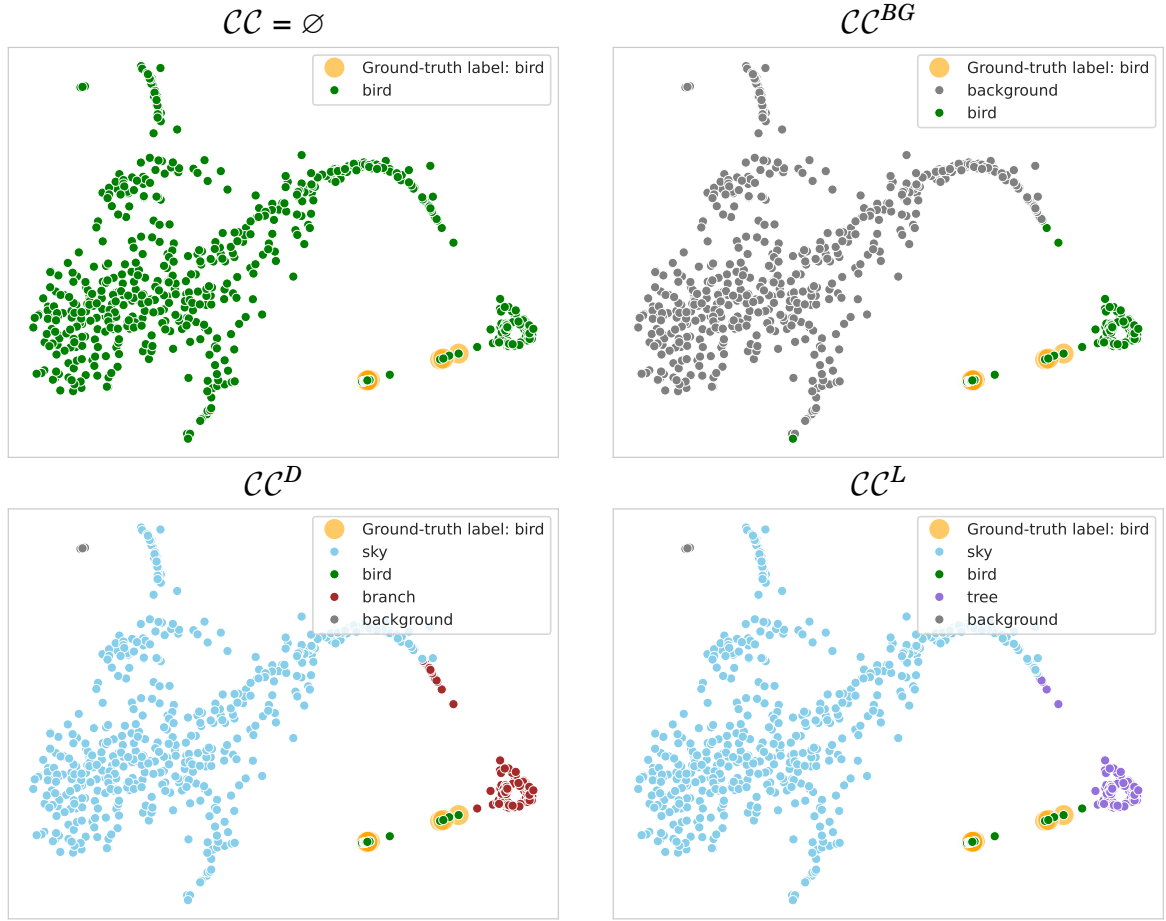
5.6.9. Replacing CC with sigmoid operation

Using a binary criterion to separate a query from its background is a natural alternative, and this could be implemented with a sigmoid.

We test using a sigmoid on CLIP similarity scores. We show the results of an experiment with CLIP-DINOiser on VOC in Fig. 5.6.6. We make the following observations: (1) none of the thresholds allow us to reach the performance of CC^{BG} , and (2) the performance is very sensitive to the threshold value. We believe this is because the CLIP space is not easily separable, as discussed in Sec. 5.6.8.



(a) Input image



(b)

Figure 5.6.5. t-SNE analysis of patch features for different CC of an image $q = \text{"bird"}$. We present patch features with their predicted closest text embedding coded in color. Text embeddings are corresponding CC of $q = \text{"bird"}$. We also mark the ground truth labels in orange. The sample is from VOC dataset.

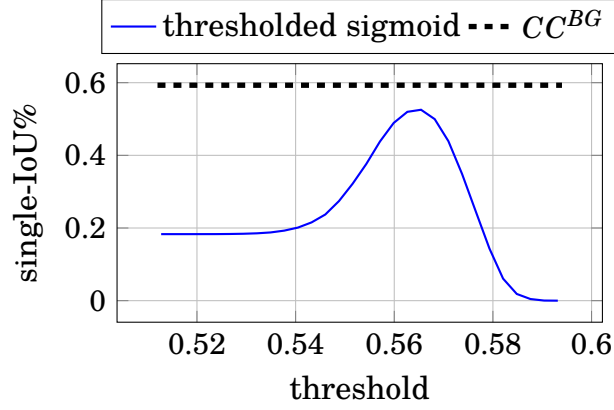


Figure 5.6.6. Sigmoid experiments. We replace softmax with sigmoid applied on individual patch-to-query prompt similarities. We show the variation of single-IoU% wrt. the threshold that is applied after sigmoid to decide on a positive vs. "background" class. To get the thresholds, we find the minimum and maximum values of the features after sigmoid and linearly sample 30 values in this range. We can see that the result is sensitive to the threshold value and does not reach the baseline of CC^{BG} .

5.6.10. Ontology-based filtering with WordNet

Method	MaskCLIP	TCL	CLIP-DINOiser
CC^D	25.2	26.0	31.6
$CC^D + \text{WordNet}$	25.2	26.4	26.3
$CC^D + \text{WordNet} - \text{sem. sim.}$	21.0	23.4	25.8

Table 5.6.5. Ontology-based (WordNet) filtering out synonyms, meronyms, hyponyms and hypernyms (at depth 1) from CC^D . Results are reported on ADE20K, as %IoU-single.

Here, we discuss our experiments using the WordNet ontology [166] for CC^D filtering. We extract synonyms, meronyms, hyponyms, and hypernyms for each query concept in-depth 1 in the WordNet ontology. From the results in Tab. 5.6.5, we observe that adding such filtering on top of our *semantic similarity* filtering brings little to no improvement, suggesting that *semantic filtering* removes most of the contrastive concepts that interfere with a query concept. Furthermore, replacing *semantic similarity* with WordNet-based filtering yields significantly worse results than our proposed CC^D .

5.6.11. Prompting the LLM

In this section, we provide more details about the LLM and the prompts used.

The LLM model. We use the recent Mixtral-8x7B-Instruct model [187], a sparse mixture of experts model (SMoE), finetuned for instruction following and released by Mistral AI. More precisely, we rely on the v0.1 version of its open weights available

via the Hugging Face transformers library. We run the LLM in 4-bit precision with flash attention to speedup inference.

The prompts used for contrastive concepts. We provide in Fig. 5.6.7 the prompt used to generate the contrastive concepts \mathcal{CC}^L and in Fig. 5.6.8 the prompt used to predict whether a concept can be seen in an image or not to filter \mathcal{CC}^D .

In these prompts, we indicate the inserted input text as $\{q\}$. We follow Mixtral-8x7B Instruct’s prompt template. In particular, we use $\langle s \rangle$ as the beginning of the string (BOS) special token, as well as $[INST]$ and $[/INST]$ as string markers to be set around the instructions.

For the generation of \mathcal{CC}^L , we also integrate a light post-processing step, ensuring that all generated lists have a unified format with coma separation. We do not apply any filtering or cleaning step to the LLM-generated results.

$\langle s \rangle [INST]$ You are a helpful AI assistant with visual abilities.

Given an input object O, I want you to generate a list of words related to objects that can be surrounding input object O in an image to help me perform semantic segmentation.

For example:

- * If the input object is 'fork', you can generate a list of words such as '["bottle", "knife", "table", "napkin", "bread"]'.
- * If the input object is 'child', you can generate a list of words such as '["toy", "drawing", "bed", "room", "playground"]'.

You should not generate synonyms of input object O, nor parts of input object O.

Generate a list of objects surrounding the input object $\{q\}$ without any synonym nor parts, nor content of it. Answer with a list of words. No explanation.

Answer: $[/INST]$

Figure 5.6.7. Prompt for \mathcal{CC}^L contrastive concept generation.

$\langle s \rangle [INST]$ Please specify whether $\{q\}$ is something that one can see.

Reply with 'yes' or 'no' only. No explanation.

Answer: $[/INST]$

Figure 5.6.8. Prompt for \mathcal{CC}^L visibility prediction.

Example of generated \mathcal{CC}^L . We present in Tab. 5.6.6 the example of \mathcal{CC}^L generated for Cityscapes dataset. We provide \mathcal{CC} for each query q in separate rows.

Query q	\mathcal{CC}_q^L
road	building, tree, car, pedestrian, sky, streetlight, sidewalk, bicycle, parked car, traffic sign
sidewalk	building, street, car, tree, people, bike, road, park, sky, lane
building	sky, tree, road, car, park, people, lane, fence, house, field
wall	door, window, floor, ceiling, painting, light, chair, table, carpet, curtain
fence	grass, tree, house, car, path, post, gate, field, flowers, animals
pole	building, wire, tree, street, sky, fence, cable, road, banner, light
traffic light	road, car, building, pedestrian, sky, streetlight, traffic sign, parking meter
traffic sign	road, street, pole, vehicle, building, sky, pedestrian, curb, lane, light
vegetation	soil, tree, grass, water, animal, fence, field, sky, rock, sun
terrain	tree, sky, building, road, mountain, river, field, fence, vehicle, person
sky	tree, building, cloud, sun, bird, airplane, mountain, sea, sunset, cityscape
person	bike, road, car, tree, building, park, cityscape, nature, animal, sports equipment
rider	bicycle, road, nature, park
car	road, tree, building, person, parking
truck	road, car, building, tree, parking
bus	road, tree, building, sky, person, car, traffic light, bicycle, parking meter, street sign
train	track, grass, sky, building, platform, tree, sign, person, car, road
motorcycle	road, person, bike, car, traffic, building, nature, parking, city, scenery
bicycle	road, tree, person, park, building, grass, basket, helmet, traffic, path

Table 5.6.6. Example of LLM-generated \mathcal{CC}^L for Cityscapes.

Part removal via LLM-prompting. We also explore the possibility of removing suggested contrastive concepts that can be *parts* of query concepts. Note that in \mathcal{CC}^L , we explicitly do it in the prompt itself (Fig. 5.6.10). Fig. 5.6.9 presents one of such examples when removing “wheel” from the \mathcal{CC}^D of query “bicycle” gives a slight improvement for MaskCLIP segmentation. However, we do not notice a particular improvement in the case of other segmentation methods since, typically, they refine the masks or feature maps to include localization priors. For example, in Fig. 5.6.9, the second row presents the same example for CLIP-DINOiser (DINOiser), where the improvement is marginal. Finally, we observe little or no quantitative improvement when applying part removal filtering on entire datasets. Therefore, we do not include it in our final method.

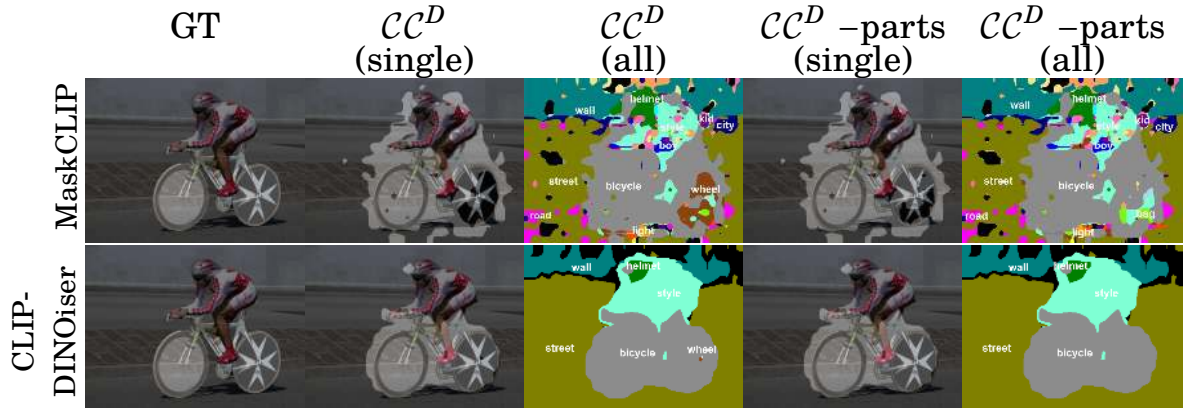


Figure 5.6.9. Part removal. We consider an example from Pascal Context with $q = \text{bicycle}$. We show the segmentation masks produced by MaskCLIP and CLIP-DINOiser for \mathcal{CC}^D , as well as for \mathcal{CC}^D when parts of objects are removed ($\mathcal{CC}^D - \text{parts}$).

<s> [INST] You are a helpful AI assistant with visual abilities.

Given an input object O, I want you to generate a list of words that are parts of an object O.

For example:

- * If the input object is 'rabbit', you can generate a list of words such as '["paw", "tail", "fur", "ears", "muzzle"]'.
- * If the input object is 'building', you can generate a list of words such as '["door", "window", "wall", "hall", "floor"]'.

Generate a list of parts of the input object {q}. Answer with a list of words. Do not give any word that is not a part of the input object. No explanation.

Answer: [/INST]

Figure 5.6.10. Prompt for part prediction.

6. Task Adaptation for Foundation Model Selection and Analysis

In the preceding chapters, we examined the exploitation of off-the-shelf representations from Vision-Language Models and their potential integration with other visual foundation models to achieve efficient downstream task performance, with particular emphasis on open-vocabulary semantic segmentation. Through CLIP-DINOiser, we demonstrated that the understanding of the complementary strengths and limitations of various foundation models can inform the development of effective task adaptation strategies, simultaneously improving downstream performance while minimizing computational and annotation costs. As we progress in our investigation, we arrive at a fundamental research question: how can we systematically evaluate and determine the optimal visual foundation model for a specific task?

The exploitation of off-the-shelf representations without substantial architectural modifications or alterations to their representation space allows downstream task adaptation to serve as an evaluation framework analogous to the linear probing methodology discussed in Sec. 2.2.3. While linear probing provides an effective evaluation strategy for different image-level and pixel-level tasks, this chapter focuses on the more complex domain of visual reasoning, specifically visual question answering (VQA). The inherent complexity of this task stems from its multimodal nature. Successful solution of the task depends not only on detailed visual scene representation but also sophisticated language understanding capabilities to process complex scene-related queries. Furthermore, the necessity of modality fusion introduces additional complexity to the adapter design. Our objective is twofold: to compare diverse visual representations using a generic design framework and to ensure fair comparison across representations, regardless of their architectural differences and model scales.

Our next contribution proposes a systematic evaluation protocol for visual representations in the context of VQA through downstream task adaptation. To enable rigorous comparison of diverse visual representations, we develop a universal reasoning module that accommodates varying input representation dimensions, thereby ensuring fair comparative studies. This evaluation framework facilitates the system-

atic analysis of differences between visual representations and their task-specific suitability. Through VQA as an exemplary task, we derive several insights that have implications for the development of more robust visual representations.

While Sec. 2.2.6 discusses prior work with similar objectives in providing evaluation frameworks for visual representations in VQA, our approach differs in its emphasis on controlled experimentation. We specifically focus on synthetic datasets, offering more precise control over experimental conditions than the approaches discussed in Sec. 2.2.6. Our methodology includes a rigorous validation of the adaptation module’s task suitability through verification using ground-truth scene representations. This work was first presented at the NeurIPS Workshop in 2022, preceding the evaluation frameworks discussed in Sec. 2.2.6.

6.1. Introduction

Visual representation learning has gained a lot of attention thanks to advanced frameworks demonstrating unprecedented results without relying on explicit supervision [15, 48, 21]. On the one hand, classical self-supervised methods [45, 191, 22, 15, 192] producing localized features (i.e., features that densely correspond to regions of the image) are exhaustively evaluated on standard tasks like image classification or object detection where they perform on par with the supervised ones. Although highly popular, these features provide an unstructured representation of a scene, which impairs their generalization capabilities [193]. On the other hand, more recent unsupervised systems [194, 195, 196, 197] aiming at learning object-centric representations (i.e., each feature is associated with an object in the image) are typically evaluated, for instance, segmentation where benchmarks are saturated. While the object-centric representations enforce the decomposition of a scene into objects, yielding an overall more structured representation, they are still limited to synthetic datasets [198]. Motivated by the recent advancements in visual representation learning, in this work, our goal is to investigate how well off-the-shelf features extract meaningful information about the objects in a given image.

Because features are usually implicit, finding explicit mappings between the latent space and the corresponding attributes is typically ambiguous and difficult. We thus propose to evaluate the feature ability to model objects through the performance of a reasoning module trained for different visual reasoning tasks. We find visual reasoning a perfect testbed to assess the quality of visual representations as it requires a holistic visual understanding [199]. However, setting various image representations side by side in a unified framework that enables a fair comparison is not straightforward. To tackle this problem, we design a new evaluation protocol

that is based on a simple transformer-based reasoning module learned on top of the frozen visual features to be evaluated. Similarly to feature evaluations that use shallow networks for predictions to decouple visual extraction from evaluation, we build a restricted reasoning module with a limited capacity in two ways. We first methodically find a reasoning module with a minimal computational capacity necessary to obtain high performances with perfect image representation given by the ground truth. Second, we limit the size of the reasoning module input to a fixed number to ensure fair comparisons among features of different sizes. Finally, we provide results in low-shot evaluation setups, which we argue are better suited to identify noisy representations yielding spurious correlations. Contrary to existing protocols [200, 201], our VQA evaluation enables us to compare off-the-shelf features, either densely extracted local features or object-centric representations.

Based on the protocol we make the following findings. Although such representations demonstrate excellent results in their dedicated benchmarks, their reasoning performances in our constrained setup are far from the ones obtained with ground truth object descriptions or from state-of-the-art performance obtained using supervised features. This suggests that off-the-shelf features do not accurately represent all the scene information necessary to solve the visual reasoning task. Secondly, we find that object-centric features are better suited for visual reasoning than local features. Although this is conceptually expected, we provide an empirical way to exhibit this behavior. Finally, we demonstrate that low-shot evaluation setups prohibit learning the correct reasoning patterns across all visual representations. However, object-centric representations seem to be better at preventing learning spurious correlations than localized ones.

To summarize our contribution is as follows:

- We propose to study visual features’ suitability to address complex reasoning by studying their performance on the VQA task.
- To allow for fair studies between different representations, we introduce a specific memory adaptation module and a restricted-size reasoning module.
- Our framework allows us to make several findings on off-the-shelf visual representations for visual reasoning.

6.2. Related work

In this section, we first briefly cover the relevant works on visual reasoning, with an emphasis on image representation and reasoning. We then give an overview of generic representation learning methods, their capabilities at solving other tasks, and their relevance to visual reasoning.

6.2.1. Visual question answering (VQA)

Good visual representations are typically used in end-to-end supervised VQA systems to achieve high accuracy [202, 76, 77]. Although most methods use Convolutional Neural Networks (CNNs) typically pre-trained on ImageNet [6] to extract local features, there are two different approaches to exploit them, leading to different levels of semantic structure in the visual representations. One is the grid-level approach, where extracted local features uniformly span a whole image [203, 204, 205, 206] and as such are used in the reasoning process to provide local visual information. Another approach is to use object-level representations, first introduced in [207], where the idea is first to detect objects and salient regions in an image and then only use features corresponding to those regions. Although very effective [208, 209], such an approach requires a pre-trained object detector and bounding box annotations for the supervision, which is not always feasible in real-case scenarios. Recently, [210] proposes 3D prototypical networks and demonstrates their effectiveness on 3D datasets for Visual Reasoning by disentangling RGB-D images into objects, their 3D locations, sizes, 3D shapes, and styles. However, these visual representations are trained, or at least fine-tuned, on the VQA objective and are thus specialized to solve this task. In contrast, in this work, we want to evaluate to what extent it is possible to answer simple visual reasoning questions using unspecialized features that have not been trained for this specific objective. Finally, the orthogonal to our line of works discards the visual representation part of VQA by using off-the-shelf image captioners to map an input image to the textual space and performing reasoning in the unimodal space, typically using large language models (LLM) [211, 212].

From the reasoning standpoint, the large body of proposed methods for VQA relies on dedicated natural-language parsers, which determine a sequence of logical steps to perform to give an answer to a particular question [213, 209, 214, 215]. The reasoning steps are carried out by small neural module networks specifically designed for the task, which are usually trained with strong supervision by using functional program annotations. Recently, self-attention mechanism [50] proved successful across many applications, arising from natural language processing, through vision [37] up to multimodal learning [216, 13, 217]. Moreover, the literature also points out that they are suitable for symbolic computations [218, 219, 220]. [221] leverages the self-attention mechanism in spatiotemporal reasoning, demonstrating at the same time the applicability of unsupervised object-centric representations for reasoning. Finally, [222] proposes a transformer-based architecture for unified object detection and reasoning, which does not use any functional program supervision but is trained to detect objects that are required for performing intermediate reasoning steps, hinting that self-attention may be sufficient to solve simple visual reasoning

questions. In this work, we also leverage self-attention as the building block for the reasoning part of our work. We show that it is indeed sufficient to obtain good accuracy when used with well-structured visual representations.

6.2.2. Generic representation learning

The common approach of reusing representations pre-trained on ImageNet [6] proved successful across multiple vision tasks [71]. On the other hand, recent advances in self-supervised learning (SSL) [45, 191, 22, 15, 192] have shown that learning image representations without labels may yield better, more generalizable representations for downstream tasks. As the first self-supervised approaches relied on discriminative learning [45, 191], they used a significant amount of computations due to large batches and memory banks requirements [192]. Less computational-intensive methods in terms of training have been recently proposed based on self-distillation. Particularly, [22] uses two neural networks, referred to as online and target networks, that interact and learn from each other. DINO [15] improves on this idea and shows that this training strategy, combined with ViT [37] as an architecture, gives high-quality image representations suitable for segmentation tasks. In this work, we examine the effectiveness of self-supervised features without explicit object structure, DINO in particular, in terms of their ability to provide effective representations for visual reasoning.

The significant body of recent works shows the effectiveness of learning visual representations through natural language guidance, which can be considered another form of self-supervision. Particularly, CLIP [13] and ALIGN [24] employ a contrastive similarity learning approach to match images with their associated captions and leverage a large dataset of image-text pairs from the Web for training. Text-aligned representations proved useful in many downstream tasks such as image captioning [223], image retrieval [224], and open vocabulary semantic segmentation [30, 29]. In our work, we study the effectiveness of visual representations in multimodal space for visual reasoning, particularly CLIP.

In parallel to popular local features extracted on a dense grid, recent works aim at learning features that can explicitly be associated with objects contained in the given image. These unsupervised approaches can be split into three categories. A first category of methods estimates latent variables which best encode pixel assignments to a fixed number of image components [225, 226, 194, 195, 227, 196]. A second category of methods leverages spatial attention [228] to extract image patches containing objects [229, 230, 231, 232]. The third category of methods explicitly learns a dictionary of RGBA images representing canonical objects called sprites, that are positioned and composed to generate the image [197, 233]. In this work,

we evaluate two different approaches from the first and third categories, namely Slot Attention [196] and DTI-Sprites [197], which demonstrated state-of-the-art results in terms of unsupervised instance segmentation. Since these features naturally uncover the object structure in the scene that seems to be critical for visual reasoning, without using explicit supervision, we investigate how they perform in visual reasoning problems and compare them to the unstructured representations described previously.

6.3. Evaluation framework

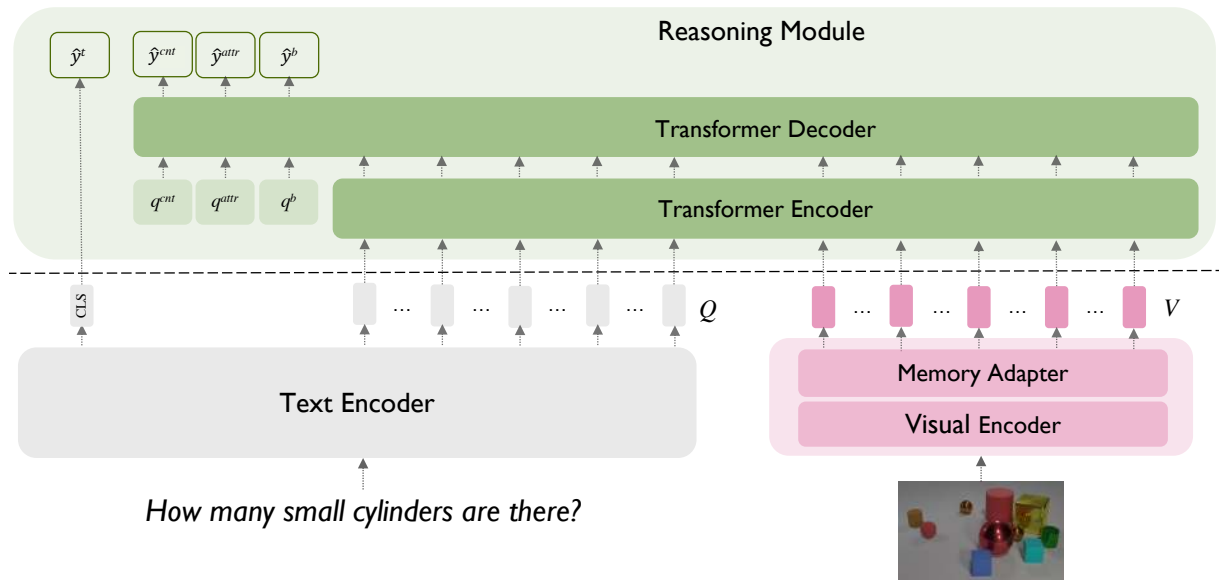


Figure 6.3.1. Our evaluation framework overview. Given an input question and the corresponding input image, we first extract text tokens Q as well as visual tokens V , which are constrained to have a fixed size through a module called *memory adapter*. These tokens are then concatenated along the sequence dimension and fed to the reasoning module. Finally, we decode the answer using several heads attached to the reasoning module and a specific head given the predicted question type. During training, only the parameters of the reasoning module are trained (above a dashed line), while the rest of the pipeline remains frozen.

In this section, we introduce our framework to evaluate image representations on the visual reasoning tasks. We first provide the main motivation for this work. We then give a general overview of the pipeline and move further with the details on how we adapt different visual representations for this study.

6.3.1. Motivation

Our goal is to examine to what extent different off-the-shelf image representations are capable of encoding the information needed for reasoning. Particularly, we are interested in investigating whether the visual features proven to be very effective for classical vision tasks are suitable for basic reasoning. We examine popular off-the-shelf features that we split into two groups: (i) the classical dense sets of features localized on a grid-like structure, which we refer to as \mathcal{LOC} , and (ii) object-centric features that can be associated with objects in the scene, denoted here as \mathcal{OBJ} . In the case of learned local features, we further distinguish 2 types of learning processes. We examine the standard approach of transferring features obtained from backbones pre-trained for classification on ImageNet [6], which we denote by \mathcal{IN} . We also study features that are obtained through more recent self-supervised learning frameworks, denoted by \mathcal{SSL} .

Examining such various visual representations rigorously imposes a challenge on the design of a fair evaluation protocol. To make features comparable, we introduce a restricted reasoning module that is limited by both a small computational capacity and a fixed-size input, which we will refer to as the reasoning module *memory* in the following. We cover these evaluation constraints in subsection 6.3.3.

6.3.2. Pipeline overview

We build our pipeline upon the *disentangling reasoning from the vision and language understanding* paradigm [213]. Our framework serves as plug’n’play for unimodal encoders and consists of 3 separate modules: text, vision, and reasoning respectively. First, given the question-image pair, we use frozen text and visual encoders to extract representations and map them to a common multimodal space. Then, the concatenated features are fed to the reasoning module predicting the answer. An overview of our pipeline is presented in Figure 6.3.1.

Text module

Since the focus of this work is evaluating image representations, we use a pre-determined and fixed text encoder to ensure a fair evaluation of the visual features. In this work, we use the RoBERTa language model introduced in [234] as the fixed text encoder. Note that this could be any other language model that produces a question representation as a sequence of text tokens.

Visual module

The visual module maps an input image to a set of visual tokens in a two-step process. First, a visual encoder extracts an image representation, which is seen as a

sequence of features. Then, to make sure that all features are comparable, we design a module called *Memory adapter* whose goal is to convert the extracted representations to a fixed-size input. We cover both processes in detail in subsection 6.3.3.

Reasoning module

Inspired by the state-of-the-art model MDETR [222], our reasoning module is a transformer encoder-decoder architecture [50] which operates on a sequence of text tokens Q and a sequence of visual tokens V . To allow our reasoning module to distinguish between modalities, we add to each token a modality-specific segment embedding which is a learnable parameter, in a similar fashion to practices in natural language processing. Additionally, in the case of local features, we add a learned positional encoding to incorporate spatial information, as they do not inherently have such information.

First, we forward the concatenated sequence of text and visual tokens through a transformer encoder with N_T layers with a N_H number of heads in the self-attention mechanism. Then, we apply the same number of layers and heads of the transformer decoder to predict the answer. Following MDETR [222], we use several heads specialized in specific types of questions. However, unlike them, we move the question type head and pool it from the output CLS token of the text encoder, as our preliminary studies showed it prevents overfitting to the language signal.

Training strategy

Compared to end-to-end supervised VQA pipelines, we train our reasoning module using question-image-answer triplets without any external supervision. Concretely, given a question-image-answer triplet (q, i, a) we predict the answer type \hat{y}_t as well as the corresponding answer encoded by \hat{y}_b , \hat{y}_{cnt} or \hat{y}_{attr} respectively associated to *True/False*, *count* and *attribute* questions. Therefore, our final loss is defined as:

$$L_{total} = L_t + L_b + L_{cnt} + L_{attr}, \quad (6.1)$$

where L_t, L_{cnt}, L_{attr} denote cross entropy losses between the ground truth y_t, y_{cnt}, y_{attr} respectively, and the predictions $\hat{y}_t, \hat{y}_{cnt}, \hat{y}_{attr}$, whereas L_b is a binary cross entropy loss between y_b and \hat{y}_b .

6.3.3. Evaluation constraints

Drawing inspiration from standard representation learning evaluations, we design a restricted reasoning module taking fixed-sized features as input to assess their quality in comparable setups. We do so by designing a minimal reasoning

transformer using ground truth information and by introducing a visual memory adapter that ensures a fixed-sized input memory to the reasoning module. Intuitively, shallow evaluation networks prevent by design the evaluation part from performing the representation learning task (e.g., considering raw pixels as features and a deep classification CNN as an evaluation module is absurd), and the visual memory adapter allows us to make features of different sizes comparable.

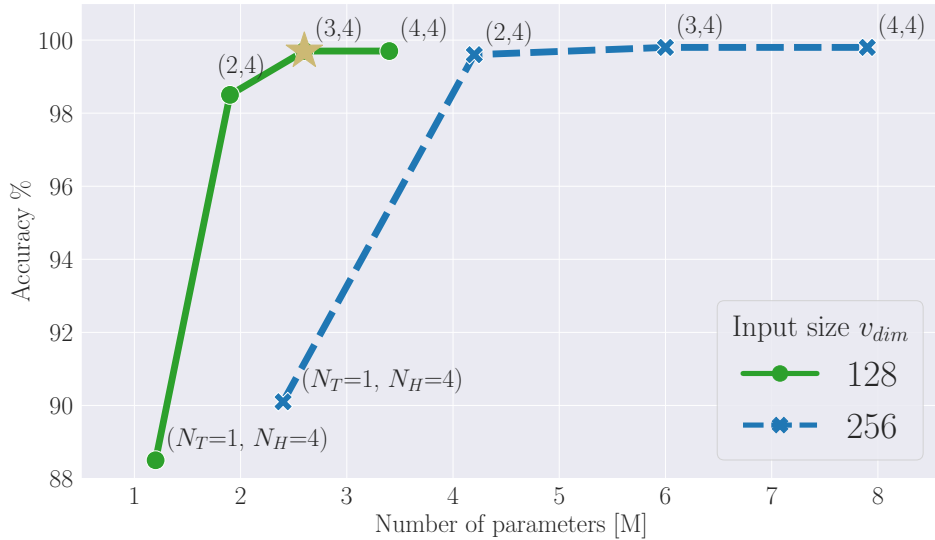


Figure 6.3.2. Accuracy on CLEVR against the number of parameters in the reasoning module when using perfect visual information. Below a minimal complexity in the reasoning module, even having perfect visual information is insufficient to solve the VQA task.

Restricted reasoning module

We propose to use the vectorized ground-truth description of a scene to find a reasoning module with minimal complexity that solves the task using these ground-truth features. In practice, we vectorize the information about each object’s presence, position, and properties and use the vector as a visual token V instead of extracting visual representations with the visual encoder. We then find the minimal set of hyperparameters of the transformer that yields a high-performance reasoning module.

More precisely, we methodically use a grid search to optimize for the number of layers N_T , the number of heads N_H , and the dimensionality of the input size v_{dim} . Figure 6.3.2 shows some of the configurations of hyperparameters and their associated accuracy. As we can see, there is a minimal computational capacity below which the task is unsolvable even with perfect visual information. We thus set the capacity of our reasoning module to that minimum, *i.e.* $N_T = 3$, $N_H = 4$, $v_{dim} = 128$.

Visual memory adapter

To ensure comparability, the reasoning module should keep a fixed-sized input memory across all representations. To fix the input memory size of V , we apply a two-step process. First, depending on whether the visual encoder is \mathcal{OBJ} or \mathcal{LOC} , the sequence length N_v may differ. Thus, we restrict the N_v to be roughly equal to the maximum number of objects in a scene. Let us assume we have a K maximum number of objects that can appear in a scene. Therefore, if the $N_v > K$, which is the case for \mathcal{LOC} due to the grid-like structure of the local features, we apply adaptive average pooling on the features until we roughly match a size of K .

Second, to ensure the reasoning module operates on compact representations of similar sizes in memory, we constrain the dimension of the visual tokens. Let us assume d_v^{min} is the minimum visual token dimension needed to solve the task, which corresponds to a number of objects' properties and their respective positions in a scene for relational reasoning. If $d_v > d_v^{min}$, we compress visual tokens using Principal Component Analysis (PCA) and decrease the dimensionality to match d_v^{min} .

Therefore, we define the minimum memory size M required to solve the task as:

$$M = K \cdot d_v^{min} \quad (6.2)$$

The memory adapter converts the output of the visual encoder to a fixed memory size within a few orders of magnitude of M by relaxing the d_v^{min} constraint since visual features are not expected to attain perfect compression of the visual information. This, in turn, implies a fixed input size for the reasoning module since the text encoder is shared in the evaluation.

6.4. Experiments

In this section, we present the results of our comparative study. We first give an overview of implementation details, including the evaluation datasets and visual encoders used, and then move on to experiments with constrained memory or training size.

6.4.1. Datasets

We conduct our analysis on 2 CLEVR-like datasets: CLEVR VQA [235] - a standard synthetic benchmark for Visual Question Answering and CLEVR-Math [236]. CLEVR world consists of 3D objects in 3 different shapes, 8 different colors, 2 sizes,

and 2 materials. Thus, there are 4 properties to encode in the visual representation. Moreover, the maximum number of objects in a scene K is 10. Therefore, the minimum memory size of visual tokens $M = 10 \times 7$ for 10 objects, 4 properties, and 3D position (x,y,z).

In CLEVR VQA, for each image, there are associated questions that ask about various aspects of the scene. These questions are designed to assess different levels of reasoning, including object attributes, spatial relationships, counting, and comparisons. Each question comes with a corresponding answer. The answers can be single-word or numbers that provide the solution to the posed question based on the information present in the image.

CLEVR-Math is an extension of CLEVR consisting of simple math problems involving addition/subtraction, represented partly by a textual description and partly by an image illustrating the scenario. The text describes actions performed in the scene that is depicted in the image. The difficulty of the tasks stems from the fact that the question posed may not be about the scene in the image but about the state of the scene before or after the actions are applied. Solving these word problems requires a combination of language, visual, and mathematical reasoning. Note that the questions in the CLEVR-Math dataset concern only one type of answer; therefore, we keep only the *count* question type head. For more details on CLEVR-Math adaptation, see Sec. 6.6.5.

The advantage of synthetic CLEVR-based datasets is the presence of ground-truth scene graphs, which fully describe objects and their relationships in the scene. We use scene graphs to curate the design and training procedure of our reasoning module, which would not be possible without the ground truth.

6.4.2. Implementation details

For our comparative study, we choose popular image representations trained with various levels of supervision. Models are always frozen during training and used solely as feature extractors.

Dense local features

We use features extracted from two popular architectures: convolutional-based ResNet-50 [36], and the transformer-based ViT proposed in [37]. For ResNet-50, we use the local features after the last convolutional layer right before the global average pooling, whereas for ViT, we use the output tokens of the last layer corresponding to image patch position (*i.e.*, the CLS token is not used). To study the influence of the level of supervision in image backbones, in the case of ResNet-50, we evaluate the local features trained in a supervised manner using the classification task on Imagenet,

denoted $\mathcal{LOC-IN}$, and self-supervised DINO features, denoted $\mathcal{LOC-SSL}$. We also account for localized features trained with language supervision by evaluating the CLIP image visual encoder. We denote CLIP features as $\mathcal{LOC-CM}$. We use the smallest model available, which is ViT-B, and check the performance in 2 different configurations, with input patches of size 16 and input patches of 32. For DINO, we report the results for ViT-S architecture as our experiments with ViT-B trained with DINO supervision showed that the larger architecture results in lower performance. For more details, see the Supplementary Material. To encode the position of local features, we use the corresponding 2D position in the feature map for ResNet-50 features and the 2D position of the corresponding patch for ViT features.

Object-centric representations

To evaluate the performance of unsupervised object-centric representations, we consider two methods: Slot Attention [196] and DTI-Sprites [197], which demonstrated state-of-the-art segmentation results on the recent CLEVRText [198] benchmark. We first train both methods on the CLEVR dataset - which does not require any supervision - and use pre-trained models as feature extractors. More details on the training process can be found in the supplementary material. In Slot Attention, we use the slots right before the decoder part as features. DTI-Sprites is an explicit \mathcal{OBJ} representation. Thus, the features correspond to predicted transforms (color, scale, position, and prototype id) for each object in a scene. We refer the reader to the original paper for further details [197]. More details on feature adaptation can be found in Sec. 6.6.2.

Memory adaptation

Since all considered representations are much larger than M , we consider 2 memory sizes in our protocol: (i) a total memory of size 100, denoted M_1 , which approximates the minimum $M = 10 \times 7$ for CLEVR dataset, and (ii) a total memory of size 1000, M_2 , to account for richer representations than the bare minimum required for solving the VQA task. For completeness, we also report results without any memory adaptation, a setup denoted by M_∞ .

In the case of \mathcal{LOC} , we follow standard practices for normalization and input image size (224×224), whereas, in the case of \mathcal{OBJ} , we follow their standard practices and use images of size 120×160 . This implicitly means that \mathcal{OBJ} features have access to fewer details than \mathcal{LOC} features, but this is compensated by their intrinsically more precise account of spatial information.

In Tab. 6.4.1, we summarize the parameters used in both the spatial pooling

method	image size	feature size (M_∞)	M_1	M_2
GT	-	10×7	10×10	10×100
DTI-Sprites	120×160	10×10	10×10	10×100
Slot Attention	120×160	11×64	11×9	11×90
ResNet-50	224×224	49×2048	16×6	16×62
ViT-S-16	224×224	194×384	16×6	16×62
ViT-B-16	224×224	194×768	16×6	16×62
ViT-B-32	224×224	49×768	16×6	16×62
Raw	192×192	9×12k	9×11	9×111

Table 6.4.1. Evaluation setups for different memory regimes. We conduct our studies in 2 memory regimes, M_1 and M_2 . We also provide the raw feature size (corresponding to a memory regime M_∞) to highlight the extent of compression and expansion applied at the adaptation stage.

and the dimension reduction strategies described in Sec. 6.3.3 to fit the 2 memory constraints for each visual encoder.

	Method	count	exist	comp num	query attr	comp attr
<i>M_1 memory regime</i>						
	GT	99.8	99.9	99.5	99.2	100.0
<i>OBJ</i>	DTI-Sprites [197]	72.6	87.8	84.8	84.3	82.4
<i>OBJ</i>	Slot Attention [196]	50.8	70.8	70.1	58.3	55.8
<i>LOC-SSL</i>	DINO ResNet-50	51.0	72.6	71.4	58.7	61.2
<i>LOC-IN</i>	ResNet-50	47.3	69.3	69.7	56.2	56.0
<i>LOC-CM</i>	CLIP	47.8	67.7	69.3	50.4	54.4
<i>LOC-SSL</i>	DINO ViT	45.2	66.9	69.2	46.3	53.1
<i>LOC</i>	Raw	44.9	64.1	67.9	39.8	51.1
<i>M_2 memory regime</i>						
	GT	100.0	100.0	99.8	99.1	100.0
<i>OBJ</i>	Slot Attention [196]	85.1	95.0	92.8	93.1	92.6
<i>OBJ</i>	DTI-Sprites [197]	72.7	88.1	86.2	84.6	82.9
<i>LOC-SSL</i>	DINO ResNet-50	71.4	88.8	82.9	85.7	84.7
<i>LOC-SSL</i>	DINO ViT	66.0	86.5	79.1	80.7	81.0
<i>LOC-IN</i>	ResNet-50	66.6	85.9	81.5	83.0	80.2
<i>LOC-CM</i>	CLIP	60.4	82.3	78.8	78.7	74.4
<i>LOC</i>	Raw	44.5	64.8	69.0	43.9	51.2
	MDETR [222]	99.3	99.9	99.4	99.9	99.9

Table 6.4.2. Results on the CLEVR VQA dataset in 2 memory size regimes. We report scores on the validation set with a detailed split by the question type. We report average accuracy on the validation set.

6.4.3. Quantitative results

In this experiment, we study the effectiveness of different visual features in solving 2 reasoning tasks. We report the question-specific categorical accuracy in Tab. 6.4.2 on CLEVR VQA and the overall accuracy in Tab. 6.4.3. We also report the overall accuracy on CLEVR-Math dataset in Tab. 6.4.4. For completeness, we report results without any memory adaptation in a setup denoted by M_∞ . As a sanity check to investigate how much visual processing our reasoning module is capable of, we also experiment with compressed raw image patches as visual tokens, denoted as Raw in the method column.

	Method	M_1	M_2	M_∞
<i>OBJ</i>	DTI-Sprites [197]	81.8	82.2	81.8
<i>OBJ</i>	Slot Attention [196]	58.9	91.4	91.4
<i>LOC-SSL</i>	DINO ResNet-50	60.4	82.4	89.2
<i>LOC-IN</i>	ResNet-50	57.1	79.0	87.7
<i>LOC-CM</i>	CLIP-32	56.6	70.9	94.4
<i>LOC-CM</i>	CLIP-16	54.6	74.0	91.3
<i>LOC-SSL</i>	DINO ViT	52.1	78.4	89.8

Table 6.4.3. Overall results on CLEVR VQA in 2 memory size regimes and with no memory restriction. We report average accuracy on the validation set.

	Method	M_1	M_2	M_∞
<i>OBJ</i>	DTI-Sprites [197]	57.3	57.3	57.3
<i>OBJ</i>	Slot Attention [196]	52.9	74.0	74.0
<i>LOC-SSL</i>	DINO ResNet-50	50.4	60.4	69.9
<i>LOC-IN</i>	ResNet-50	50.1	59.0	68.8
<i>LOC-CM</i>	CLIP-32	51.6	61.6	81.8
<i>LOC-SSL</i>	DINO ViT	51.1	62.4	79.1

Table 6.4.4. Overall results on CLEVR-Math in 2 memory size regimes and with no memory restriction.

At comparable size, are visual features as good as a ground-truth representation?

We focus here on the M_1 setup, which ensures results comparable to the one obtained using ground truth. In the case of CLEVR VQA dataset, it is clear that all

of the methods except for DTI-Sprites fail to attain an accuracy significantly different from the Raw baseline. This suggests that they do not encode visual information in a compact way that is comparable to perfect visual information. For CLEVR-Math dataset DTI-Sprites also performs the best out of all compared visual features, however, overall the scores are much lower compared to the VQA task, which indicates that the math reasoning task itself is much more difficult. We attribute the overall best performance of DTI-Sprites in this very restricted memory size regime to its inherent explicit representation nature. The original memory size of a DTI-Sprites visual token is the closest to the ground truth as the method explicitly encodes all the visual properties needed to fully reconstruct a scene. This aspect plays in its favour in this experiment.

Are object-centric representations more suitable than localized ones for this task?

We compare representations at similar memory regimes M_1 and M_2 in the VQA task. For the *exist* questions, *OBJ* features are outperforming *LOC* ones. This is expected and quantitatively indicates that structuring the scene into a set of objects is much more suitable for reasoning. When it comes to *comp num*, *query attr* and *comp attr*, Slot Attention performs significantly better compared to other methods. We argue this can be attributed to the object-centric nature of the representation that facilitates comparisons among objects and focuses on describing their properties. Note that, even though *OBJ* were trained on CLEVR contrary to *LOC* features, we tried fine-tuning DINO features on the CLEVR dataset but obtained worse results. We hypothesize they are not suited for synthetic datasets like CLEVR, which is smaller and much less diverse than standard large-scale SSL datasets. The experiments on CLEVR-Math reveal a similar tendency of object-centric representations performing better in M_1 and M_2 .

Do visual representations contain sufficient information for solving VQA?

In the more relaxed memory constraint setup M_2 , all the studied visual representations enable the reasoning module to learn to solve VQA to some extent, with Slot Attention clearly outperforming the rest. The gap between Slot Attention and localized features is less significant in the case of no memory restriction setup, with CLIP features yielding the overall best performance. This suggests that even if the visual information is not encoded in a way that can be heavily compressed, these features contain significant semantic information. Interestingly, the Raw baseline does not lead to significant accuracy, which validates that our setup does not allow the reasoning module to perform visual processing and also that all visual

representations extract some meaningful information from the image, albeit not in a compact way.

Does image-language pretraining yield better representations for visual reasoning?

Looking at the results with no memory restriction on both datasets, CLIP features perform the best among \mathcal{LOC} as well as \mathcal{OBJ} . However, we observe a significant CLIP’s performance drop compared to other visual features when given the memory restriction. This may indicate that text-image space in CLIP, through accommodating the text alignment, is less compact. We leave verification of this hypothesis for future work. Additionally, to check the influence of the language model, we run experiments with the original CLIP text encoder. The results in Tab. 6.4.5 indicate that using an aligned CLIP text encoder yields worse results compared to RoBERTa in all memory regimes and even when no memory restriction is given. This result holds for all types of questions. We attribute this observation to the fact that CLIP was trained on short textual prompts, whereas questions in CLEVR are complex and mainly built of multiple sentences, thus requiring long sequence modeling.

Method	M_1	M_2	M_∞
CLIP-16 + RoBERTa	53.2	74.8	91.3
CLIP-16 + CLIP Language Encoder	50.8	73.8	76.8

Table 6.4.5. The study of the influence of the text encoder on the CLIP performance on the CLEVR VQA dataset. We report average accuracy over all the questions in the validation set.

Finally, there is still a significant gap between off-the-shelf features and the state-of-the-art MDETR [222], which can obtain performances comparable to having perfect visual information.

6.4.4. Low-shot evaluation study

We study the effectiveness of visual features in solving VQA with a limited number of samples. The idea behind this experiment is to test the generalization capacity of the features as we argue that accurate and discriminative features should generalize better to unseen samples. Besides, having a small training set forbids the reasoning module to brute-force memorize a significant portion of the combinations of questions/scenes and answers and thus uncovers the minimum training set size needed to learn reasoning patterns. Fig. 6.4.1 shows a comparison for varying fractions of the training set for both the M_1 and M_2 setups.

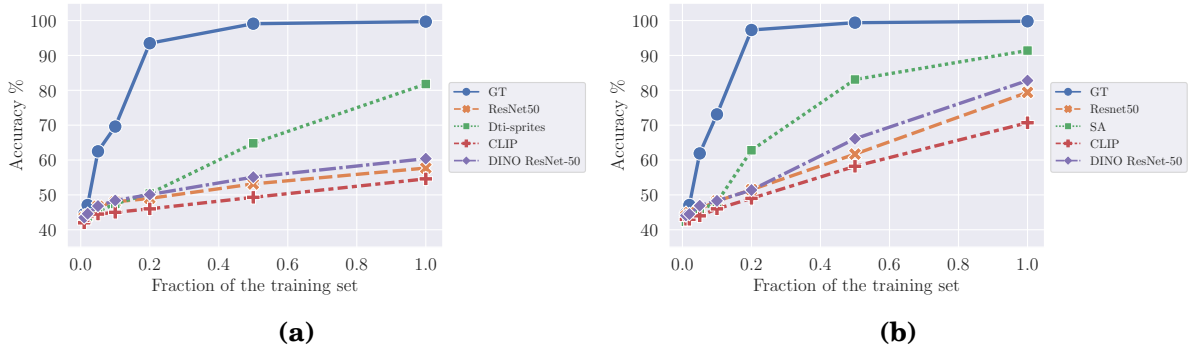


Figure 6.4.1. Results of the low-shot VQA experiments on CLEVR VQA dataset. We visualize the results for 2 memory size regimes, M_1 (top) M_2 (bottom), and the visual encoders with the best overall performance.

Are visual representations suited for low-shot reasoning?

We observe that starting at 20% of the full training set and with perfect visual information corresponding to the ground truth, it is possible to infer reasoning patterns. However, at this training set size, none of the considered visual representations enables visual reasoning, regardless of the memory constraint.

Which visual representations are better suited for learning to reason from a few examples?

Looking at the M_1 setup, we can see that DTI-Sprites is able to obtain higher accuracy with much fewer examples than the other methods, whereas all achieve comparable very low training losses. This indicates that explicit representations enable faster discovery of relevant information than implicit representations.

Do object-centric representations prevent learning spurious correlation?

As we can see from the M_2 results in Figure 6.4.1, both OBJ and LOC representations exhibit similar behavior when the memory is less constrained. Given that they all reach similar very high training accuracy, this indicates that structuring the representation into objects may not be as good at preventing learning spurious correlations as is the distinction between explicit and implicit representations.

6.5. Conclusions

We investigated to what extent off-the-shelf representations model the information necessary to perform visual reasoning. To that end, we designed a new feature evaluation protocol based on VQA, which aims at disentangling as much as

possible the vision from the reasoning part. Indeed, we constrained the computation capacity as well as the visual memory size to decouple representation learning from evaluation and make sure features are comparable. Using our evaluation protocol, we make three key findings: (i) off-the-shelf visual representations are far from being able to structure visual information in a compact manner, (ii) object-centric representations seem to be better at encoding the critical information necessary for reasoning, and (iii) limiting the training set size has a dramatic impact on the learning of spurious correlations for all visual representations regardless of their type.

While these findings contrast with the excellent performances that off-the-shelf features usually obtain in simpler vision tasks, they also show that having representations that encode object properties is a promising first step towards unsupervised visual reasoning. Finally, once scaled to a real-world setting, object-centric representations could provide a framework for learning generalizable visual features.

6.6. Appendix

6.6.1. Visual encoders extraction and adaptation details

Dense local features. We use features extracted from two popular architectures: convolutional-based ResNet-50 [36] and the transformer-based ViT proposed in [37]. For ResNet-50, we use the local features after the last convolutional layer right before the global average pooling. In contrast, for ViT, we use the output tokens of the previous layer corresponding to the image patch position (*i.e.*, the CLS token is not used). To encode the position of local features, we use the corresponding 2D position in the feature map for ResNet-50 features and the 2D position of the corresponding patch for ViT features.

For ResNet-50 pre-trained on ImageNet, we use model weights available in torch-vision package¹. For both DINO ResNet-50 and DINO ViT, we use weights provided in the original DINO repository². For CLIP features we use implementation³ from HuggingFace library [237].

Object-centric representations. To evaluate the performance of unsupervised object-centric representations, we consider two methods: Slot Attention [196] and DTI-Sprites [197], which demonstrated state-of-the-art segmentation results on the recent CLEVRText benchmark [198]. We first train both methods on the CLEVR

¹ <http://pytorch.org/vision/stable/index.html>

² <http://github.com/facebookresearch/dino>

³ <http://huggingface.co/transformers/clip>

dataset - which does not require any supervision - and use pre-trained models as feature extractors.

In Slot Attention, we use the slots right before the decoder part as features. We use the implementation available in CLEVRText benchmark repository ⁴. We train the model on the original CLEVR VQA dataset, preserving the original train/validation splits. To keep a similar resolution as in the case of the multi-object segmentation task, we feed input images resized to 120×160 . We do not apply any center cropping to make sure all the objects in the scenes are clearly visible. Following [198], we use 11 slots. We also maintain the original learning rate, batch size, and optimizer settings, as well as 500k iterations of training.

We use original implementation of DTI-Sprites ⁵. Similar to Slot Attention, we train the model on images resized to 120×160 with no center cropping. To account for the smaller resolution compared to the original multi-object segmentation task, we increase the representation expressiveness in the backbone by changing adaptive average pooling to 4×4 , instead of the originally proposed 2×2 . We train the model with 10 layers, corresponding to a maximum number of objects in the CLEVR dataset. We also increased the number of prototypes to 8 since we observed that the training did not lead to obtaining a complete set of prototypes in the dataset when using only 6. We train the model with the original batch size and learning rate for 760k iterations.

6.6.2. Memory adaptation details

To match the memory constraints in 2 setups for all visual encoders, we use PCA implementation in Scikit-learn [238]. We first extract features for training and validation sets and apply PCA offline.

6.6.3. Training details

We implement our framework in PyTorch [239]. For the text encoder, if not indicated otherwise, we use RoBERTa ⁶ model available in HuggingFace library [237]. In the case of all visual encoders, we follow the same training strategy. We train the reasoning module for 40 epochs with a batch size of 64. We use adamW [240] optimizer with a learning rate of $1e-4$, weight decay of $1e-4$, and linear warmup for the first 10k iterations. We then decrease the learning rate at epochs 30th and 35th by a factor of 10.

⁴ <http://github.com/karazijal/clevrtex>

⁵ <http://github.com/monniert/dti-sprites>

⁶ <http://huggingface.co/roberta-base>

6.6.4. DINO ViT architecture comparison

We provide the results for 2 different DINO ViT architectures (small - S, base - B) in Table 6.6.1 alongside the original feature size and a number of parameters of each model. We use 16 patch-size versions in both cases. We observe a significant performance decrease when using the larger model, even when no memory restriction is performed. When comparing the same architecture trained with CLIP (CLIP-16 in Table 6.4.3), we note that the reasoning module should be able to learn from a descriptor of such size. We, therefore, conclude that DINO-ViT-B features, which proved useful for object segmentation, are not as efficient in reasoning over scenes with multiple objects.

method	# params	feature size	M_1	M_2	M_∞
DINO ViT-S	21M	384	52.1	78.4	89.8
DINO ViT-B	85M	756	53.5	69.7	72.8

Table 6.6.1. DINO ViT performance comparison on CLEVR VQA dataset.

6.6.5. CLEVR-Math experiments

Here, we explain the modifications in our framework for CLEVR-Math dataset. CLEVR-Math is a CLEVR-like dataset that introduces a task of mathematical reasoning on multi-modal input. The questions concern only numerical reasoning and therefore require only *count* answers. To accommodate this task, we simply remove all the heads in our reasoning module while keeping the *count* head. We also apply the procedure of finding the restricted reasoning module, as described in 6.3.3 and find optimal hyperparameters for CLEVR-Math to be: $N_T = 2$, $N_H = 4$, $v_{dim} = 128$, at which with perfect visual information our reasoning module scores 99.9% overall accuracy.

7. Task adaptation in Real-world Scenarios

In our investigations so far we have demonstrated the adaptability of visual foundation models across different downstream tasks, reducing both human annotation requirements and computational overhead. We now examine the practical implications in real-world contexts. The final part of this thesis explores whether visual foundation models can successfully address complex real-world challenges.

One critical problem in real-world applications is content personalization. This is particularly evident in web platforms, where users expect tailored recommendations. The ability to tailor content to individual user preferences and behavioral patterns has evolved from a desirable feature to an essential component to maintain user engagement and satisfaction across digital services. This challenge is particularly relevant for Booking.com, an online travel agency that aggregates accommodation properties globally. A crucial aspect of the platform is the effective presentation of properties, where the goal is to optimize the information presented to users searching for accommodation options. This includes creating visual previews comprising selected images from larger collections—summaries that significantly influence users’ decision-making processes. We show examples of such previews in Fig. 7.0.1.

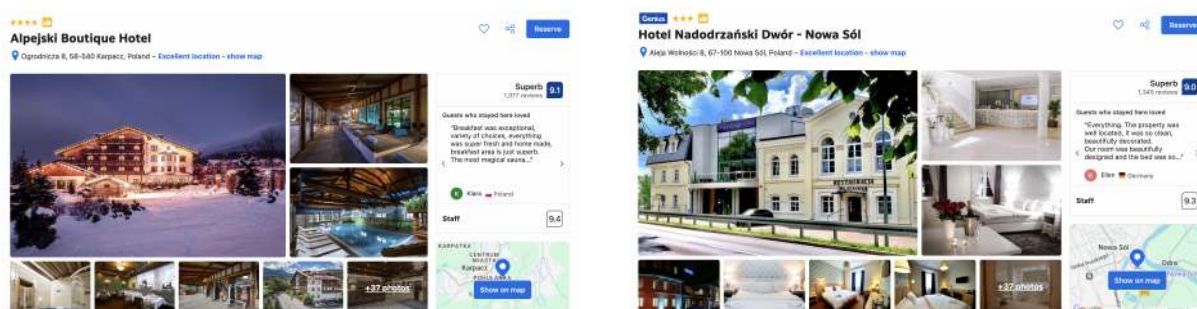


Figure 7.0.1. Examples of property pages on Booking.com

In this chapter, we discuss some practical aspects of adapting a Vision-Language Model to personalize image collection summarization. While traditional approaches to this task focus on creating concise visual summaries through subset selection, web platforms face an additional complexity: accommodating diverse user preferences and priorities when viewing identical content.

Gathering annotations for such a personalization task could be particularly costly. That is why, in our work, we seek solutions with the data available at hand to avoid a tedious annotation process. Booking.com users can rate properties and share their past experiences with hotels through textual reviews. We leverage this information to personalize summaries for future users of the platform. We do this by identifying key aspects that matter to particular user groups in properties and including those insights when selecting subsets of images to present. Our approach employs a proprietary VLM, which is a variant of CLIP, to establish connections between visual content and textual user feedback. As a result, our cross-modal strategy creates more relevant and personalized visual summaries without requiring additional manual annotations.

7.1. Introduction

Visual content is one of the key aspects when evaluating and deciding upon a place to stay on the Booking.com platform. Throughout their journey, platform users browse through visual content for four main reasons: (1) To get an accurate and realistic idea of what to expect, (2) To assess the quality of the property, (3) To build trust and remove doubts that they are making the right booking decision, (4) To look for a content that matches their travel intent. When looking for their next trip, users might be overwhelmed by the amount of information they are exposed to, both visual and textual. Image galleries can contain up to hundreds of images. Hence, we aim to focus the platform’s users on the visual content that is most relevant to them, given their current personal context. We achieve that by summarizing each property with a subset of visually informative, high-quality, segment-personalized images.

Most of the works in the image collection summarization area focus on a generic summarization problem, where the main objective is to select a *diverse* set of images. Only recently, some of the efforts have been made in a so-called *guided summarization* [241], where the aim is to get a diverse yet *representative* subset of images corresponding to a specific query. In our work, personalization can be seen as a variant of a query-based approach. However, the queries are not explicit. User intents can not easily be translated into specific queries, making *personalized image gallery summarization* more complex. In this work, inspired by recent advances in multimodal learning, we solve the above-mentioned challenge and develop a method for personalized image collection summarization with textual guidance. We focus on entire groups of users, which we further refer to as **user segments**. We leverage millions of reviews corresponding to properties on the Booking.com platform for enhanced user segment personalization. We do that by extracting key topics

mentioned in the reviews, and since they significantly differ across segments of users (see Fig. 7.1.1) we adjust the image selection accordingly. As the main challenges of our task, we identify the following:

Personalization modeling. It is not apparent how to obtain the personalization data for our task and avoid costly annotations. Therefore, we focus on the entire segments of users and leverage the textual reviews with the metadata available on the Booking.com platform.

User intents extraction from reviews. Reviews available on the platform can be a very rich source of information. Users of the platform can help future travellers to make the best choices by sharing their experiences. However, in practice, the reviews tend to be very noisy. Therefore, an essential aspect of our approach is to extract relevant pieces of information for a visual summary from free-form text. Thus, the extracted signal has to be adequate for matching the semantic content of images in a gallery.

Matching text and images. Finally, matching text with images requires representing all of them in a joint multi-modal space. There has been a recent surge of methods that leverage free-form text to reduce the need for costly annotations [242, 243, 244] yielding better performances when cast into multi-modal problems [245, 246, 247, 248, 249, 250], or obtaining better image representations [251, 13]. Most importantly, CLIP [13] shows that image-text large-scale pretraining gives the ability to learn the generalizable image representations and enables zero-shot image-text matching, which we leverage in our approach.

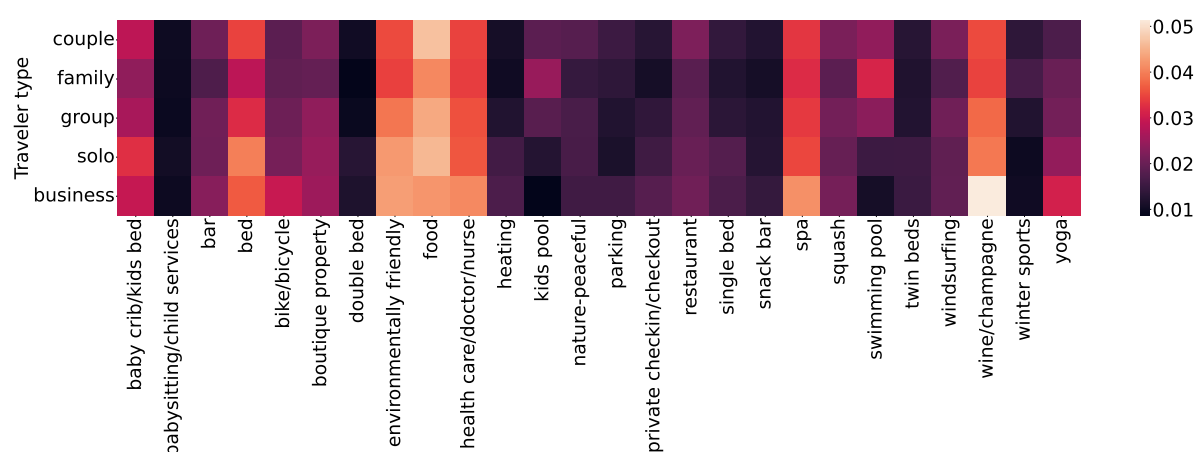


Figure 7.1.1. Heatmap of most popular topics (x-axis) extracted from reviews for different traveller types at Booking.com. We note that the ranking of the most mentioned topics differs among traveller segments, making reviews a valuable signal for user segment personalization.

The main contribution of our work is as follows:

- We introduce an unsupervised method for image collection summarization personalized for entire segments of users using textual guidance extracted from reviews. Our approach leverages text and image representations in multi-modal space.
- We extend previously introduced evaluation metrics for image collection summarization to the segment-personalization use case when no ground-truth annotations are given.
- In our experiments, we conduct human perceptual studies alongside the quantitative evaluation using our proposed metrics and show that reviews provide an adequate signal for segment personalization.

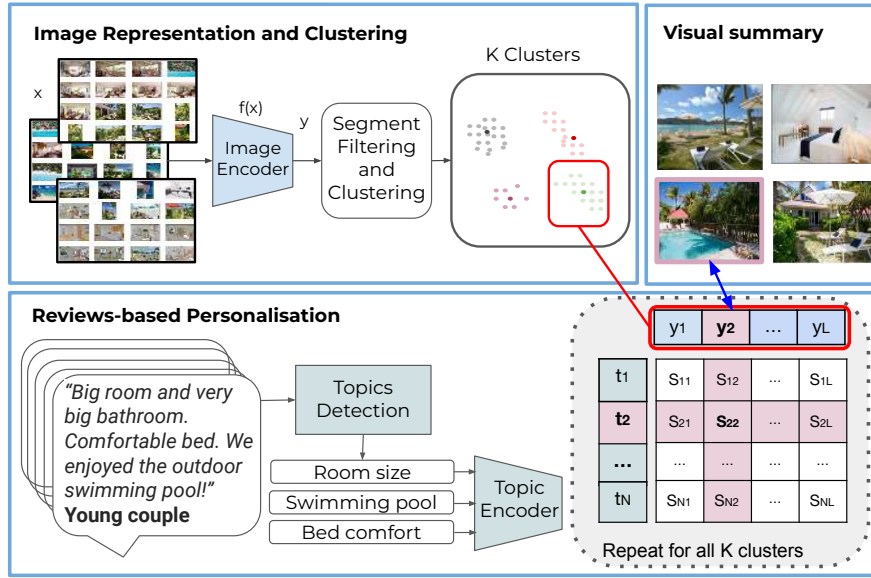


Figure 7.1.2. Overview of our method. First, we extract image embeddings and cluster them to obtain K ($K=4$ in this case) semantically separated groups of images. Then, for each cluster, we calculate the similarities between all the images within the cluster and the topics extracted from reviews of the specific segment u (here $u=Couple$). Finally, the selected images are the ones with the highest similarity to any of the topics.

7.2. Method

In this section, we define a task of personalized image collection summarization for segments of users and describe in detail all the building blocks of *CrossSummarizer*. We also provide the metrics which we use to evaluate our method quantitatively. We explain how we adapt standard metrics for image collection summarization to our segment-personalized use case.

7.2.1. Task Definition

Given a collection of images G , the goal of our method is to select a subset of K images $G_s \subset G$ that best corresponds to a given user segment $u \in U$. We consider two types of user segments:

- Traveller types: *Solo, Couple, Group, Family, Business*
- Trip types: *Beach, Ski, City, Nature Active, Nature Peaceful*

Our personalized summary aims to cover the essential aspects of the whole image gallery while selecting images that show details relevant to the specified user segment u .

Each of the images in G has a corresponding set of visual classes $c \in C$, and for each user segment u , we manually define a subset of $C_u \subset C$ of relevant classes (for details on how we define C see Appendix 7.6.1). Moreover, we have access to each property’s textual reviews, labelled with the corresponding segment u .

However, reviews can mention many aspects in just a short piece of text (see Fig. 7.1.2). Therefore, we represent each review as a set of topics T it covers. Below we explain all of the steps of our method in detail.

7.2.2. Image Embeddings & Filtering

Let $f(x_i) \rightarrow y_i \in \mathbb{R}^D$ be a feature representation of an image $x_i \in G$. In this work, we use a recently proposed MuMIC [243] for image representation $f(x)$, which extends the image-text contrastive pretraining proposed in CLIP [13] to a multi-label case. MuMIC learns a multi-modal embedding space by jointly training an image encoder and text encoder to maximize the cosine similarity of the image and label-text embeddings of real labels. The model applies tempered sigmoid-based Binary Cross-Entropy (BCE) loss on each class and then mean-reduces it (see Eq. 7.1), optimizing across all classes. Given a batch of images $\{x_i \in G, i = 1..N\}$ and their associated ground-truth multi-label vector $\{\hat{w}_i \in \mathbb{R}^{|C|}, i = 1..N\}$

$$\begin{aligned} \ell_{BCE} = & -\frac{1}{N|C|} \sum_{i=1}^N \sum_{j=1}^{|C|} (p_j \hat{w}_{ij}) \cdot \log \sigma(w_{ij}) \\ & + (1 - \hat{w}_{ij}) \cdot \log(1 - \sigma(w_{ij})) \end{aligned} \quad (7.1)$$

where $\sigma(\cdot)$ is the Tempered Sigmoid function; w_{ij} are the original output logits (before applying temperature scaling), which is the pairwise image-text cosine similarity between image x_i and class j ; and p_j is the positive sample weight of class j . A higher p_j indicates that positive samples are given greater weight, increasing the penalty for identifying false negatives.

We use MuMIC to both extract image embedding as well as to obtain classes C per each image $x \in G$. We then filter out images that are not relevant to given u by keeping only the subset of images with at least one class $c \in C_u$.

7.2.3. Clustering

For the clustering phase we use *KMedoids* algorithm [252] implementation ¹. We choose *KMedoids* as it is robust to outliers and gives high flexibility in choosing K .

We run clustering on the image embeddings using *Cosine Similarity (CosSim)* as a similarity metric, such that

$$\text{CosSim}(u, v) = \left\langle \frac{u}{\|u\|}, \frac{v}{\|v\|} \right\rangle, \quad \text{with } u, v \in \mathbb{R}^D \quad (7.2)$$

where $\langle \cdot, \cdot \rangle$ is the inner product in \mathbb{R}^D .

Finally, we obtain resulting cluster assignments for each of the images.

7.2.4. Text2topic: Topics Detection Model

We use a recently proposed Text2Topic [253] model, a topic detection model, to extract user segment preferences and personalize the subset of images based on their topics of interest. We first filter the reviews by u and then detect the topics associated with these reviews. To get the topics, we train a classification model with 45 travel-domain topics (see the subset of the topics in the heatmap in Fig. 7.1.1) using cross-encoder transformer-based architecture [254], which relies on BERT [43]. We train the model with 15,663 positive pairs of reviews and topics and sample X5 negative topics per review. We use the Binary Cross Entropy loss function on the [CLS] embedding vector of the crossed input [REVIEW] [SEP] [TOPIC] to get the probability the topic is mentioned in the review.

At inference, we run the model on pairs of reviews and each of the 45 topics to get the probability scores. Finally, we get the topics T with a probability greater than 0.5 to match with the review.

7.2.5. Matching Images to Topics

Having obtained clusters of images and a list of topics T for given u , $T_u \subset T$ we select the final subset of images G_s . We do it by first computing the *confidence matrix*

¹ <https://github.com/scikit-learn-contrib/scikit-learn-extra>

Algorithm 2: Pseudocode for selecting images

Result: G_s - selected images from the gallery

Inputs: A_K - Image-to-cluster assignment $x \in G \mapsto \{1..K\}$

S - Confidence matrix from Eq. (7.3)

procedure SelectImages(L_k, S)

```
 $G_s \leftarrow \emptyset$  // Selected images
 $\Omega_T \leftarrow \{1..|T|\}$  // Indices of active topics
for Cluster  $k \in \{1..K\}$  do
    // Image indices for cluster k
     $\Omega_k \leftarrow \{j : A_K(x_j) = k, \forall j \in \{1..|G|\}\}$ 

    // Compute best matches within k
     $i^*, j^* \leftarrow \underset{(i,j) \in \Omega_T \times \Omega_k}{\operatorname{argmax}} S_{ij}$ 

     $\Omega_T \leftarrow \Omega_T \setminus \{i^*\}$  // Update active topics
     $G_s \leftarrow G_s \cup \{x_{j^*}\}$  // Add selected image
return  $G_s$ 
```

$S \in \mathbb{R}^2$ of images G being aligned with topics T_u . We follow [243] and use tempered sigmoid, formulated as:

$$S_{ij} = \sigma(\exp(\gamma) \cdot \langle t_i, y_j \rangle) \quad (7.3)$$

where γ is the log-parameterized multiplicative scalar, and $t_i, y_j \in \mathbb{R}^D$ are respectively a topic from T_u and a feature representation $f(x_j)$ of an image x_j .

We then select the final images by iterating over clusters and selecting the pair (t_i, y_j) with the highest similarity within a cluster of representations. The pseudo-code for this selection is given in the Algorithm 2.

7.2.6. Evaluation Metrics

To ensure our generated summaries are diverse and adequately correspond to user segments' interests, we define multiple evaluation metrics. Following [255], we use *Coverage*, *Representativeness* and *Diversity* metrics. However, we apply some modifications to match our use case. More specifically, we make sure each one of the metrics is normalized across samples. Our motivation lies in a large variety of galleries across properties regarding image redundancy and coverage of relevant user segment aspects within images.

Diversity. Let us denote $d(x_i, x_j)$ as a distance between images x_i and x_j . The *Diversity (Div)* metric measures to what extent the diversity in terms of distance

between embeddings of images in G_s is similar to the one in the original gallery G . We define Div as:

$$Div = \frac{\max_{(x_i, x_j) \in G_s \times G_s} d(x_i, x_j)}{\max_{(x_l, x_m) \in G \times G} d(x_l, x_m)}, \quad (7.4)$$

where $d(\cdot, \cdot)$ is computed in image embedding space as:

$$d(x_i, x_j) = 1 - \text{CosSim}(f(x_i), f(x_j)). \quad (7.5)$$

Split	Per property statistics		Totals		
	#reviews	#images	#properties	#reviews	#images
Small	between 30 and 150	between 50 and 100	3230	252k	232k
Big	151 <	100 <	3151	1.5M	458k

Table 7.2.1. Dataset split in detail. Overall we collected more than 6000 samples from the platform. Through stratified sampling, we make sure the distribution in terms of location, rating, and accommodation type reflects the real distribution.

Representativeness. Then, let us denote $\mu_G, \mu_{G_s} \in \mathbb{R}^D$ as the mean vectors of the original gallery representations and the selected subset, respectively. *Representativeness* (*Repr*) is defined as:

$$Repr = \text{CosSim}(\mu_G, \mu_{G_s}) \quad (7.6)$$

With the generated summaries, we wish both vectors to be similar in representation, such that their CosineSimilarity is close to 1.

Coverage. We also measure *Coverage* (*Cov*) in the semantic space by using classes associated with images in G . We adopt the *Probabilistic coverage* suggested in [255] and use the probabilities $P(c|x) \in \mathbb{R}$ for a given image x to represent a particular class $c \in C_u$.

$$Cov = \frac{1}{|C_u|} \sum_{c \in C_u} \frac{P_{G_s}(c)}{P_G(c)}, \quad (7.7)$$

where $P_G(c) = \max_{x \in G} P(c|x)$. Note that in our work, we use the MuMIC model to obtain probabilities since it was trained in-domain. However, this could be any off-the-shelf multi-label image classification method.

With our *Coverage* metric, we mainly focus on measuring the accuracy of the person-

alization step. Therefore, instead of taking all of the classes in C , we only consider the ones corresponding to a given user segment u , C_u .

Reviews Coverage. Finally, to measure how well the generated summaries correspond to user segment topics from reviews, we calculate the topics coverage of selected images. Similarly to *Coverage*, we use the confidence matrix $S \in \mathbb{R}^2$ as probabilities of a topic t_j being aligned with the set of selected images G_s . *Reviews Coverage (RCov)* is therefore given by:

$$RCov = \frac{1}{|T_u|} \sum_{i=1}^{|T_u|} \frac{\max_{j \in \Omega_{G_s}} S_{ij}}{\max_{j \in \Omega} S_{ij}}, \quad (7.8)$$

where $\Omega = \{1..|G|\}$ and $\Omega_{G_s} = \{j : x_j \in G_s, \forall j \in \Omega\}$ are respectively the range of indices of the images in G and the subset of indices of images that are only in $G_s \subset G$.

7.3. Experiments

In this section, we provide the experimental results obtained through both offline evaluation and user studies conducted internally at Booking.com. We also describe our experimental setup, including details on the dataset collected at the Booking.com platform and baseline models we compare our CrossSummarizer against.

7.3.1. Experimental Setup

Dataset. We conduct our experiments on the Booking.com dataset consisting of properties. We collected over 6000 real properties from the platform by carefully curating the sampling to adequately represent a real distribution in terms of geographical location, types of accommodation as well as travellers' experience with a given property. We do it by applying a stratified sampling technique with a country, type of accommodation, and property rating being the factors. We do not share our dataset, however, we note that the examples can easily be downloaded (both images and reviews) since the data is publicly available on the Booking.com site.

Each of the samples in our dataset consists of a set of uploaded by property owners images, which correspond to a gallery, and a set of reviews of past travellers' experiences. Alongside sampled reviews, we also include metadata about the type of traveller that authored a particular review.

Since samples in our dataset significantly vary in amounts of both images and reviews, we split the dataset into two groups according to the size of galleries and the number of reviews. Precisely:

- **Small** - properties with the size of a gallery between 50 and 100 photos and 30 - 150 reviews,
- **Big** - gallery sizes of $100 < \text{photos}$ and $150 < \text{reviews}$.

More details on precise numbers of our dataset split are given in Tab. 7.2.1. In our experiment, we report results separately for the two aforementioned splits.

Baselines. To the best of our knowledge, none of the proposed methods for *image collection summarization* such as [241, 256] nor the *multimodal* ones [249, 247, 257] tackle the *personalization case*. For evaluation purposes, we implement multiple baselines, which we describe in detail below.

Default Clustering (Def) - we compare against a simple no-personalization approach by running clustering on image embeddings and omitting the filtering step. The selected K images are resulting cluster centres. A similar approach was proposed in [258] for videos.

Clustering with personalization (Clust-W/P) - we implement the approach without topic-based refinement and select cluster centres as the summarization. The approach differs from the *Clustering* setting by an additional filtering phase based on relevance to the particular user segment classes being associated with images in the gallery (see Sec. 7.2.2).

Topic-based personalization (TopicBased) - we also implement the approach based only on the topic’s similarity with images, without a clustering step. We extract image and topic embeddings and apply user segment filtering. We calculate the similarity matrix between all the image and topic embeddings and choose top K scores in the matrix. We make sure each image gets selected only once and apply the selection process iteratively. The pseudo-code is similar to the final approach, as we simply take argmax on the entire confidence matrix S , which is being decreased with each iteration by selected images’ columns.

Note that for all of the approaches, we use the same image representations obtained with MuMIC, which was trained on Booking.com multi-label classification dataset as described in [243]. For the filtering phase, we create a mapping of MuMIC’s classes for each user segment beforehand and use the mapping at inference time.

7.3.2. Offline Evaluation

Evaluation protocol We first run the evaluation procedure offline internally at Booking.com. For all the methods relying on K Medoids algorithm, we set the same

random seed for a fair comparison. We run experiments with a fixed $K = 9$ corresponding to the current Booking.com setting.

	Small				Big			
Method	<i>Div</i> ↑	<i>Repr</i> ↑	<i>Cov</i> ↑	<i>RCov</i> ↑	<i>Div</i> ↑	<i>Repr</i> ↑	<i>Cov</i> ↑	<i>RCov</i> ↑
<i>No Personalization</i>								
Default Clustering [258]	0.947	0.931	0.516	0.581	0.925	0.929	0.430	0.488
<i>Personalization</i>								
Clustering with Personalization	0.903	0.931	0.578	0.656	0.932	0.928	0.487	0.521
Topic-Based Personalization	0.903	0.733	0.365	0.649	0.874	0.775	0.261	0.707
CrossSummarizer (Cross-modal)	0.963	0.903	0.617	0.729	0.950	0.885	0.524	0.677

Table 7.3.1. Quantitative evaluation. We report the results for two different dataset splits: small and big galleries and the number of reviews.

Quantitative results The results of our experiments are presented in Tab. 7.3.1. Looking at *Cov* and *RCov*, we observe that our cross-modal approach outperforms the baselines in the segment personalization task, especially for the **Small** dataset split. Compared to the Def setting, we observe approximately 0.1 gains in *Cov* for both dataset splits and a significant improvement over *RCov*. For the **Big** dataset split, we report a higher *RCov* for the TopicBased approach. This is expected since the approach is based on maximizing the similarity between topics and images, which corresponds to the Eq. 7.8.

Moreover, our cross-modal approach outperforms the rest of the methods in terms of *Diversity*, indicating that the personalization step produces more diverse summarization.

Relying only on reviews, which corresponds to the results of a Topic-based method, gives a significantly lower *Representativeness* and poor visual user segment *Coverage*. *Coverage* gain of our CrossSummarizer over the Clust-W/P approach also emphasizes the need for using two modalities in our task. Overall, we observe a clear trade-off between *Diversity* and *Representativeness*. CrossSummarizer finds a sweet spot between the two, giving an excellent segment personalization result at the same time.

Qualitative results. Alongside the quantitative evaluation, we also provide some qualitative results. Fig. 7.4.1 shows the visual comparison between the two baseline approaches and our method. The example shows a non-personalized Def result and the personalized results obtained with Clust-W/P and CrossSummarizer approach for *Ski* trip type and $K = 8$.

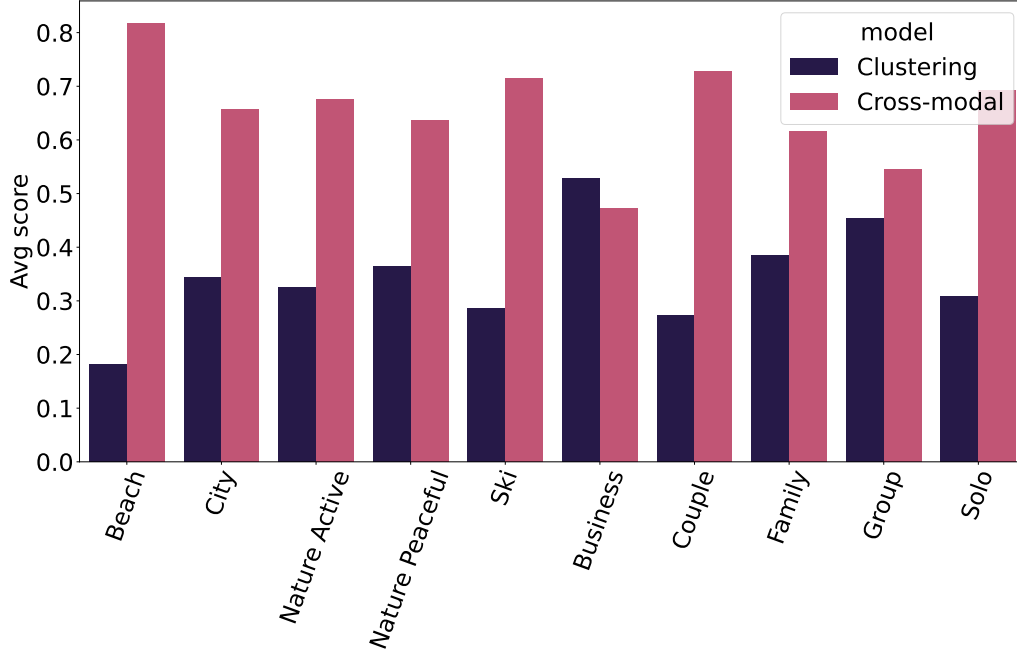


Figure 7.3.1. Results of our user studies. We report the results separately for each user segment (x-axis).

All of the approaches give a set of diverse images without any redundant photos. However, our cross-modal model provides the best personalization result. We highlight the photos that are relevant to the user segment: a picture of a skier and a photo of a sauna. We can also see that contrary to the Clust-W/P approach, with our CrossSummarizer, the selected image of a property from outside (bottom right) was taken in the wintertime (expected for *Ski* trip type).

7.3.3. User Studies

Experimental setup. In addition to offline evaluation, we conduct human perceptual studies in the form of an anonymized paired test. We compare two methods: Clust-W/P approach and CrossSummarizer. We manually select 210 samples of properties with galleries that contain photos relevant to the specified user segment. We split the samples across 5 participants. The samples are uniformly distributed over the dataset split (**Small, Big**) and user segment types.

The participants were asked to answer the question: *Which of the models (A or B) gives a better summary for a given user segment?* We assign a score of 1 for a model that performs better than the other and 0 otherwise. We also allow for a 0.5 score in case of a tie.

Results. The conducted user studies indicate the superiority of our CrossSummarizer approach with an average score of 0.66 ± 0.38 for all user segments over

Clust-W/P 0.34 ± 0.38 . Additionally, in Fig. 7.3.1, we provide detailed results for each user segment. We observe that CrossSummarizer obtains higher average scores for most of the segments in our experiments, except for only *Business* traveller type. To confirm the statistical significance of our studies, we perform the paired T-test [259] with a null hypothesis of *average scores for models A and B being equal* and the alternative hypothesis of *model A scoring lower than model B*. The results ($t(209) = -5.537; p < 5e - 8$) let us reject the null hypothesis and accept the alternative hypothesis.

7.4. Application

This section covers some practical aspects of our approach and how it is leveraged at Booking.com. Two essential parts are first run offline, which are the Text2Topic model for topic extraction and the MuMIC model for image embedding extraction and image multi-class annotations. The results are then stored in the database and accessed at runtime. It takes approximately 170 ms for the MuMIC model to run on a batch of 100 images and 244 ms for the Text2Topic model to run prediction on a batch of 100 reviews. We leverage GPU computation for this purpose.

7.4.1. Deployment & Maintenance

The model is served with Amazon SageMaker and deployed on the Booking.com Content Intelligence Platform (CIP) [243]. CIP is a stream processing platform based on Apache Flink. It consumes real-time events from Kafka topics (e.g. images uploaded by Booking.com partners) and generates model-based predictions. The same architectural design allows CIP to be also used for backfilling purposes. Backfilling refers to the enrichment of historical data with newly deployed model predictions. We leverage the mentioned design by simulating historical data events and pushing them to Kafka.

CIP is designed to achieve high prediction throughput while keeping a low latency. It achieves that by leveraging Apache Flink’s asynchronous I/O operator to perform concurrent asynchronous HTTP calls to a model endpoint. However, this optimization mechanism relies on the assumption that each model prediction can be made independently. This assumption does not hold for summarization models where a group of events should be sent to the model together in a single prediction query.

We built upon Apache Flink’s windowing mechanism to implement the grouping of events that should be sent to the model endpoint. Whenever a new call to the

model should be triggered, an event containing the request metadata (e.g. a hotel id) is sent to Kafka. Then, the matching image data is fetched by issuing calls to an external service holding image data. Those calls are executed independently and concurrently. Following that, images are grouped in the same window. Images are accumulated within the window as soon as they are fetched, and the window is closed after a predefined period (e.g. 3 seconds). Then, the images are sent together to the model endpoint for prediction.

7.4.2. Application example

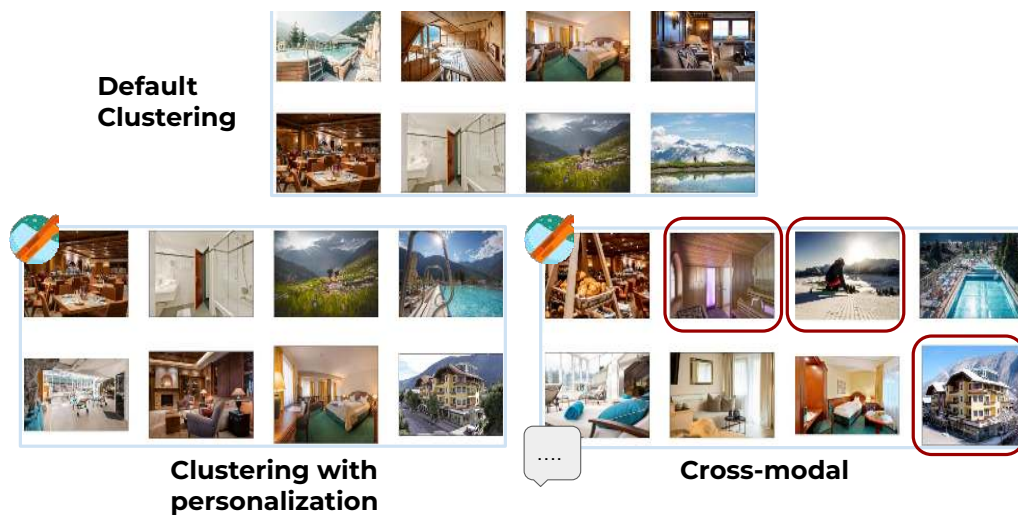


Figure 7.4.1. Example results of summaries of $K = 8$ images for one of the properties. We compare Def (top), and personalized methods (bottom) on this property’s visual summarization. The presented personalization examples here are for a *Ski* trip type. We highlight in red the selected images which are relevant to this user segment.

The applications of our image collection summarization approach at Booking.com are three-fold:

- Image subset selection optimization for large collections of images when given a constraint on a number of images or smaller displays.
- Visual content personalization based on traveler type.
- Visual content personalization based on trip type.

Fig. 7.4.1 shows some qualitative results of our CrossSummarizer for the third bullet point. Using this personalization approach, we expect to reduce the friction from the decision-making process as users will have a better understanding of the property at an earlier step.

Our proposed model is currently under experimentation. When deploying the personalized CrossSummarizer model, we compare it with the current model, which

is produced by the Def model, through A/B test experimentation on the CTR (click to ratio) metric.

We also note that personalizing summaries of collections of images is a relevant task for most e-commerce websites. Hence, our method could also be applied to any other personalization task, such as product recommendations where a multi-modal input is available (reviews + images).

7.5. Conclusions & Limitations

We presented a method for personalized image collection summarization for entire segments of users. Our approach is capable of taking into account users' intents when producing summaries of large image collections. As the personalization signal, we use other travelers' experiences with properties, which we extract from the reviews. We implemented and tested our method on the Booking.com platform and our experiments, including human perceptual study, indicate that our proposed approach yields good results on the diversity and representativeness axes. The comparison with other baselines indicates that our proposed method performs the best in terms of personalization. Future works include A/B tests of our model in a production environment to measure the real-world impact.

The main limitation of our method is handling samples that are considered a *cold-start* zone, e.g. having a limited number of reviews but complete image galleries. This, however, can be addressed by leveraging information from other properties of a similar profile, which we leave for future work.

7.6. Additional material

7.6.1. Definition of relevant image classes

We provide more details on how we defined visual classes for each of the user segments considered in this work. We first obtain a list of relevant classes in the Booking.com context from domain experts, i.e. product managers who, based on the statistics of the content of image galleries, decided upon the most frequent classes, leading to a dataset with 120 classes. Having obtained the list of visual classes, we then ask 3 independent experts to associate classes with users. In Tab. 7.6.1, we present exemplary classes for each considered segment.

User Segment	Image classes
Solo	Bar, Entertainment Center, Lobby or reception, TV/Multimedia, Public transport
Couple	Private dining area, Bed, Sea view, Fireplace, Bathtub, Jacuzzi/Hot tub
Family	Aqua park, BBQ facilities, Children playground (outdoors), Dishwasher, Game room
Group	Buffet, Kitchen or kitchenette, Billiard, Water sport, Seating area (sofa, living room, etc.)
Business	Business traveler, Business/Conference Room, Desk (for work), Lobby or reception, Ironing facilities
Beach	Beach, Sea view, Sun umbrella, Sunbed, Sunset
Ski	Heating (not air conditioner), Mountain view, Winter, Fireplace, Skiing
Nature Peaceful	Mountain view, River/Lake view, Natural landscape, Yoga, Garden/garden view
Nature Active	Yoga, Cycling/Biking, Horse-riding, Birds eye view, Water sport
City	Landmark (attractions, sightseeing), Neighborhood/Street, City view, Car, Public transport

Table 7.6.1. Exemplary image classes per user segment.

8. Final Remarks

In this chapter, we summarize the contributions presented in this thesis and give an overview of some of the open problems and challenges in adapting visual foundation models.

8.1. Summary of contributions

In this thesis, we have explored how pre-trained image-text aligned visual foundation models, or VLMs, can be effectively utilized for downstream tasks with minimal additional supervision. We presented five key contributions: (1) CLIP-DIY, which extends CLIP to open- vocabulary semantic segmentation through modified inference rather than retraining; (2) CLIP- DINOiser, which enhances CLIP representations with complementary model-obtained localization priors via a lightweight adaptation module; (3) an analysis of how textual prompts and pre-training data distribution understanding can improve downstream performance on the example of open-vocabulary semantic segmentation; (4) an evaluation framework using Visual Question Answering to determine which foundation models best suit specific applications; and (5) a real-world application demonstrating personalized image collection summarization using a VLM’s multimodal capability with minimal annotations. Together, these findings demonstrate promising directions for adapting visual foundation models to complex visual understanding tasks efficiently while minimizing computational and annotation requirements.

8.2. Open problems

Understanding pre-training datasets. As discussed in Sec. 2.1.3, data curation proved to be important in developing stronger visual foundation models. Chapter 5 extends this insight by demonstrating how analyzing concept distribution patterns within a VLM’s pre-training dataset can address specific challenges in open-vocabulary semantic segmentation. However, our contribution merely initiates a deeper investigation into the complex cross-modal alignment between textual concepts and visual patch features. Future research directions include exploration

of hierarchical relationships between concepts in the CLIP representation space, including synonymous terms and part-whole (meronymic) relationships, which could substantially enhance open-vocabulary method performance while providing deeper insights into the representation space. Furthermore, developing an evaluation methodology that quantifies the effectiveness with which patch-level representations maintain concept relationships and hierarchical structures represents a promising avenue for research. Such an assessment framework could serve as a valuable tool for guiding the development of improved CLIP models.

Leveraging the complementarity of different paradigms. In Chapter 3 and Chapter 4, we address open-vocabulary semantic segmentation by exploiting the complementary strengths of self-supervised features and image-text-aligned representations. Recent research work has explored hybrid foundation models that combine contrastive image-text pre-training with self-supervision methodologies [60, 59, 62]. While these hybrid approaches demonstrate enhanced feature quality when evaluated on fine-grained downstream tasks with full fine-tuning in certain setups, their effectiveness in zero-shot scenarios remains less definitive. Current evidence indicates that zero-shot evaluations of patch representations for open-vocabulary semantic segmentation using these hybrid approaches have not yet surpassed the performance of, e.g., CLIP-DINOiser, which utilizes both models, CLIP and DINO, trained separately with their respective objectives. We believe that ongoing evaluation of feature performance for fine-grained tasks, including the contributions presented in this work, will be valuable in guiding the future development of more sophisticated and powerful hybrid foundation models.

Fair evaluation protocols. In Chapter 6, we developed an adaptation protocol for Visual Question Answering that establishes a controlled evaluation environment for comparative analysis of visual representation models. As VQA inherently requires comprehensive supervision, our methodological framework incorporated fine-tuning with a full set of image-question-answer triplets. The design of rigorous evaluation protocols—particularly those that involve parametric optimization—necessitates careful consideration of multiple experimental variables to ensure methodological validity. The memory constraint mechanism implemented in our experimental design represents one approach to standardizing the dimensionality of the representation, thereby facilitating direct comparability. A more comprehensive evaluation paradigm would additionally analyze representations extracted from various layers or parts of a foundation model, similar to our ablative study in Sec. 4.6.2. Significantly, the assumption of training procedure invariance across representation architectures may be methodologically unsound, as different visual encoders may

exhibit optimal performance under distinct optimization regimes. This observation suggests that standardized evaluation frameworks should potentially incorporate architecture-specific training hyperparameter optimization to assess each model’s true representational capacity.

Real-world challenges. In Chapter 7, we demonstrate the practical application of visual foundation models to real-world challenges through our work on personalized visual summaries. It is important to note that the domain of hotel imagery examined in this study closely aligns with the images sourced from the Internet used to train CLIP, thus representing a relatively favorable transfer scenario. This thesis does not explore adaptation techniques for more specialized visual domains such as medical imaging, biological microscopy, or satellite-based earth monitoring, areas that present substantially different visual distributions from CLIP’s training data. Although we hypothesize that several methodologies introduced in our research could potentially extend to these specialized domains, a systematic investigation of such cross-domain adaptation strategies remains a promising direction for future research.

Bibliography

- [1] A. Torralba, P. Isola, and W.T. Freeman. *Foundations of Computer Vision*. Adaptive Computation and Machine Learning series. MIT Press, 2024.
- [2] Andre Esteva, Katherine Chou, Serena Yeung, Nikhil Naik, Ali Madani, Ali Mottaghi, Yun Liu, Eric Topol, Jeff Dean, and Richard Socher. Deep learning-enabled medical computer vision. *npj Digital Medicine*, 2021.
- [3] Claudine Badue, Rânik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius B Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago M Paixao, Filipe Mutz, et al. Self-driving cars: A survey. *Expert systems with applications*, 2021.
- [4] Roberto Gozalo-Brizuela and Eduardo C Garrido-Merchán. A survey of generative ai applications. *arXiv preprint arXiv:2306.02781*, 2023.
- [5] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.
- [6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. *CVPR*, 2009.
- [7] David G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004.
- [8] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *CVPR*, 2005.
- [9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *NeurIPS*, 2012.
- [10] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CVPR*, 2015.
- [11] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015.
- [12] Zihao Zhao, Yuxiao Liu, Han Wu, Yonghao Li, Sheng Wang, Lin Teng, Disheng Liu, Zhiming Cui, Qian Wang, and Dinggang Shen. Clip in medical imaging: A comprehensive survey. *arXiv preprint arXiv:2312.07353*, 2023.
- [13] Alec Radford, Jong Wook Kim, Chris Hallacy, A. Ramesh, Gabriel Goh, Sand-

- hini Agarwal, Girish Sastry, Amanda Askill, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.
- [14] Jianzong Wu, Xiangtai Li, Shilin Xu, Haobo Yuan, Henghui Ding, Yibo Yang, Xia Li, Jiangning Zhang, Yunhai Tong, Xudong Jiang, et al. Towards open vocabulary learning: A survey. *T-PAMI*, 2024.
 - [15] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.
 - [16] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICLR*, 2021.
 - [17] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*, 2022.
 - [18] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. *NeurIPS*, 2021.
 - [19] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Re-visiting unreasonable effectiveness of data in deep learning era. In *ICCV*, 2017.
 - [20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.
 - [21] Xinlei Chen*, Saining Xie*, and Kaiming He. An empirical study of training self-supervised vision transformers. In *ICCV*, 2021.
 - [22] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *NeurIPS*, 2020.
 - [23] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.
 - [24] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, 2021.
 - [25] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of

- large language models. *ICLR*, 2022.
- [26] Reece Shuttleworth, Jacob Andreas, Antonio Torralba, and Pratyusha Sharma. Lora vs full fine-tuning: An illusion of equivalence. *arXiv preprint arXiv:2410.21228*, 2024.
 - [27] Bjoern H. Menze, Andras Jakab, Stefan Bauer, Jayashree Kalpathy-Cramer, Keyvan Farahani, Justin Kirby, Yuliya Burren, Nicole Porz, Johannes Slotboom, Roland Wiest, Levente Lenczi, Elizabeth Gerstner, Marc-André Weber, Tal Arbel, Brian B. Avants, Nicholas Ayache, Patricia Buendia, D. Louis Collins, Nicolas Cordier, Jason J. Corso, Antonio Criminisi, Tilak Das, Hervé Delingette, Çağatay Demiralp, Christopher R. Durst, Michel Dojat, Senan Doyle, Joana Festa, Florence Forbes, Ezequiel Geremia, Ben Glocker, Polina Golland, Xiaotao Guo, Andac Hamamci, Khan M. Iftekharuddin, Raj Jena, Nigel M. John, Ender Konukoglu, Danial Lashkari, José António Mariz, Raphael Meier, Sérgio Pereira, Doina Precup, Stephen J. Price, Tammy Riklin Raviv, Syed M. S. Reza, Michael Ryan, Duygu Sarikaya, Lawrence Schwartz, Hoo-Chang Shin, Jamie Shotton, Carlos A. Silva, Nuno Sousa, Nagesh K. Subbanna, Gabor Szekely, Thomas J. Taylor, Owen M. Thomas, Nicholas J. Tustison, Gozde Unal, Flor Vasseur, Max Wintermark, Dong Hye Ye, Liang Zhao, Binsheng Zhao, Darko Zikic, Marcel Prastawa, Mauricio Reyes, and Koen Van Leemput. The multimodal brain tumor image segmentation benchmark (brats). *IEEE Transactions on Medical Imaging*, 2015.
 - [28] Chong Zhou, Chen Change Loy, and Bo Dai. Extract free dense labels from clip. In *ECCV*, 2022.
 - [29] Monika Wysoczanska, Michael Ramamonjisoa, Tomasz Trzcinski, and Oriane Simeoni. Clip-diy: Clip dense inference yields open-vocabulary semantic segmentation for-free. In *WACV*, 2024.
 - [30] Monika Wysoczańska, Oriane Siméoni, Michaël Ramamonjisoa, Andrei Bursuc, Tomasz Trzciński, and Patrick Pérez. CLIP-DINOiser: Teaching CLIP a few DINO tricks. In *ECCV*, 2024.
 - [31] Monika Wysoczańska, Antonin Vobecky, Amaia Cardiel, Tomasz Trzciński, Renaud Marlet, Andrei Bursuc, and Oriane Siméoni. Test-time contrastive concepts for open-world semantic segmentation with vision-language models. *arXiv preprint arXiv:2407.05061*, 2024.
 - [32] Monika Wysoczańska, Tom Monnier, Tomasz Trzciński, and David Picard. Toward unsupervised visual reasoning: Do off-the-shelf features know how to reason? *IEEE Access*, 2024.
 - [33] Monika Wysoczanska, Moran Beladev, Karen Lastmann Assaraf, Fengjun Wang, Ofri Kleinfeld, Gil Amsalem, and Hadas Harush Boker. Tell me what is

good about this property: Leveraging reviews for segment-personalized image collection summarization. *AAAI*, 2024.

- [34] Jacek Komorowski, Monika Wysoczanska, and Tomasz Trzcinski. Egonn: Egocentric neural network for point cloud based 6dof relocalization at the city scale. *IEEE Robotics and Automation Letters*, 2022.
- [35] Jacek Komorowski, Monika Wysoczańska, and Tomasz Trzcinski. Minkloc++: Lidar and monocular image fusion for place recognition. In *IJCNN*, 2021.
- [36] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.
- [37] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.
- [38] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.
- [39] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.
- [40] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.
- [41] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020.
- [42] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *ICLR*, 2022.
- [43] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *ACL*, 2019.
- [44] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. iBOT: Image bert pre-training with online tokenizer. In *ICLR*, 2022.
- [45] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, 2018.
- [46] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2021.
- [47] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

- [48] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *TMLR*, 2024.
- [49] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. *ICLR*, 2023.
- [50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.
- [51] Golnaz Ghiasi, Xiuye Gu, Yin Cui, and Tsung-Yi Lin. Scaling open-vocabulary image segmentation with image-level labels. In *ECCV*, 2022.
- [52] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, et al. Simple open-vocabulary object detection. In *ECCV*, 2022.
- [53] Seokju Cho, Heeseong Shin, Sunghwan Hong, Anurag Arnab, Paul Hongsuck Seo, and Seungryong Kim. CAT-Seg: Cost aggregation for open-vocabulary semantic segmentation. In *CVPR*, 2024.
- [54] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, 2021.
- [55] Hu Xu, Saining Xie, Po-Yao Huang Xiaoqing Ellen Tan, Russell Howes, Shang-Wen Li Vasu Sharma, Gargi Ghosh, Luke Zettlemoyer, and Christoph Feichtenhofer. Demystifying clip data. In *ICLR*, 2024.
- [56] Alex Fang, Albin Madappally Jose, Amit Jain, Ludwig Schmidt, Alexander T Toshev, and Vaishaal Shankar. Data filtering networks. In *The Twelfth International Conference on Learning Representations*, 2024.
- [57] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023.
- [58] Xiaohua Zhai, Xiao Wang, Basil Mustafa, Andreas Steiner, Daniel Keysers, Alexander Kolesnikov, and Lucas Beyer. Lit: Zero-shot transfer with locked-image text tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18123–18133, 2022.
- [59] Muhammad Ferjad Naeem, Yongqin Xian, Xiaohua Zhai, Lukas Hoyer, Luc

- Van Gool, and Federico Tombari. Silc: Improving vision language pretraining with self-distillation. In *ECCV*, 2024.
- [60] Cijo Jose, Théo Moutakanni, Dahyun Kang, Federico Baldassarre, Timothée Darcet, Hu Xu, Daniel Li, Marc Szafraniec, Michaël Ramamonjisoa, Maxime Oquab, et al. Dinov2 meets text: A unified framework for image-and pixel-level vision-language alignment. *arXiv preprint arXiv:2412.16334*, 2024.
- [61] Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.
- [62] Kevis-Kokitsi Maninis, Kaifeng Chen, Soham Ghosh, Arjun Karpur, Koert Chen, Ye Xia, Bingyi Cao, Daniel Salz, Guangxing Han, Jan Dlabal, et al. Tips: Text-image pretraining with spatial awareness. *ICLR*, 2025.
- [63] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *ECCV*, 2020.
- [64] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.
- [65] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *ICLR*, 2019.
- [66] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. *NeurIPS*, 2017.
- [67] Shoufa Chen, Chongjian GE, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. In *NeurIPS*, 2022.
- [68] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *ECCV*, 2022.
- [69] Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. Exploring visual prompts for adapting large-scale models. *arXiv preprint arXiv:2203.17274*, 2022.
- [70] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for

- generic visual recognition. In *ICLR*, 2014.
- [71] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *CVPR*, 2014.
 - [72] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary object detection via vision and language knowledge distillation. *arXiv preprint arXiv:2104.13921*, 2021.
 - [73] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? In *ECCV*, 2020.
 - [74] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.
 - [75] Junbum Cha, Jonghwan Mun, and Byungseok Roh. Learning to generate text-grounded mask for open-world semantic segmentation from only image-text pairs. In *CVPR*, 2023.
 - [76] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *NeurIPS*, 2022.
 - [77] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICLR*, 2023.
 - [78] Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, et al. Pali: A jointly-scaled multilingual language-image model. *arXiv preprint arXiv:2209.06794*, 2022.
 - [79] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *NeurIPS*, 2023.
 - [80] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
 - [81] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
 - [82] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines

- with visual instruction tuning. In *Proceedings of the IEEE / CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306, 2024.
- [83] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90% chatgpt quality, 2023. Version 1.5.
 - [84] Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *arXiv preprint arXiv:2406.16860*, 2024.
 - [85] Shijie Hao, Yuan Zhou, and Yanrong Guo. A brief survey on semantic segmentation with deep learning. In *Neurocomputing*, 2020.
 - [86] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *CVPR*, 2021.
 - [87] Bowen Cheng, Alex Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. In *NeurIPS*, 2021.
 - [88] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results.
 - [89] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *CVPR*, 2019.
 - [90] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.
 - [91] Kanchana Ranasinghe, Brandon McKinzie, Sachin Ravi, Yinfei Yang, Alexander Toshev, and Jonathon Shlens. Perceptual grouping in contrastive vision-language models. In *ICCV*, 2023.
 - [92] Jishnu Mukhoti, Tsung-Yu Lin, Omid Poursaeed, Rui Wang, Ashish Shah, Philip HS Torr, and Ser-Nam Lim. Open vocabulary semantic segmentation with patch-aligned contrastive learning. In *CVPR*, 2023.
 - [93] Quande Liu, Youpeng Wen, Jianhua Han, Chunjing Xu, Hang Xu, and Xiaodan Liang. Open-world semantic segmentation via contrasting and clustering vision-language embedding. In *ECCV*, 2022.
 - [94] Jiarui Xu, Shalini De Mello, Sifei Liu, Wonmin Byeon, Thomas Breuel, Jan Kautz, and Xiaolong Wang. Groupvit: Semantic segmentation emerges from text supervision. In *CVPR*, 2022.
 - [95] Oriane Siméoni, Gilles Puy, Huy V. Vo, Simon Roburin, Spyros Gidaris, Andrei Bursuc, Patrick Pérez, Renaud Marlet, and Jean Ponce. Localizing objects

- with self-supervised transformers and no labels. In *BMVC*, 2021.
- [96] Yangtao Wang, Xi Shen, Shell Xu Hu, Yuan Yuan, James L. Crowley, and Dominique Vaufreydaz. Self-supervised transformers for unsupervised object discovery using normalized cut. In *CVPR*, 2022.
 - [97] Xudong Wang, Rohit Girdhar, Stella X Yu, and Ishan Misra. Cut and learn for unsupervised object detection and instance segmentation. In *CVPR*, 2023.
 - [98] Gyungin Shin, Samuel Albanie, and Weidi Xie. Unsupervised salient object detection with spectral cluster voting. In *CVPRW*, 2022.
 - [99] Adam Bielski and Paolo Favaro. Move: Unsupervised movable object segmentation and detection. In *NeurIPS*, 2022.
 - [100] Oriane Siméoni, Chloé Sekkat, Gilles Puy, Antonin Vobecky, Éloi Zablocki, and Patrick Pérez. Unsupervised object localization: Observing the background to discover objects. In *CVPR*, 2023.
 - [101] Yongqin Xian, Subhabrata Choudhury, Yang He, Bernt Schiele, and Zeynep Akata. Semantic projection network for zero-and few-label semantic segmentation. In *CVPR*, 2019.
 - [102] Hang Zhao, Xavier Puig, Bolei Zhou, Sanja Fidler, and Antonio Torralba. Open vocabulary scene parsing. In *ICCV*, 2017.
 - [103] Maxime Bucher, Tuan-Hung Vu, Mathieu Cord, and Patrick Pérez. Zero-shot semantic segmentation. In *NeurIPS*, 2019.
 - [104] Zhangxuan Gu, Siyuan Zhou, Li Niu, Zihan Zhao, and Liqing Zhang. Context-aware feature generation for zero-shot semantic segmentation. In *ACM MM*, 2020.
 - [105] Naoki Kato, Toshihiko Yamasaki, and Kiyoharu Aizawa. Zero-shot semantic segmentation via variational mapping. In *ICCVW*, 2019.
 - [106] Ping Hu, Stan Sclaroff, and Kate Saenko. Uncertainty-aware learning for zero-shot semantic segmentation. In *NeurIPS*, 2020.
 - [107] Peike Li, Yunchao Wei, and Yi Yang. Consistent structural relation learning for zero-shot segmentation. In *NeurIPS*, 2020.
 - [108] Giuseppe Pastore, Fabio Cermelli, Yongqin Xian, Massimiliano Mancini, Zeynep Akata, and Barbara Caputo. A closer look at self-training for zero-label semantic segmentation. In *CVPR*, 2021.
 - [109] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.
 - [110] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.
 - [111] Jianzong Wu, Xiangtai Li, Shilin Xu, Haobo Yuan, Henghui Ding, Yibo Yang, Xia Li, Jiangning Zhang, Yunhai Tong, Xudong Jiang, Bernard Ghanem,

- and Dacheng Tao. Towards open vocabulary learning: A survey. *CoRR*, abs/2306.15880, 2023.
- [112] Huaishao Luo, Junwei Bao, Youzheng Wu, Xiaodong He, and Tianrui Li. Seg-CLIP: Patch aggregation with learnable centers for open-vocabulary semantic segmentation. In *ICML*, 2023.
 - [113] Yongming Rao, Wenliang Zhao, Guangyi Chen, Yansong Tang, Zheng Zhu, Guan Huang, Jie Zhou, and Jiwen Lu. Denseclip: Language-guided dense prediction with context-aware prompting. In *CVPR*, 2022.
 - [114] Mert Yuksekgonul, Federico Bianchi, Pratyusha Kalluri, Dan Jurafsky, and James Zou. When and why vision-language models behave like bags-of-words, and what to do about it? In *ICLR*, 2023.
 - [115] Jian Ding, Nan Xue, Gui-Song Xia, and Dengxin Dai. Decoupling zero-shot semantic segmentation. In *CVPR*, 2022.
 - [116] Jilan Xu, Junlin Hou, Yuejie Zhang, Rui Feng, Yi Wang, Yu Qiao, and Weidi Xie. Learning open-vocabulary semantic segmentation models from natural language supervision. In *CVPR*, 2023.
 - [117] Pengzhen Ren, Changlin Li, Hang Xu, Yi Zhu, Guangrun Wang, Jianzhuang Liu, Xiaojun Chang, and Xiaodan Liang. Viewco: Discovering text-supervised segmentation masks via multi-view semantic consistency. In *ICLR*, 2023.
 - [118] Gyungin Shin, Weidi Xie, and Samuel Albanie. Reco: Retrieve and co-segment for zero-shot transfer. In *NeurIPS*, 2022.
 - [119] Huy V. Vo, Patrick Pérez, and Jean Ponce. Toward unsupervised, multi-object discovery in large-scale image collections. In *ECCV*, 2020.
 - [120] Van Huy Vo, Elena Sizikova, Cordelia Schmid, Patrick Pérez, and Jean Ponce. Large-scale unsupervised object discovery. In *NeurIPS*, 2021.
 - [121] Luke Melas-Kyriazi, Christian Rupprecht, Iro Laina, and Andrea Vedaldi. Deep spectral methods: A surprisingly strong baseline for unsupervised semantic segmentation and localization. In *CVPR*, 2022.
 - [122] Xinlong Wang, Zhiding Yu, Shalini De Mello, Jan Kautz, Anima Anandkumar, Chunhua Shen, and Jose M. Alvarez. Freesolo: Learning to segment objects without annotations. In *CVPR*, 2022.
 - [123] Andrii Zadaianchuk, Matthäus Kleindessner, Yi Zhu, Francesco Locatello, and Thomas Brox. Unsupervised semantic segmentation with self-supervised object-centric representations. In *ICLR*, 2023.
 - [124] Pitchaporn Rewatbowornwong, Nattanat Chatthee, Ekapol Chuangsuwanich, and Supasorn Suwajanakorn. Zero-guidance segmentation using zero segment labels. In *ICCV*, 2023.
 - [125] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn,

- Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.
- [126] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade W Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa R Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5b: An open large-scale dataset for training next generation image-text models. In *NeurIPS Datasets and Benchmarks Track*, 2022.
- [127] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 2020.
- [128] Nikita Araslanov and Stefan Roth. Single-stage semantic segmentation from image labels. In *CVPR*, 2020.
- [129] Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan. Learning to detect salient objects with image-level supervision. In *CVPR*, 2017.
- [130] Alex Fang, Gabriel Ilharco, Mitchell Wortsman, Yuhao Wan, Vaishaal Shankar, Achal Dave, and Ludwig Schmidt. Data determines distributional robustness in contrastive language image pre-training (CLIP). In *ICML*, 2022.
- [131] Prasanna Mayilvahanan, Thaddäus Wiedemer, Evgenia Rusak, Matthias Bethge, and Wieland Brendel. Does CLIP’s generalization performance mainly stem from high train-test similarity? *arXiv preprint arXiv:2310.09562*, 2023.
- [132] Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, Mohd Omama, Tao Chen, Shuang Li, Ganesh Iyer, Soroush Saryazdi, Nikhil Keetha, Ayush Tewari, et al. Conceptfusion: Open-set multimodal 3d mapping. In *RSS*, 2023.
- [133] Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. In *CVPR*, 2023.
- [134] Runnan Chen, Youquan Liu, Lingdong Kong, Xinge Zhu, Yuexin Ma, Yikang Li, Yuenan Hou, Yu Qiao, and Wenping Wang. Clip2scene: Towards label-efficient 3d scene understanding by clip. In *CVPR*, 2023.
- [135] Mahyar Najibi, Jingwei Ji, Yin Zhou, Charles R Qi, Xichen Yan, Scott Ettinger, and Dragomir Anguelov. Unsupervised 3d perception with 2d vision-language distillation for autonomous driving. In *ICCV*, 2023.
- [136] Antonín Vobecký, Oriane Siméoni, David Hurych, Spyros Gidaris, Andrei Bursuc, Patrick Perez, and Josef Sivic. Pop-3d: Open-vocabulary 3d occupancy

- prediction from images. In *NeurIPS*, 2023.
- [137] Ahmed Abdelreheem, Ivan Skorokhodov, Maks Ovsjanikov, and Peter Wonka. Satr: Zero-shot semantic segmentation of 3d shapes. In *ICCV*, 2023.
 - [138] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lurf: Language embedded radiance fields. In *ICCV*, 2023.
 - [139] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *CVPR*, 2023.
 - [140] Gyungin Shin, Weidi Xie, and Samuel Albanie. Namedmask: Distilling segmenters from complementary foundation models. In *CVPRW*, 2023.
 - [141] Laurynas Karazija, Iro Laina, Andrea Vedaldi, and Christian Rupprecht. Diffusion models for zero-shot open-vocabulary segmentation. *ECCV*, 2024.
 - [142] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *CVPR*, 2018.
 - [143] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.
 - [144] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *IJCV*, 2019.
 - [145] Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and Rene Ranftl. Language-driven semantic segmentation. In *ICLR*, 2022.
 - [146] Feng Liang, Bichen Wu, Xiaoliang Dai, Kunpeng Li, Yinan Zhao, Hang Zhang, Peizhao Zhang, Peter Vajda, and Diana Marculescu. Open-vocabulary semantic segmentation with mask-adapted clip. In *CVPR*, 2023.
 - [147] Yiwu Zhong, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luwei Zhou, Xiyang Dai, Lu Yuan, Yin Li, et al. Regionclip: Region-based language-image pretraining. In *CVPR*, 2022.
 - [148] Jun Chen, Deyao Zhu, Guocheng Qian, Bernard Ghanem, Zhicheng Yan, Chenchen Zhu, Fanyi Xiao, Sean Chang Culatana, and Mohamed Elhoseiny. Exploring open-vocabulary semantic segmentation from clip vision encoder distillation only. In *ICCV*, 2023.
 - [149] Oriane Siméoni, Éloi Zablocki, Spyros Gidaris, Gilles Puy, and Patrick Pérez. Unsupervised object localization in the era of self-supervised vits: A survey. *arXiv preprint arXiv:2310.12904*, 2023.
 - [150] Matthew Walmer, Saksham Suri, Kamal Gupta, and Abhinav Shrivastava. Teaching matters: Investigating the role of supervision in vision transformers.

- In *CVPR*, 2023.
- [151] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.
 - [152] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014.
 - [153] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark, 2020.
 - [154] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.
 - [155] Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan. Learning to detect salient objects with image-level supervision. In *CVPR*, 2017.
 - [156] Chuan Yang, Lihe Zhang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang. Saliency detection via graph-based manifold ranking. In *CVPR*, 2013.
 - [157] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results.
 - [158] Jianping Shi, Qiong Yan, Li Xu, and Jiaya Jia. Hierarchical image saliency detection on extended cssd. *T-PAMI*, 2015.
 - [159] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *CVPR*, 2021.
 - [160] Karan Desai, Gaurav Kaul, Zubin Aysola, and Justin Johnson. RedCaps: Web-curated image-text data created by the people, for the people. In *NeurIPS Datasets and Benchmarks*, 2021.
 - [161] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. 2018.
 - [162] Walid Bousselham, Felix Petersen, Vittorio Ferrari, and Hilde Kuehne. Grounding everything: Emerging localization properties in vision-language transformers. In *CVPR*, 2024.
 - [163] Dimity Miller, Niko Sünderhauf, Alex Kenna, and Keita Mason. Open-set recognition in the age of vision-language models. In *ECCV*, 2024.
 - [164] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *NeurIPS*, 2022.
 - [165] James Urquhart Allingham, Jie Ren, Michael W Dusenberry, Xiuye Gu, Yin Cui, Dustin Tran, Jeremiah Zhe Liu, and Balaji Lakshminarayanan. A sim-

- ple zero-shot prompt weighting technique to improve prompt ensembling in text-image models. In *ICML*, 2023.
- [166] Christiane Fellbaum. *WordNet: An electronic lexical database*. MIT press, 1998.
 - [167] Yunhao Ge, Jie Ren, Andrew Gallagher, Yuxiao Wang, Ming-Hsuan Yang, Hartwig Adam, Laurent Itti, Balaji Lakshminarayanan, and Jiaping Zhao. Improving zero-shot generalization and robustness of multi-modal models. In *CVPR*, 2023.
 - [168] Yuqi Lin, Minghao Chen, Wenxiao Wang, Boxi Wu, Ke Li, Binbin Lin, Haifeng Liu, and Xiaofei He. CLIP is also an efficient segmenter: A text-driven approach for weakly supervised semantic segmentation. In *CVPR*, 2023.
 - [169] Sarah Pratt, Rosanne Liu, and Ali Farhadi. What does a platypus look like. In *ICCV*, 2023.
 - [170] Sachit Menon and Carl Vondrick. Visual classification via description from large language models. In *ICLR*, 2023.
 - [171] Tanmay Gupta, Alexander Schwing, and Derek Hoiem. ViCo: Word embeddings from visual co-occurrences. In *ICCV*, 2019.
 - [172] Zehao Xiao, Jiayi Shen, Mohammad Mahdi Derakhshani, Shengcai Liao, and Cees GM Snoek. Any-shift prompting for generalization over distributions. In *CVPR*, 2024.
 - [173] Reza Esfandiarpour, Cristina Menghini, and Stephen H Bach. If CLIP could talk: Understanding vision-language model representations through their preferred concept descriptions. *arXiv preprint arXiv:2403.16442*, 2024.
 - [174] Manli Shu, Weili Nie, De-An Huang, Zhiding Yu, Tom Goldstein, Anima Anandkumar, and Chaowei Xiao. Test-time prompt tuning for zero-shot generalization in vision-language models. In *NeurIPS*, 2022.
 - [175] Yuqi Lin, Minghao Chen, Kaipeng Zhang, Hengjia Li, Mingming Li, Zheng Yang, Dongqin Lv, Binbin Lin, Haifeng Liu, and Deng Cai. Tagclip: A local-to-global framework to enhance open-vocabulary multi-label classification of clip without training. *arXiv preprint arXiv:2312.12828*, 2023.
 - [176] Qihang Yu, Ju He, Xueqing Deng, Xiaohui Shen, and Liang-Chieh Chen. Convolutions die hard: Open-vocabulary segmentation with single frozen convolutional CLIP. In *NeurIPS*, 2023.
 - [177] Ronghang Hu, Marcus Rohrbach, and Trevor Darrell. Segmentation from natural language expressions. In *ECCV*, 2016.
 - [178] Henghui Ding, Chang Liu, Suchen Wang, and Xudong Jiang. VLT: Vision-language transformer and query generation for referring segmentation. *TPAMI*, 2023.

- [179] Zhaoqing Wang, Yu Lu, Qiang Li, Xunqiang Tao, Yandong Guo, Mingming Gong, and Tongliang Liu. CRIS: CLIP-driven referring image segmentation. In *CVPR*, 2022.
- [180] Chang Liu, Henghui Ding, and Xudong Jiang. GRES: Generalized referring expression segmentation. In *CVPR*, 2023.
- [181] Xinpeng Chen, Lin Ma, Jingyuan Chen, Zequn Jie, Wei Liu, and Jiebo Luo. Real-time referring expression comprehension by single-stage grounding network. *arXiv preprint arXiv:1812.03426*, 2018.
- [182] Jiajun Deng, Zhengyuan Yang, Tianlang Chen, Wengang Zhou, and Houqiang Li. TransVG: End-to-end visual grounding with transformers. In *ICCV*, 2021.
- [183] Yue Liao, Si Liu, Guanbin Li, Fei Wang, Yanjie Chen, Chen Qian, and Bo Li. A real-time cross-modality correlation filtering method for referring expression comprehension. In *CVPR*, 2020.
- [184] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *ECCV*, 2024.
- [185] Vishaal Udandaraao, Ameya Prabhu, Adhiraj Ghosh, Yash Sharma, Philip H. S. Torr, Adel Bibi, Samuel Albanie, and Matthias Bethge. No "zero-shot" without exponential data: Pretraining concept frequency determines multimodal model performance. In *NeurIPS*, 2024.
- [186] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. LAION-400M: Open dataset of CLIP-filtered 400 million image-text pairs. In *NeurIPS*, 2021.
- [187] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, L lio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Th ophile Gervet, Thibaut Lavril, Thomas Wang, Timoth e Lacroix, and William El Sayed. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.
- [188] Mengde Xu, Zheng Zhang, Fangyun Wei, Han Hu, and Xiang Bai. Side adapter network for open-vocabulary semantic segmentation. In *CVPR*, 2023.
- [189] Karsten Roth, Jae Myung Kim, A Koepke, Oriol Vinyals, Cordelia Schmid, and Zeynep Akata. Waffling around for performance: Visual classification with random words and broad concepts. In *ICCV*, 2023.

- [190] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. <https://github.com/facebookresearch/detectron2>, 2019.
- [191] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *ICLR*, 2020.
- [192] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *CVPR*, 2020.
- [193] Lukas Schott, Julius Von Kügelgen, Frederik Träuble, Peter Vincent Gehler, Chris Russell, Matthias Bethge, Bernhard Schölkopf, Francesco Locatello, and Wieland Brendel. Visual representation learning does not generalize strongly within the same domain. In *ICLR*, 2022.
- [194] Christopher P. Burgess, Loïc Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matthew M. Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *CoRR*, abs/1901.11390, 2019.
- [195] Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Chris Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-Object Representation Learning with Iterative Variational Inference. In *ICML*, 2019.
- [196] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *NeurIPS*, 2020.
- [197] Tom Monnier, Elliot Vincent, Jean Ponce, and Mathieu Aubry. Unsupervised Layered Image Decomposition into Object Prototypes. In *ICCV*, 2021.
- [198] Laurynas Karazija, Iro Laina, and Christian Rupprecht. Clevrtex: A texture-rich benchmark for unsupervised multi-object segmentation. In *NeurIPS*, 2021.
- [199] Spyridon Mouselinos, Henryk Michalewski, and Mateusz Malinowski. Measuring CLEVRness: Black-box testing of visual reasoning models. In *ICLR*, 2022.
- [200] Aimen Zerroug, Mohit Vaishnav, Julien Colin, Sebastian Musslick, and Thomas Serre. A benchmark for compositional visual reasoning. In *NeurIPS*, 2022.
- [201] Michal Nazarczuk and Krystian Mikolajczyk. Shop-vrb: A visual reasoning benchmark for object perception. In *ICRA*, 2020.
- [202] Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, Alexander Kolesnikov, Joan Puigcerver, Nan Ding, Keran Rong, Has-

- san Akbari, Gaurav Mishra, Linting Xue, Ashish V Thapliyal, James Bradbury, Weicheng Kuo, Mojtaba Seyedhosseini, Chao Jia, Burcu Karagol Ayan, Carlos Riquelme Ruiz, Andreas Peter Steiner, Anelia Angelova, Xiaohua Zhai, Neil Houlsby, and Radu Soricut. PaLI: A jointly-scaled multilingual language-image model. In *ICLR*, 2023.
- [203] Kan Chen, Jiang Wang, Liang-Chieh Chen, Haoyuan Gao, Wei Xu, and Ram Nevatia. Abc-cnn: An attention based convolutional neural network for visual question answering. *arXiv preprint arXiv:1511.05960*, 2015.
- [204] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: visual reasoning with a general conditioning layer. *AAAI*, 2018.
- [205] Aisha Urooj Khan, Hilde Kuehne, Kevin Duarte, Chuang Gan, Niels Lobo, and Mubarak Shah. Found a reason for me? weakly-supervised grounded visual question answering using capsules. *CVPR*, 2021.
- [206] Drew A Hudson and Christopher D Manning. Compositional attention networks for machine reasoning. *arXiv preprint arXiv:1803.03067*, 2018.
- [207] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. *CVPR*, 2018.
- [208] Zhonghao Wang, Kai Wang, Mo Yu, Jinjun Xiong, Wen-mei Hwu, Mark Hasegawa-Johnson, and Humphrey Shi. Interpretable visual reasoning via induced symbolic space. In *CVPR*, 2021.
- [209] Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B. Tenenbaum, and Jiajun Wu. The Neuro-Symbolic Concept Learner: Interpreting Scenes, Words, and Sentences From Natural Supervision. In *ICLR*, 2019.
- [210] Mihir Prabhudesai, Shamit Lal, Darshan Patil, Hsiao-Yu Tung, Adam W Harley, and Katerina Fragkiadaki. Disentangling 3d prototypical networks for few-shot concept learning. In *ICLR*, 2021.
- [211] Zhenwei Shao, Zhou Yu, Meng Wang, and Jun Yu. Prompting large language models with answer heuristics for knowledge-based visual question answering. In *CVPR*, 2023.
- [212] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. An empirical study of gpt-3 for few-shot knowledge-based vqa. In *AAAI*, 2022.
- [213] Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Joshua B. Tenenbaum. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. In *NeurIPS*, 2018.
- [214] David Mascharka, Philip Tran, Ryan Soklaski, and Arjun Majumdar. Trans-

- parency by design: Closing the gap between performance and interpretability in visual reasoning. *CVPR*, 2018.
- [215] Huayi Zhan, Peixi Xiong, Xin Wang, Xin WANG, and Lan Yang. Visual question answering by pattern matching and reasoning. *Neurocomputing*, 2022.
 - [216] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *ECCV*, 2020.
 - [217] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, 2019.
 - [218] Guillaume Lample and François Charton. Deep learning for symbolic mathematics. In *ICLR*, 2020.
 - [219] Christopher Hahn, Frederik Schmitt, Jens U. Kreber, Markus Norman Rabe, and Bernd Finkbeiner. Teaching temporal logics to neural networks. In *ICLR*, 2021.
 - [220] Jorge Pérez, Pablo Barceló, and Javier Marinkovic. Attention is turing-complete. *JMLR*, 2021.
 - [221] David Ding, Felix Hill, Adam Santoro, Malcolm Reynolds, and Matt Botvinick. Attention over learned object embeddings enables complex visual reasoning. *NeurIPS*, 2021.
 - [222] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. Mdetr-modulated detection for end-to-end multi-modal understanding. In *ICCV*, 2021.
 - [223] Ron Mokady, Amir Hertz, and Amit H Bermano. Clipcap: Clip prefix for image captioning. *arXiv preprint arXiv:2111.09734*, 2021.
 - [224] Alberto Baldrati, Marco Bertini, Tiberio Uricchio, and Alberto Del Bimbo. Effective conditioned and composed image retrieval combining CLIP-based features. In *CVPR*, 2022.
 - [225] Klaus Greff, Rupesh Kumar Srivastava, and Jürgen Schmidhuber. Binding via Reconstruction Clustering. In *ICLR Workshops*, 2016.
 - [226] Klaus Greff, Sjoerd van Steenkiste, and Jürgen Schmidhuber. Neural Expectation Maximization. In *NeurIPS*, 2017.
 - [227] Martin Engelcke, Adam R. Kosior, Oivi Parker Jones, and Ingmar Posner. GENESIS: Generative Scene Inference and Sampling with Object-Centric Latent Representations. In *ICLR*, 2020.
 - [228] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial Transformer Networks. In *NeurIPS*, 2015.
 - [229] S. M. Ali Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepes-

- vari, Koray Kavukcuoglu, and Geoffrey E Hinton. Attend, Infer, Repeat: Fast Scene Understanding with Generative Models. In *NeurIPS*, 2016.
- [230] Adam R. Kosiosek, Hyunjik Kim, Ingmar Posner, and Yee Whye Teh. Sequential Attend, Infer, Repeat: Generative Modelling of Moving Objects. In *NeurIPS*, 2018.
- [231] Eric Crawford and Joelle Pineau. Spatially Invariant Unsupervised Object Detection with Convolutional Neural Networks. In *AAAI*, 2019.
- [232] Zhixuan Lin, Yi-Fu Wu, Skand Vishwanath Peri, Weihao Sun, Gautam Singh, Fei Deng, Jindong Jiang, and Sungjin Ahn. SPACE: Unsupervised Object-Oriented Scene Representation via Spatial Attention and Decomposition. In *ICLR*, 2020.
- [233] Dmitriy Smirnov, Michael Gharbi, Matthew Fisher, Vitor Guizilini, Alexei A. Efros, and Justin Solomon. MarioNette: Self-Supervised Sprite Learning. In *NeurIPS*, 2021.
- [234] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.
- [235] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017.
- [236] Adam Dahlgren Lindström and Savitha Sam Abraham. Clevr-math: A dataset for compositional language, visual and mathematical reasoning, 2022.
- [237] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface’s transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.
- [238] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *JMLR*, 2011.
- [239] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 2019.
- [240] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

- [241] Suraj Kothawade, Vishal Kaushal, Ganesh Ramakrishnan, Jeff Bilmes, and Rishabh Iyer. Prism: A rich class of parameterized submodular information measures for guided data subset selection. *AAAI*, 2022.
- [242] Medhini Narasimhan, Arsha Nagrani, Chen Sun, Michael Rubinstein, Trevor Darrell, Anna Rohrbach, and Cordelia Schmid. Tl; dw? summarizing instructional videos with task relevance and cross-modal saliency. In *ECCV*, 2022.
- [243] Fengjun Wang, Sarai Mizrahi, Moran Beladev, Guy Nadav, Gil Amsalem, Karen Lastmann Assaraf, and Hadas Harush Boker. Memic–multimodal embedding for multi-label image classification with tempered sigmoid. In *AAAI*, 2023.
- [244] Jonathan C. Stroud, David A. Ross, Chen Sun, Jia Deng, Rahul Sukthankar, and Cordelia Schmid. Learning video representations from textual web supervision. *CoRR*, 2020.
- [245] Haopeng Li, Qiuhong Ke, Mingming Gong, and Rui Zhang. Video summarization based on video-text modelling. *CoRR*, abs/2201.02494, 2022.
- [246] Medhini Narasimhan, Anna Rohrbach, and Trevor Darrell. Clip-it! language-guided video summarization. *CoRR*, abs/2107.00650, 2021.
- [247] Zhengkun Zhang, Xiaojun Meng, Yasheng Wang, Xin Jiang, Qun Liu, and Zhenglu Yang. Unims: A unified framework for multimodal summarization with knowledge distillation. *AAAI*, 2022.
- [248] Junnan Zhu, Haoran Li, Tianshang Liu, Yu Zhou, Jiajun Zhang, and Chengqing Zong. MSMO: Multimodal summarization with multimodal output. In *EMNLP*, 2018.
- [249] Junnan Zhu, Yu Zhou, Jiajun Zhang, Haoran Li, Chengqing Zong, and Changliang Li. Multimodal summarization with guidance of multimodal reference. *AAAI*, 2020.
- [250] Monika Wysoczańska and Tomasz Trzciński. Multimodal dance recognition. In *VISAPP*, 2020.
- [251] Pedro Morgado, Nuno Vasconcelos, and Ishan Misra. Audio-visual instance discrimination with cross-modal agreement. *arXiv preprint arXiv:2004.12943*, 2020.
- [252] Hae-Sang Park and Chi-Hyuck Jun. A simple and fast algorithm for k-medoids clustering. *Expert Systems with Applications*, 2009.
- [253] Fengjun Wang, Moran Beladev, Ofri Kleinfeld, Elina Frayerman, Tal Shachar, Eran Fainman, Karen Lastmann Assaraf, Sarai Mizrahi, and Benjamin Wang. Text2topic: Multi-label text classification system for efficient topic detection in user generated content with zero-shot capabilities, 2023.
- [254] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using

- siamese bert-networks. In *EMNLP*, 2019.
- [255] Rishabh K. Iyer, Pratik Dubal, Kunal Dargan, Suraj Kothawade, Rohan Mahadev, and Vishal Kaushal. Vis-dss: An open-source toolkit for visual data selection and summarization. *CoRR*, abs/1809.08846, 2018.
- [256] Sebastian Tschiatschek, Rishabh K Iyer, Haochen Wei, and Jeff A Bilmes. Learning mixtures of submodular functions for image collection summarization. In *NeurIPS*, 2014.
- [257] Haoran Li, Junnan Zhu, Tianshang Liu, Jiajun Zhang, and Chengqing Zong. Multi-modal sentence summarization with modality attention and image filtering. In *IJCAI*, 2018.
- [258] Youssef Hadi, Fedwa Essannouni, and Rachid Oulad Haj Thami. Video summarization by k-medoid clustering. *SAC*, 2006.
- [259] Student. The probable error of a mean. *Biometrika*, 1908.

List of Figures

1.2.1	The outline of the thesis. We examine how visual foundation models adapt to downstream applications beyond image classification. First, we address computational costs when adapting VLMs for Open-Vocabulary Semantic Segmentation. Next, we explore the evaluation and selection of optimal VFMs for specific tasks, using Visual Question Answering as an example. Finally, we investigate VLMs' capabilities in navigating complex real-world scenarios like content personalization.	19
2.1.1	Image-text alignment via contrastive learning. Source [13]. . .	29
2.2.1	LLaVa 1.5 architecture - a generic framework for MLLMs. Source [82].	32
3.1.1	Simple and accurate semantic segmentation for-free using CLIP-DIY. Our method processes regions of an image <i>separately, in parallel</i> to produce per-patch-per-class scores, given a set of prompts that can take <i>any length</i> . CLIP-DIY does not require re-training and can, therefore, immediately adapt to any new vocabulary.	35
3.2.1	Overview of CLIP-DIY, our two-step pipeline for segmentation map extraction. We note Φ_V and Φ_T the CLIP encoders for image and text, respectively. (1) An input image is partitioned into smaller patches, and each of them is fed <i>independently</i> to the image encoder, yielding a vector of per-class similarity to an <i>arbitrary-length</i> vocabulary of classes. (2) Patches are then aggregated back before upsampling to produce dense similarity maps. (3) An objectness score obtained by an off-the-shelf foreground-background segmentation method such as FOUND [100] is used to guide the prediction of the final segmentation.	38

3.3.1	Multi-scale maps ρ_s of CLIP-DIY. Our method produces multi-scale predictions of dense logits that capture information at different levels of granularity. While the coarsest ($s = 0$) scale pools global information from the image, finer scales induce more localized and potentially discover different object classes. While our method can incorporate as many scales as needed, very fine resolutions such as scale 4 and above do not generate meaningful information.	40
3.4.1	Qualitative open-vocabulary segmentation results. We compare our method against CLIPpy [91] and TCL (with PAMR post-processing) [75]. Our method consistently outperforms two other methods by producing accurate segmentation masks. TCL and CLIPpy also both suffer from hallucinating classes based on context, such as the aeroplane and sheep in 1st column, or zebra in 5th column. All pixels annotated in black are from the background class.	45
3.4.2	Qualitative ablation study. In the right column we show the results when progressively adding more scales in the multi-scale approach. Running with 3 scales enables segmenting most of the chair.	47
3.4.3	Failure cases. Our method is not robust in a few cases, such as (a) incomplete or (b) coarse labeling. Another failure mode of our method is when there is an ambiguity in the foreground class (c-d).	48
3.6.1	Qualitative segmentation results on PASCAL VOC. We compare our method against CLIPpy [91] and TCL (with PAMR post-processing) [75]. Our method consistently outperforms two other methods by producing accurate segmentation masks.	51
3.6.2	Qualitative segmentation results on COCO. We compare our method against CLIPpy [91] and TCL [75] (with PAMR [128] post-processing) [75]. Our method consistently outperforms two other methods by producing accurate segmentation masks. TCL and CLIPpy also both suffer from hallucinating or producing noisy masks.	52
3.6.3	More examples of in-the-wild open-world segmentation. We compare the segmentation produced by our method with the results of TCL [75] (with PAMR [128] post-processing). While both methods are able to detect and locate each class, including distinguishing between pink elephant and grey elephant , TCL largely over-segments objects.	53

3.6.4	Failure cases. We show examples from both datasets. Our method at times produces incomplete masks when saliency focuses only on parts of the scene such as in (a) and (c), ambiguous classification in (b), as well confusion when classes overlap (d).	54
4.1.1	Examples of open-vocabulary semantic segmentation results obtained with our method CLIP-DINOiser on ‘in-the-wild’ images vs. those of MaskCLIP [28]. Our method improves MaskCLIP features with a smart pooling strategy, which does <i>not alter the original</i> open-vocabulary properties. We use self-supervised DINO [15] as a guide to <i>teach CLIP</i> [54] to produce DINO-like localization features through two light convolutional layers. Our method, which achieves state-of-the-art results, only requires a <i>single forward</i> pass through CLIP model and our two layers. In addition to the correct prompts (light grey row), we list the irrelevant prompts predicted (in yellow) that we query in all images shown here.	56
4.3.1	We present in (a) is our <i>guided pooling</i> strategy defined in Eq. (4.1). The $N \times N$ affinity matrix is computed from patch features and is used to refine MaskCLIP features (bottom left). In (b) we compare our results with F^+ (right) versus those obtained with MaskCLIP features (middle).	61
4.3.2	Overview of CLIP-DINOiser which leverages the quality of self-supervised features to improve the notoriously noisy MaskCLIP feature maps. We use DINO as a teacher which ‘teaches’ CLIP how to extract localization information. We train (left) a $\text{conv}3 \times 3$ layer to reproduce the patch correlations obtained with DINO. At inference (right), an input image is forwarded through the frozen CLIP image backbone and MaskCLIP projection. The produced features are then improved with our <i>pooling</i> strategy which is guided by correlations predicted with the trained convolutional layer applied on CLIP. With this light ‘DINOising’ process, we obtain ‘DINOised’ features which are matched against the prompts features to produce CLIP-DINOiser outputs.	62
4.3.3	Comparison of the affinity maps between a <i>seed</i> (one on the ‘plant’ and the other on a ‘pillow’) and the other patch features when using features of MaskCLIP, DINO and ours after training.	63

4.3.4	We present in (a) a comparison of <i>objectness</i> mask generated by FOUND [100] and with our layer using CLIP features. We carefully define the fusion operation and the simple training strategy of the $\text{conv}1 \times 1$ again using DINO as a teacher in Sec. 4.3.5. In (b) is an overview of our <i>background filtering</i> which is applied when a ‘background’ prompt is provided and helps reduce hallucinations. .	64
4.4.1	Qualitative open-vocabulary segmentation results. We compare ours against CLIP-DIY [29], TCL [75] and MaskCLIP [28]. For a fair comparison, we do not apply post-processing. All pixels annotated in black are from the background class. We observe that our method achieves more accurate results both in terms of localization and class assignment.	68
4.6.1	Visualization of correlation and segmentation obtained with different embeddings of DINO: <i>query</i> , <i>key</i> and <i>value</i> . The predicted prompts are: <i>sky, tree, train, ground, fence, grass</i>	71
4.6.2	Visual ablations of the impact of our pooling method. Examples from ADE20K (top), PASCAL Context (middle), and Cityscapes (bottom) datasets.	73
4.6.3	Visual ablations of the impact of background detection. We show examples from COCO Object (top, middle) and PASCAL VOC (bottom). We note with ‘bkg’ our background refinement.	74
4.6.4	In-the-wild comparative examples between MaskCLIP (top) and CLIP-DINOiser (bottom). While MaskCLIP generates noisy masks when prompted with <i>false positive</i> classes our method is robust and produces cleaner masks.	75
4.6.5	Failure cases of our method. From left to right: input RGB image, ground truth (GT) masks, masks predicted by CLIP-DINOiser, predicted text prompts. We discuss these failure cases in Sec. 4.6.6.	75

5.1.1	Illustration of our proposed open-world scenario and benefits of contrastive concepts (CC). We investigate open-world segmentation, where only one (or a few) visual concepts are to be segmented (2 nd column), while all concepts that can occur in an image are unknown. Contrasting the query with “background” allows us to obtain a coarse segmentation [91, 30] (3 rd column), but is not enough to catch all pixels <i>not</i> corresponding to the query when they are related or co-occur frequently in the VLM training set. Our automatically-generated <i>contrastive concepts</i> (CC) (4 th column) help to separate and disentangle pixels of the query (right column, generated CC in text boxes), therefore achieving better segmentation. 77	
5.3.1	Statistics about “background” in metadata of web-crawled datasets. (a) Frequency of some of the concepts from VOC dataset in LAION-400M caption samples. Examples of images in web-crawled data with a caption including the words “background” (b) or “in the background” (c). 83	
5.3.2	Overview of our method. We propose two solutions to generate CC automatically, the first one (<i>top-left</i>) based on LLM prompting (CC^L) and the second one, CC^D that relies on the distribution of co-occurring concepts in a pre-training dataset of a VLM (<i>top-right</i>). Both methods can be effectively integrated into various open-vocabulary segmentation methods. 84	
5.3.3	An abbreviated version of the prompt we use to generate CC^L . 86	
5.4.1	Illustration of IoU-single metric. We show the difference with the standard mIoU metric (dataset-driven mIoU), where all the concepts present on an image are considered at once. On the contrary, our IoU-single considers each of the present concepts separately to measure the single-class segmentation ability of open-vocabulary semantic segmenters. 87	
5.4.2	Qualitative results. We show segmentation examples from ADE20K (1 st and 2 nd row) and Context (3 rd and 4 th row), with segments produced by CLIP-DINOiser. For CC^D and CC^L , we additionally show the joint segmentation of all contrastive classes (all). 91	
5.4.3	In the wild examples. We visualize results for MaskCLIP and CLIP-DINOiser for query concepts beyond \mathcal{T} . The closest neighbour to a query is presented below each example (grey row). 92	

5.6.1	Failure cases of our method. We show examples of CLIP-DINOiser when one of the methods fails to generate accurate \mathcal{CC} . In the first example \mathcal{CC}^L suggests "blanket" for "bed" which typically covers the query concept. In the second row, both methods fail to provide "floor" to contrast with "rug". Finally, in the third example, both methods fail to generate "person" to contrast with "bedclothes", however, \mathcal{CC}^L suggest "pyjamas", which results in a better segmentation.	97
5.6.2	More qualitative results of CLIP-DINOiser with different \mathcal{CC} . Here we focus on cases where \mathcal{CC}^D and \mathcal{CC}^L give different results. For "boat" (2 nd row), \mathcal{CC}^L gives a better result providing a good \mathcal{CC} ("dock"). On the other hand, for "skyscraper" (3 rd row), \mathcal{CC}^D yields slightly better results suggesting "sky" and not "cloud". Note that in this last example, \mathcal{CC}^{BG} completely fails, possibly due to a difficult (uncommon) angle of view.	97
5.6.3	Number of \mathcal{CC} vs performance. We compare the number of \mathcal{CC} against the performance of CLIP-DINOiser for each class used in our evaluations (considering all datasets). Performance is reported with per class IoU-single %.	98
5.6.4	Distribution of maximum patch similarities with text prompts. We plot histograms for 100 images of VOC (a) and ADE20K (b) of patch similarities in MaskCLIP.	99
5.6.5	t-SNE analysis of patch features for different \mathcal{CC} of an image $q = \text{"bird"}$. We present patch features with their predicted closest text embedding coded in color. Text embeddings are corresponding \mathcal{CC} of $q = \text{"bird"}$. We also mark the ground truth labels in orange. The sample is from VOC dataset.	100
5.6.6	Sigmoid experiments. We replace softmax with sigmoid applied on individual patch-to-query prompt similarities. We show the variation of single-IoU% wrt. the threshold that is applied after sigmoid to decide on a positive vs. "background" class. To get the thresholds, we find the minimum and maximum values of the features after sigmoid and linearly sample 30 values in this range. We can see that the result is sensitive to the threshold value and does not reach the baseline of \mathcal{CC}^{BG}	101
5.6.7	Prompt for \mathcal{CC}^L contrastive concept generation.	102
5.6.8	Prompt for \mathcal{CC}^L visibility prediction.	102

5.6.9	Part removal. We consider an example from Pascal Context with $q = \text{bicycle}$. We show the segmentation masks produced by MaskCLIP and CLIP-DINOiser for \mathcal{CC}^D , as well as for \mathcal{CC}^D when parts of objects are removed ($\mathcal{CC}^D - \text{parts}$).	103
5.6.10	Prompt for part prediction.	104
6.3.1	Our evaluation framework overview. Given an input question and the corresponding input image, we first extract text tokens Q as well as visual tokens V , which are constrained to have a fixed size through a module called <i>memory adapter</i> . These tokens are then concatenated along the sequence dimension and fed to the reasoning module. Finally, we decode the answer using several heads attached to the reasoning module and a specific head given the predicted question type. During training, only the parameters of the reasoning module are trained (above a dashed line), while the rest of the pipeline remains frozen.	110
6.3.2	Accuracy on CLEVR against the number of parameters in the reasoning module when using perfect visual information. Below a minimal complexity in the reasoning module, even having perfect visual information is insufficient to solve the VQA task. . . .	113
6.4.1	Results of the low-shot VQA experiments on CLEVR VQA dataset. We visualize the results for 2 memory size regimes, M_1 (top) M_2 (bottom), and the visual encoders with the best overall performance.	121
7.0.1	Examples of property pages on Booking.com	125
7.1.1	Heatmap of most popular topics (x-axis) extracted from reviews for different traveller types at Booking.com. We note that the ranking of the most mentioned topics differs among traveller segments, making reviews a valuable signal for user segment personalization.	127
7.1.2	Overview of our method. First, we extract image embeddings and cluster them to obtain K ($K=4$ in this case) semantically separated groups of images. Then, for each cluster, we calculate the similarities between all the images within the cluster and the topics extracted from reviews of the specific segment u (here $u = \text{Couple}$). Finally, the selected images are the ones with the highest similarity to any of the topics.	128

7.3.1	Results of our user studies. We report the results separately for each user segment (x-axis).	136
7.4.1	Example results of summaries of $K = 8$ images for one of the properties. We compare Def (top), and personalized methods (bottom) on this property’s visual summarization. The presented personalization examples here are for a <i>Ski</i> trip type. We highlight in red the selected images which are relevant to this user segment.	138

List of Tables

3.4.1	Zero-shot open-vocabulary segmentation. Comparison of our approach to the state of the art (under the mIoU metric). While our method does not need any additional training it performs significantly better than the current SOTA on PASCAL VOC (+4.9) and performs on par with its competitors on COCO object, ranking 3rd on COCO. We mark with [†] results from [75]. All methods are evaluated considering that background is a class of the dataset. We note with * when more than one backbone was used, we refer here to CLIP-like backbones. GroupViT and ViewCo use a 12 Transformer layers backbone following [13], noted 12T.	43
3.4.2	Ablation studies. We find that while all components are improving the performance of our method, objectness is particularly critical. Numbers are reported using a ViT-B/32 backbone.	44
3.4.3	Ablation of our multi-scale approach. To validate our multi-scale design, we report the results of our method with different sets of scales, by progressively adding more fine-grained scales. We find empirically that after scale 3, adding more scales gives similar or worse results. We also report results without the first scale $s = 0$ (global scale), which leads to worse results. Numbers are reported using a ViT-B/32 backbone.	46

3.4.4	Comparison of different objectness methods used for objectness-guided fusion. We find that the version of FOUND that was trained <i>in the original paper</i> performs the best, and therefore keep this method as our objectness guide.	46
3.6.1	CLIP-DIY zero-shot performance (IoU) on the 21 classes from Pascal VOC. The background class is denoted as \square	54
4.4.1	Open-vocabulary semantic segmentation quantitative comparison using the mIoU metric. We separate in two columns the evaluation datasets: those without a ‘background’ prompt and those with (noted ‘W/ bkg prompt’), as discussed in Sec. 4.4.1. We report all methods without post-processing. We note with * methods for which we computed scores; we obtained MaskCLIP* scores with OpenCLIP [54] and mark with † the use of MaskCLIP refinement. The first and second best methods are, respectively, bold and <u>underlined</u> . We specify if a method assumes prior access to names of concepts (‘Concept spec.’) and if it employs a frozen backbone (❄). We specify what additional data is used at training (‘Extra data’) (‘I’ stands for images and ‘IT’ for image/text aligned data). Our CLIP-DINOiser only needs 1k images from ImageNet to be trained. ‘SD’ stands for Stable Diffusion [154]. We refer to Sec. 4.4.1 for more details on baselines and we detail the datasets used for training by each method in the supplementary material.	67
4.4.2	Impact of the pooling strategy (a) and background detection (b) on diverse datasets reported with the mIoU metric.	69
4.6.1	Performance with different training datasets. When using random splits extracted from ImageNet (noted ‘IN’), we report the average score and standard deviation computed over training with three random splits (of 1k or 10k) extracted in ImageNet. In (a), we report the scores on the datasets without ‘background’ class, and in (b) with.	71

4.6.2	Results of single object discovery and unsupervised saliency detection obtained when following FOUND [100] protocol. We compute the single object discovery scores on classic VOC benchmarks [88] and 20k images of COCO (noted ‘C20k’) following [100] and use the CorLoc metric. We report the mIoU metric for unsupervised saliency detection and provide all results with the post-processing bilateral solver. We note ‘DUT-O.’ DUT-OMRON [156] and ‘DUTS-T.’ stands for DUTS-TEST [155].	72
4.6.3	Details on additional data required for training.	73
5.4.1	Benefits of \mathcal{CC} measured in IoU-single. ‘*’ indicates that the method’s original background handling is applied, if any and provided it gives the best results. Note that CAT-Seg input resolution is 640x640, whereas it is 448x448 for all the other methods. We note \mathcal{CC}^{PI} the unrealistic setup where we have access to all of the dataset classes and use them as systematic contrastive concepts (except for VOC, as its annotations do not cover all pixels). Please note that \mathcal{CC}^{BG} is our baseline.	88
5.4.2	Results w/ mIoU.	89
5.4.3	Ablation studies. (a) The impact of filtering steps: ‘co-occ.’ is the co-occurrence-based filtering; ‘no abs.’ is the removal of abstract concepts; ‘sem. sim.’ is the semantic-similarity filtering. (b) Relevance of adding “background” to \mathcal{CC}^L . (c) Varying the pre-training dataset.	90
5.6.1	Results with standard mIoU metric when employing different contrastive concept generation strategies. ‘*’ denotes our implementation, ‘†’ denotes results from TCL [75], and ‘MaskCLIP (+keys)’ denotes keys refinement proposed in the original paper [28]. Training datasets include CC12M [159], RedCaps [160], ImageNet [6], CC3M [161].	95
5.6.2	Results on all datasets with our IoU-single. ‘*’ denotes the result when the original background handling gives the best results.	96
5.6.3	Parameter study of γ and δ. Selection (marked in grey) of the hyperparameters γ and δ with IoU-single on 100 randomly-selected images in ADE20k training dataset.	98
5.6.4	Selection of β with classic mIoU on 100 randomly-selected images in the VOC training dataset. Results are reported for \mathcal{CC}^L	98

5.6.5	Ontology-based (WordNet) filtering out synonyms, meronyms, hyponyms and hypernyms (at depth 1) from \mathcal{CC}^D . Results are reported on ADE20K, as %IoU-single.	101
5.6.6	Example of LLM-generated \mathcal{CC}^L for Cityscapes.	103
6.4.1	Evaluation setups for different memory regimes. We conduct our studies in 2 memory regimes, M_1 and M_2 . We also provide the raw feature size (corresponding to a memory regime M_∞) to highlight the extent of compression and expansion applied at the adaptation stage.	117
6.4.2	Results on the CLEVR VQA dataset in 2 memory size regimes. We report scores on the validation set with a detailed split by the question type. We report average accuracy on the validation set. . .	117
6.4.3	Overall results on CLEVR VQA in 2 memory size regimes and with no memory restriction. We report average accuracy on the validation set.	118
6.4.4	Overall results on CLEVR-Math in 2 memory size regimes and with no memory restriction.	118
6.4.5	The study of the influence of the text encoder on the CLIP performance on the CLEVR VQA dataset. We report average accuracy over all the questions in the validation set.	120
6.6.1	DINO ViT performance comparison on CLEVR VQA dataset.	124
7.2.1	Dataset split in detail. Overall we collected more than 6000 samples from the platform. Through stratified sampling, we make sure the distribution in terms of location, rating, and accommodation type reflects the real distribution.	132
7.3.1	Quantitative evaluation. We report the results for two different dataset splits: small and big galleries and the number of reviews. .	135
7.6.1	Exemplary image classes per user segment.	140