Towards Generalized Multimodal Foundation Model

by

Xueyan Zou

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

(Computer Sciences)

at the

UNIVERSITY OF WISCONSIN–MADISON

2024

Date of final oral examination: 04/15/2024

The dissertation is approved by the following members of the Oral Committee:
Pedro Morgado, Assistant Professor, Electrical and Computer Engineering
Yudong Chen, Assistant Professor, Computer Sciences
Jianwei Yang, Principle Research Scientist, Microsoft
Yong Jae Lee (Advisor), Associate Professor, Computer Sciences

Acknowledgments

I would like to express my deepest gratitude to everyone who has contributed to the completion of this doctoral thesis. This research journey has been a transformative and enlightening experience, and I am fortunate to have received guidance, support, and encouragement from all individuals and institutions.

Firstly, I would like to express my immense gratitude to my advisor, Professor Yong Jae Lee. His knowledge, guidance, and patience with high standards have been the cornerstone throughout my Ph.D. career. I appreciated the opportunity he gave me to directly enter research life after undergraduate studies. With his supervision from every perspective, I gradually became a mature researcher.

Secondly, along the way, I met many unbelievably great mentors. Their hands-on mentoring truly nurtured my research interests, tastes, and skills. I would like to sincerely thank Jianwei Yang for giving me the opportunity to ascend to a new level of research. Additionally, I am grateful to Fanyi Xiao, who cultivated and supported me when I was a junior student for the very first publication. Furthermore, I would like to thank Zhiding Yu, who introduced me to the field of computer vision research. Lastly, I want to thank Guanya Shi, whose passion for AI ten years ago inspired me to switch my major from economics to computer science.

Thirdly, during my years as a Ph.D. student, many collaborators have constantly supported my research. I would like to express my deep appreciation for them, including but not limited to Hao Zhang, Feng Li, Shilong Liu, Yuheng Li, Haotian Liu, Utkarsh Ojha, Mingyu Ding, Xiaoyu Xiang, Junyi Wei, and others.

Finally, my heartfelt appreciation goes to my family, especially my parents and grandparents, although we haven't met for five years. Their selfless love, and my deep love for them, remain one of my strongest supports. Additionally, I would like to thank all my roommates during my Ph.D., including Yinling Zhang, Shuyun Yuan, and Meng Li. Their emotional support and deep trust in me have helped me through many difficult days. I would also like to appreciate the people I have loved along the way; they have taught me to see the world from a perspective different from research. Lastly, I would like to thank myself for never giving up during those hard days in the past six years.

I'm also hugely grateful to my dissertation committee, Prof. Pedro Morgado, Prof. Yudong Chen, Dr. Jianwei Yang, and Prof. Yong Jae Lee. Their expertise, constructive

feedback, and insights have greatly improved my thesis.

To everyone who helped with this thesis, whether I mentioned you or not, your support and encouragement have been invaluable. I am deeply grateful for all of you being part of my life.

Contents

List of Tables

List of Figures

Abstract

This thesis contributes significantly to the field of multimodal foundation models by developing a generalist approach to visual understanding that spans various levels of granularity and modality. Previously, the study of visual understanding has progressed from classical image recognition, characterized by discrete categorization using human-defined labels, to a more integrated approach that combines insights from natural language understanding. This evolution has facilitated advancements in open-vocabulary visual understanding, highlighting the synergy between textual and visual data as it occurs in human perception.

Recent developments have seen a shift towards creating comprehensive, specialized models for individual tasks. However, the focus has increasingly turned towards constructing a generalist model capable of handling multiple tasks across different granularities and modalities. Such generalist models boast not only parameter efficiency, allowing a single model architecture to adapt to various tasks, but also data efficiency, where the model parameters are optimized across diverse datasets.

The recent advancement of this research trend is the creation of steerable models that align closely with human cognitive processes, enabling direct and intuitive interactions between humans and AI systems. Throughout my doctoral studies, I have engaged with these evolving trends, progressively working towards the realization of effective and efficient generalist models. This thesis details these developments, emphasizing both the theoretical advancements and practical applications in the field of visual understanding.

Chapter 1

Introduction

The burgeoning field of visual understanding has evolved from classical image understanding, where machines were confined to a language of their own, to the advent of steerable models that delve into the human linguistic realm. This roadmap is consistent with the human's ambitious AGI goal. However, along the formidable road, we must confront several key challenges before developing the "giant" foundation model: (1) How could machines understand the visual world precisely, efficiently, and consistently. (2) How to align the machine's knowledge space of the visual world with the human knowledge space. (3) Spanning the visual understanding task, how can we develop a generalized system with mutual benefits. (4) How do we bridge human communication with machine language.

Along the roadmap, my research vision is to distill human and world knowledge into a unified foundation model. In pursuit of this vision, my research agenda is centered around **building powerful machine models that are seamlessly aligned with human knowledge space supporting multimodality input and output that are easily accessible to assist human life**. These agendas are supported by driving deep neural networks to model the visual world, align with human knowledge, generalize to multiple tasks, and interact with humans. To that end, my Ph.D. research concentrates on four areas spanning the entire spectrum from modeling to deployment:

Classical Image Understanding My Ph.D. initially focuses on classical image understanding problems concentrated on image classification, object detection, and instance segmentation. These foundational tasks delve into the basic building blocks of how deep neural networks recognize images, objects, and pixels, marking the evolution from convolutional networks to transformers in this field. One significant challenge in this area, particularly when using convolutional networks, is the aliasing problem during feature downsampling, which leads to distortion of high-frequency information and inconsistent results. In my publication [218, 219], we introduced an adaptive anti-aliasing module that utilizes a convolution layer to predict a blurring kernel for each spatial location on the feature map, effectively preserving high-frequency information

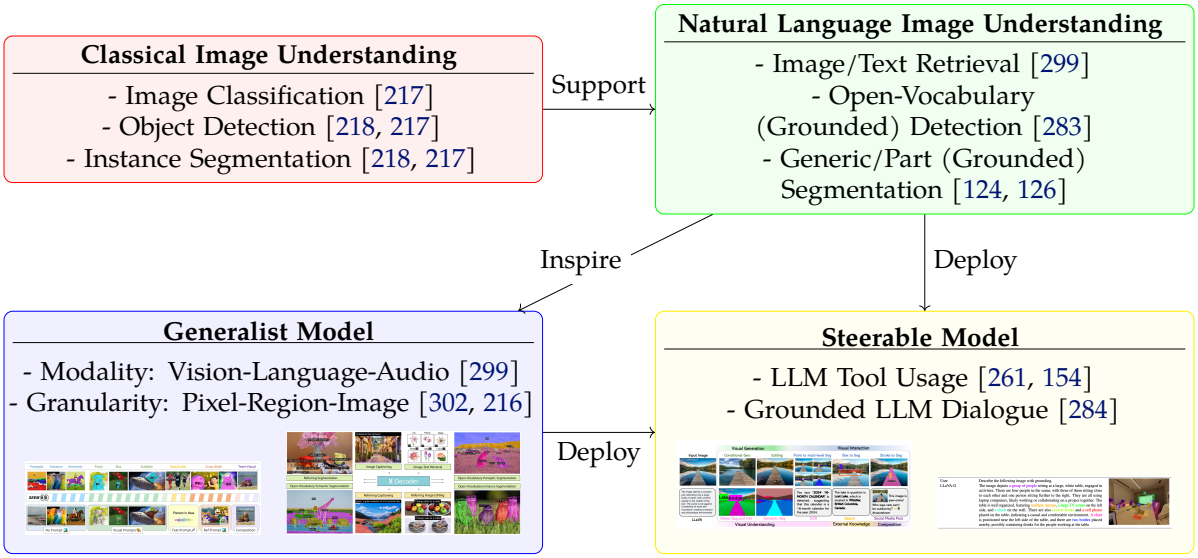


Figure 1.1: The connection between my research agendas to the overview of Ph.D. research.

and thereby enhancing accuracy and consistency across classification, detection, and segmentation tasks. Furthermore, with the development of transformers in image classification and object detection, there emerged a need for an end-to-end training system for instance segmentation. Where people are using the two-stage method with anchors and roi-align for segmentation and detection. Addressing this, my work [217] developed a transformer-based model capable of end-to-end processing for object detection, instance segmentation, and edge detection by employing a similarity mapping between object queries and image features, thereby contributing to the end-to-end training system for segmentation.

Natural Language Image Understanding After establishing a strong understanding of the visual world through deep neural networks, where models cluster images, pixels, etc. with similar traits and map them to predefined category IDs, the field has progressed towards integrating this machine-based understanding with natural language. This integration has led to the emergence of works in image-text retrieval, and open-vocabulary detection and segmentation. My work X-Decoder [299], explores these directions within a single framework, which is elaborated upon in the following section. Additionally, our project, OpenSEED [283], investigates object and pixel-level open vocabulary understanding, showing SoTA performance, particularly in pixel-level tasks. This is achieved by training open vocabulary object detection models to increase the vocabulary size and using conditioned mask decoding to bridge object detection with instance segmentation, thus the mapping from pixel to semantic is enriched. We have also ventured into more nuanced granularity with Semantic-SAM [126], aligning both instance and part-level objects with semantic meanings. This approach has addressed the challenge of ambiguity in mapping a query to the correct granularity level (instance,

part), and further map those queries with natural language. In [124], we further fuse the vision and language embedding space to enable querying instances, part through visual context.

Generalist Model As specialist models in various domains of visual understanding evolve, it becomes evident that there is a significant overlap in the training methodologies and model architectures across tasks. My Ph.D. mainly focuses on creating a generalist model for multi-modality understanding, specifically focusing on pixel, region, and image level performance. This observation motivates me to develop X-Decoder [299], the first generalist model that can handle different levels of visual understanding (image, pixel, region) without additional overhead for multimodal input and output. We create X-Decoder through a transformer decoder, where different tasks share the same set of weights by adjusting self-attention masks between learnable queries and features. This framework has not only achieved 9 state-of-the-art results in open-vocabulary segmentation benchmark but has also laid the groundwork for subsequent innovations.

Building upon the X-Decoder, my work evolved to include human interaction in the model. In this context, I build SEEM [302], a generalist model that integrates human instruction for segmentation tasks, allowing users to query segments in an image using points, scribble, box, etc. SEEM achieves competitive performance across interactive segmentation, generic segmentation, referring segmentation, and video object segmentation on 9 datasets with minimum 1/100 supervision compared with SAM and other models. Moreover, the model demonstrates strong compositional abilities as well as referring image segmentation capability that is generalizable to video object segmentation. This evolution is further enhanced by integrating large language model (LLM) embeddings in the FIND framework, which unifies the tasks of X-Decoder and SEEM. FIND [216] is the first work that interfaces the pre-trained LLM weight with visual features for understanding tasks. In addition to the multi-modal capability, FIND also supports interleave segmentation, grounding, and retrieval. As shown in Fig. 1.2, the modality and granularity generalization procedure has been shown in a 2d coordinate system.

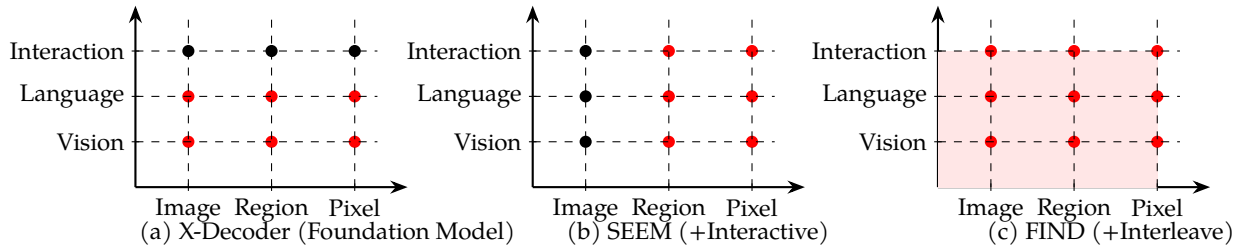


Figure 1.2: Visualization of task coverage of X-Decoder, SEEM, FIND span granularity (x-axis), and modality (y-axis). X-Decoder is the first foundation model building along the line, while SEEM extends to the interactive section, and FIND covers the whole interleave space.

Steerable Model After the vision and language model are well aligned with each other, I started to delve into the human-accessible interface for vision-language foundation models. The interface enables users to interact with the information encapsulated within these models, leading to the concept of steerable models. These models are designed to act as AI-assisted tools, allowing users to communicate with the foundation model through conversation. This approach is part of a growing trend to merge vision models with large language models (LLMs), facilitating information queries via LLMs. Building on this concept, I have developed LLaVA-Grounding [284] and LLaVA-Plus [154]. These are general-purpose multimodal assistants trained end-to-end to systematically enhance the capabilities of large multimodal models. LLaVA-Grounding, in particular, brings an innovative dimension by providing pixel-level grounding chat capabilities. It also enables a visual chat feature that supports interactive visual communication, marking a significant advancement in the realm of steerable models with multi-modal LLMs. For LLaVA-Plus, it supports multimodal large language models with tool usage capability. This area of research is rapidly evolving, with ongoing efforts to further explore and refine the potential of these models. It is the first attempt reported to combine end-to-end training and tool-chaining methods.

In addition to this line of research, I have also worked on generation models for video inpainting [303].

1.1 Related Work

In the realm of image classification, substantial advancements have been driven by deep learning models, predominantly Convolutional Neural Networks (CNNs). Seminal models like AlexNet [113] and later, more sophisticated architectures such as VGG [205] and ResNet [79], have set foundational benchmarks in the field. These models employ deep layers and innovative training techniques to capture complex image features at various scales, dramatically enhancing classification accuracy. More recently, the introduction of attention mechanisms, specifically through the Transformer architecture, has been explored in models like Vision Transformer (ViT) [52], which adapts the self-attention-based architecture originally used in natural language processing to image classification tasks. This shift represents a significant move towards non-convolutional, purely attention-based models that manage to perform at or above par with established CNNs on standard benchmarks like ImageNet. These developments illustrate a pivotal transition in image classification methodologies, focusing on deeper, more computationally efficient architectures capable of leveraging large-scale datasets.

Later, people start to focus on instance segmentation. Instance segmentation in computer vision has evolved from the early classification of bottom-up segments [64, 73, 44, 74] to sophisticated frameworks like the RCNN series, culminating in the widely adopted

Mask R-CNN [75]. Efforts to refine these models have focused on enhancing efficiency and effectiveness through lighter, anchor-less, one-stage detectors [12, 220, 81] and advanced boundary delineation techniques using polygons [19, 140] or polar representations [251]. Recent developments include MaskFormer [37] and Mask2Former [34], which utilize a dense prediction head similar to our approach, though our research primarily addresses instance edge detection combined with bounding box and instance segmentation. However, adapting these transformer-based models for such tasks has proven challenging due to low detection accuracy, indicating a need for significant methodological adjustments. Additionally, our method employs a point supervised edge detection loss, differing from approaches that sample points on predicted feature maps [104, 36] or use sparse sampling for panoptic tasks [136], highlighting a unique application of point supervision in our work.

While image classification and segmentation models evolve, integrating with natural language models becomes a new trend. Vision-language (VL) pretraining has effectively enhanced performance across various VL tasks, evolving from using transformers with pre-extracted object features [30, 134, 289, 5] to employing end-to-end transformers that process raw image pixels [101, 128, 54]. Recent studies have explored the benefits of large-scale image-text data in visual representation learning, including applications in zero-shot image classification, action recognition, and image generation [241, 238, 206, 187, 93, 277, 271, 138]. Moreover, these pre-trained models are expanding into region-level tasks like phrase grounding and open-vocabulary object detection [97, 68, 294, 172], with proposed unified frameworks integrating image-text pairs with region-level data [16, 130, 285, 266, 53]. The developments and applications of VL pretraining are comprehensively reviewed in [58].

Moreover, with a comprehensive understanding of vision and language, the generalist and steerable model attracts new attention. Recent advancements in foundation models have significantly impacted various domains, including computer vision [277], natural language processing [226, 46, 13, 176], and multimodal interactions [3, 127, 267]. Notable examples include GPT-3 [13], which excels in natural language tasks, and Florence [277, 249], adaptable to multiple computer vision applications. The model Flamingo [3] integrates vision and language models through cross-attention, while BLIP-2 [127] enhances vision-language pre-training with an innovative two-stage approach. These developments suggest the potential for unifying language and vision models in a shared embedding space, moving beyond traditional multi-modal methods that integrate separate model outputs for enhanced interaction.

1.2 Contribution and Thesis Outline

This doctoral thesis makes significant contributions to the field of multi-modal foundation model, with a specific focus on four key research topics: *Classical Image Understanding*, *Natural Language Image Understanding*, *Generalist Model*, and *Steerable Model*. The central contribution of this thesis lies in the development of novel methodologies and theoretical insights to address these challenges. In the thesis, we will focus on classical image understanding and the generalist approach.

In terms of the thesis outline: Chapter I investigates classical image understanding with a focus on the work [218] and [217]. Later in Chapter II, we start to focus on the methods building towards the generalist model, such as image understanding spanning the granularity from pixel, region, to image and modality from vision to language. In this section, we focus on the works including [298, 302, 216].

Part I

Classical Image Understanding

Chapter 2

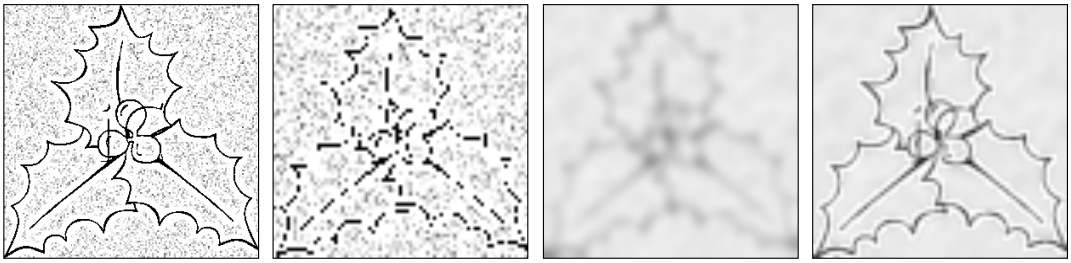
Delving Deeper into Anti-aliasing in ConvNets

Publication Statement. This chapter is joint work with Xueyan Zou, Fanyi Xiao, Zhiding Yu, Yuheng Li, Yong Jae Lee. The paper version of this chapter appeared in IJCV and BMVC20 [218, 219].

Aliasing refers to the phenomenon that high frequency signals degenerate into completely different ones after sampling. It arises as a problem in the context of deep learning as downsampling layers are widely adopted in deep architectures to reduce parameters and computation. The standard solution is to apply a low-pass filter (e.g., Gaussian blur) before downsampling [291]. However, it can be suboptimal to apply the same filter across the entire content, as the frequency of feature maps can vary across both spatial locations and feature channels. To tackle this, we propose an adaptive content-aware low-pass filtering layer, which *predicts separate filter weights for each spatial location and channel group* of the input feature maps. We investigate the effectiveness and generalization of the proposed method across multiple tasks, including image classification, semantic segmentation, instance segmentation, video instance segmentation, and image-to-image translation. Both qualitative and quantitative results demonstrate that our approach effectively adapts to the different feature frequencies to avoid aliasing while preserving useful information for recognition.

2.1 Introduction

Deep neural networks have led to impressive breakthroughs in visual recognition, speech recognition, and natural language processing. On certain benchmarks such as ImageNet and SQuAD, they can even achieve "human-level" performance [173, 77, 214, 189]. However, common mistakes that these networks make are often quite *unhuman*



(a) Input (b) 4x Down (c) Gaussian + 4x Down (d) Adaptive + 4x Down

Figure 2.1: **Toy example demonstrating the effect of adaptive filtering for anti-aliasing.** (a) Input image. (b) Result of direct downsampling. (c) Result of downsampling after applying a single Gaussian filter tuned to match the frequency of the noise. (d) Result of downsampling after applying spatially-adaptive Gaussian filters (stronger blurring for background noise and weaker for edges).[1]

like. For example, a tiny shift in the input image can lead to drastic changes in the output prediction of convolutional neural networks (ConvNets) [200, 8, 214]. This phenomenon was demonstrated to be in part due to *aliasing* when downsampling in ConvNets [291].

Aliasing refers to the phenomenon that high frequency information in a signal is distorted during subsampling [65]. The Nyquist theorem states that the sampling rate must be at least twice the highest frequency of the signal in order to prevent aliasing. Without proper anti-aliasing techniques, a subsampled signal can look completely different compared to its input. Below is a toy example demonstrating this problem on 1D signals:

$$001100110011 \xrightarrow[\text{maxpool}]{\text{k=2, stride=2}} 010101 \tag{2.1}$$

$$011001100110 \xrightarrow[\text{maxpool}]{\text{k=2, stride=2}} 111111 \tag{2.2}$$

Here k is the kernel size (1×2). Because of aliasing, a one position shift in the original signal leads to a completely different sampled signal (bottom) compared to the original sampled one (top). As downsampling layers in ConvNets are critical for reducing parameters and inducing invariance in the learned representations, the aliasing issue accompanying these layers will likely result in a performance drop as well as undesired shift variance in the output if not handled carefully.

[1]We generate the background impulse noise using a Bernoulli distribution (with $P = 0.5$) per pixel location with a normal distribution determining the impulse noise magnitude. We then overlay the foreground image over the background noise. For (c), the fixed filter value is generated by $g(x, y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$, where σ is the standard deviation of the Gaussian filter, and (x, y) is the index of the filter location with $(0, 0)$ as the filter center. For (d), the filter "strength" is varied by σ as well as the kernel size.

To tackle this, Zhang [291] proposed to insert a Gaussian blur layer before each downsampling module in ConvNets. Though simple and effective to a certain degree, we argue that the design choice of applying a universal Gaussian filter is not optimal – as signal frequencies in a natural image (or feature map) generally vary throughout spatial locations and channels, different blurring filters are needed in order to satisfy the Nyquist theorem to avoid aliasing. For example, the image in Fig. 2.1 (a) contains high frequency impulse noise in the background and relatively lower frequency edges in the foreground. Directly applying a downsampling operation produces discontinuous edges and distorted impulse noise shown in (b) due to aliasing. By applying a Gaussian filter before downsampling, we can avoid aliasing as shown in (c). However, as the high frequency impulse noise needs to be blurred more compared to the lower frequency edges, when using a single Gaussian filter tuned for the impulse noise, the edges are over-blurred leading to significant information loss. To solve this issue, what we need is to apply different Gaussian filters to the foreground and background separately, so that we can avoid aliasing while preserving useful information, as in (d).

With the above observation, we propose a *content-aware anti-aliasing* module, which adaptively predicts low-pass filter weights for different spatial locations. Furthermore, as different feature channels can also have different frequencies (e.g., certain channels capture edges, others capture color blobs), we also predict different filters for different channels. In this way, our proposed module adaptively blurs the input content to avoid aliasing while preserving useful information for downstream tasks. To summarize, our contributions are:

- We propose a novel adaptive and architecture independent low-pass filtering layer in ConvNets for anti-aliasing.
- We propose novel evaluation metrics, which measure shift consistency for semantic and instance segmentation tasks; i.e., a method's robustness to aliasing effects caused by shifts in the input.
- We conduct experiments on image classification (ImageNet), semantic segmentation (PASCAL VOC and Cityscapes), instance segmentation (MS-COCO), video instance segmentation (YoutubeVIS), and domain generalization (ImageNet to ImageNet VID, COCO to YoutubeVIS). The results show that our method outperforms competitive baselines with a good margin on both accuracy and shift consistency.
- We demonstrate intuitive qualitative results, which show the interpretability of our module when applied to different spatial locations and channel groups.

This chapter expands upon our previous conference paper [301] with the following new contributions:

- We propose a novel consistency metric for video instance segmentation and evaluate the robustness of our approach to video natural perturbation on the YoutubeVIS dataset (Section 2.4.4 and 2.4.8.2).
- We conduct experiments on the image-to-image translation task using pix2pixHD [236] as a baseline. Results in Section 2.4.5 and 2.4.8.3 show that our approach can generate more realistic images both qualitatively and quantitatively.
- We identify our adaptive filtering layer as a variant of the sliding window self-attention in vision transformers (Section 2.3.4).
- We give more comprehensive related work analysis in Section 2.2.
- We discuss some limitations of our approach in Section 2.5.

2.2 Related Work

Anti-aliasing Aliasing is a well-known problem in signal processing, and lowpass filters are often designed according to the Nyquist theorem to counter it [201, 186]. In addition, the phenomenon has been studied under the scope of invariance in pattern recognition [243, 1, 15, 135]. More recently, it has been shown that aliasing also widely exists in deep neural networks and has non-negligible effect on the network predictions. For example, Zhang [291] made the observation that network predictions are not consistent to shifting inputs and pointed out that these phenomena are caused by aliasing when a feature map is downsampled. Our subsequent work [301] further proposed *adaptive* filtering layers in place of the fixed low-pass filtering layers proposed in [291] to better address the shift inconsistency problem. Recently, several concurrent works have either addressed aliasing issues in GANs using a continuous interpretation [100], or target the design of truly shift-invariant convnets with adaptive polyphase sampling [20]. Anti-aliasing is also highly related to geometric transformation invariance, which is explored in several recent works [292, 118, 11, 197].

Network Robustness Current deep neural networks are vulnerable to input perturbations without special training recipes. These perturbations can be malicious such as adversarial attacks [212, 114], or naturally occurring such as input translation [163, 10, 269, 291], natural perturbations [200], domain gaps [174, 123], or out-of-distribution samples [119, 120]. One underlying reason is that networks tend to pick up superficial patterns instead of learning truly compositional representations [60], and their vulnerability to input perturbations can also lead to prediction inconsistencies. Adversarial defense methods via novel training pipelines [162, 141], losses [98] and architectures [250] have been proposed to obtain adversarially robust networks. [163, 10] propose new algorithms to learn more shift-invariant representations. In addition, data

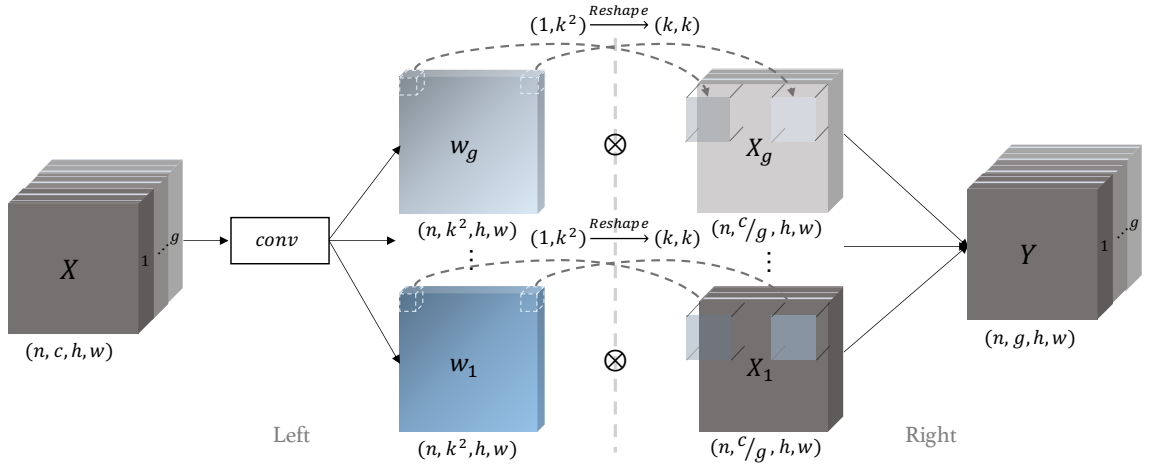


Figure 2.2: **Method overview.** (Left) For each spatial location and feature channel group in the input X, we predict a $k \times k$ filter w. (Right) We apply the learned filters on X to obtain content aware anti-aliased features. See text for more details.

augmentation is an effective way to improve network robustness [286, 291, 278] and generalization. Finally, domain generalization methods (e.g., [228, 227, 88]) have been proposed to increase a model's robustness to domain differences in the data.

Image Filtering Low-pass filters like box [196] and Gaussian [65] are classic *content agnostic* smoothing filters; i.e., their filter weights are fixed regardless of spatial location and image content. Bilateral [180] and guided [76] filters are *content aware* as they can simultaneously preserve edge information while removing noise. Recent works integrate such classic filters into deep networks [291, 250]. However, directly integrating these modules into a neural network requires careful tuning of hyperparameters subject to the input image (e.g., σ_s and σ_r in bilateral filter or r and ϵ in guided filter). [210, 94] introduced the dynamic filtering layer, whose weights are predicted by convolution layers conditioned on pre-computed feature maps. Our method differs from them in two key aspects: 1) our filter weights vary across both spatial and channel groups, and 2) we insert our low-pass filtering layer before every downsampling layer for anti-aliasing, whereas the dynamic filtering layer is directly linked to the prediction (last) layer in order to incorporate motion information for video recognition tasks. Finally, [232] introduces an adaptive convolution layer for upsampling, whereas we focus on downsampling with an adaptive low-pass filtering layer.

Applications The application of anti-aliasing covers a variety of visual recognition tasks, ranging from classification [45], dense prediction [75, 22], video analysis [263] to generation tasks [236]. We find that anti-aliasing techniques are especially effective for dense prediction tasks including instance segmentation [75, 12] and semantic segmentation [158, 24]. These tasks require precise modeling of object boundaries, so that

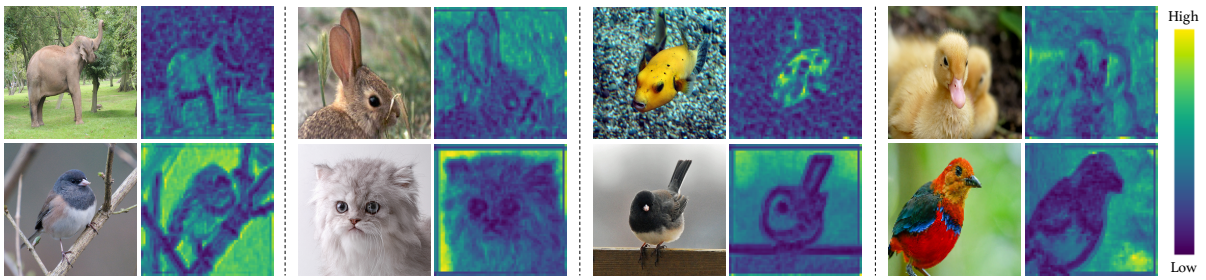


Figure 2.3: **Variance of the learned filter weights across spatial locations.** Low variance corresponds to more blur, while high variance corresponds to less blur. Our model correctly learns to blur high frequency content (e.g., edges) more to prevent aliasing, and blur low frequency content less to preserve useful information.

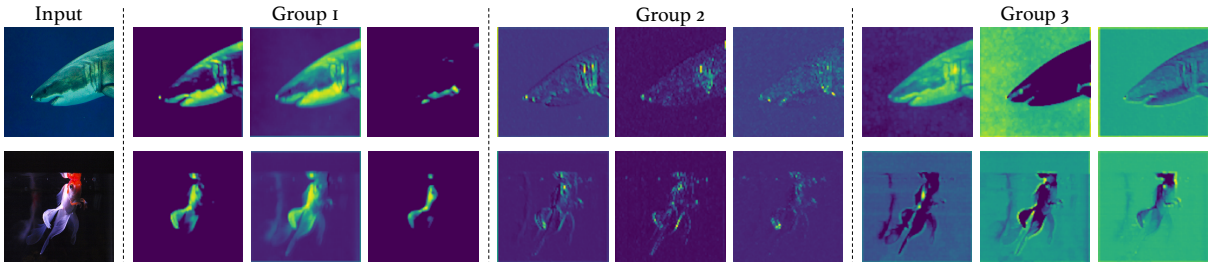


Figure 2.4: **Visualization of predicted feature maps within and across groups.** The features within each group are more similar to each other than to those in other groups. Each group captures a different aspect of the image (e.g., edges, color blobs).

pixels from the same object instance can be correctly grouped together. Thus, while blurring can help reduce aliasing, it can also be harmful to these tasks (e.g., when the edges are blurred too much or not blurred enough hence resulting in aliasing). We investigate the effect of anti-aliasing in these pixel-level tasks, whereas our closest work, [291], focused mainly on image classification. In addition, video insconsistency caused by motion blur, natural perturbations, etc. has also been widely observed [200, 69, 131]. We specifically explore the consistency problem in video instance segmentation [263] to demonstrate the effectiveness of our approach. Finally, generative models also have sampling operations in their encoder and/or decoder architecture [236, 193, 181]. Thus, we also investigate our approach in this area.

2.3 Approach

To enable anti-aliasing for ConvNets, we apply the proposed *content-aware anti-aliasing* module before each downsampling operation in the network. Inside the module, we first generate low-pass filters for *different spatial locations and channel groups* (Fig. 5.2 left), and then apply the predicted filters back onto the input features for anti-aliasing (Fig. 5.2 right).

2.3.1 Spatial adaptive anti-aliasing.

As frequency components can vary across different spatial locations in an image, we propose to learn different low-pass filters in a content-aware manner across spatial locations. Specifically, given an input feature X that needs to be down-sampled, we generate a low-pass filter $w_{i,j}$ (e.g., a 3×3 conv filter) for each spatial location (i, j) on x. With the predicted low-pass filter $w_{i,j}$, we can then apply it to input X:

$$Y_{i,j} = \sum_{p,q \in \Omega} w_{i,j}^{p,q} \cdot X_{i+p,j+q}, \tag{2.3}$$

where $Y_{i,j}$ denotes output features at location (i, j) and Ω points to the set of locations surrounding (i, j) on which we apply the predicted smooth filter. In this way, the network can learn to blur higher frequency content more than lower frequency content, to reduce undesirable aliasing effects while preserving important content as much as possible.

2.3.2 Channel-grouped adaptive anti-aliasing.

Different channels of a feature map can capture different aspects of the input that vary in frequency (e.g., edges, color blobs). Therefore, in addition to predicting different filters for each spatial location, it can also be desirable to predict different filters for each *feature channel*. However, naively predicting a low-pass filter for each spatial location and channel can be computationally very expensive. Motivated by the observation that some channels will capture similar information [247], we group the channels into k groups and predict a single low-pass filter $w_{i,j,g}$ for each group g. Then, we apply $w_{i,j,g}$ to the input X:

$$Y_{i,j}^{g} = \sum_{p,q \in \Omega} w_{i,j,g}^{p,q} \cdot X_{i+p,j+q}^{c}, \tag{2.4}$$

where g is the group index to which channel c belongs. In this way, channels within a group are learned to be similar, as shown in Fig. 2.4.

2.3.3 Learning to predict filters.

To dynamically generate low-pass filters for each spatial location and feature channel group, we apply a convolutional block (conv + batchnorm) to the input feature $X \in R^{n \times c \times h \times w}$ to output $w \in R^{n \times g \times k^2 \times h \times w}$, where g denotes the number of channel groups and each of the k^2 channels corresponds to an element in one of $k \times k$ locations in the filters. For grouping, we group every c/g consecutive channels, where c is the total number of channels. Finally, to ensure that the generated filters are low-pass, we

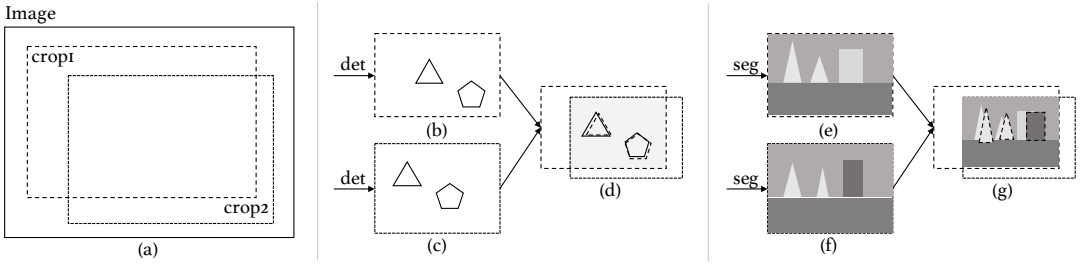


Figure 2.5: **Our new consistency metrics.** (b,c,d): mean Average Instance Segmentation Consistency (mAISC). (e,f,g): mean Average Semantic Segmentation Consistency (mASSC). Both metrics first crop two patches from the input image (a) and then perform detection/segmentation (det/seg) on its content (b,c,e,f). Then, the overlapping part from the two patches are selected out (d,g) for evaluating the consistency score.

constrain their weights to be positive and sum to one by passing it through a softmax layer.

2.3.4 Analyzing the predicted filters.

In this section, we analyze the behavior of our learned filters. First, we analyze how the filters spatially adapt to different image content. For this, we compute the variance of the learned filter weights across different spatial locations. A $k \times k$ average filter with $1/k^2$ intensity in each element will have zero variance whereas an identity filter with one in the center and zeros everywhere else will have high variance. From Fig. 2.3, one can clearly see that when the image content has high frequency information (e.g., elephant background trees, bird contours), the learned filters' variance tends to be smaller; i.e., more blur is needed to prevent aliasing. Conversely, the filters' variance is larger when the content is relatively smoother (e.g., background in bird images); i.e., less blur is needed to prevent aliasing. In this way, the learned filters can reduce aliasing during sampling while preserving useful image content as much as possible.

We next analyze how the filters adapt to different content across different feature groups. Fig. 2.4 shows this effect; e.g., group 1 captures relatively low frequency information with smooth areas, while group 2 captures higher frequency information with sharp intensity transitions. In this way, the learned filters can adapt to different frequencies across feature channels, while saving computational costs by learning the same filter per group.

2.3.5 Relation with self-attention

Recently, the transformer architecture [226] has emerged as a state-of-the-art alternative to convolutional networks on various vision tasks including classification [52], detection [18], and segmentation [252]. To deal with the transformer's quadratic complexity to input length, more efficient architectures such as the SwinTransformer [157] and

methods	Filter Size	accuracy Top-1 Abs	Top-5 Abs	Delta	consistency Abs	Delta	generalization Abs	Delta
ResNet-101 [79]	-	77.7	93.8	-	90.6	-	67.6	-
LPF [291]	3 x 3	78.4	94.1	+ 0.7	91.6	+ 1.0	68.8	+1.2
	5 x 5	77.7	93.9	+ 0.0	91.8	+ 1.2	67.0	- 0.6
Ours	3 x 3	**79.0**	**94.4**	**+ 1.3**	91.8	+ 1.2	**69.9**	**+2.3**
	5 x 5	78.6	94.3	+ 0.9	**92.2**	**+ 1.6**	69.1	+1.5

Table 2.1: **Image classification accuracy, consistency on ImageNet [45], and domain generalization results ImageNet → ImageNet VID [45].** We compare to strong ResNet-101 [79] and LPF (low-pass filter) [291] baselines. Our method shows consistent improvement in accuracy, consistency, and generalization.

LongFormer [9] have been proposed. Their key idea is to apply both sparse global and local attention. In this section, we show that our proposed anti-aliasing module can be interpreted as a form of sliding window local attention.

Given feature map X with dimension $h \times w \times d$, a sliding window local attention will apply local self-attention within each $k \times k$ feature patch window. It can be represented by the following equation:

$$Attention(x_c) = softmax(\frac{\phi_q(x_c)\phi_k(x)^T}{\sqrt{d}})\phi_v(x) \tag{2.5}$$

where x is the feature patch with size $k \times k \times d$, x_c is the center point of the feature patch x, and ϕ represents linear projection. The self-attention layer will first compute the cross similarity between x_c to each feature point in x (k^2 total), apply a softmax to normalize the similarity values to sum to one, and finally, use the resulting weights to compute a weighted sum over the projected values ($\phi_v(x)$) of the k^2 points.

In the above equation, we can consider replacing the linear projections ($\phi_q(\cdot)$ and $\phi_k(\cdot)$) and the dot product between them, with a conv layer to compute the summing weights:

$$Attention(x_c) = softmax(conv(x))\phi_i(x) \tag{2.6}$$

where ϕ_i is identity projection. This equation exactly represents our proposed anti-aliasing module.

In both cases (Eqns. 2.5 and 2.6), the output is a weighted sum of its input value tensor, and demonstrates that our approach can be viewed as a form of sliding window self-attention.

2.4 Experiments

We first introduce our experimental settings and propose consistency metrics for image classification, instance segmentation, and semantic segmentation. We compare to strong baselines including ResNet [79], Deeplab v3+ [24], Mask R-CNN on large

	Mask				Box			
method	mAP	Delta	mAISC	Delta	mAP	Delta	mAISC	Delta
Mask R-CNN [75]	36.1	-	62.9	-	40.1	-	65.1	-
LPF [291]	36.8	+ 0.7	66.0	+ 4.1	40.9	+ 0.8	68.8	+ 3.7
Ours	**37.2**	**+ 1.1**	**67.0**	**+ 5.1**	**41.4**	**+ 1.3**	**69.8**	**+ 4.7**

Table 2.2: **Instance segmentation results on MS COCO. We compare to Mask R-CNN [75] and LPF [291].** Our approach consistently improves over the baselines on both mask and box detection accuracy. Our model performs especially well on shift consistency, with a 5.1 and 4.7 point improvement over Mask R-CNN on mAISC mask and box, respectively.

	PASCAL VOC				Cityscapes			
method	mIOU	Delta	mASSC	Delta	mIOU	Delta	mASSC	Delta
Deeplab v3+ [24]	78.5	-	95.5±0.11	-	78.5	-	96.0±0.10	-
LPF [291]	79.4	+ 0.9	95.9±0.07	+ 0.4	78.9	+ 0.4	96.1±0.05	+ 0.1
Ours	**80.3**	**+ 1.8**	**96.0±0.13**	**+ 0.5**	**79.5**	**+ 1.0**	**96.3±0.07**	**+ 0.3**

Table 2.3: **Semantic segmentation on PASCAL VOC 2012 [55] and Cityscapes [43]. We compare to Deeplab v3+ [24] and LPF [291].** Our approach leads to a large improvement in accuracy on PASCAL VOC and Cityscapes (1.8 point and 1.0 point, respectively). Under the mASSC consistency metric, our approach also shows improvement upon the two baselines. The results are averaged over three runs.

scale datasets including ImageNet, ImageNet VID [45], MS COCO [145], PASCAL VOC [55] and Cityscapes [43]. We also conduct ablation studies on our design choices including number of groups, parameter counts, as well as filter types. Finally, we present qualitative results demonstrating the interpretability of our anti-aliasing module.

2.4.1 Image Classification

Experimental settings We evaluate on ILSVRC2012 [45], which contains 1.2M training and 50K validation images for 1000 object classes. We use input image size of 224×224, SGD solver with initial learning rate 0.1, momentum 0.9, and weight decay 1e-4. Full training schedule is 90 epochs with 5 epoch linear scaling warm up. Learning rate is reduced by 10x every 30 epochs. We train on 4 GPUs, with batch size 128 and batch accumulation of 2. For fair comparison, we use the same set of hyperparameters and training schedule for both ResNet-101, LPF [291] baselines as well as our method. The number of groups is set to 8 according to our ablation study. We extend the code base introduced in [291].

Consistency metric We use the consistency metric defined in [291], which measures how often the model outputs the same top-1 class given two different shifts on the same test image:

$$Consist = \mathbb{E}_{X,h_1,w_1,h_2,w_2}\, \mathbb{I}\{F(X_{h_1,w_1}) = F(X_{h_2,w_2})\} \tag{2.7}$$

where $\mathbb{E}$ and $\mathbb{I}$ denote expectation and indicator function (outputs $1/0$ with true/false inputs). X is the input image, h_1, w_1 (height/width) and h_2, w_2 parameterize the shifts and $F(\cdot)$ denotes the predicted top-1 class.

Results and analysis As shown in Table 2.1, our adaptive anti-aliasing module outperforms the baseline ResNet-101 without anti-aliasing with a 1.3 point boost (79.0 vs 77.7) in top-1 accuracy on ImageNet classification. More importantly, when comparing to LPF [291], which uses a fixed blurring kernel for anti-aliasing, our method scores 0.6 points higher (79.0 vs 78.4) on top-1 accuracy. Furthermore, our method not only achieves better classification accuracy, it also outputs more consistent results (+0.2/+0.4 consistency score improvements for 3×3 and 5×5 filter sizes) compared to LPF. These results reveal that our method preserves more discriminative information for recognition when blurring feature maps.

2.4.2 Domain Generalization

Experimental settings ImageNet VID is a video object detection dataset, which has 30 classes that overlap with 284 classes in ImageNet (some classes in ImageNet VID are the super class of ImageNet). It contains $3862/1315$ training/validation videos. We randomly select three frames from each validation video, and evaluate Top-1 accuracy on them to measure the generalization capability of our model which is pretrained on ImageNet (i.e. it has never seen any frame in ImageNet VID). As a video frame may contain multiple objects in different classes, we count a prediction as correct as long as it belongs to one of the ground-truth classes.

Results and analysis Table 2.1 reveals that our method generalizes better to a different domain compared to the ResNet-101 baseline ($+2.3\%$ points increase in top-1 accuracy for 3×3 filter) and LPF model ($+1.1\%$) which adopts a fixed blur kernel. We hypothesize that the better generalization capability comes from the fact that we learn a representation that is less sensitive to downsampling (i.e., more robust to shifts). This is particularly useful for video frames, as they can be thought of as having natural shift perturbations of the same content across frames [200].

2.4.3 Instance Segmentation

Experimental settings In this section, we present results on MS-COCO for instance segmentation [145]. MS-COCO contains 330k images, 1.5M object instances and 80 categories. We use Mask R-CNN [75] as our base architecture. We adopt the hyperparameter settings from the implementation of [168]. When measuring consistency, we first resize images to 800×800 and then take a crop of 736×736 as input.

Method	mAVISC $\alpha = 0.5$	Delta	$\alpha = 0.6$	Delta	$\alpha = 0.7$	Delta	$\alpha = 0.8$	Delta
Mask R-CNN [75]	90.09	-	89.29	-	88.14	-	87.29	-
LPF [291]	90.11	+ 0.02	88.96	- 0.33	88.38	+ 0.24	87.32	+ 0.03
Ours	**90.71**	**+ 0.62**	**89.61**	**+ 0.32**	**88.79**	**+ 0.65**	**87.68**	**+ 0.39**

Table 2.4: **Video instance segmentation consistency on YoutubeVIS [263].** We evaluate video instance segmentation consistency for IOU thresholds (α) ranging from 0.5 to 0.8. Our approach consistently increases video consistency with a good margin (+0.62/+0.32/+0.65/+0.39) for all IOU thresholds, whereas LPF increases it with a relatively smaller margin (+0.02/+0.32/+0.03) or can even decrease video consistency (-0.33 when $\alpha = 0.6$).

Consistency metric (mAISC) We propose a new mean Average Instance Segmentation Consistency (mAISC) metric to measure the shift invariance property of instance segmentation methods. As shown in Fig. 2.5, given an input image (a), we randomly select two crops (b) and (c), and apply an instance segmentation method on them separately. $M(b)$ and $M(c)$ denote the predicted instances in the overlapping region of image (b) and (c). To measure consistency, for any given instance m_b in $M(b)$ we find its highest overlapping counterpart m_c in $M(c)$. If the IOU between m_b and m_c is larger than a threshold (0.9 in our experiments), we regard m_b as a positive (consistent) sample in $M(b)$. (A sample m_c from $M(c)$ can only be considered a counterpart of any instance in $M(b)$ once.) We compute the final mAISC score as the mean percentage of positive samples in $M(b)$ over all input image pairs.

Results and analysis We evaluate mAP and mAISC for both mask and box predictions. As shown in Table 2.2, while simply applying a fixed Gaussian low-pass filter improves mAP by +0.7/+0.8 points for mask/box, our adaptive content-aware anti-aliasing module is more effective (further +0.4/+0.5 point improvement over LPF for mask/box). This demonstrates that it is important to have different low-pass filters for different spatial locations and channel groups. More interestingly, by introducing our adaptive low-pass filters, mAISC increases by a large margin (+5.1/+4.7 for mask/box over the baseline, and +1.0/+1.0 over LPF). This result demonstrates that 1) an anti-aliasing module significantly improves shift consistency via feature blurring, and 2) edges (higher frequency) are better preserved using our method (compared to LPF) during downsampling which are critical for pixel classification tasks.

2.4.4 Semantic Segmentation

Experimental settings We next evaluate on PASCAL VOC2012 [55] and Cityscapes [43] semantic segmentation with Deeplab v3+ [24] as the base model. We extend implementations from [82, 83] and [225]. For Cityscapes, we use syncBN with a batch size of 8. As for PASCAL VOC, we use a batch size of 16 on two GPUs without syncBN.

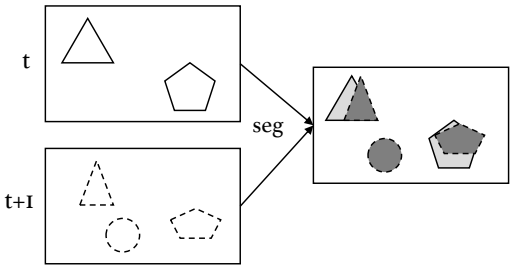


Figure 2.6: **Video instance segmentation consistency metric.** For any two consecutive frames, if the object is detected in both frames, we record it as a positive pair.

We report better performance compared to the original implementation for DeepLab v3+ on PASCAL VOC. For Cityscapes, our ResNet-101 backbone outperforms the Inception backbone used in [22].

Consistency metric (mASSC) We propose a new mean Average Semantic Segmentation Consistency (mASSC) metric to measure shift consistency for semantic segmentation methods. Similar to mAISC, we take two random crops (e,f) from the input image (a) in Fig. 2.5. We then compute the Semantic Segmentation Consistency between the overlapping regions X and Y of the two crops:

$$Consist(X, Y) = \mathbb{E}_{i\in[0,h)}\mathbb{E}_{j\in[0,w)}\, \mathbb{I}[S(X)_{i,j} = S(Y)_{i,j}] \tag{2.8}$$

where $S(X)_{i,j}$ and $S(Y)_{i,j}$ denote the predicted class label of pixel (i, j) in X and Y, and h, w is the height and width of the overlapping region. We average this score for all pairs of crops in an image, and average those scores over all test images to compute the final mASSC.

Results and analysis As shown in Table 2.3, our method improves mIOU by 1.8 and 1.0 points on PASCAL VOC and Cityscapes compared to the strong baseline of DeepLab v3+. Furthermore, our method also consistently improves the mASSC score (+0.5 and +0.3 for VOC and Cityscapes) despite the high numbers achieved by the baseline method (95.5/96.0). Finally, to measure the variance of our mASSC results, we report the standard deviation over three runs with different random seeds.

2.4.5 Video Consistency

Experimental settings We next validate our method's generalization to video data and its robustness to natural perturbations in video. For this, we perform the video instance segmentation task on the YoutubeVIS dataset [263] using the model trained in Section 2.4.2. We only evaluate on the 20 overlapping classes between COCO and YoutubeVIS.

datasets	Cityscapes						Facades			
methods	FID ↓	Delta	mIoU	Delta	mAcc	Delta	PSNR	Delta	SSIM	Delta
pix2pixHD [236] / pix2pix [91]	52.21	-	71.23	-	78.97	-	60.33	-	1.08	-
LPF [291]	52.68	+0.47	67.61	-3.62	75.61	-3.36	61.14	+0.81	1.37	+0.29
Ours [301]	**50.21**	**-2.0**	**71.99**	**+0.67**	**80.23**	**1.26**	**61.50**	**+1.17**	**1.41**	**+0.33**

Table 2.5: **Image-to-Image translation results.** On the Cityscapes dataset, the generated images of LPF have worse performance on both image quality and semantic segmentation, while the images generated by our approach tend to be more realistic (FID) and semantically accurate (mIoU, mAcc). On the Facades dataset, for shifted image pairs, our approach generates more consistent images compared to the baseline approaches for both pixel (PSNR) and patch (SSIM) metrics.

Since the validation set of YoutubeVIS does not have ground-truth annotation for all frames, we randomly select 260 videos in the training set to validate video consistency.

Consistency metric (mAVISC) To measure an instance segmentation model's robustness to natural perturbations in video, we propose a new mean Average Video Instance Segmentation Consistency (mAVISC) metric. For each video sequence, for all pairs of consecutive frames, and for each object that appears in each pair of frames, we first determine whether the object is detected according to a predetermined IOU threshold. If so, we record it as a positive pair, as shown in Fig. 2.6. Below is the equation for computing mAVISC:

$$\frac{1}{NM_iQ_i}\sum_{i=1}^{N}\sum_{j=1}^{M_i}\sum_{t=1}^{Q_i}\mathbb{I}\{\mathbb{I}\{IOU(GT_{i,j,t},P_{i,j,t})>\alpha\}= \mathbb{I}\{IOU(GT_{i,j,t+1},P_{i,j,t+1})>\alpha\}\} \tag{2.9}$$

where N, M_i, Q_i is the number of video sequences, objects in the i'th video, and frames in the i'th video, respectively. GT represents the ground truth video object bounding boxes, P represents the bounding box predictions, and α is the IOU threshold to determine whether the ground truth object is detected.

Results Table 2.4 shows video consistency results on the YoutubeVIS dataset for Mask R-CNN, Mask R-CNN with LPF, and Mask R-CNN with our approach using the proposed mAVISC metric. We evaluate on IOU thresholds ranging from $\alpha = 0.5$ to $\alpha = 0.8$. (We do not include $\alpha = 0.9$ because at this very strict threshold, there are too few correct detections for any method, making difficult to make reliable conclusions.) As shown in Table 2.4, our approach consistently increases video consistency with a good margin (+0.62/+0.32/+0.65/+0.39) across all IOU thresholds, where LPF increases with fairly small margin (+0.02/+0.32/+0.03) or even decreases video consistency (-0.33 when $\alpha = 0.6$).

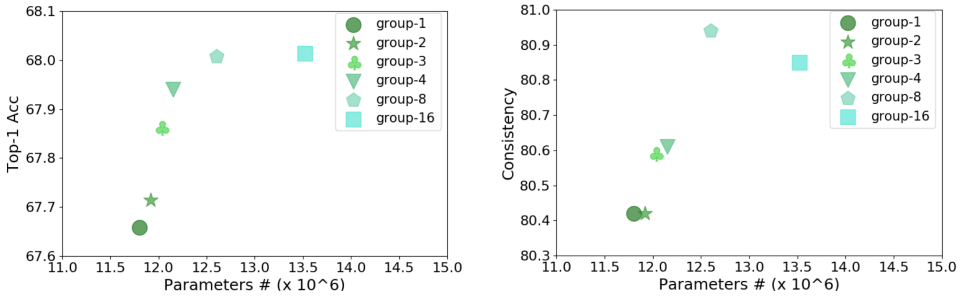


Figure 2.7: **Effect of number of groups on top-1 accuracy and consistency.** As the group number increases, both Top-1 accuracy and consistency first increase then decrease. The performance saturates with group number 8.

2.4.6 Image-to-Image Translation

Experiment Settings We evaluate image-to-image translation on the Cityscapes [43] and Facades [224] datasets using Pix2PixHD [236] and Pix2Pix [91] as the baseline models, respectively. On Cityscapes, following [236], we use 2976 images for training and 500 images for evaluation. On Facades, we use a total of 400 images for training and evaluation following [91]. For both Pix2Pix [91] and Pix2PixHD [236], we insert our module before each downsampling layer and upsampling layer following [291]. For downsampling, we simply insert our adaptive module with $stride = 2$. For upsampling, we first use nearest neighbor interpolation to upsample the feature map and then apply our adaptive filtering layer with $stride = 1$. We follow all the training settings from [236, 91].

On the Cityscapes dataset, we focus on image generation quality as well as our model's generalization capability to the segmentation task. We use mIoU, mAcc, and FID to evaluate the generated image quality. For mIoU and mACC, we first run the DeepLab V3+ semantic segmentation model (trained in Section 2.4.3) on the generated images, following [236]. We compare the resulting segmentation maps with the ground truth segmentation maps. For FID, we use the publicly available codebase at `https://github.com/mseitzer/pytorch-fid` to compare the distributions of the generated image features and the real image features. On the Facades dataset, we follow [291, 100] to evaluate the shift consistency of the image generation model. To evaluate the similarity of two shifted images, we compute both PSNR and SSIM to evaluate both pixel-wise and patch-wise similarity.

Results In Table 2.5, we first compare pix2pixHD [236], pix2pixHD together with LPF [291], and pix2pixHD with our approach on the Cityscapes dataset. Overall, our approach generates more realistic images (e.g., FID score decreases by 2 points) and has better mIOU and mAcc scores than both pix2pixHD and LPF. In addition, we compare pix2pix [91], pix2pix together with LPF [291], and pix2pix with our approach on the

methods	top-1 Acc	consistency
ResNet	66.5	79.1
Gaussian	66.7	79.8
Image Adaptive	66.7	78.7
Spatial Adaptive	67.7	80.3
Ours	**68.0**	**80.9**

Table 2.6: **Filter ablations.** Gaussian blur is better than no blur (ResNet). Learning the blur filter globally (Image Ada.), spatially (Spatial Ada.), and over channels (Ours) progressively does better.

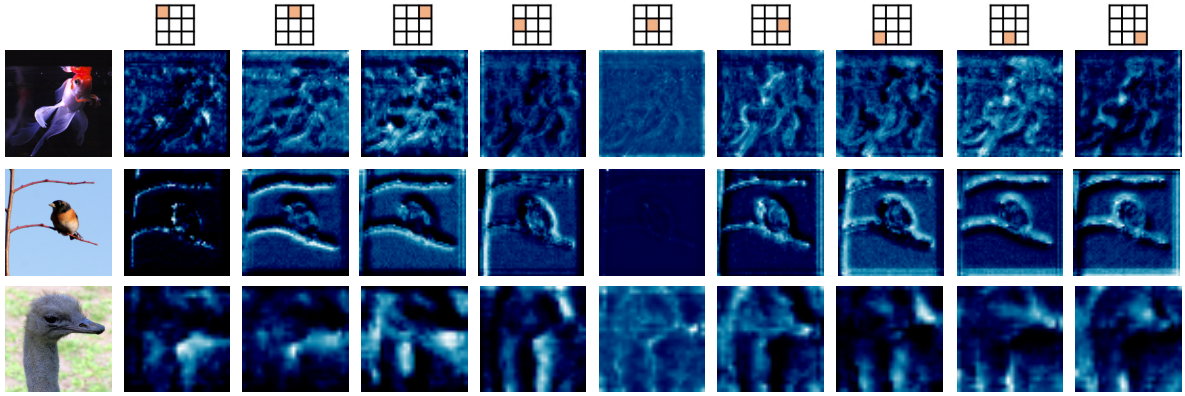


Figure 2.8: **Visualization of learned filter weights at each spatial location.** We can see that the learned filter weight is adaptive to different visual content. Specifically, our model tends to "grow" edges so that it is easier for them to be preserved. For example, the learned filter tends to integrate more information from left to right (see center-left and bottom-left weights in the second row of this figure) on the vertical tree branch and thus grow it to be thicker. This way, it is easier for the tree branch contours to be preserved after downsampling.

Facades dataset. The results show that our model is more consistent on image shift compared to the baseline approaches.

2.4.7 Ablation Studies

Experimental settings For efficiency, we perform all ablation studies using ResNet-18 with input image size 112×112 and batch size 200 on ImageNet. All other hyperparameters are identical to those used in Sec. 2.4.1.

Number of channel groups. We vary the number of channel groups and study its influence on image classification accuracy. As shown in Fig. 2.7, the trend is clear – increasing the number of groups generally leads to improved top-1 accuracy. This demonstrates the effectiveness of predicting different filters across channels. However, there exists a diminishing return in this trend – the performance saturates when the group number goes beyond 8. We hypothesize this is caused by overfitting.

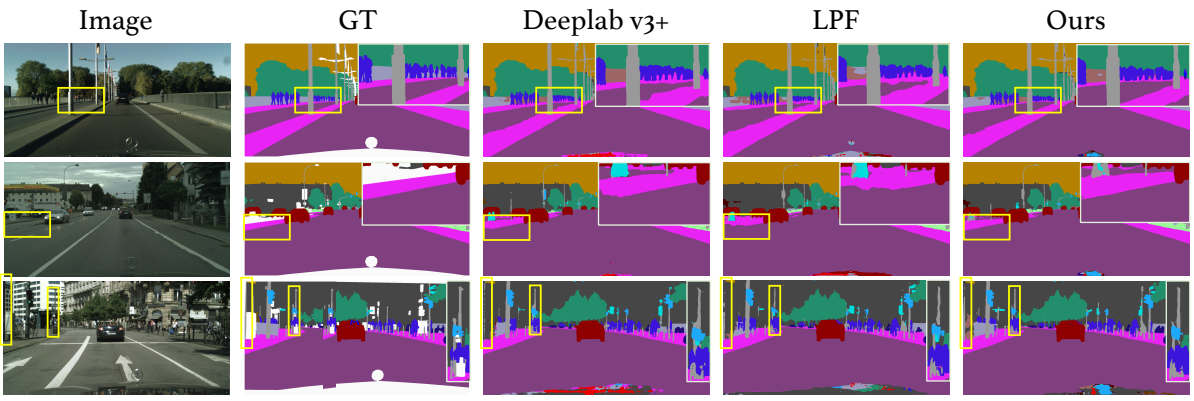


Figure 2.9: **Qualitative results for semantic segmentation on Cityscapes.** In the first row, within the yellow box region, our method clearly distinguishes the road edge compared to Deeplab v3+ and LPF. Similar behavior (better segmented road contours) is also observed in the second row. This holds for other objects as well – the light pole has better delineation compared to both baselines in the third row.

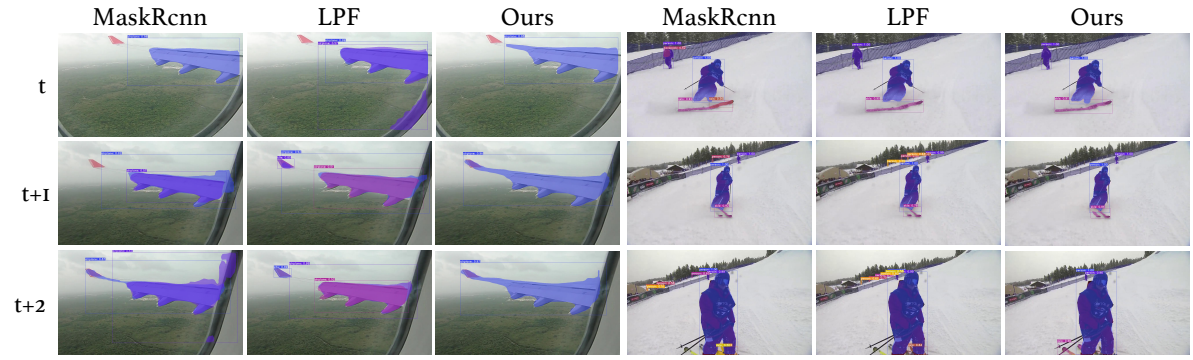


Figure 2.10: **Qualitative results of video instance segmentation consistency.** As shown in the first three columns, our method produces more consistent instance segmentation of the airplane wing whereas Mask-RCNN and LPF produce more inconsistent results that fluctuate over frames. In the last three columns, we observe the existence of redundant detections for both Mask-RCNN and LPF.

Number of parameters. We further compare the effects of directly increasing the number of parameters in the base network *vs* adding more groups in our content-aware low-pass filters. To increase the number of parameters for the base network, we increase the base channel size in ResNet-18. We find that directly increasing the number of parameters barely improves top-1 accuracy – when the number of parameters increases from 12.17M to 12.90M, top-1 accuracy increases only by 0.1%. Also, with similar (or less) number of parameters, our method yields a higher performance gain compared to naively increasing network capacity (68.0% *vs* 67.7% top-1 accuracy for 12.60M *vs* 12.90M parameters). This shows that our adaptive anti-aliasing method does not gain performance by simply scaling up its capacity.

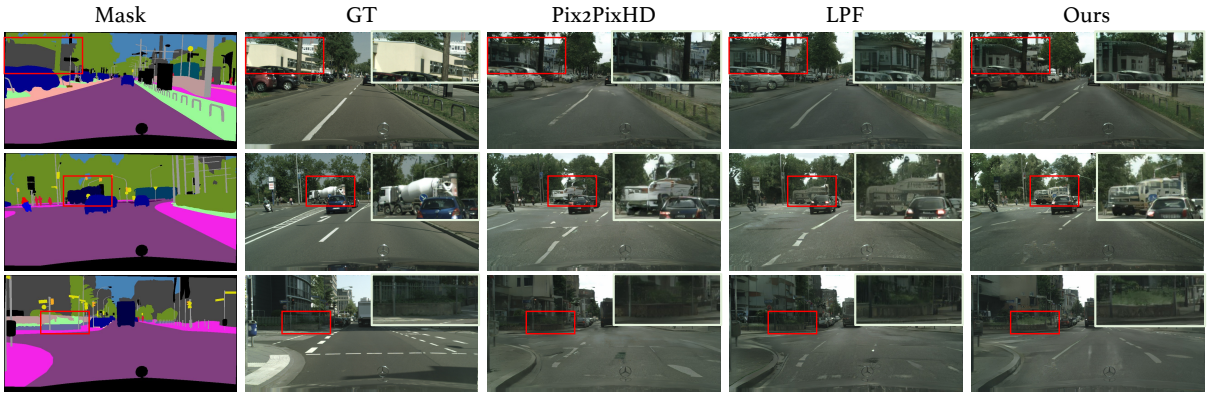


Figure 2.11: **Qualitative results for image to image translation on Cityscapes.** In the first row, our approach generates clear boundaries along the roof. In contrast, the other methods produce blurry boundaries. In the second row, our approach not only produces a clear edge on the car, it also generates a very tiny traffic light (see region inside the red rectangle). The other two methods fail in this situation. In the last row, our approach clearly identifies the boundary between the wall and bushes whereas the other two approaches' produce very blurry and dark generations.

Type of filter. In Table 2.6, we ablate our pixel adaptive filtering layers with various baseline components. Applying the same low-pass filter (Gaussian, Image Adaptive) across the entire image performs better than the vanilla ResNet-18 without any anti-aliasing. Here, Image Adaptive refers to the baseline which predicts a single low-pass filter for the entire image. By adaptively learning a spatially variant low-pass filter, performance improves further (Spatial Adaptive). Overall, our method achieves the best performance which demonstrates the benefits of predicting filters that are both spatially varying and channel adaptive.

Overhead. Finally, with our spatial/channel adaptive filtering added, the number of parameters increases by 2.9-7.8% for ResNet models (e.g., 4% for R-101, 4.5M to 4.63M). As for runtime, on a RTX2070 GPU, our method (R-101 backbone) takes 6.4 ms to forward a 224x224 image whereas a standard ResNet-101 takes 4.3 ms.

Type of Backbone. We compare Top-1 accuracy and Consistency with two additional backbone networks, VGG [205] and DenseNet-121 [86], on the Cifar-10 dataset [112]. For VGG, our approach achieves 94.0 Top-1 accuracy and 97.2 Consistency, and for DenseNet, our approach achieves 95.6 Top-1 accuracy and 97.4 Consistency. Similar to the ResNet101 results, our approach improves Top-1 accuracy with a good margin compared to the baseline network, which does not have any anti-aliasing (+0.6 for VGG and +1.7 for DenseNet) as well as LPF (+0.4 for VGG and +1.1 for DenseNet). Our method's consistency is also improved upon the baseline network (+0.6 for VGG and +0.1 for DenseNet) although it does not outperform the LPF method (-0.4 for VGG

and -0.9 for DenseNet). As Cifar-10 has relatively low resolution (32^2 pixels) images in comparison with ImageNet (224^2 pixels), there can be a trade-off between accuracy and consistency. Specifically, we find that decreasing the content frequency for anti-aliasing to improve shift consistency may have a side effect on classification accuracy when the image resolution is already very low. Thus, the consistency performance may not be improved as much in comparison with higher resolution images such as those in ImageNet, as we had shown in Table 2.1.

2.4.8 Qualitative Results

2.4.8.1 Semantic segmentation.

We show qualitative results for semantic segmentation in Fig. 2.9 to demonstrate that our module better preserves edge information. For example, in the first row, within the yellow box region, our method clearly distinguishes the road edge compared to Deeplab v3+ and LPF. Similar behavior (better segmented road contours) is also observed in the second row. This holds for other objects as well – the light pole has better delineation compared to both baselines in the third row.

2.4.8.2 Low-pass filter weights.

To further understand our adaptive filtering module, we visualize the low-pass filter weights for each spatial location. As shown in Fig. 2.8, our model tends to "grow" edges so that it's easier for them to be preserved. For example, the learned filter tends to integrate more information from left to right (see center-left and bottom-left weights in Fig. 2.8 in the second row) on the vertical tree branch and thus grow it to be thicker. This way, it's easier for tree branch contours to be preserved after downsampling.

2.4.8.3 Video instance segmentation consistency

In addition to image results, we also show qualitative results on a video dataset. In Section 2.4.4, we quantitatively demonstrated that our method provides additional robustness to natural perturbations. Here we show qualitative results to illustrate its effectiveness. In Fig. 2.10, each row represents a different time stamp. In the left airplane example, we can observe that while all three methods can detect the airplane's wing, the detections of Mask R-CNN [75] and LPF [291] fluctuate over time (e.g. multiple detections on the airplane's wing) whereas our detections are quite stable. In the right skiing example, both Mask R-CNN and LPF generate lots of redundant detections compared to our approach that is likely caused by the aliasing effects of downsampling.

2.4.8.4 Image-to-Image translation

Finally, we show qualitative results of applying our approach to generative models. In Fig. 2.11, we compare with pix2pixHD [236] and pix2pixHD together with LPF [291] on image-to-image translation using the Cityscapes dataset. We find that our adaptive filters are better at preserving boundaries in image generation. In the first row, our approach generates clear boundaries along the roof. In contrast, the other methods produce blurry boundaries. In the second row, our approach not only produces a clear edge on the car, it also generates a very tiny traffic light (see region inside the red rectangle). The other two methods fail in this situation. In the last row, our approach clearly identifies the boundary between the wall and bushes whereas the other two approaches' produce very blurry and dark generations. We attribute this property to the fact that with LPF or the original conv filters, the filter weights are fixed at all spatial locations. This means that it will be difficult for neighbouring in the higher resolution output to have different values within a small local region. And this could potentially cause the unclear boundary effect shown in Fig. 2.11.

2.5 Limitations

We have shown in this chapter that our approach is effective for various discriminative and generative tasks. However, it also has some limitations. First, although both the computation and parameter overhead is marginal, with our current implementation, GPU memory overhead is not negligible as it involves the unfold function in PyTorch which is memory intensive. Second, we empirically found the optimal group number of filter weights to be 8 for our tasks. However, it may not be optimal for other tasks and thus is a hyperparameter that needs to be tuned.

2.6 Conclusion

In this chapter, we proposed an adaptive content-aware low-pass filtering layer, which predicts separate filter weights for each spatial location and channel group of the input. We quantitatively demonstrated the effectiveness of the proposed method across multiple tasks and qualitatively showed that our approach effectively adapts to the different feature frequencies to avoid aliasing while preserving useful information for recognition. Despite some of the limitations observed in Section 2.5, we believe our work can be a promising foundation for exploring anti-aliasing on other tasks (e.g., video recognition) as well as other forms of input noise.

This work was supported in part by ARO YIP W911NF17-1-0410, NSF CAREER IIS-2150012, NSF IIS-2204808, NSF CCF-1934568, GCP research credit program, and AWS ML research award.

Chapter 3

End-to-End Instance Edge Detection

Publication Statement. This chapter is joint work with Haotian Liu, Yong Jae Lee. The paper version of this chapter appeared in arXiv 2022.

Edge detection has long been an important problem in the field of computer vision. Previous works have explored category-agnostic or category-aware edge detection. In this chapter, we explore edge detection in the context of object *instances*. Although object boundaries could be easily derived from segmentation masks, in practice, instance segmentation models are trained to maximize IoU to the ground-truth mask, which means that segmentation boundaries are not enforced to precisely align with ground-truth edge boundaries. Thus, the task of instance edge detection itself is different and critical. Since precise edge detection requires high resolution feature maps, we design a novel transformer architecture that efficiently combines a FPN and a transformer decoder to enable cross attention on multi-scale high resolution feature maps within a reasonable computation budget. Further, we propose a light weight dense prediction head that is applicable to both instance edge and mask detection. Finally, we use a penalty reduced focal loss to effectively train the model with point supervision on instance edges, which can reduce annotation costs. We demonstrate highly competitive instance edge detection performance compared to state-of-the-art baselines, and also show that the proposed task and loss are complementary to instance segmentation and object detection.

3.1 Introduction

We address the problem of instance edge detection. Unlike category-agnostic [167, 7, 253] or category-aware (semantic) edge detection [275, 276], instance edge detection requires predicting the semantic edge boundaries of *each object instance*. This problem is fundamental and can be of great importance to a variety of computer vision tasks

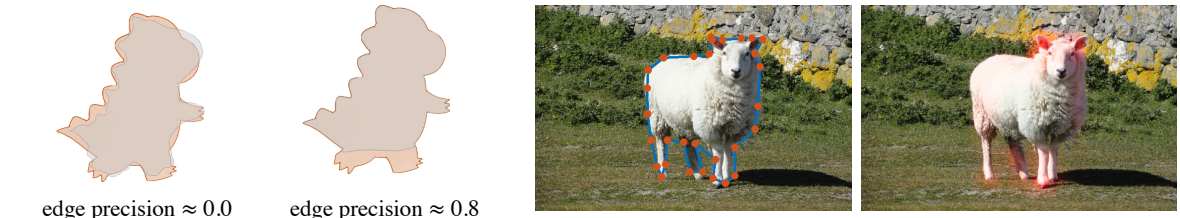


Figure 3.1: (a,b) Although closely related, instance segmentation and edge detection are not fully invertible; i.e., a method that performs well on instance segmentation will not necessarily perform well on edge detection. In this example, although the mask IoUs in (a) and (b) are the same, their edge precision are very different. (c) Although the annotated points (red) are on the object's boundary, the edges (blue) that connect them do not align well to the object's boundary. (d) Cross attention weights between object queries and image feature in DETR [18].

including segmentation, detection/recognition, tracking and motion analysis. In particular, instance edge detection can be critical for applications that require precise object boundary localization such as autonomous driving or robot grasping.

Instance segmentation is closely related to instance edge detection. After all, in theory, an instance's boundary can be trivially extracted from the output of any standard instance segmentation algorithm [75, 12]. However, in practice, this naive solution does not produce good results [40, 33]. Since an instance segmentation algorithm is trained to correctly predict *all* pixels that belong to an object, and since there are relatively few pixels on an instance's contour than inside of it, the model has no strong incentive to accurately localize the instance boundaries. As shown in Fig. 3.1 (a), although the predicted mask may have high quality pixel alignment with the ground truth mask, its boundary may not be well-aligned with the ground truth instance edge. Thus, instance edge detection itself is a unique and important task.

Meanwhile, the recent transformer [226] based DETR [18] object detector has drawn significant attention as it greatly simplifies the detection pipeline by achieving end-to-end learning without ROI pooling, NMS, and anchor modules. Moreover, several transformer based object detection models [18, 169, 25] have shown that object boundaries produce high responses in the attention maps (Fig. 3.1 (d)), which suggests that transformer based architectures can be suitable for instance edge detection. However, DETR is not directly applicable to instance edge detection in several ways. First, the quadratic complexity on self-attention over feature maps prevents DETR from using high resolution feature maps in its transformer decoder. However, high resolution features maps are critical for dense prediction tasks such as edge detection. Second, since the output dimension (length) of edges can vary per object instance (unlike box coordinates or classes), it is not straightforward to produce instance edge outputs directly from an object query.

To address these difficulties, we first propose a multi-scale transformer decoder that takes in both encoder features and FPN features. The first several layers in the transformer decoder take in the feature maps computed from the transformer encoder while the last two layers take in the high resolution FPN feature maps. In this way, object queries interact with different resolution feature maps in a coarse to fine grained manner. To enable the model to output instance edge detections, we introduce a light weight dense prediction head that computes a simple matrix multiplication between object queries and high resolution feature maps to produce binary output maps (whose spatial resolution is the same as the feature maps) where predicted 1/0's indicate edges/non-edges.

By changing only the loss function, we show that our method can perform either edge detection or instance segmentation without any modification to the architecture. At the same time, since instance edge detection and segmentation are closely related, if we do perform both tasks together (with separate heads for each task), we show that they provide complementary benefits to each other.

Finally, one key challenge with instance edge detection is its annotation requirement; i.e., labeling all pixels along an object instance's contour can be extremely expensive. We therefore propose to train our instance edge detector using only *point supervision*. Similar to how instance segmentation methods are trained with keypoint-based polygon masks [19, 2], we use a sparse set of keypoint annotations along the object's boundary. For instance segmentation, this results in a 4.7x speed up over annotating all points [19]. However, due to the sparsity, simply connecting adjacent keypoints to 'complete the edge' (as done in BMask R-CNN [40]) can often lead to incorrect annotations, as shown in Fig. 3.1 (c). We therefore instead train the edge detector using the keypoints with a penalty reduced loss along the edges.

3.1.0.1 Contributions.

(1) We introduce a novel transformer model with a multi-scale transformer decoder and dense prediction head for instance edge detection, which achieves highly competitive results on the COCO and LVIS datasets compared to related state-of-the-art baselines. (2) We demonstrate that our model can perform object detection, instance segmentation, and instance edge detection in a single pass, and show complementary benefits for each task. (3) We show that we can efficiently train our instance edge detection model with only point supervision using a penalty reduced focal loss.

3.2 Related Work

3.2.0.1 Edge Detection.

Edge detection has a rich history and has been studied since at least the 1980s. Early pioneering methods include the Sobel filter [106], zero-crossing [166, 221], and Canny edge detector [17]. The early 2000s saw approaches driven by information theory such as statistical edges [110], Pb [167] and gPB [7]. The advent of deep learning in the last decade introduced highly effective approaches like HED [253, 156], and the focus shifted from texture based edge detection [7] to category agnostic semantic edge detection [73]. More recently, researchers have begun to focus on *semantic aware* edge detection [275, 276] whose goal is to accurately localize the boundary between semantic classes (but not between instances). In this work, we explore edge detection in the *semantic and instance aware* setting [40] to localize object *instance* boundaries.

3.2.0.2 Instance Segmentation.

Instance segmentation is now a classic task in computer vision, where early methods [64, 73, 44, 74] resorted to classifying bottom-up segments. The development of the RCNN framework [78, 63, 191] led to Mask R-CNN [75], a strong performing and simple architecture, which greatly increased the popularity of instance segmentation. Since then, researchers have explored various directions to improve efficiency [12, 182] and effectiveness [89, 104, 40]. Recent trends are towards developing light weight detectors that contain only one-stage [12], without anchors [220], ROI-align and NMS [81]. Further, instead of formulating instance segmentation as a dense pixel prediction task [158, 75], some approaches [19, 140] predict polygon points for each instance to focus more on the object boundaries. Another uses a polar representation [251].

The recent concurrent works, MaskFormer [37] and Mask2Former [34], are similar to our work in regards to the dense prediction head and pixel decoder. However, our main focus is on solving the different instance edge detection task (while also performing bounding box detection and instance segmentation), whereas those works focus on segmentation. Importantly, we find that simply changing their models to perform bounding box and mask prediction in the transformer decoder results in low detection accuracy, suggesting that sophisticated changes to the methods would be needed to do well on those tasks.

Finally, related to our point supervised edge detection loss, [104, 36] also perform point supervised learning, however, they sample points on the model's predicted feature maps instead of using ground truth annotations. Similarly, although [136] performs sparse sampling on output mask, it is designed for the panoptic segmentation task, which is different from our goal of instance edge detection.

3.2.0.3 Transformers.

The Transformer was first introduced in [226], and has become the state-of-the-art architecture for natural language processing tasks [215, 279]. However, despite its high accuracy, the transformer architecture suffers from slow convergence [151] and quadratic computation and memory consumption [105, 9] necessitating a high number of GPUs and up to weeks for training. Recently, the transformer has begun to be explored for visual recognition tasks including image classification [52], detection [18], image generation [95], etc. Since image data typically has longer input sequences (pixels) than text data, the computation and memory problem is arguably more critical in this setting. To address this, researchers have proposed methods [287, 237, 157] that reduce both computation and memory complexity, allowing the transformer to perform dense prediction tasks [252, 37, 209, 229]. Apart from the efficiency problem, the vision transformer also suffers from long training times especially for object detection; specifically, 500 epochs for DETR [18] to achieve the same performance as Faster R-CNN [191] with only 50 epochs. As such, many follow-up works to DETR aim at improving its convergence speed via additional priors [297, 59, 169] or by reducing transformer density [297]. In particular, [169] achieves significant improvements by introducing a conditional spatial query. In this work, we extend the DETR framework [169, 18] to instance edge detection and segmentation.

3.3 Method

3.3.1 Problem Definition

Given an input image I, the task of instance edge detection is to correctly predict the boundaries of each object instance $G_E = \{e_0, e_1, ..., e_n\}$ together with its category label $G_C = \{l_0, l_1, ..., l_n\}$, where n is the number of instances in I.

3.3.2 Method Overview

Our method overview is shown in Fig. 5.2. It comprises four main components: (1) a backbone network, which extracts a hierarchical combination of features (Sec. 3.3.2.1) and a transformer encoder network that perform global attention over low resolution feature maps; (2) a feature pyramid network (FPN) [143], which fuses the feature maps of different levels (Sec. 3.3.2.2); (3) a multi-scale transformer decoder, which takes in the different resolution feature maps and a set of object queries and performs cross-attention between them (Sec. 3.3.2.3); and (4) light weight dense prediction heads, which perform either instance edge detection or segmentation (Sec. 3.3.2.4), along with prediction heads for box detection and classification. We also introduce our point based

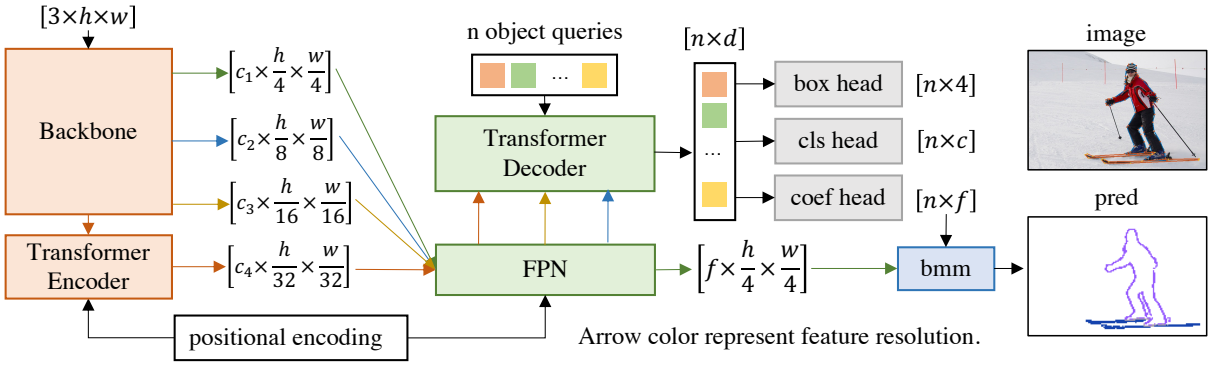


Figure 3.2: **Method overview.** Our model consists of four main components: a backbone feature extractor, FPN, multi-scale transformer decoder, and task heads. The output dimension for each module is indicated in $[\cdot]$, and bmm denotes batch matrix multiplication.

instance edge detection loss in Sec. 3.3.3 and analyze relationship between object edge and segmentation in Sec. 3.3.5. *The multi-scale transformer decoder, light weight dense prediction head, point based instance edge detection loss, and object edge and segmentation relationship analysis are the main technical contributions.*

3.3.2.1 Backbone Feature Extractor

Given an input image I with shape $[3, h, w]$, the feature backbone extracts a set of feature maps with shape $[c_i, h/r_i, w/r_i]$ for $c_i \in [256, 512, 1024, 2048]$ and $r_i \in [4, 8, 16, 32]$. We set the feature extractor to be a ResNet [79] together with a transformer encoder [18] with self-attention [226].

3.3.2.2 Feature Pyramid Network

The final output of the transformer encoder is $1/32$ of the original image, which is too low for edge detection. Thus, we integrate a feature pyramid network (FPN) [143] to increase the resolution by fusing higher resolution feature maps from the backbone and self-attention features. The output of the FPN includes feature maps with $[1/4, 1/8, 1/16, 1/32]$ of the original image resolution. We also add positional encodings [226] to the projected features, which will enable the object queries (explained shortly in Sec. 3.3.2.3) to better localize objects and their boundaries.

3.3.2.3 Transformer Decoder

Given n input object queries each with d dimensions (i.e., size $[n, d]$), the transformer decoder first applies self-attention [226] to allow the object queries to interact with each other to remove redundant predictions. It then applies cross attention [226] between the object queries Q with shape $[n, d]$ and the feature map F (i.e., size $[d, h/i, w/i]$ for $i \in (32, 16, 8)$ with (h, w) as the image size) from the FPN. Note that the cross attention

operation has a time and memory complexity of $O(nd(hw)^2 + nd^2(hw))$. Thus, in order to leverage high resolution feature maps without a large computation overhead, we use a feature resolution of $1/32$ in the first four layers and $1/16$ and $1/8$ in the last two layers of transformer decoder. In this way, the object queries can attend to the features in a coarse to fine-grained manner for improved dense prediction performance.

3.3.2.4 Dense Prediction Head

Our design for edge prediction is motivated by three observations: (1) Without training with any dense pixel-level labels, and instead, with only box supervision, the cross attention maps computed between the object queries and encoder features have the nature to focus on instance edges [18, 169, 25] (Fig. 3.1 (d)). (2) The encoder features within the same object *instance* have similar representations [18]. (3) By directly taking a weighted combination of the high-resolution feature maps along channel dimension, it leads to mask predictions that can clearly follow the boundaries of the instances [12]. These three observations suggest that convolving the feature maps with each object query could lead to accurate pixel-level instance edge predictions.

Given the transformer decoded object queries Q with shape $[n, d]$, and image features F from the FPN with shape $[f, h/4, w/4]$, we first predict f weight coefficients for each object query with a simple linear projection:

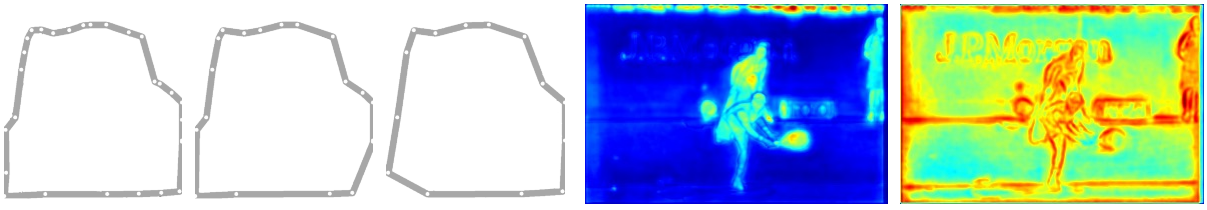
$$Q' = sigmoid(linear(d, f)(Q)) \tag{3.1}$$

where $linear(i, j)$ indicates linear projection from dimension i to j. The result is a coefficient for each query; i.e., coefficient tensor with shape $[n, f]$. This operation corresponds to the 'coef head' shown in Fig. 5.2.

Then, to predict the edge map for each object query, we apply a 1×1 convolution to the feature maps F using the object query Q' coefficients as filter weights. This is equivalent to applying a batch matrix multiplication between Q' and F:

$$O_i = sigmoid(Q'_i \times F), \forall i \tag{3.2}$$

where i is the index of the object query, Q'_i has shape $[1, f]$, F has shape $[f, h, w]$, and O_i has shape $[h, w]$. Note that all object queries are multiplied with the same set of features maps. This dense prediction head is general and very light weight, and is applicable to any object instance based pixel classification tasks. For example, we can easily obtain mask segmentations by only changing the edge detection loss function to a mask segmentation loss.



(a) ratio = 1/1 (b) ratio = 2/3 (c) ratio = 1/2 (d) Mask Feature (e) Edge Detection

Figure 3.3: (a,b,c) show the annotated points that are regarded as positive samples and the tunnels (in gray) that are inside the penalty reduced regions. We show these with different fractions of annotated points. (d,e) Feature map norm on channel dimension for the layer preceding the final output layer (pixel-wise mask/edge for instance/edge detection).

3.3.3 Point Supervised Focal Loss

As dense labeling of all pixels along an object instance's contour can be extremely expensive, we train our instance edge detector using only point supervision along the object's boundary, similar to how instance segmentation methods are trained with keypoint-based polygon masks [19, 2]. Note that simply connecting adjacent keypoints to 'complete the edge' as done in BMask R-CNN [40] will lead to incorrect annotations that are not on the ground-truth edge (see Fig. 3.1 (c)).

To address this, we design a novel training objective to account for the sparse keypoint annotation. Specifically, we build upon the penalty-reduced pixel-wise logistic regression with focal loss [117], which was designed to reduce the penalty in slightly mis-predicted corners of a bounding box (since those slightly shifted boxes will also localize the object well). In our case, we can use this loss to account for slightly mis-predicted keypoints, but we also need to deal with a different issue, which is that a large portion of the ground-truth edges are not annotated at all. To handle the latter, we construct the ground-truth in the following way.

We first connect the ground-truth keypoints to create edges, and then blur the result with a small 3×3 kernel (e.g., a Gaussian or a box filter). This creates a "tunnel" whose values are greater than 0. We set these values to 0.7, and the original keypoints as 1, as shown in Fig. 3.3 (a-c). The lower values for the tunnels account for the uncertainty in ground-truth edge location for the non-keypoints. While we could also take continuous values that degrade as a function of distance to the keypoints and edges, we find this simple approach to work well in practice.

Formally, we use the ground-truth maps Y as targets in our extension of the penalty-reduced pixel-wise logistic regression with focal loss [117]:

$$L_k = \frac{-1}{N} \sum_{cxy} \begin{cases} Y_{cxy}(1-\widehat{Y}_{cxy})^{\alpha} \log(\widehat{Y}_{cxy}) & \text{if } Y_{cxy} \geq \gamma \\ (1-Y_{cxy})^{\beta}(\widehat{Y}_{cxy})^{\alpha} \log(1-\widehat{Y}_{cxy}) & \text{else} \end{cases} \tag{3.3}$$

where α and β are hyper-parameters of focal loss [144], and N is the number of annotated keypoints inside an image. We set $\alpha = 2$ and $\beta = 4$ following [117] and set $\gamma = 0.7$. $\widehat{Y}_{cxy}$ and Y_{cxy} denotes the prediction and ground truth value at location c, x, y. With this loss, the model is encouraged to accurately predict the annotated edge points, while also predicting edge points inside the 'tunnels' that connect those keypoints. To complement the point supervised focal loss and to get sharper boundaries [40], we also add the dice loss [170] for edge detection.

3.3.4 Overall Objective

Our final objective combines the following: for edge detection, we use our point supervised focal loss as well as dice loss [170] between the matched prediction and ground truth edge pairs. For bounding box regression, we apply L1 and generalized IoU loss [192]. For classification, and to match each object query to a ground truth box, we use the paired matching loss from DETR [18]. Finally, when generalizing our architecture to instance segmentation, we follow [220], and use the dice loss [170] and sigmoid focal loss.

3.3.5 Relation to Instance Segmentation

Instance edge detection and instance segmentation are highly correlated tasks as their ground truths are fully invertible. However, since the ratio of pixels on the boundary over the inner pixels for an instance mask is very small, an instance segmentation model will have less preference to correctly predict the edge boundaries compared to the inner pixels. In contrast, an edge detection model would fully focus on correctly predicting the edge boundaries as the inner pixels would be labeled as background.

In this section, we provide a quantitative analysis on this difference. In particular, we perform the analysis using the dice loss [170], but the conclusion holds for other losses as well. We choose the dice loss because it is used in many state-of-the-art instance segmentation methods [220, 37, 34] due to its explicit accounting of the imbalance in foreground versus background pixels, which largely improves segmentation performance.

Given predicted (either edge or mask) instance map p and ground truth map y, each with shape $[h, w]$, the dice loss $L(p, y)$ is:

$$[0.9]L(p, y) = 1 - (2\sum_{j=1}^{hw} p_j y_j)/(\sum_{j=1}^{hw} p_j^2 + \sum_{j=1}^{hw} y_j^2) \tag{3.4}$$

Its partial derivative with respect to a prediction p_i at pixel i is:

$$[0.9]\frac{\partial L(p, y))}{\partial p_i} = -2(y_i(\sum_{j=1}^{hw} p_j^2 + \sum_{j=1}^{hw} y_j^2) - 2p_i \sum_{j=1}^{hw} p_j y_j)/(\sum_{j=1}^{hw} p_j^2 + \sum_{j=1}^{hw} y_j^2)^2 \tag{3.5}$$

Next, let us consider the case in which pixel i is on the instance boundary ($y_i = 1$) but the model incorrectly predicts it as background ($p_i = 0$). We would like to analyze the impact of such incorrect boundary predictions when the objective is mask segmentation versus edge detection. Since a neural network's weights are updated according to their gradient direction and magnitude, the absolute gradient value of a prediction on a single pixel can measure how much it influences the training procedure (given same loss function and prediction value). We can therefore take the ratio between the absolute gradient value of the boundary pixel's prediction when the objective is edge detection (with predictions and ground truth denoted as p'_i and y'_i, respectively) over that when the objective is mask segmentation:

$$[0.9]|\frac{\partial L(p', y'))}{\partial p'_i}|/|\frac{\partial L(p, y))}{\partial p_i}| = (\sum_{j=1}^{hw} p_j^2 + \sum_{j=1}^{hw} y_j^2)/(\sum_{j=1}^{hw} p_j'^2 + \sum_{j=1}^{hw} y_j'^2) = \alpha \gg 1 \tag{3.6}$$

For the same object instance, since its mask will be at least as big as its boundary (and typically much larger), we will have $\sum_{j=1}^{hw} y_j^2 \gg \sum_{j=1}^{hw} y_j'^2$, and also $\sum_{j=1}^{hw} p_j^2 \gg \sum_{j=1}^{hw} p_j'^2$ for any reasonably performing model, where y', p' denote edge ground truth and prediction maps, and y, p denote mask ground truth and prediction maps. Thus, we can easily conclude that the gradient magnitude of edge detection will be much larger than that of instance segmentation for pixel predictions on the object boundary, as shown on the right hand side of Eq. 3.6.

In other words, for pixels on the object boundary, the instance edge detection objective enforces a stronger influence than the instance mask segmentation objective. In addition, if we think this about the problem more intuitively, edge detection feature maps will have high response only on the boundary pixels for each object (Fig.3.3 (e)) whereas instance segmentation will train towards predicting all pixels within each object (Fig.3.3 (d)). And since the ratio of pixels on the boundary over the inner pixels is very small, the model will have less preference to correctly predict the edge boundaries than the inner pixels in mask segmentation.

Thus, we argue that instance edge detection itself is an important task to explore, distinct from instance segmentation, especially for applications that require precise object boundary localization e.g., self driving or robot grasping.

3.4 Experiments

In this section, we first explain the datasets and evaluation metrics used for evaluating instance edge detection. We then present our implementation details. We further describe our key baselines, and compare to them both quantitatively and qualitatively. Finally, we ablate our model with various baseline components.

3.4.0.1 Datasets.

We train our model on MS COCO [145] and evaluate on both COCO as well as LVIS [70] as the boundary annotations in LVIS are much more precise, as shown in Fig. 3.4 (right).

MS COCO [145] contains 118K images for training, and 5K images for evaluation with around 1.5M object instances and 80 categories. The annotation contains bounding box, category labels, and keypoint-based mask polygons. All instances in the dataset are exhaustively annotated.

LVIS [70] contains 164K images and 2.2M high-quality instance segmentation masks for over 1000 entry-level object categories. Its images are a subset of the images from MS COCO. We keep all the annotated instances that overlap with MS COCO and re-label the categories in the same way as COCO for evaluation.

3.4.0.2 Evaluation Metrics.

As well-established problems, both semantic aware [275, 276] and agnostic [7, 167] edge detection have standard evaluation pipelines. We use the same standard ODS (optimal dataset scale) and OIS (optimal image scale) metrics to evaluate instance edge detection.

Briefly, an edge thinning step is typically applied to produce (near) pixel-wide edges. Then, bipartite matching is used to match the predicted edges PD with the ground-truth edges GT (see Fig. 3.4 left). Candidate matches are those whose distance is within a small pre-defined distance proportional to the image size. Then, precision p and recall r are computed, where precision measures the number of predicted edge points that are matched to a ground truth edge, and recall measures the number of ground truth edge points that are matched to a predicted edge. The F-measure is then computed as $2 \cdot p \cdot r/(p + r)$. ODS is the best F-measure using the global optimal threshold across the entire validation set. OIS is the aggregate F-measure when the optimal threshold is chosen for each image. (We provide more details in the supplementary document.)

In addition, a recent paper [33] proposes the 'Boundary IoU' to supplement mask mAP for evaluating the boundary of instance segmentation. However, we argue that this is an imprecise measurement on edge, as it blurs the boundary and computes the IoU between the thick boundary and ground truth edges.

3.4.0.3 Implementation Details.

Training: We adopt the DETR framework proposed in [18] and replace the transformer decoder with the conditional decoder from Conditional DETR [169] to accelerate model training by ∼6x. For MS COCO, we train all models on 4 NVIDIA 3090 Ti GPUs with per GPU batch size of 2. For the experiments in Table 3.1, we use the multi-scale transformer decoder with 6 consecutive layers. For each layer, the input feature resolution is $[1/32, 1/32, 1/32, 1/32, 1/16, 1/8]$ corresponding to the original image size.

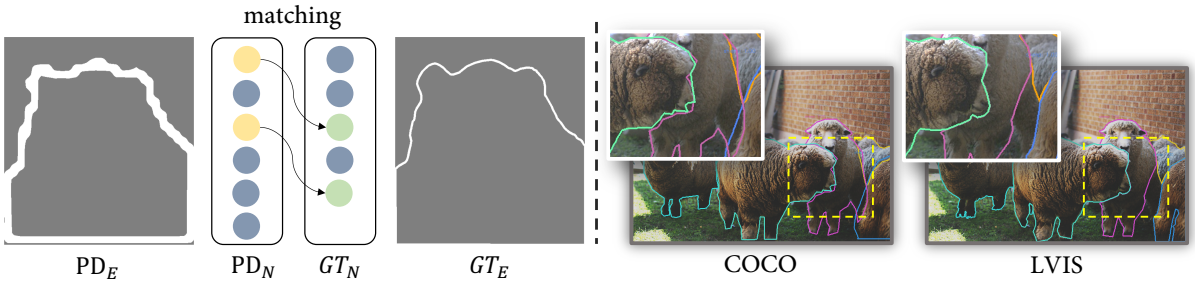


Figure 3.4: (Left) Bipartite matching between edge detections and ground-truth annotated edges. (Right) Annotations from COCO vs. LVIS. It clearly shows that LVIS has more fine-grained boundary annotations than COCO.

				COCO					LVIS	
	Backbone	*Epochs*	*#Param*	*ODS*	*OIS*	AP^{bdry}	AP^{box}	AP^{mask}	*ODS*	*OIS*
Mask R-CNN	R-50	50	44M	62.9	62.9	23.2	40.9	**37.0**	63.8	64.3
BMask R-CNN	R-59	12	47 M	44.1	47.3	**23.5**	38.6	36.6	45.5	46.1
BMask R-CNN	R-50	50	47M	44.2	47.2	22.9	37.6	35.0	45.9	46.7
Ours (Mask)	R-50	50	46M	56.0	56.4	18.4	42.8	34.0	60.0	60.4
Ours (Edge)	R-50	50	46M	63.1	63.8	-	42.6	-	66.2	67.9
Ours (Edge + Mask)	R-50	50	47M	**63.6**	**64.5**	21.7	**43.0**	35.0	**66.6**	**68.3**
Mask R-CNN	R-101	50	63M	**63.7**	63.7	24.4	42.6	38.3	64.7	65.2
BMask R-CNN	R-101	12	66M	44.7	48.2	24.7	40.6	38.0	46.5	47.0
BMask R-CNN	R-101	50	66M	44.8	47.7	24.3	40.0	36.7	46.7	46.1
Ours (Edge)	R-101	50	65M	63.6	**64.4**	-	**44.3**	-	**67.1**	**68.7**

Table 3.1: Edge detection, object detection, and instance segmentation results on MS COCO and LVIS.

The dense prediction head takes in the last layer output of the FPN network with a size of $1/4$ of the original resolution. In Table 3.2, 3.3, to reduce computation cost, we use single scale transformer decoder with a $1/32$ input resolution. And the dense prediction head takes in FPN features with $1/8$, $1/4$ of the original image resolution respectively. Most of the hyperparameters follow the implementation in [169].

Evaluation: We evaluate our approach on three different tasks including object detection, instance segmentation, and edge detection. For object/instance detection, we follow [145] and use mAP metric for evaluation. For instance edge detection, we use ODS and OIS following [253]. The reason for not including AP for edge detection is because the boundary of instance segmentation will only have a probability range from [0.5,1], which will introduce errors in computing the AP score. To compensate for this, we use AP boundary [33] to evaluate our model when there is a mask output.

3.4.0.4 Baselines.

The most related work to ours that simultaneously predicts instance mask and boundaries is BMask R-CNN [40], which learns a separate instance edge detection head in parallel with the mask and box heads in Mask R-CNN [75]. In addition, since instance edges can be computed from instance segmentation masks, we also compare to the boundaries of the masks produced by Mask R-CNN [75]. Instance edge is derived from

instance mask by applying a laplacian filter on the binary mask. This baseline is used to demonstrate that this way of computing instance edges is insufficient due to the bias in the mask segmentation objective, which rewards accurate prediction of interior pixels in the ground-truth mask more than those that are on the boundary (since they are relatively much fewer).

3.4.1 Quantitative Results

In Table 3.1, we compare our approach with various state-of-the-art baselines for edge detection, object detection, and instance segmentation tasks using the COCO and LVIS datasets. For BMask R-CNN, we use the authors' publicly available codebase. For Mask R-CNN, we use Detectron2 [248] to train and evaluate the baseline model. For all the baseline methods, we re-train the model with 50 epochs schedule. Despite multiple attempts, we could not get BMask R-CNN trained with 50 epochs to outperform its 12 epochs model, which is why we report both of them in the table.

Edge detection On the COCO dataset, our approach achieves the best results under ODS/OIS edge detection metrics compared to BMask R-CNN and Mask R-CNN. Surprisingly, we achieve ∼18% better performance than BMask R-CNN, which is our closest baseline. When taking a closer look at the qualitative results in Fig. 6.5, the reason becomes clear. For example, in the second column of Fig. 6.5, using the same edge probability threshold, the thickness of the predicted instance boundaries for BMask R-CNN varies widely. This indicate that the model lacks a unified treatment for all instances, and thus it is harder to find a single threshold that works well for all instances in all images. The quantitative results on OIS and ODS again prove this hypothesis: the OIS improves by around 3-4 points for BMask R-CNN while it does not change a lot for all other models. In addition, because we are directly thresholding the predicted masks for Mask R-CNN and our mask variant (Ours Mask) to obtain edge detections, their OIS and ODS remain nearly constant under all mask settings. Apart from our better performance compared to the edge detection method of BMask R-CNN, our approach also performs better than instance segmentation methods (especially on lvis dataset with accurate ground truth): Mask R-CNN and our mask variant (Ours Mask). This is mainly due to two reasons: (1) The baseline mask predictions are inaccurate along boundaries. (2) The baseline mask can have holes inside. These observations are further illustrated in Sec. 3.4.3.

On the LVIS dataset, the results are consistent with those on the COCO dataset. However, in general all methods achieve better results using LVIS annotations. And our approach performs extremely well on LVIS dataset in comparison with Mask R-CNN with $\sim$ 3-4 higher on both scale of models (R-50, R-101). This is also explainable if we

	COCO			LVIS	
	ODS	*OIS*	AP^{box}	*ODS*	*OIS*
contour GT	59.0	59.3	41.0	62.5	63.1
point GT	**63.0**	**63.7**	**41.5**	**66.7**	**67.9**

	COCO			LVIS	
ratio	*ODS*	*OIS*	AP^{box}	*ODS*	*OIS*
1/2	58.4	59.4	41.3	61.7	63.9
2/3	61.7	62.5	41.1	64.7	66.4
1/1	63.0	63.7	41.5	66.7	67.9

Table 3.2: (Left) Varying the type of ground truth training target for edge detection. (Right) Varying the number of ground truth edge points used for training. With ratio 1, the average number of keypoints across all instances is 23.

take a look at Fig. 3.4: LVIS has more precise boundary annotations than COCO. The predictions are usually aligned better with these more accurate annotations.

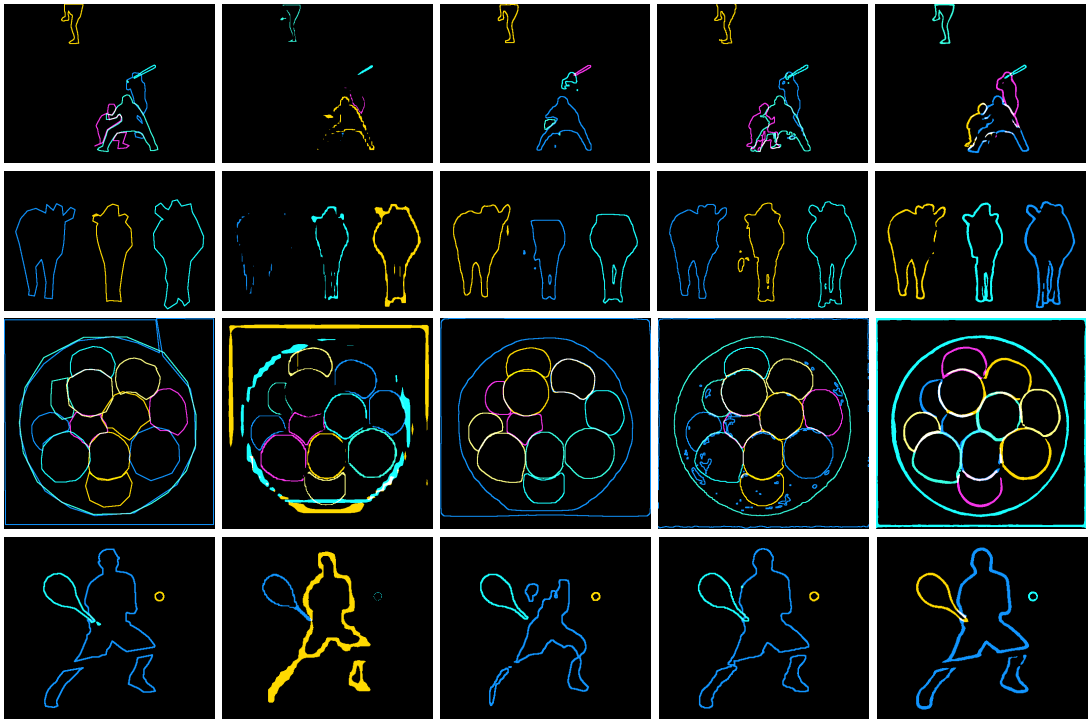
Object detection Our approach also achieves the best result on box mAP with ∼2-4 points higher on ResNet 50 backbone with 50 epochs compared to both Mask R-CNN and BMask R-CNN. When training with a larger ResNet 101 backbone, this improvement also holds with a consistent performance gain of ∼2-4 points.

Instance segmentation Finally, we compare with the baselines on the instance segmentation task using our model with the dense prediction head plus mask loss. It performs ∼2 points worse than Mask R-CNN and BMask R-CNN. One hypothesis is that training an object query containing both mask and box information has a divergent effect; e.g., object query for box detection should have the ability to locate the extreme points of an object, whereas instance segmentation requires the query to focus on the full object.

3.4.2 Ablation Study

Type of ground truth. We first study the effect of our point supervised training objective, which models the uncertainty in the edges that are not labeled by the keypoints by assigning them a softer target score. We compare to the training objective used in BMask R-CNN, which simply connects the keypoints to create ground-truth edges, and applies both a weighted binary cross-entropy loss and the dice loss [170]. As shown in Table 3.2 (left), training with our point supervision objective (point) produces significantly better edge detection performance on both COCO and LVIS datasets compared to the baseline (contour). Furthermore, the improvements on edge detection also lead to a 0.5 improvement in box mAP, which demonstrates their complementary relationship.

Number of edge points. A key advantage of training an edge detector with point supervision is the large reduction in annotation effort that is required. We therefore investigate how the number of annotated edge points affects instance edge detection performance. Specifically, we sample the number of end points that are used for training from 1/1 to 2/3 to 1/2 of the full original set of annotated keypoints. As shown in



(1) GT (2) BMask Rcnn (3) Mask Rcnn (4) Ours Mask (5) Ours Edge

Figure 3.5: Qualitative results comparing to the baselines. The thickness of predicted edges varies largely for BMask R-CNN. In addition, the mask boundary quality of Mask R-CNN is not as good as that of Ours Edge and Ours Mask. Finally, there exist redundant predictions or holes in our mask variant (Ours Mask) predictions.

Table 3.2 (right), by decreasing the annotation by 1/3 and 1/2, both ODS and OIS decreases as expected but not by a large amount.

Annotation types. We also ablate our DETR based dense prediction framework under different types of annotations (box, mask, and edge). As shown in Table 3.3, by adding mask and edge objectives, box prediction improves by 0.2 and 0.4 points respectively. We can also conclude from the table that training with an edge objective leads to a much better edge detection result in comparison with training with a mask objective. This further proves our argument that instance edge detection is different from instance segmentation. Last but not least, by simply adding an edge objective to mask objective, AP box further improves by 0.3 points, and AP mask improves by a good margin with 0.6 point.

			COCO			
box	*mask*	*edge*	AP^{box}	AP^{mask}	*ODS*	*OIS*
✓			40.9	-	-	-
✓	✓		41.1	34.5	56.3	56.4
✓		✓	41.3	-	63.4	64.2
✓	✓	✓	**41.6**	**35.1**	**63.6**	**64.5**

Table 3.3: Varying annotation types (bounding boxes, masks, and edges) for model training.

3.4.3 Qualitative Results

For a fair comparison, all qualitative results use the models trained with a ResNet 50 backbone with 1x schedule. We threshold the mask probability with 0.5 to obtain the

binary mask together with their boundaries. And for edge detection methods, we use 0.7 as a threshold to filter out noisy predictions. As stated in Sec. 3.4.1, we can observe clear reasons for why our approach achieves better performance for edge detection. For example, in the second row of Fig. 6.5, while the blue cow predicted by BMask R-CNN is nearly thresholded out, the edge of the yellow cow remains very thick. This is the primary reason for BMask R-CNN's low performance. Further, the third column of Fig. 6.5 shows the results of Mask R-CNN, which clearly indicate that it is usually unable to predict the boundaries well (e.g. the blue cow in second row, the person in fourth row) compared with our mask and edge models. Last but not least, although our mask variant (Ours Mask) usually generates high quality boundaries, when the mask is large, redundant predictions or holes can appear in the mask as shown in the fourth column.

3.5 Conclusion and Limitations

We introduced a novel point supervised transformer model for edge detection. In an extension to the DETR object detector, we introduce a multi-scale transformer decoder and a dense prediction head that could be easily applied to both instance segmentation and edge detection. Although our approach achieves good results for object and edge detection, it does not perform as well on instance segmentation. This is likely because of the divergent objective function – for the same object query, instance segmentation requires focusing on the whole object but edge/object detection requires focusing more on object boundaries.

3.6 Summary

With a clear understanding and exploration of how each kind of specialist works, another question naturally emerges. How can we have a generalist model that can integrate those models into a single framework? So in the next chapter, we are going to give a clear demonstration of the works that we build towards the generalist model.

Part II

Generalist Model

Chapter 4

Generalized Decoding for Pixel, Image, and Language

Publication Statement. This chapter is joint work with Zi-Yi Dou, Jianwei Yang, Zhe Gan, Linjie Li, Chunyuan Li, Xiyang Dai, Harkirat Behl, Jianfeng Wang, Lu Yuan, Nanyun Peng, Lijuan Wang, Yong Jae Lee, Jianfeng Gao. The paper version of this chapter appeared in CVPR 2023 [298].

We present *X-Decoder*, a generalized decoding model that can predict pixel-level segmentation and language tokens seamlessly. *X-Decoder* takes as input two types of queries: (i) generic non-semantic queries and (ii) semantic queries induced from text inputs, to decode different pixel-level and token-level outputs in the same semantic space. With such a novel design, *X-Decoder* is the first work that provides a unified way to support all types of image segmentation and a variety of vision-language (VL) tasks. Without any pseudo-labeling, our design enables seamless interactions across tasks at different granularities and brings mutual benefits by learning a common and rich pixel-level understanding. After pretraining on a mixed set of a limited amount of segmentation data and millions of image-text pairs, *X-Decoder* exhibits strong transferability to a wide range of downstream tasks in both zero-shot and finetuning settings. Notably, it achieves (1) state-of-the-art results on open-vocabulary segmentation and referring segmentation on seven datasets; (2) better or competitive finetuned performance to other generalist and specialist models on segmentation and VL tasks; and (3) flexibility for efficient finetuning and novel task composition (*e.g.*, referring captioning and image editing shown in Fig. 6.1).

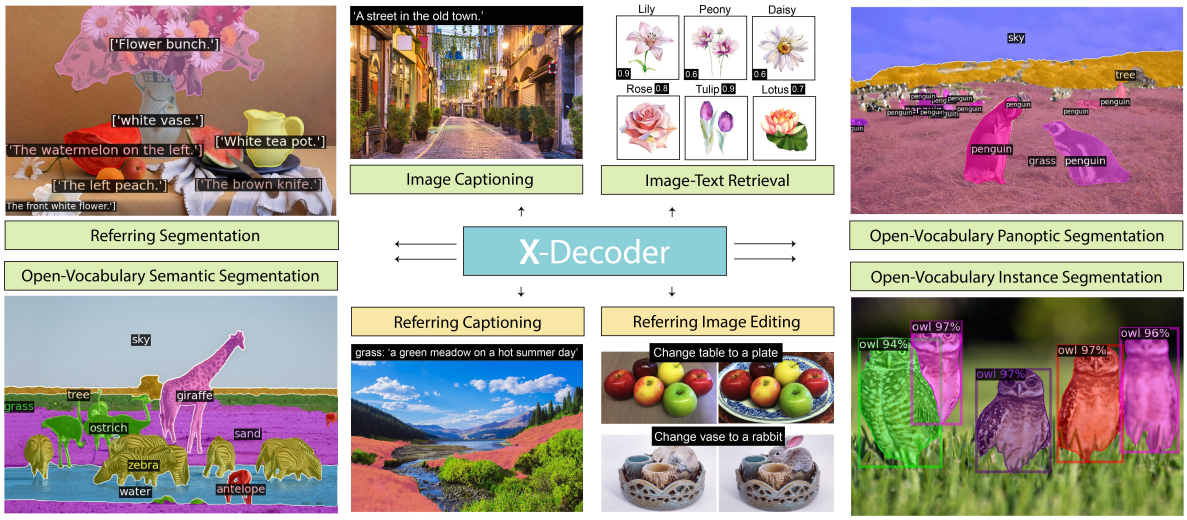


Figure 4.1: With one suite of parameters, X-Decoder after pretraining supports all types of image segmentation tasks ranging from open-vocabulary instance/semantic/panoptic segmentation to referring segmentation, and vision-language tasks including image-text retrieval, and image captioning (labeled in green boxes). It further empowers composite tasks like referring captioning using X-Decoder itself and image editing collaborating with generative models such as Stable Diffusion [195] (labeled in yellow boxes).

4.1 Introduction

Visual understanding at different levels of granularity has been a longstanding problem in the vision community. The tasks span from image-level tasks (*e.g.*, image classification [45], image-text retrieval, image captioning [28], and visual question answering (VQA) [6]), region-level localization tasks (*e.g.*, object detection and phrase grounding [184]), to pixel-level grouping tasks (*e.g.*, image instance/semantic/panoptic segmentation [158, 102, 72]). Until recently, most of these tasks have been separately tackled with specialized model designs, preventing the synergy of tasks across different granularities from being exploited. In light of the versatility of transformers [226], we are now witnessing a growing interest in building general-purpose models that can learn from and be applied to a diverse set of vision and vision-language tasks, through multi-task learning [85, 71], sequential decoding [233, 265, 26, 160], or unified learning strategy [277, 260, 271, 285]. While these works have shown encouraging cross-task generalization capabilities, most target the unification of image-level and region-level tasks, leaving the important pixel-level understanding underexplored. In [26, 160], the authors attempt to unify segmentation into a decoding of a coordinate sequence or a color map, which, however, produces suboptimal performance and limited support for open-world generalization.

Arguably, understanding images down to the pixel level is one of the most important

yet challenging problems in that: (1) pixel-level annotations are costly and undoubtedly much more scarce compared to other types of annotations; (2) grouping every pixel and recognizing them in an open-vocabulary manner is less studied; and (3) more importantly, it is non-trivial to learn from data at two substantially different granularities while also obtaining mutual benefits. Some recent efforts have attempted to bridge this gap from different aspects. In [35], Chen *et al.* propose a unified architecture Mask2Former that tackles all three types of segmentation tasks but in a closed set. To support open vocabulary recognition, a number of works study how to transfer or distill rich semantic knowledge from image-level vision-language foundation models such as CLIP [187] and ALIGN [93] to specialist models [61, 50, 190]. However, all these initial explorations focus on specific segmentation tasks of interest and do not show generalization to tasks at different granularities. In this work, we take one step further to build a generalized decoder called *X-Decoder*[1] towards the unification of pixel-level and image-level vision-language understanding, as shown in Figure 6.1.

A generalized decoding framework. We formulate all tasks including pixel-level image segmentation, image-level retrieval, and vision-language tasks into a generic decoding procedure. Specifically, *X-Decoder* is built on top of a vision backbone and a transformer encoder for extracting multi-scale image features, following the framework of Mask2Former [35]. The key novelty lies in the decoder design. First, it takes two sets of queries as input: (i) generic non-semantic queries that aim to decode segmentation masks for universal segmentation, similar to Mask2Former [35], and (ii) newly introduced textual queries to make the decoder language-aware for a diverse set of language-related vision tasks. Second, it predicts two types of outputs: pixel-level masks and token-level semantics, and their different combinations can seamlessly support all tasks of interest. Third, we use a single text encoder to encode the textual corpus involved in all tasks, including concepts in segmentation, phrases in referring segmentation, tokens in image captioning and questions in VQA, etc. As a result, our *X-Decoder* can naturally facilitate the synergy across tasks and advocate the learning of a shared visual-semantic space, while respecting the heterogeneous nature of different tasks.

An end-to-end learning paradigm. With our generalized decoder design, we propose an end-to-end pretraining method to learn from all granularities of supervision. We unite three types of data: panoptic segmentation, referring segmentation, and image-text pairs. Unlike previous works that use pseudo-labeling techniques to extract fine-grained supervision from image-text pairs [285, 61], *X-Decoder* directly groups and proposes a few meaningful segmentation candidates, so that it can map the regions easily to the contents described in the captions on the fly. Meanwhile, the referring segmentation task bridges generic segmentation and image captioning by sharing the

[1]Here, 'X' denotes versatile, and also represents 'piXel'.

latent queries and semantic queries during decoding.

Strong zero-shot and transfer ability to a wide range of segmentation and VL tasks. Pre-trained with a limited amount of segmentation data and millions of image-text pairs (4m images), our *X-Decoder* supports a diversity of tasks in a zero-shot and open-vocabulary manner. Concretely, our model can be directly applied for all three types of segmentation tasks in a wide range of domains, establishing new state-of-the-art on ten settings of seven datasets. When transferred to specific tasks, our model also exhibits consistent superiority to previous works. Finally, we observe some intriguing properties in our model that it can support some novel task compositions and efficient finetuning, thanks to the flexibility endowed by our model design.

4.2 From Specialist to Generalist Models

4.2.1 Pixel-Level Understanding

Pixel-level image understanding, also known as image segmentation, has been a long-standing problem [57, 179].

Generic Segmentation. There are mainly three well-defined tasks for pixel-level understanding, including semantic [158], instance [72], and panoptic [102] segmentation. Semantic segmentation cares about the per-pixel semantic within an image [158, 23, 32], whereas instance segmentation groups pixels of the same semantic meaning into objects. Models for both tasks have evolved from CNN-based architectures [158] to transformer-based ones [32], and from two-stage models [75], one-stage models [12, 220] to the recent query-based approaches [51, 217]. With the capability of per-pixel and instance-level understanding, a natural step was taken to formulate panoptic segmentation [102, 229, 35]. Most recently, Mask2Former [35] proposed to address all three tasks with a unified encoder-decoder architecture. Nevertheless, all these works cope with a limited number of categories. In MSeg [115], the authors manually merge different datasets, which is still limited to being a closed set.

Open-Vocabulary Segmentation. Recently, a number of works opt to transfer or distill the rich visual-semantic knowledge from foundation models [187, 93] to specific segmentation tasks. Prominent examples include LSeg [121], OpenSeg [61], and [90]. Instead of using existing models, GroupViT [254] performed language-image pretraining from scratch with a bottom-up grouping ViT [52], while DenseCLIP [190] demonstrated the superiority of foundation models in finetuning settings compared with supervised models.

Referring Segmentation by nature is open-vocabulary. Models are usually designed specifically to learn from target datasets using various multimodal fusion strategies [84, 148, 165, 268, 272, 244]. Since the emergence of vision transformers, works like LAVT [264]

enhance the cross-modal interactions from the very beginning, which led to SoTA on RefCOCO [272], RefCOCO+ [272] and G-Ref [164, 175]. CLIPSeg [161] extended the textual query to a visual query and showed superior performance not only on referring segmentation but also on semantic segmentation.

In this work, we propose *X-Decoder*, which is the first model to tackle generic and referring segmentation tasks all in one model. Furthermore, the generalized decoder jointly learns from segmentation data and image-text pairs end-to-end, and thus can augment the synergy across tasks for rich pixel-level and image-level understanding.

4.2.2 Vision-Language Understanding

Vision-language (VL) pretraining has proven to be effective for various VL tasks [159, 213, 211, 129]. The field has evolved from a transformer fusion model [30, 134, 289] with pre-extracted object features [5] to end-to-end transformers [101, 128, 54], that directly learn from raw image pixels. Recently, researchers [241, 238, 206] have found that image-text data at scale can be helpful for visual representation learning (*e.g.*, , enabling zero-shot image classification [187, 93], action recognition [277, 271], and image generation [138]). VL pre-trained models can be further extended to region-level tasks, such as phrase grounding and open-vocabulary object detection [97, 68, 294, 172], and unified frameworks that aim to combine image-text pairs with region-level data have also been proposed [16, 130, 285, 266, 53]. A comprehensive review on this topic is provided in [58].

We are clearly witnessing a trend from building specialist models to generalist ones. Early efforts [85, 71] build a multi-task learning paradigm to accommodate a diversity of tasks. However, the interactions among different tasks in these works are less studied, and the combination usually leads to performance degradation compared with specialist models. Recently, a number of works aim to reformulate the tasks into a unified sequential decoding process [26, 265, 233, 160, 109]. In this work, instead of developing a unified interface for vision and VL tasks, our *X-Decoder* builds a generalized decoding paradigm that can seamlessly connect the tasks by taking the common (*e.g.*, , semantic) but respecting the natural differences (*e.g.*, , spatial mask *v.s.* sequential language), leading to significant improvements for different segmentation and VL tasks across the board.

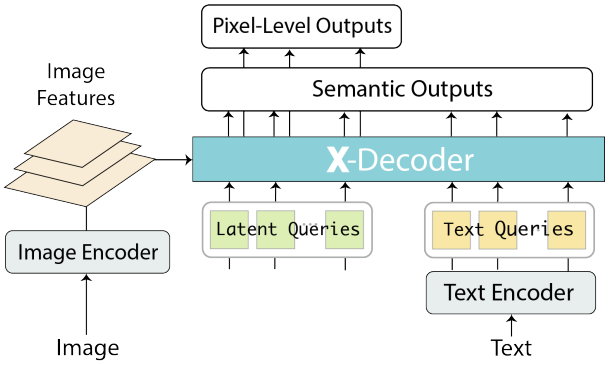


Figure 4.2: Overall pipeline for our model. It consists of an image encoder, a text encoder and our own designed *X-Decoder*.

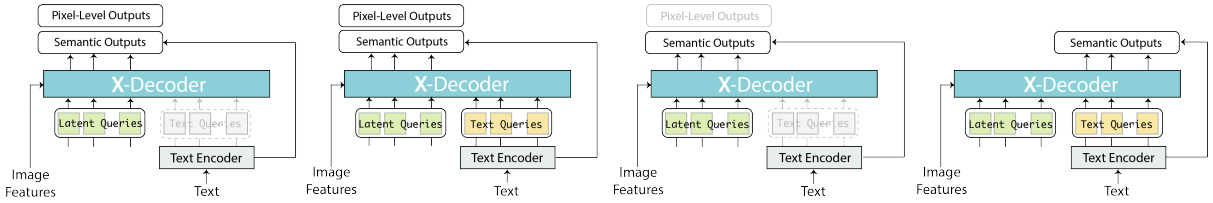


Figure 4.3: Unifying four different types of tasks with our proposed *X-Decoder*. From left to right, they are: (a) generic semantic/instance/panoptic segmentation; (b) referring segmentation; (c) image-text retrieval and (d) image captioning and VQA. The components with white text indicate not applied.

4.3 X-Decoder

4.3.1 Formulation

Our model follows the generic design of encoder-decoder architecture as shown in Fig. 4.2. Given an input image $\mathbf{I} \in \mathcal{R}^{H \times W \times 3}$, we first use an image encoder $\mathbf{Enc}_I$ to extract features $\mathbf{Z}$. Afterwards, we use the text encoder $\mathbf{Enc}_T$ to encode a textual query $\mathbf{T}$ into $\mathbf{Q}_t = \langle q_t^1, \cdots, q_t^n \rangle$ of length n. The visual features, textual queries and the m non-semantic or latent queries $\mathbf{Q}_h = \langle q_h^1, \cdots, q_h^m \rangle$ are fed to our *X-Decoder* to predict the outputs:

$$\langle \mathbf{O}^p_{\{h,t\}}, \mathbf{O}^s_{\{h,t\}} \rangle = \mathbf{XDec}\left(\langle \mathbf{Q}_h, \mathbf{Q}_t \rangle; \mathbf{Z}\right) \tag{4.1}$$

where $\mathbf{O}^p_{\{h,t\}}$ and $\mathbf{O}^s_{\{h,t\}}$ are the pixel-level masks and token-level semantics for latent and textual queries, respectively. In the above formula, we note three critical designs to empower the generalization ability of our to a variety of vision and vision-language tasks.

We define two types of queries and outputs for *X-Decoder*. As discussed earlier, the queries for the decoder are categorized into latent queries $\mathbf{Q}_h$ and text queries $\mathbf{Q}_t$, which undertake generic vision and vision-language tasks, respectively. Likewise, the output is categorized into pixel-level masks and semantic embeddings. By simply using

different combinations, we can adapt our *X-Decoder* to various tasks with the same suite of parameters.

We employ a single text encoder $\mathbf{Enc}_T$ to encode the textual corpus from all tasks. The common text encoder is used to encode referring phrases, text descriptions, image captions in the task of referring segmentation, image-text retrieval and image captioning, respectively. Furthermore, we reformulate the mask classification in segmentation into a mask-text matching problem between $\mathbf{O}^{\mathbf{s}}$ and the textual embeddings of prompted textual concepts similar to [260, 61]. Sharing the text encoder for all textual corpus could maximally exchange knowledge from different tasks and learn a richer and more coherent semantic space.

We fully decouple the image and text encoder. In many previous unified encoder-decoder models [97, 265, 26], the image and text are fused in the encoder side. This design makes it intractable not only for global image-text contrastive learning [187, 260], but also generative pretraining [231]. In contrast, by fully decoupling the image and text encoder and using the outputs all as queries, *X-Decoder* can learn from both intra-image supervisions and inter-image ones, which is essential to learn stronger pixel-level representations and support different granularity of tasks.

4.3.2 Unification of Tasks

Based on the above designs, *X-Decoder* can be used to seamlessly unify different vision and vision-language tasks, simply with different combinations of queries as inputs.

Generic Segmentation. For this task, there are no textual queries as inputs. Hence, Eq. (4.1) becomes:

$$\langle \mathbf{O}_h^p, \mathbf{O}_h^s \rangle = \mathbf{XDec}(\mathbf{Q}_h; \mathbf{Z}) \tag{4.2}$$

where $\mathbf{O}_h^p$, $\mathbf{O}_h^s$ correspond and have the same size to the latent queries $\mathbf{Q}_h$. For generic segmentation, our X-Decoder resembles Mask2former [35] but with open-vocabulary capacity since it transforms mask classification into a mask-text matching problem.

Referring Segmentation. It requires both latent and text queries as inputs:

$$\langle \mathbf{O}_h^p, \mathbf{O}_h^s \rangle = \mathbf{XDec}(\langle \mathbf{Q}_h, \mathbf{Q}_t \rangle; \mathbf{Z}) \tag{4.3}$$

and only uses the decoded outputs corresponding to the latent queries. Compared with Eq. (4.2), Eq. (4.3) can be considered as language-conditioned generic segmentation.

Image-Text Retrieval. The decoupled image and text encoder in our *X-Decoder* makes it straightforward for inter-image retrieval tasks. Specifically, we only feed the latent queries to the decoder and obtain the semantic representation of an image:

$$\mathbf{O}_h^s = \mathbf{XDec}\left(\mathbf{Q}_h; \mathbf{Z}\right) \tag{4.4}$$

where the last (m-th) token in $\mathbf{O}_h^s$ is used as the image representation to compute the similarities to texts.

Image Captioning and VQA. For both tasks, *X-Decoder* takes both latent and text queries and decodes the outputs:

$$\mathbf{O}_t^s = \mathbf{XDec}\left(\langle \mathbf{Q}_h, \mathbf{Q}_t \rangle; \mathbf{Z}\right) \tag{4.5}$$

where $\mathbf{O}_t^s$ correspondingly has equal size to $\mathbf{Q}_t$, and no masks are predicted. There are two slight differences between the two tasks. First, the caption prediction follows a causal masking strategy while VQA does not. Second, we use all the outputs in $\mathbf{O}_t^s$ for captioning, but only the last one to predict the answer for VQA.

The adaptation of our *X-Decoder* to each task is further depicted in Fig. 4.3. Based on this unification, we can pretrain our jointly with all tasks using a proper combination of queries and losses, and further finetune for individual tasks without any extra heads [2]. As discussed earlier, a lineup of works exploited a sequential decoding interface for the unification [42, 234, 26, 26, 265, 160]. However, in this work, we advocate the unification by *functionality* rather than interface, namely, we maximally share the common parts of different tasks while keeping the remaining unchanged for individual tasks.

4.3.3 Unified Architecture

We follow Mask2Former [35] to build our decoder architecture. Given an image $\mathbf{I} \in \mathcal{R}^{H \times W \times 3}$, we extract hierarchical visual features from L layers:

$$\mathbf{Z} = \mathbf{Enc}_I(\mathbf{I}) = \langle \mathbf{z}_l \rangle_{l=1}^{L} \tag{4.6}$$

where $\mathbf{z}_l \in \mathcal{R}^{H_l \times W_l \times d}$ and $\{H_l, W_l\}$ is the size of feature map at level l and d is the feature dimension. These hierarchical feature maps are important for pixel-level understanding at different scales.

One Decoder XDec for All Tasks. Given the visual features $\mathbf{Z}$, *X-Decoder* uses a stack of transformer layers to refine the queries and render the outputs. At layer l, it first cross-attends the visual features and then performs self-attention among latent and text queries:

$$\langle \hat{\mathbf{Q}}_h^{l-1}, \hat{\mathbf{Q}}_t^{l-1} \rangle = \mathbf{CrossAtt}(\langle \mathbf{Q}_h^{l-1}, \mathbf{Q}_t^{l-1} \rangle; \mathbf{Z}) \tag{4.7}$$

$$\langle \mathbf{Q}_h^{l}, \mathbf{Q}_t^{l} \rangle = \mathbf{SelfAtt}(\langle \hat{\mathbf{Q}}_h^{l-1}, \hat{\mathbf{Q}}_t^{l-1} \rangle) \tag{4.8}$$

In Eq. (4.7), we let all queries cross-attend the visual features. For latent queries, we use a masked cross-attention mechanism as in [35], and full attention for the textual

[2] VQA is not used for pretraining following common practice.

queries. In Eq. (4.8), we specifically design the self-attention mechanism: (i) we use the last latent query to extract the global image representation and the remaining for generic segmentation; (ii) for image captioning, each textual query can attend itself, its predecessors and all latent queries; (iii) for referring segmentation, latent queries attend all text queries to use it as the language condition. Based on these rules, the resulting self-attention in our *X-Decoder* is shown in Fig. 4.4.

Method	Type	Generic Segmentation						Referring	Retrieval				Captioning		VQA	
		ADE			COCO			g-Ref	COCO-Karpathy		F30k-Karpathy		COCO-Karpathy		VQAv2-test	
		PQ	mAP	mIoU	PQ	mAP	mIoU	cIoU	IR@1	TR@1	IR@1	TR@1	CIDEr	BLEU	dev	std
Mask2Former (T) [35]	Segmentation	39.7	26.4	47.7	53.2	43.3	63.2	-	-	-	-	-	-	-	-	-
Mask2Former (B) [35]		⋆	⋆	53.9	56.4	46.3	67.1	-	-	-	-	-	-	-	-	-
Mask2Former (L) [35]		48.1	34.2	56.1	57.8	**48.6**	67.4	-	-	-	-	-	-	-	-	-
Pano/SegFormer (B) [139, 252]		⋆	⋆	51.0	55.4	⋆	⋆	-	-	-	-	-	-	-	-	-
kMaX-DeepLab (L) [273]		48.7	⋆	54.8	**58.1**	⋆	⋆	-	-	-	-	-	-	-	-	-
LAVT (B) [264]		-	-	-	-	-	-	61.2	-	-	-	-	-	-	-	-
UNITER (B) [30]	Vision Language (VL)	-	-	-	-	-	-	-	50.3	64.4	72.5	85.9	-	-	72.7	72.9
UNITER (L) [30]		-	-	-	-	-	-	-	52.9	65.6	75.6	87.3	-	-	73.8	74.0
VinVL (B) [289]		-	-	-	-	-	-	-	58.1	74.6	⋆	⋆	129.3	38.2	76.0	76.1
VinVL (L) [289]		-	-	-	-	-	-	-	**58.8**	75.4	⋆	⋆	130.8	38.5	76.5	76.6
ALBEF-4M (B) [128]		-	-	-	-	-	-	-	56.8	73.1	82.8	94.3	⋆	⋆	74.5	74.7
METER-Swin (B) [54]		-	-	-	-	-	-	-	54.9	73.0	79.0	92.4	⋆	⋆	76.4	76.4
UViM (L) [109]	General Purpose	⋆	⋆	⋆	45.8 [1]	⋆	⋆	-	-	-	-	-	-	-	-	-
UniT (T) [85]		-	-	-	-	-	-	-	-	-	-	-	-	-	67.6	⋆
GPV (T) [71]		-	-	-	-	-	-	-	-	-	-	-	102.3 [2]	⋆	62.5	⋆
UniTAB (B) [265]		-	-	-	-	-	-	-	-	-	-	-	119.8	36.1	70.7	71.0
Pix2Seq v2 (B) [26]		-	⋆	-	-	38.2	-	-	-	-	-	-	⋆	34.9	-	-
Unified-IO (B) [160]		-	⋆	-	-	⋆	-	-	-	-	-	-	⋆	⋆	61.8	⋆
Unified-IO (L) [160]		-	⋆	-	-	⋆	-	-	-	-	-	-	⋆	⋆	67.8	⋆
GLIPv2 (T) [285]		-	⋆	-	-	-/42.0	-	⋆	-	-	-	-	122.1	⋆	71.6	71.8
GLIPv2 (B) [285]		-	⋆	-	-	-/45.8	-	⋆	-	-	-	-	128.5	⋆	73.1	73.3
GLIPv2 (H) [285]		-	⋆	-	-	-/48.9	-	⋆	-	-	-	-	131.0	⋆	74.6	74.8
X-Decoder (T)		41.6	27.7	51.0	52.6	41.3/-	62.4	59.8 | 61.9	49.3	66.7	74.4	89.1	122.3	37.8	70.6	70.9
X-Decoder (B)		46.8	33.5	54.6	57.0	47.4/-	66.7	62.4 | 64.5	54.5	71.2	80.8	93.2	129.0	39.6	74.1	74.2
X-Decoder (L)		**49.6**	**35.8**	**58.1**	57.9	**48.6/-**	**67.8**	**64.6 | 64.6**	58.6	**76.1**	**84.4**	**94.4**	**132.1**	**40.2**	**76.8**	**77.0**

Table 4.1: **Task-specific transfer** of *X-Decoder* to different segmentation and VL tasks. Note: "⋆" denotes the model has the capability for the task but does not have number reported. "-" means the model does not have the ability for the specific task. "model name" means the model does not have task specific finetune. "1" is the reported pretrained number for UViM, the corresponding X-Decoder (L) has pretrained PQ 56.9. "2" is the reported coco test2014 value for GPV. "a|b" means "pretrain|finetune". "a/b" indicate "val/test".

As we illustrated in Sec. 4.3.2, *X-Decoder* always produces the masks only for the m latent queries, i.e., $\mathbf{O}^p_h = \{o^p_1, \cdots, o^p_m\} \in \{0,1\}^{m\times H\times W}$ for all the latent queries. As for the semantic outputs, *X-Decoder* predicts the outputs for both latent and text queries, i.e., $\mathbf{O}^s_{\{h,t\}} = \{o^s_1, \cdots, o^s_{m+n}\} \in \mathcal{R}^{(m+n)\times d}$, to cover both mask recognition and caption generation.

One Encoder Enc_T for All Texts. Given the raw text such as a phrase or caption, we convert it to discrete tokens using an off-the-shelf tokenizer and then send it to the text encoder [187]. We apply causal masking to ensure its outputs are compatible with caption decoding. For segmentation, we follow [187, 260] to convert the class name into a phrase with a text prompt (*e.g.*, , "dog" $\rightarrow$ "an image of dog"), and encode the phrase as above.

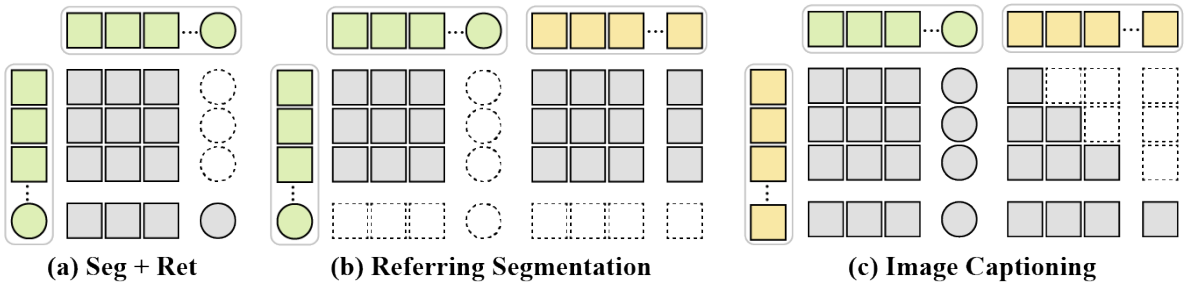


Figure 4.4: Interaction among latent queries (green), between latent and text queries (yellow) for (a) Generic segmentation and image/text retrieval (b) referring segmentation and (c) image captioning. The square latent query is designated for image-text retrieval.

4.3.4 End-to-End Pre-training

We train our *X-Decoder* in an end-to-end manner with two types of losses corresponding to the outputs.

Semantic Loss. There are three losses on the semantic outputs corresponding to three tasks. For image-text retrieval, we compute the image-language contrastive loss as [187]. We take the last valid token feature of $\mathbf{Q}_t$ from the text encoder to represent text as $\hat{q}_t$ and take the last (m-th) entry in $\mathbf{O}_h^s$ derived from *X-Decoder* as $\hat{o}^s$, and obtain B pairs of features for a minibatch of B image-text pairs. Afterwards, we compute the dot-product between these $B \times B$ feature pairs to obtain affinity matrix $\mathbf{S}_{it} \in \mathcal{R}^{B \times B}$, and compute the bidirectional cross-entropy loss:

$$\mathcal{L}_{it} = \mathbf{CE}(\mathbf{S}_{it}, \mathbf{y}_{it}) + \mathbf{CE}(\mathbf{S}_{it}^T, \mathbf{y}_{it}) \tag{4.9}$$

where $\mathbf{y}_{it}$ are the class labels corresponding to diagonal entries in $\mathbf{S}_{it}$, and $\mathbf{S}_{it}^T$ is the transpose of $\mathbf{S}_{it}$.

For mask classification, we encode all C class names including "background" into C text queries and take the last valid token feature from each to represent the concept. Afterward, we take the decoder outputs corresponding to the first $(m-1)$ latent queries and compute the dot-product between these outputs and concept embeddings to obtain an affinity matrix $\mathbf{S}_{cls} \in \mathcal{R}^{(m-1) \times C}$ and compute the loss $\mathcal{L}_{cls} = \mathbf{CE}(\mathbf{S}_{cls}, \mathbf{y}_{cls})$, with the corresponding ground-truth class $\mathbf{y}_{cls}$ guided by Hungarian Matching [18].

For image captioning, we first extract the embeddings for all tokens in the vocabulary of size V from the text encoder. Given the last n semantic outputs from *X-Decoder*, we compute the dot-product with all token embeddings to obtain an affinity matrix $\mathbf{S}_{cap} \in \mathcal{R}^{n \times V}$. Then we compute the cross-entropy loss $\mathcal{L}_{cap} = \mathbf{CE}(\mathbf{S}_{cap}, \mathbf{y}_{cap})$, with the ground-truth next-token id $\mathbf{y}_{cap}$.

Mask Loss. Given the $\mathbf{O}_h^p$ derived from m latent queries, we use the computed correspondence based on Hungarian Matching [18] and follow [35] to use binary cross-entropy

loss $\mathcal{L}_{bce}$ and dice loss $\mathcal{L}_{dice}$ to compute the loss for masks.

Finally, we combine the above four losses to pretrain our model with segmentation and image-text pair data.

4.4 Experiments

Datasets and Settings. We pretrain *X-Decoder* on three types of data including panoptic segmentation, image-text pairs (itp), and referring segmentation. For panoptic and referring segmentation, we use COCO2017 [145] with segmentation annotations and exclude the validation sets of Ref-COCOg UMD [272] and COCO Karpathy [270]. In total, there are 104k images for segmentation pretraining, out of which 30k images are with referring segmentation annotations. For image-text pairs, we use the standard 4M corpora, including Conceptual Captions [202], SBU Captions [178], Visual Genome [111], and COCO Captions [29]. We broadly evaluate our models on all tasks covered by pretraining. In particular, we benchmark on 10 settings of 7 datasets covering a wide range of domains on zero-shot segmentation. Moreover, we finetune and report results on VQA for fine-grained visual reasoning.

Implementation Details. Our visual encoder follows [35] to use 100 latent queries and 9 decoder layers, and we add one additional latent query for image-level task. However, we do not adopt a deformable encoder as it does not generalize well to open-vocabulary settings. We adopt Focal-T [259] and DaViT-B/L [49] as the vision encoder and a transformer text encoder with causal masking [187, 277] as language encoder. The models are pre-trained on large-scale image-text data [277] (Base or Large) or UniCL [260] for the tiny model.

4.4.1 Task-Specific Transfer

Without any architecture change except adding a head for VQA, we directly finetune *X-Decoder* to demonstrate its task transfer capability. Table 4.1 presents the comparisons with previous specialized and generalized models.

Comparison with segmentation models. We list the most recent published models for individual tasks, including Mask2Former [35], Panoptic SegFormer [139], KMaX-DeepLab [273] for generic segmentation, and LAVT [264] for referring segmentation. Notably, our 25 epoch finetuned *X-Decoder* (L) ***establishes a new SoTA on ADE20k dataset*** that outperforms the current SoTA KMaX-DeepLab (L) on ADE Panoptic Segmentation (our model trained with 1024 resolution achieves 51.0 PQ), as well as Instance Segmentation SoTA, Mask2Former-L. On COCO, our model attains comparable or better performance to Mask2Former and kMaX-DeepLab. Finally, we compare with LAVT [264] on COCO G-ref. It is worth pointing out that with lightweight finetuning,

Model	COCO (p/s)			ITP	Fix	EM	Pseudo	ADE-150			A-847	VOC	PC-59	PC-459	SUN	SCAN-20		SCAN-41	Cityscapes				BDD	
	m	cls	cap					PQ	mAP	mIoU	mIoU	mIoU	mIoU	mIoU	mIoU	mIoU	PQ	mIoU	mIoU	mAP	PQ	mIoU	PQ
MSeg (B) [115]	✓	✓	✗	✗	✗	✗	✗	33.7	32.6	19.1	⋆	73.4	43.4	⋆	29.6	33.4	⋆	⋆	46.9	24.8	51.1	44.9	⋆
GroupViT (S)	✗	✗	✗	✓	✗	✓	✗	-	-	⋆	⋆	52.3	22.4	⋆	⋆	⋆	-	⋆	⋆	-	-	⋆	-
LSeg+ (B) [121]	✓	✓	✗	✗	✓	✓	✗	-	-	18.0	3.8	⋆	46.5	7.8	⋆	⋆	-	⋆	⋆	-	-	⋆	-
ZegFormer (B) [47]	✓	✓	✗	✗	✓	✓	✗	-	-	⋆	8.1	80.7	⋆	⋆	⋆	⋆	-	⋆	⋆	-	-	⋆	-
OpenSeg (B) [68]	✓	✗	✓	✗	✓	✓	✓	-	-	21.1	6.3	70.3	45.9	9.0	⋆	⋆	-	⋆	⋆	-	-	⋆	-
OpenSeg (B) [68]	✓	✗	✓	✓	✓	✓	✓	-	-	26.4	8.1	70.2	44.8	11.5	⋆	⋆	-	⋆	⋆	-	-	⋆	-
MaskCLIP (L) [50]	✓	✓	✗	✗	✓	✓	✗	15.1	6.0	23.7	8.2	⋆	45.9	10.0	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆
X-Decoder-Seg (B)	✓	✓	✗	✗	✗	✗	✗	15.3	8.3	19.5	2.9	95.7	63.5	13.3	33.0	41.6	32.5	22.4	47.3	22.8	35.2	44.1	14.1
X-Decoder-Seg$^+$ (B)	✓	✓	✓	✗	✗	✗	✗	16.9	9.5	23.8	4.6	97.8	64.7	12.1	32.2	35.1	33.8	18.5	47.6	**25.9**	36.9	42.7	16.6
X-Decoder (T)	✓	✓	✓	✓	✗	✗	✗	18.8	9.8	25.0	6.4	96.2	62.9	12.3	34.5	37.8	30.7	21.7	47.3	16.0	37.2	42.4	16.4
X-Decoder (B)	✓	✓	✓	✓	✗	✗	✗	21.1	11.7	27.2	8.2	**97.9**	**65.1**	14.7	39.6	40.3	35.4	24.8	50.8	22.3	**39.5**	45.1	17.1
X-Decoder (L)	✓	✓	✓	✓	✗	✗	✗	**21.8**	**13.1**	**29.6**	**9.2**	97.7	64.0	**16.1**	**43.0**	**49.5**	**39.5**	**29.7**	**52.0**	24.9	38.1	**47.2**	**17.8**

Table 4.2: **One suite of model weights** for open-vocabulary image segmentation. Note: "ITP" means image-text pairs. "Fix" indicates whether contains fixed text/image encoder. "EM" means whether the model has extra modules that are designed for open-vocabulary settings (e.g. Adaptor, class agnostic proposal, and etc.). "Pseudo" means whether the method uses an extra step to extract pseudo label image-text pairs. "gray" color means a fully supervised approach. "light purple" color means a semi-supervised learning approach. "FL-in21k" means the backbone is pretained with in21k data using a FocalNet backbone. For COCO, different methods use different supervisions of mask (m), class label (cls) and caption (cap). "⋆ and -" follows Table 4.1

our tiny model already outperforms LAVT-Base (**61.9** *v.s.* **61.2**). Further increasing the model size can bring additional gains by **2.6** and **2.7** points respectively, which helps to ***set a new record on this benchmark*** *in the published literature*.

Comparison with VL models. We compare with a set of VL models on image-text retrieval, image captioning and VQA in Table 4.1. *X-Decoder* achieves competitive performance across the board. Specifically, *X-Decoder* outperforms UNITER [30] and rivals VinVL [289] on COCO retrieval, and even beats all the baselines on Flickr30k [184]. Unlike all these works, the image and text encoders are fully decoupled in *X-Decoder*, which leads to a much faster inference speed. On captioning and VQA, our models also demonstrate superior performance to their counterparts. For example, it outperforms VinVL by **1.3** and **1.7** on CIDEr and BLEU, respectively. Note that most of these works use sophisticatedly designed training objectives, such as masked data modeling, image-text matching and hard-negative mining [128, 230, 53]. In contrast, is pretrained with image-text contrastive and image captioning, along with the segmentation losses. The simplicity and effectiveness imply a great potential for using as a general pretraining paradigm for VL.

Comparison with generalist models. We further compare with prior arts that explore general-purpose vision models. Limited works report the generic segmentation performance. Our model outperforms UViM [109] and Pix2Seq v2 [26] significantly on COCO panoptic (**56.9** *v.s.* **45.8**) and instance segmentation (**46.7** ***v.s.*** **38.2**), respectively (The X-Decoder (L) have the same zero-shot and finetuning performance). With the same amount of segmentation data, these margins strongly justify our model design, , unifying functionality *without* any tweaks for individual tasks. When compared with GLIPv2 [285], our model achieves comparable performance. Note that GLIPv2 uses over 10M pretraining data, including around 2M with box supervision. Despite the

Method	C.E.	M.E	Q.E	Dec.	#Param	ADE PQ	ADE mAP	ADE mIoU	Cityscapes PQ	Cityscapes mAP	Cityscapes mIoU
Mask2Former (T) [35]	-	-	-	-	-	39.7	26.4	47.7	63.9	39.1	80.5
Pano/SegFormer (T) [139, 252]	-	-	-	-	-	36.4	⋆	46.5	⋆	⋆	⋆
kMaX-DeepLab (T) [273]	-	-	-	-	-	41.5	⋆	45.0	64.3	38.5	79.7
Mask2Former (S) [35]	-	-	-	-	-	⋆	⋆	51.3	64.8	40.7	81.8
Mask2Former (B) [35]	-	-	-	-	-	⋆	⋆	53.9	66.1	42.8	82.7
Mask2Former (L) [35]	-	-	-	-	-	48.1	34.9	56.1	66.6	43.6	82.9
X-Decoder (L)	✓	✗	✗	✗	0.26M	44.3	33.2	54.6	65.1	41.4	81.7
	✓	✓	✗	✗	1.05M	43.9	33.2	53.9	64.8	41.2	81.2
	✓	✓	✓	✗	1.15M	44.0	32.8	54.0	64.6	41.1	81.5
	✓	✓	✓	✓	38.3M	**47.0**	**35.1**	**56.0**	**65.6**	**42.2**	**81.7**

Table 4.3: Performance with different efficient finetuning strategies for *X-Decoder* large, and comparisons with fully-finetuned models. Note: C.M denotes class embedding, M.E. denotes mask embedding, Q.E. denotes query embedding, #Param means the number of parameters tuned.

Model	COCO PQ	COCO mAP	COCO mIoU	ADE PQ	ADE mAP	ADE mIoU	COCO-Karparthy IR@1	COCO-Karparthy IR@1	COCO-Karparthy CIDEr	g-Ref cIoU
X-Decoder	**51.4**	**40.5**	**62.8**	14.7	**9.6**	**23.4**	30.7	48.5	**82.0**	**59.7**
* text: [yny]	51.4	39.8	61.7	14.7	9.4	22.2	29.9	46.9	78.6	57.7
* text: [nyy]	51.4	38.6	61.7	15.2	9.4	23.1	30.3	47.5	78.9	59.4
* latent: [yyn]	50.9	39.6	62.0	**15.5**	9.4	22.8	29.8	47.6	81.1	57.6

Table 4.4: **Ablation of query interaction** in *X-Decoder*. [x,x,x] denotes whether attend [object latent query, image latent query, text query]

Model	COCO PQ	COCO mAP	COCO mIoU	ADE PQ	ADE mAP	ADE mIoU	COCO-Karparthy IR@1	COCO-Karparthy TR@1	COCO-Karparthy CIDEr	g-Ref cIoU
* bs 1024	50.9	39.5	62.4	15.2	10.0	24.6	30.6	48.1	85.0	58.0
* bs 768	51.0	39.5	62.4	15.4	10.0	24.2	29.0	46.8	78.6	58.8
* bs 512	50.7	39.3	62.0	14.9	9.7	24.3	27.4	43.8	76.1	58.6

Table 4.5: **Ablation of VL batch size.** We mark the significant drop metrics in green.

Model	COCO PQ	COCO mAP	COCO mIoU	ADE PQ	ADE mAP	ADE mIoU	COCO-Karparthy IR@1	COCO-Karparthy TR@1	COCO-Karparthy CIDEr	g-Ref cIoU
X-Decoder	**51.4**	**40.5**	62.8	14.7	**9.6**	**23.4**	**30.7**	**48.5**	**82.0**	**59.7**
- Retrieval	51.4	40.4	62.6	*14.0*	9.2	*21.8*	n/a	n/a	*78.8*	59.2
- Captioning	*51.1*	40.4	**63.2**	15.0	9.6	23.2	*29.9*	48.1	n/a	*57.7*
- Referring	*51.1*	*39.7*	*62.3*	**15.2**	*8.9*	22.6	30.0	*47.6*	*78.8*	n/a

Table 4.6: **Ablation of pretraining tasks** by removing one at a time. We bold the best entry and underline the worst entry in each column.

Model	COCO PQ	COCO mAP	COCO mIoU	ADE PQ	ADE mAP	ADE mIoU	COCO-Karparthy IR@1	COCO-Karparthy TR@1	COCO-Karparthy CIDEr	g-Ref cIoU
Full Datasets	50.9	39.5	62.4	15.2	10.0	24.6	30.6	48.1	85.0	58.0
- coco	50.9	39.9	62.2	15.3	9.8	24.4	27.4	38.2	32.6	59.4
- cc3m	51.2	39.7	62.6	15.5	10.1	24.6	31.0	50.0	81.2	58.3
- vg	51.1	39.8	62.4	14.6	9.7	23.8	36.1	56.1	107.1	58.3
- sbu	51.1	39.8	62.4	15.3	9.5	24.6	30.3	48.3	81.2	58.3

Table 4.7: **Ablation of VL datasets** in *X-Decoder*. A single VL dataset is removed in each row. And we mark the metrics that significantly drop/increase in green/red.

huge gap in pretraining data, *X-Decoder* outperforms GLIPv2 on both captioning and VQA.

Efficient Finetuning. Finally, we study whether our pretrained *X-Decoder* can be finetuned for segmentation with a low cost. In Table 4.3, we show that we can simply finetune the class embedding layer, mask embedding layer or the whole decoder to reach a decent segmentation performance and surpass the fully finetuned tiny SoTA models like kMaX-DeepLab [273]. These results imply an efficient way of using our pretrained *X-Decoder* models.

4.4.2 Zero-Shot Transfer

Without any change in model weights, *X-Decoder* can be directly applied to various segmentation tasks and datasets after pretraining. In Table 4.2, we evaluate our model

Figure 4.5: An example of region retrieval as a showcase of task composition of image-text retrieval and referring segmentation.

in a zero-shot manner on seven commonly used segmentation datasets in 10 different settings from diverse domains, including common indoor, outdoor and self-driving scenarios. We report PQ, mAP and mIoU for generic segmentation quantitatively.

Comparison with baselines. We build two *X-Decoder* variants: (1) *X-Decoder*-Seg, which is only trained with COCO panoptic segmentation using a text encoder for class names; and (2) *X-Decoder*-Seg$^+$, where we take the heuristic way to extract noun phrases from COCO captions and use them as extra supervision on top of the matched decoder outputs. First, *X-Decoder*-Seg shows clear advantages on open-vocabulary segmentation over MSeg [115], that manually conducts label mapping across different datasets. Second, the extra supervision from COCO captions improves model performance on 9 out of 15 metrics, which indicates the benefit of joint learning with image-level supervision. Third, when pretraining with the full *X-Decoder*, the performance is significantly boosted. Notably, the mIoU metric is improved by **7.4**, **3.4** and **2.6** on SUN, ADE-150 and PC-459, respectively.

Comparison with state-of-the-art. We further compare with the most advanced methods for open-vocabulary image segmentation in Table 4.2. Clearly, our models achieve the best results across all datasets. Among the base-sized models, *X-Decoder* (B) outperforms OpenSeg (B) [61] on two challenging datasets, ADE-150 and PC-459 for semantic segmentation. Scaling *X-Decoder* to large size further improves mIoU by **2.4** and **1.4** on these two datasets. Among prior arts, MaskCLIP [50] is the first proposed for open-vocabulary panoptic segmentation by combining Mask2Former with CLIP models. With COCO caption supervisions, our simple baseline *X-Decoder*-Seg$^+$ already performs comparably. The full version of our tiny model *X-Decoder* (T) surpasses MaskCLIP across the board except A-847. We note that these comparisons are not strictly fair in terms of supervision, settings and models used. However, these results demonstrate the effectiveness of our *X-Decoder* to learn from the different granularity of supervisions *end-to-end* for open-vocabulary segmentation, which leads to ***new SoTA on 10 settings of 7 datasets across three segmentation tasks***.

4.4.3 Model Inspection

Pretraining Tasks. By default, we exploit four pre-training tasks including generic and referring segmentation, captioning and retrieval. In Table 4.6, we keep the generic segmentation while ablating the importance of the other pre-training tasks. Accordingly, we have the following observations:

Image-text retrieval help open-vocabulary segmentation: On ADE, mIoU decreases from 23.4 to 21.8 and PQ by 0.7 without image-text retrieval. As both tasks share semantic space, improved visual-semantic alignment boosts recognition of novel concepts.

Image captioning helps referring segmentation and vice versa: COCO g-Ref drops **2.0** pts without training with image captioning, and CIDEr falls **3.2** pts without training with referring tasks. This indicate sharing a text encoder and joint training enhances text input understanding.

Image captioning and retrieval can mutually benefit each other: Removing captioning in pretraining, R@1 drops by **0.8**; and CIDEr decreases **3.2** pts without retrieval task. It indicates *X-Decoder* fosters generative and contrastive learning synergy.

Query Interactions. The interaction among tasks is highly dependent on the interaction between latent and text queries. We have described how the queries interact with each other by default in Fig. 4.4. Here, we investigate how our model behaves with different interactions in Tab. 4.4:

Image captioning requires both fine-grained and global image information: Comparing rows 1-3 in Tab. 4.4, CIDEr score drops significantly when information flow from global latent queries or other latent queries to text queries is cut off (**82.0** $\rightarrow$ **78.6** and **78.9**, respectively).

Language-condition is important for ref-segmentation: In the last row of Tab. 4.4, turning off text-to-latent query interaction significantly decrease ref-segmentation (**59.7** $\rightarrow$ **57.6**) performance, indicating generic segmentation can't be converted to referring segmentation using post-hoc matching with referring texts easily.

VL Batch Size & Dataset The default batch size of VL task is 1024, we explore the gradual decreasing of VL batch size in Tab. 4.5. In addition, each VL dataset is removed to investigate the pre-trained performance on different tasks.

Decreasing VL batch size hurts VL tasks and open-vocab Segmentation performance: As shown in Tab. 4.5, decreasing the VL task batch size from 1024 to 256 significantly hurts the retrieval and captioning performance as well as minor influence on open-vocabulary settings.

VG dataset hurts pretraining VL tasks performance but improves open-vocab segmentation: As shown in Table 4.7, removing the visual genome from the pretraining VL dataset significantly improves captioning task with 22.1 points in pretraining caused by the different annotation style of that dataset.

4.4.4 Task Composition

X-Decoder has the unique benefit of task interaction. We demonstrate our model can perform region-based retrieval (Fig. 4.5) and referring-based captioning (Fig. 6.1) without any architecture/weight change. As shown in Fig. 4.5, given a set of animal images and text query, our model first retrieves the correct image and then grounds the query in pixel level. Further, our model can be easily adapted to referring captioning by localizing a given word and then modulating the predicted mask in the cross-attention layers for text queries. This will allow the text queries to focus on the grounded region only. Thus lead to the generated caption that focus on the specific area. Lastly, we also integrate X-Deocder with diffusion model for referring image editing and inpainting. This has been demonstrated in Fig. 6.1.

4.5 Conclusion

We introduce *X-Decoder*, a versatile model for pixel and image-level vision-language understanding. Its unified design enables generic segmentation, referring segmentation, and VL tasks with strong generalizability and SoTA/Comparable performance. We hope this work can shed a light on the design of the next-generation general-purpose vision system.

Chapter 5

Segment everything everywhere all at once

Publication Statement. This chapter is joint work with Jianwei Yang, Hao Zhang, Feng Li, Linjie Li, Jianfeng Wang, Lijuan Wang, Jianfeng Gao, Yong Jae Lee. The paper version of this chapter appeared in NeurIPS 2023 [302].

In this work, we present ***SEEM***, a promptable and interactive model for segmenting everything everywhere all at once in an image, as shown in Fig. 5.1. In *SEEM*, we propose a novel decoding mechanism that enables diverse prompting for all types of segmentation tasks, aiming at a universal segmentation interface that behaves like large language models (LLMs). More specifically, *SEEM* is designed with four desiderata: i) Versatility. We introduce a new visual prompt to unify different spatial queries including points, boxes, scribbles and masks, which can further generalize to a different referring image; ii) Compositionality. We learn a joint visual-semantic space between text and visual prompts, which facilitates the dynamic composition of two prompt types required for various segmentation tasks; iii) Interactivity. We further incorporate learnable memory prompts into the decoder to retain segmentation history through mask-guided cross-attention from decoder to image features; and iv) Semantic-awareness. We use a text encoder to encode text queries and mask labels into the same semantic space for open-vocabulary segmentation. We conduct a comprehensive empirical study to validate the effectiveness of *SEEM* across diverse segmentation tasks. Notably, our single *SEEM* model achieves competitive performance across interactive segmentation, generic segmentation, referring segmentation, and video object segmentation on 9 datasets with minimum 1/100 supervision. Furthermore, *SEEM* showcases a remarkable capacity for generalization to novel prompts or their combinations, rendering it a readily universal image segmentation interface.

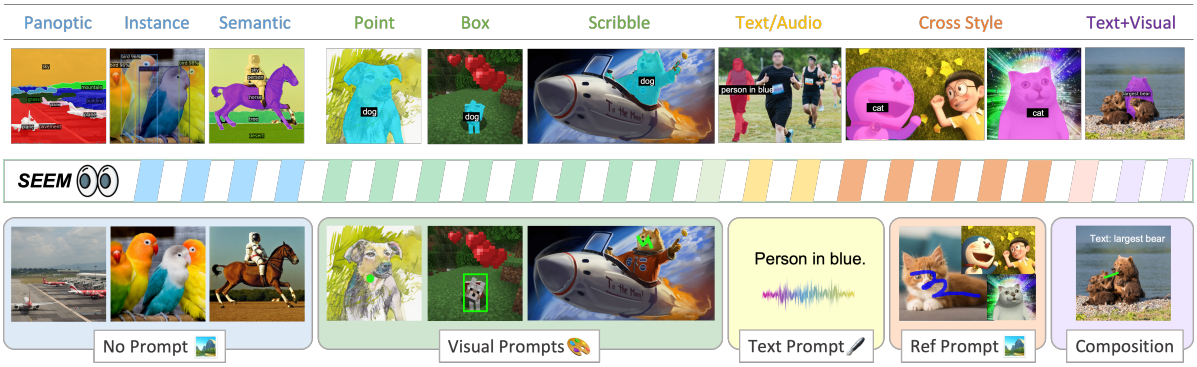


Figure 5.1: *SEEM* supports generic segmentation tasks—including semantic, instance, and panoptic segmentation—in an open-set fashion when no prompt is provided. *SEEM* also enables the use of visual, textual, and referring region prompts in flexbile combinations, making it a promptable and interactive segmentation interface.

5.1 Introduction

Image segmentation is arguably the most important yet challenging problem in computer vision. In the past, we have witnessed significant progress in a wide range of segmentation tasks including instance, semantic and panoptic segmentation [203, 21, 102, 75, 229, 35, 125]. Most recently, we are observing a clear trend toward more flexible segmentation models in different aspects: 1) From closed-set to open-vocabulary segmentation. Many recent works proposed to either leverage contrastive learning methods or pretrained multi-modal foundation models (*e.g.*, CLIP [187]) to make the segmentation models more transferable to unseen concepts [61, 50, 298, 255]; 2) From generic to referring segmentation. In addition to generic segmentation that segments an image thoroughly given a predetermined set of concepts, language-based referring segmentation provides a user-friendly way of segmenting a specific region referred by an arbitrary text phrase [148, 268, 244, 87, 150]; and 3) From one-shot to interactive segmentation. In practice, segmentation models do not necessarily produce satisfactory masks in one round. As such, people are also studying how to progressively refine the segmentation results through intimate interactions between humans and models [207, 39, 152, 27].

Despite the aforementioned efforts taken to design more powerful and feasible segmentation models, we are still lacking a universal segmentation interface that is capable of accommodating various types of human prompts and tackling different segmentation tasks as studied in individual works. In contrast, Large Language Models (LLMs) have already emerged as such a universal interaction interface for language tasks, from early models like GPT-3 [13] and T5 [188], to conversational agent [176] augmented by advanced prompting [204, 293, 132] and chain-of-thought [242, 108, 199]. In this work, we strive for a universal interface for *segmenting everything everywhere all at once* in an

image. On this interface, we are targeted at unifying all segmentation tasks with a single model in a promptable manner. To achieve this goal, we propose a new prompting scheme in mask decoder that has four important properties: *versatility*, *compositionality*, *interactivity*, and *semantic-awareness*. Specifically, we propose to encode points, masks, text, boxes, and even a referred region from another image into prompts in the same joint visual-semantic space. As such, our model can deal with any combination of the input prompts, leading to strong compositionality. To enable interactivity, we further introduce memory prompts for condensing the previous segmentation information followed by communication with other prompts. As for semantic awareness, our model can provide an open-set semantic label to any output segmentation.

With the proposed prompting scheme, we build a segment-everything-everywhere model called ***SEEM*** comprised of a simple Transformer encoder-decoder architecture [18, 35] with an extra text encoder [298, 283]. In *SEEM*, the decoding process emulates a generative LLM but with a multimodality-in-multimodality-out interface. An image encoder and text encoder are used as the prompt encoder to encode all types of queries, which are fed into the decoder. Concretely, we encode all spatial queries, namely, points, boxes, scribbles and masks into *visual prompts* by pooling their corresponding visual features from the image encoder, and use the text encoder to convert text queries into *text prompts*. By training on diverse segmentation tasks, our model learns to deal with various prompts, align the visual and text prompts, and promote their synergy via cross-attention between them. As a result, our single model after pretraining attains competitive performance across all segmentation tasks. Since the prompts of all 5 different types are mapped to the *joint visual-semantic space*, we can feasibly combine prompts to resolve the ambiguity to obtain better segmentation results and enable zero-shot adaptation to unseen user prompts. Furthermore, our model can immediately generalize to the case of using an exemplar image segment as the prompt and video object segmentation in a zero-shot fashion. In addition to its strong generalization capability, *SEEM* is also more efficient for interactive segmentation compared with the counterparts like SimpleClick [283]. Since we take the prompts as input to the decoder, when doing multi-round interactions with humans, our model only needs to run the feature extractor once at the beginning and lightweight decoding each per round. To the end, we build a segmentation interface with a single pre-trained model that can segment every object with semantics (everything), cover every pixel in the image (everywhere), and support all possible compositions of prompts (all at once). In summary, our contributions are threefold:

- We design a new prompting scheme that can encode various user intents into prompts in a *joint visual-semantic space*, enabling strong flexibility for various segmentation tasks and generalization capability to unseen prompts or their combinations.

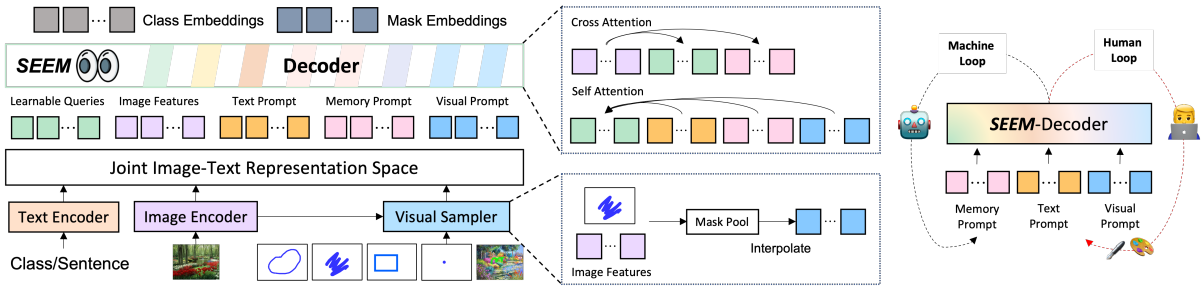
- We build *SEEM*, a universal and interactive segmentation interface that integrates the newly designed prompting mechanism into a lightweight decoder for *all* segmentation tasks, leading to a model possessing properties of versatility, compositionality, interactivity, and semantic awareness.
- We conduct extensive experiments and visualizations to show that our model has strong performance on many segmentation tasks including open-vocabulary generic segmentation, interactive segmentation, referring segmentation, and segmentation tasks with combined prompts.

5.2 Related Work

Interactive segmentation. Interactive segmentation is the task of segmenting objects by interactively taking user inputs. It has been a longstanding problem and has achieved considerable progress [137, 66, 256, 152, 27, 103]. Generally, the interaction types can take various forms, such as clicks, boxes, polygons, and scribbles, among which click-based interaction models are the most prevalent. Concurrent to our work, SAM [103] proposed a promptable segmentation model trained on 11 million images and 1.1 billion masks. It takes user interactions as prompts for general segmentation. Though SAM demonstrates strong zero-shot performance, it produces segmentations without semantic meaning. In addition, its prompt types are limited to points, boxes, and text, whereas our model can also take in a referred region from another image as a prompt.

Generic segmentation. Segmentation of visual concepts has been a persistent challenge in the field of computer vision, as evidenced by its extensive literature [57, 56, 304, 171]. Generic segmentation techniques encompass several subtasks, including instance segmentation, semantic segmentation, and panoptic segmentation [75, 21, 102], each focusing on a different semantic level. For example, semantic segmentation aims to identify and label each pixel within an image based on its corresponding semantic class [23, 35, 158]. On the other hand, instance segmentation involves grouping pixels that belong to the same semantic class into separate object instances [75, 12, 125]. Recently, the Detection Transformer (DETR)[18], a model based on the Transformer [226] architecture, has made significant advances in segmentation [139, 35, 125, 92, 282] tasks. However, these approaches cannot recognize objects absent in the training set, which constrains the model to a limited vocabulary size.

Unified vision models. Unified vision models [298, 258, 160, 103, 240] have recently drawn a lot of attention because of their advantage in generalizing to various tasks and flexibility. These models can deal with multiple vision tasks or data distributions. Among them, some [298, 258, 160] train multiple tasks together with only one model and thus can deal with all training tasks without finetuning on each target task. On



(a) Model Architecture (b) Human-Model Interaction

Figure 5.2: Overview of *SEEM*- Decoder. (a) *SEEM* encodes image, text, and human inputs into *joint visual-semantic space* as queries, features, and prompts, and then decodes queries to class and mask embeddings. (b) With the benefit of *SEEM* decoder, the machine loop enables memorizing history mask information, and the human loop provides new corrections to the next round.

the other hand, SAM [103] and SegGPT [240] propose training strategies that enable their models to handle new tasks and data distributions in a zero-shot manner. The second approach is more favorable since there is no need to resolve conflicts among tasks during training.

5.3 Method

5.3.1 Model Design

SEEM employs a generic encoder-decoder architecture but also employs a sophisticated interaction scheme between queries and prompts, as shown in Fig. 5.2 (a). Given an input image $\mathbf{I} \in \mathcal{R}^{H \times W \times 3}$, an image encoder is first used to extract image features $\mathbf{Z}$. Then, *SEEM*-Decoder predicts the masks $\mathbf{M}$ and semantic concepts $\mathbf{C}$ based on the query outputs $\mathbf{O}_h^m$ (mask embeddings) and $\mathbf{O}_h^c$ (class embeddings), which interact with text, visual, and memory prompts $\langle \mathbf{P}_t, \mathbf{P}_v, \mathbf{P}_m \rangle$:

$$\langle \mathbf{O}_h^m, \mathbf{O}_h^c \rangle = \mathbf{Decoder}(\mathbf{Q}_h; \langle \mathbf{P}_t, \mathbf{P}_v, \mathbf{P}_m \rangle | \mathbf{Z}) \tag{5.1}$$

$$\mathbf{M} = \mathbf{MaskPredictor}(\mathbf{O}_h^m) \tag{5.2}$$

$$\mathbf{C} = \mathbf{ConceptClassifier}(\mathbf{O}_h^c) \tag{5.3}$$

where $\mathbf{Q}_h$ is the learnable queries, and $\mathbf{P}_t$, $\mathbf{P}_v$, $\mathbf{P}_m$ represent the text prompts, visual prompts, and memory prompts, respectively. During training, $\mathbf{Q}_h$ is duplicated for generic, referring, and interactive segmentation, as shown in Fig. 5.3. The corresponding prompts interact with their queries through self-attention. The learnable queries can freely interact with all prompts at inference time, thereby enabling zero-shot composition. Our design is inspired by the successful practice in X-Decoder [298]. However, we highlight the differences in Eq. (5.1), marked in red, which allow for a universal model

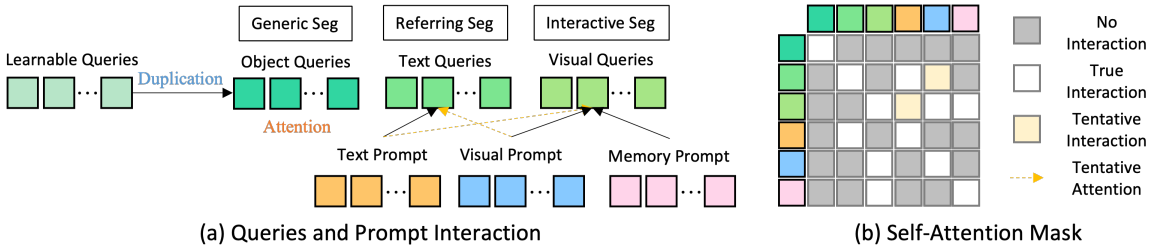


Figure 5.3: Queries and prompt interaction during training and evaluation. (a) Learnable queries are duplicated as object, grounding, and visual queries with the same set of weights for each task. (b) Attention mask between any two kinds of tokens (denoted as qpm in Algorithm. 2). Tentative means the interaction is not trained but able to do inference without any modification.

for image segmentation with the following properties:

Versatile. In *SEEM*, we introduce visual prompts $\mathbf{P}_v$ to handle *all* non-textual inputs, such as points, boxes, scribbles, *and* a referred region from another image. These non-textual queries are beneficial to disambiguate the user's intent when textual prompts alone fail to identify the correct segment. For interactive segmentation, previous works either convert spatial queries to masks and feed them into the image backbone [152] or use different prompt encoders for each input type (points, boxes) [103]. The first approach can be too heavy in applications because each interaction requires the image to go through the feature extractor. The second approach is hard to generalize to unseen prompts. To address these limitations, we propose a visual sampler (Fig. 5.2 (a)) to convert all kinds of non-textual queries to visual prompts that lie in the same visual embedding space:

$$\mathbf{P}_v = \mathbf{VisualSampler}(\mathrm{s}, \hat{\mathbf{Z}}) \tag{5.4}$$

where $\hat{\mathbf{Z}}$ is the feature maps extracted from either the target image (*i.e.*, $\hat{\mathbf{Z}} = \mathbf{Z}$) or a referred image, and $s \in \{\text{points}, \text{box}, \text{scribbles}, \text{polygons}\}$ are the sampling locations specified by the user. We first pool the corresponding region from the image feature through point sampling [35]. For all visual prompts, we interpolate at most 512 point feature vectors uniformly from the region specified by the prompt. A notable merit of our proposed method is that the visual prompts are naturally well-aligned with the textual prompts, as our model continuously learns a common visual-semantic space through panoptic and referring segmentation.

Compositional. In practice, a user may cast their intent using different or combined prompt types. Hence, a compositional approach to prompting is essential for real-world applications. However, we confront two issues during model training. First, the training data usually only covers a single type of interaction (e.g., none, textual, visual). Second, although we use visual prompts to unify all non-textual prompts and align them with textual prompts, their embedding spaces remain inherently different. To mitigate this problem, we propose to match prompts of different types with different

outputs. Considering that visual prompts $\mathbf{P}_v$ come from image features while textual prompts $\mathbf{P}_t$ come from the text encoder, we select matched output indices for visual and textual prompts by matching them with the mask embeddings $\mathbf{O}_h^m$ or class embeddings $\mathbf{O}_h^c$, respectively:

$$ID_v \leftarrow \mathbf{Match}(\mathbf{O}_h^m \cdot \mathbf{P}_v + \mathbf{IoU}_{mask}) \tag{5.5}$$

$$ID_t \leftarrow \mathbf{Match}(\mathbf{O}_h^c \cdot \mathbf{P}_t + \mathbf{IoU}_{mask}) \tag{5.6}$$

where $\mathbf{IoU}_{mask}$ is the IoU between ground-truth and predicted masks. The proposed separate matching method outperforms approaches that only match with either $\mathbf{O}_h^m$ or $\mathbf{O}_h^c$ for all prompts.

After training, our model becomes familiar with all prompt types and supports a variety of compositions, such as no prompts, one prompt type, or both visual and textual prompts using the same model and weights. *In particular, the visual and textual prompts can be simply concatenated and fed to SEEM-Decoder, even though it was never trained in this way.*

Interactive. Interactive segmentation usually cannot be completed in one shot and requires multiple interaction rounds for refinement, similar to conversational agents like ChatGPT. In *SEEM*, we propose a new type of prompt called *memory prompts* $\mathbf{P}_m$ and use them to convey the knowledge of the masks from the previous iteration to the current one. Unlike previous works that use a network to encode the previous mask [152, 103], we introduce no extra module but simply a few memory prompts. These memory prompts encode the history information by using a mask-guided cross-attention layer [35]:

$$\mathbf{P}_m^l = \mathbf{MaskedCrossAtt}(\mathbf{P}_m^{l-1}; \mathbf{M}_p | \mathbf{Z}) \tag{5.7}$$

where $\mathbf{M}_p$ is the previous mask, and $\mathbf{Z}$ is the image feature map. In this way, cross-attention only takes effect inside the regions specified by the previous mask. The updated memory prompts $\mathbf{P}_m^l$ then interact with the other prompts via self-attention to convey the historical information for the current round.

Semantic-aware. Different from previous class-agnostic interactive segmentation works such as Simple Click [152] and the concurrent work SAM [103], our model produces semantic labels to masks for all kinds of prompt combinations in a zero-shot manner, since our visual prompt features are aligned with textual features in a *joint visual-semantic space*. As shown in Fig. 5.3, semantic labels are directly computed using $\mathbf{O}_h^c$ (output of visual queries) and the text embedding. Although **we do not train with any semantic labels for interactive segmentation**, the calculated logits are well-aligned, benefiting from the *joint visual-semantic space*.

Algorithm 1 Pseudo code for SEEM.

```
# Inputs: Image(img)[B,3,H,W]; Pos_Mask(pm), Neg_Mask(nm)[B,1,H,W]; Text(txt)[abc...];
# Variables: Learnable Queries(Q_h); Attention Masks between Q and P(qpm)
# Functions: Img_Encoder(),Text_Encoder(),Visual_Sampler(),feature_attn(),prompt_attn(),output();

def init( ):
    Q_o,Q_t,Q_v = Q_h.copy();# Initialize object, text and visual queries.
    F_v,P_t = Img_Encoder(img), Text_Encoder(txt);# F_v and P_t denote image feature, text prompt.
    P_v = Visual_Sampler(F_v, pm, nm);# Sample visual prompt from image feature, pos/neg mask.
def SEEM_Decoder(F_v,Q_o,Q_t,Q_v,P_v,P_t,P_m):
    Q_o,Q_t,Q_v = feature_attn(F_v,Q_o,Q_t,Q_v);# Cross attend queries with image features.
    Q_o,Q_t,Q_v = prompt_attn(qpm,Q_o,Q_t,Q_v,P_v,P_t,P_m);# Self attend queries and prompts.
    O_m,O_c,P_m = output(F_v,Q_o,Q_t,Q_v);# Compute mask and class outputs.
def forward(img,pm,nm,txt):
    F_v,Q_o,Q_t,Q_v,P_v,P_t = init(); P_m = None;# Initialize variables.
    for i in range(max_iter):
        O_m,O_c,P_m = SEEM_Decoder(F_v,Q_o,Q_t,Q_v,P_v,P_t,P_m)
```

5.3.2 Model Pipeline and Loss Functions

We summarize the training and evaluation pipeline of the proposed method with Pytorch-style pseudo-code in Algorithm 2. *SEEM* is trained with a linear combination of losses for panoptic segmentation, referring segmentation, and interactive segmentation:

$$\begin{aligned}\mathcal{L} =& \alpha\mathcal{L}_{\text{c\_CE\_pano}} + \beta\mathcal{L}_{\text{m\_BCE\_pano}} + \gamma\mathcal{L}_{\text{m\_DICE\_pano}} + a\mathcal{L}_{\text{c\_CE\_ref}} + b\mathcal{L}_{\text{m\_BCE\_ref}} \\ &+c\mathcal{L}_{\text{m\_DICE\_ref}} + a\mathcal{L}_{\text{c\_CE\_iseg}} + b\mathcal{L}_{\text{m\_BCE\_iseg}} + c\mathcal{L}_{\text{m\_DICE\_iseg}}\end{aligned} \tag{5.8}$$

Where $\alpha = 2, \beta = \gamma = 5, a = 0.2, b = c = 2$, CE, BCE, and DICE denotes cross-entropy, binary cross entropy and dice loss, respectively.

5.4 Experiments

Datasets and Settings. *SEEM* is trained on three tasks: panoptic segmentation, referring segmentation, and interactive segmentation. Panoptic and interactive segmentation are trained on COCO2017 [145] with panoptic segmentation annotations. Following [298], we exclude the validation set of Ref-COCOg [272], resulting in 107K segmentation images in total. For referring segmentation, we use a combination of Ref-COCO, Ref-COCOg, and Ref-COCO+ for COCO image annotations. We evaluate generic segmentation (instance/panoptic/semantic), referring segmentation, and interactive segmentation.

Implementation Details and Evaluation Metrics. Our model framework follows X-Decoder [298] except the decoder. That is, we have a vision backbone, a language backbone, an encoder, and *SEEM*-Decoder. For the vision backbone, we use FocalT [259], DaViT-d3 (B), and DaViT-d5 (L) [49]. For the language encoder, we adopt a UniCL or Florence text encoder [260, 277]. For all segmentation tasks, we use standard evaluation metrics: PQ (Panoptic Quality) for panoptic segmentation, AP (Average Precision)

Table 5.1: **One model** for segmentation on a wide range of segmentation tasks. *SEEM* is the first model to simultaneously support generic segmentation, referring segmentation, and interactive segmentation, as well as prompt compositionality. (#Concurrent work. - indicates the model does not have capability for the task, * indicates do not have reported number.)

Method	Segmentation Data	Type	Generic Segmentation COCO			Referring Segmentation RefCOCOg			Interactive Segmentation PascalVOC					
			PQ	mAP	mIoU	cIoU	mIoU	AP50	5-NoC85	10-NoC85	20-NoC85	5-NoC90	10-NoC90	20-NoC90
Mask2Former (T) [35]	COCO (0.12M)		53.2	43.3	63.2	-	-	-	-	-	-	-	-	-
Mask2Former (B) [35]	COCO (0.12M)		56.4	46.3	67.1	-	-	-	-	-	-	-	-	-
Mask2Former (L) [35]	COCO (0.12M)		57.8	48.6	67.4	-	-	-	-	-	-	-	-	-
Pano/SegFormer (B) [139]	COCO (0.12M)	Segmentation	55.4	*	*	-	-	-	-	-	-	-	-	-
LAVT (B) [264]	Ref-COCO (0.03M)		-	-	-	61.2	*	*	-	-	-	-	-	-
PolyFormer (B) [150]	Ref-COCO+VG+... (0.16M)		-	-	-	69.3	*	*	-	-	-	-	-	-
PolyFormer (L) [150]	Ref-COCO+VG+... (0.16M)		-	-	-	71.1	*	*	-	-	-	-	-	-
RITM (<T) [207]	COCO+LVIS (0.12M)		-	-	-	-	-	-	*	*	2.19	*	*	2.57
PseudoClick (<T) [153]	COCO (0.12M)		-	-	-	-	-	-	*	*	1.94	*	*	2.25
FocalClick (T) [27]	COCO (0.12M)		-	-	-	-	-	-	*	*	2.97	*	*	3.52
FocalClick (B) [27]	COCO (0.12M)	Interactive	-	-	-	-	-	-	*	*	2.46	*	*	2.88
SimpleClick (B) [152]	COCO+LVIS (0.12M)		-	-	-	-	-	-	1.75	1.93	2.06	1.94	2.19	2.38
SimpleClick (L) [152]	COCO+LVIS (0.12M)		-	-	-	-	-	-	1.52	1.64	1.72	1.67	1.84	1.96
SimpleClick (H) [152]	COCO+LVIS (0.12M)		-	-	-	-	-	-	1.51	1.64	1.76	1.64	1.83	1.98
UViM (L) [109]	COCO (0.12M)		45.8	*	*	-	-	-	-	-	-	-	-	-
Pix2Seq v2 (B) [26]	COCO (0.12M)		-	38.2	-	-	-	-	-	-	-	-	-	-
X-Decoder (T) [298]	COCO (0.12M)		52.6	41.3	62.4	59.8	*	*	-	-	-	-	-	-
X-Decoder (B) [298]	COCO (0.12M)		56.2	45.8	66.0	64.5	*	*	-	-	-	-	-	-
X-Decoder (L) [298]	COCO (0.12M)		56.9	46.7	67.5	64.6	*	*	-	-	-	-	-	-
UNINEXT (T) [258]	Image+Video (3M)		-	44.9	-	70.0	*	*	-	-	-	-	-	-
UNINEXT (L) [258]	Image+Video (3M)		-	49.6	-	73.4	*	*	-	-	-	-	-	-
Painter (L) [239]	COCO+ADE+NYUv2 (0.16M)	Generalist	43.4	*	*	-	-	-	-	-	-	-	-	-
#SegGPT (L) [240]	COCO+ADE+NYUv2 (0.16M)		34.4	*	*	-	-	-	-	-	-	-	-	-
#SAM (B) [103]	SAM (11M)		-	-	-	-	-	-	2.47	2.65	3.28	2.23	3.13	4.12
#SAM (L) [103]	SAM (11M)		-	-	-	-	-	-	1.85	2.15	2.60	2.01	2.46	3.12
#SAM (H) [103]	SAM (11M)		-	-	-	-	-	-	1.82	2.13	2.55	1.98	2.43	3.11
SEEM (T)	COCO+LVIS (0.12M)		50.8	39.7	62.2	60.9	65.7	74.8	1.72	2.30	3.37	1.97	2.83	4.41
SEEM (B)	COCO+LVIS (0.12M)		56.1	46.4	66.3	65.0	69.6	78.2	1.56	2.04	2.93	1.77	2.47	3.79
SEEM (L)	COCO+LVIS (0.12M)		57.5	47.7	67.6	65.6	70.3	78.9	1.51	1.95	2.77	1.71	2.36	3.61
SEEM (T)	COCO+LVIS (0.12M)		-	-	-	70.4	71.7	82.1	1.72	2.28	3.32	1.97	2.82	4.37
SEEM (B)	COCO+LVIS (0.12M)	Composition	-	-	-	76.2	77.8	87.8	1.56	2.03	2.91	1.77	2.46	3.76
SEEM (L)	COCO+LVIS (0.12M)		-	-	-	75.1	76.9	86.8	1.52	1.97	2.81	1.72	2.38	3.64

Table 5.2: **One model** for all kinds of mask interactions. *SEEM* has strong generalization capability on different input mask types.

Method	COCO					Open Image					ADE				
	Point 1-IoU	Stroke 1-IoU	Scribble 1-IoU	Polygon 1-IoU	Box 1-IoU	Point 1-IoU	Stroke 1-IoU	Scribble 1-IoU	Polygon 1-IoU	BoX 1-IoU	Point 1-IoU	Stroke 1-IoU	Scribble 1-IoU	Polygon 1-IoU	BoX 1-IoU
SimpleClick (B)	49.0	33.1	65.1	48.6	42.5	48.6	29.5	54.2	49.5	42.7	47.0	19.0	52.1	48.3	37.2
SimpleClick (L)	38.9	33.9	68.8	39.2	34.7	37.5	29.1	59.8	35.2	31.2	36.8	16.4	56.4	41.7	29.5
SimpleClick (H)	59.0	37.3	71.5	45.3	52.4	54.1	32.6	64.7	39.9	49.3	52.8	18.4	58.3	46.8	41.8
SAM (B)	58.6	22.8	34.2	44.5	50.7	62.3	28.4	39.2	45.8	53.6	51.0	21.9	31.1	31.0	**58.8**
SAM (L)	64.7	44.4	57.1	60.7	50.9	65.3	45.9	55.7	57.8	52.4	57.4	45.8	53.1	45.8	58.7
SAM (H)	65.0	27.7	30.6	37.8	50.4	67.7	26.5	29.9	41.9	52.1	58.4	20.4	22.2	28.3	58.5
SEEM (T)	78.9	81.0	81.2	72.2	73.7	67.1	**69.4**	**69.5**	63.1	**60.9**	65.4	67.3	67.3	59.0	53.4
SEEM (B)	81.7	82.8	83.5	76.0	75.7	**67.6**	69.0	68.7	**64.2**	60.3	**66.4**	**68.6**	**67.7**	**60.5**	53.6
SEEM (L)	**83.4**	**84.6**	**84.1**	**76.5**	**76.9**	66.8	67.8	67.6	62.4	60.1	65.5	66.6	66.3	58.1	54.1

for instance segmentation, and mIoU (mean Intersection over Union) for semantic segmentation. For interactive segmentation, we follow previous works [152, 146] to simulate user clicks by comparing the predicted segmentation with the ground-truth one in an automatic way. After one click on the image to generate the predicted mask, the next click is placed at the center of the area with the largest segmentation error. We use the Number of Clicks (NoC) metric to evaluate interactive segmentation performance, which measures the number of clicks needed to achieve a certain Intersection over Union (IoU), i.e., 85% and 90%, denoted as NoC@85 and NoC@90, respectively. We also vary the number of maximum clicks indicated by K-NoC@90 (K=5, 10, 20), and evaluate the mean IoU on the single click denoted as 1-IoU to study the performance on different constraints. More qualitative evaluation with stroke, scribble, polygon, and box as prompts are illustrated in the supplementary material.

Table 5.3: **Zero-shot** video object segmentation. Without training with video or pairwise image data, our approach is able to do video object segmentation in a zero-shot manner. (#Concurrent work.)

Method	Segmentation Data	Type	Refer-Type	Zero-Shot	Single Image	DAVIS17 JF	DAVIS17 J	DAVIS17 F	DAVIS16-Interactive JF	DAVIS16-Interactive J	DAVIS16-Interactive F	YouTube-VOS 2018 G	YouTube-VOS 2018 Js	YouTube-VOS 2018 Fs	YouTube-VOS 2018 Ju	YouTube-VOS 2018 Fu
With Video Data																
AGSS [142]	VOS+DAVIS (0.1M)	Video	Mask	✗	✗	67.4	64.9	69.9	-	-	-	71.3	71.3	65.5	75.2	73.1
AGAME [96]	(Synth)VOS+DAVIS (0.11M)		Mask	✗	✗	70.0	67.2	72.7	-	-	-	66.0	66.9	*	61.2	*
SWEM [147]	Image+VOS+DAVIS (0.25M)		Mask	✗	✗	84.3	81.2	87.4	-	-	-	82.8	82.4	86.9	77.1	85.0
XMem [38]	Image+VOS+DAVIS (0.25M)		Mask	✗	✗				-	-	-	86.1	85.1	89.8	80.3	89.2
SiamMask [235]	COCO+VOS (0.21M)		Box	✗	✗	*	54.3	58.5	69.8	71.7	67.8	*	60.2	58.2	45.1	47.7
MiVOS [39]	BL30K+VOS+DAVIS (4.88M)		Mask/Scribble	✗	✗	84.5	81.7	87.4	91.0	89.6	92.4	82.6	81.1	85.6	77.7	86.2
ReferFormer-B [245]	RefCOCO(+/g)+VOS+DAVIS (0.13M)		Text	✗	✗	61.1	58.1	64.1	-	-	-	*	*	*	*	*
TAM-L [262]	XMem+SAM (11.2M)	Generalist	Multiple Points	✗	✗	-	-	-	88.4	87.5	89.4	-	-	-	-	-
UNINEXT-T [258]	Image+Video (3M)		Mask	✗	✗	74.5	71.3	77.6	-	-	-	77.0	76.8	81.0	70.8	79.4
UNINEXT-L [258]	Image+Video (3M)		Mask	✗	✗	77.2	73.2	81.2	-	-	-	78.1	79.1	83.5	71.0	78.9
UNINEXT-L [258]	Image+Video (3M)		Text	✗	✗	66.7	62.3	71.1	-	-	-	*	*	*	*	*
Without Video Data																
Painter-L [239]	COCO+ADE+NYUv2 (0.16M)	Generalist	Mask	✓	✗	34.6	28.5	40.8	-	-	-	24.1	27.6	35.8	14.3	18.7
#SegGPT-L [240]	COCO+ADE+VOC+... (0.25M)		Mask	✓	✗	75.6	72.5	78.6	-	-	-	74.7	75.1	80.2	67.4	75.9
#PerSAM-L [290]	SAM+DAVIS (11M)		Mask	✗	✓	60.3	56.6	63.9	-	-	-	*	*	*	*	*
SEEM-T				✓	✓	60.4	57.6	63.3	62.7	58.9	66.4	51.4	55.6	44.1	59.2	46.9
SEEM-B	COCO+LVIS (0.12M)		Mask/Single Point	✓	✓	62.8	59.5	66.2	67.2	63.6	70.9	53.8	60.0	44.5	63.5	47.2
SEEM-L				✓	✓	58.9	55.0	62.8	62.2	58.3	66.0	50.0	57.2	38.2	61.3	43.3

Table 5.4: **Ablation study** on interaction strategy. "#Iter" denotes the maximum training iteration on interactive segmentation in a single forward. "Negative" means adding negative tokens during interactive segmentation. "Scratch" means the model trains from scratch.

Ablation	Fix	#Iter	Pos	Neg	COCO PQ	COCO mAP	COCO mIoU	Referring Segmentation cIoU	Referring Segmentation mIoU	Referring Segmentation AP@50	Pascal VOC NoC50	Pascal VOC NoC90	DAVIS17 JF	DAVIS17 J	DAVIS17 F
Baseline	Y	0	✓	✗	50.7	39.5	60.8	57.9	63.3	71.6	1.74	5.43	59.6	55.8	63.5
- LVIS	✓	2	✓	✓	51.0	39.8	62.2	58.6	63.9	72.6	1.57	4.91	59.5	55.9	63.1
+ Negative	✓	0	✓	✓	50.9	39.8	61.4	58.8	64.0	72.6	1.81	5.41	60.1	56.3	63.9
+ Scratch	✗	3	✓	✓	50.2	39.5	60.7	51.4	59.2	67.0	1.45	4.41	60.6	57.7	63.4
+ Iter	✓	1	✓	✓	50.7	39.7	60.5	58.3	63.4	71.3	1.76	5.14	59.2	55.4	63.0
	✓	2	✓	✓	50.5	39.5	61.0	58.0	63.2	71.6	1.78	5.20	59.6	56.2	63.0
	✓	3	✓	✓	50.4	39.5	61.0	58.0	63.0	71.5	1.55	4.67	59.9	56.4	63.5
	✓	5	✓	✓	50.6	39.4	60.9	58.4	63.4	71.6	1.54	4.59	59.7	56.3	63.1

5.4.1 Main Results

Generic segmentation With one suite of parameters pre-trained on all the segmentation tasks, we are able to evaluate its performance on generic segmentation datasets. As shown in Table 5.1, *SEEM* maintains competitive panoptic, instance, and semantic segmentation performance against strong baselines. Compared with generalist models such as UViM [109], Pix2Seqv2 [26] and especially the recent model Painter [239] and SegGPT [240], our approach significantly outperforms those methods on generic segmentation with a margin around 10 points on panoptic segmentation metrics.

Referring segmentation As shown in Table 5.1, compared with other referring segmentation and generalist models, *SEEM* achieves competitive performance. Notably, by adding a visual compositional prompt, referring segmentation performance is improved with a large margin by 10.5 cIoU, 6.0 mIoU, and 9.3 AP50 points for the tiny model. And this gap is retained for the base and large model. Specifically, this number is computed by class embeddings $\mathbf{O}_h^c$ (Output-Q-Textual). The margin is even larger when computed with mask embeddings $\mathbf{O}_h^m$ (Output-Q-Visual) as shown in Table 5.5. Further, we benchmark the vanilla composition (Ensemble) that directly combines visual and text mask output probabilities as shown in Table 5.5 row 2.

Interactive segmentation As shown in Table 5.1, our approach achieves compa-

rable performance with the specialized models, e.g. RITM, SimpleClick, and better performance than SAM [103] (B) which is trained with $\times 100$ more segmentation data than ours. Notably, unlike existing interactive models, *SEEM* is the first interface that supports not only classical segmentation tasks but also a wide range of user input types, including text, points, scribbles, boxes, and images, providing strong compositional capabilities as shown in Table 5.2,5.5.

Table 5.5: The term 'Text/Visual Prompt' refers to the modality of information utilized in the study. 'Output Query' is indicative of the type of query employed to predict the output. 'Composition Approach' specifies the method through which text and visual information are integrated.

Text	Visual	Output	Composition	Focal-Tiny			Davit-Base			Davit-Large		
Prompt	Prompt	Query	Approach	cIoU	mIoU	AP@50	cIoU	mIoU	AP@50	cIoU	mIoU	AP@50
Y	N	Text	N/A	58.4	63.4	71.6	63.0	68.2	76.7	62.4	67.6	75.3
Y	Y	All	Ensemble	63.0	60.0	66.9	69.3	66.6	74.3	68.9	65.5	72.7
Y	Y	Text	Self-Attn	66.5	69.6	78.8	75.0	76.9	86.3	73.2	76.5	85.9
N	Y	Visual	N/A	70.7	71.8	81.3	75.4	77.8	87.4	**75.2**	78.2	87.7
Y	Y	Visual	Self-Attn	**71.5**	**72.8**	**82.2**	**75.9**	**78.3**	**87.7**	74.9	**78.4**	**87.7**

User input type of interactive segmentation In Table 5.2, we compare 1-IoU of *SEEM* with other strong baselines SimpleClick and SAM with 5 common types of prompts on three datasets. 1-IoU indicates the mean IoU of all images with a single click. The prompt types include point, stroke, scribble, and box. The results show that our *SEEM* achieves the best performance in the extremely limited number of clicks over all three datasets.

Video object segmentation Without any modification, our model is able to do (interactive) video object segmentation in a zero-shot manner through the visual prompt (by replacing the current image visuals prompt with the visual prompts from another image). As shown in Table 5.3, without any observation of DAVIS/VOS dataset [257, 185], our approach is able to achieve close performance in a zero-shot manner with a fully supervised method on DAVIS17 dataset [185]. Meanwhile, our model is able to do interactive video object segmentation on DAVIS16-Interactive [185] and achieves comparable performance with the supervised baselines with one single click of the first frame.

5.4.2 Ablation Study

We conduct an ablation study on all the training segmentation tasks and zero-shot video object segmentation, dissecting each component of our model. The results are presented in Table 5.4.

LVIS mask annotation will improve interactive segmentation results. We replace the COCO mask with an overlap IoU larger than 0.7 with LVIS mask during training. This will improve the performance on interactive segmentation with 0.3 and 0.2 point gain on NoC0.9 and NoC0.85.

Figure 5.4: Click/scribble-based segmentation. *SEEM* supports arbitrary formats of clicks or scribbles by users. Moreover, it simultaneously gives the semantic label for the segmented mask, which is not possible in SAM [103].

Figure 5.5: Text to mask or text referring segmentation. The referred text is shown on the masks. *SEEM* adapts to various types of input images in the domain of cartoons, movies, and games.

Training from scratch only hurts referring segmentation performance. We compare the *SEEM* model trained with X-Decoder pre-trained checkpoint or the checkpoint initialized with UniCL or Florence vision and language backbone (+Scratch). It indicates that training from scratch will slightly improve the performance on interactive segmentation but hurt the referring segmentation performance.

Increase interactive training iterations does help. As shown in Table 5.4, increasing the training iteration (the first N-1 iteration is without gradient) from 1 to 5 will gradually improve the interactive segmentation performance from 5.41 to 4.59 on NoC0.9. As the computation cost increases with more clicks, we use iteration 3 for the main paper results.

5.4.3 Qualitative Results

We further qualitatively evaluate *SEEM*. Based on the proposed prompting scheme and decoder design, with the same suite of parameters, *SEEM* supports a wide range of visual input types.

Visual prompt interactive segmentation. In Fig. 5.4, we show the visualization of using *SEEM* to segment objects in an interactive way. The user can segment objects

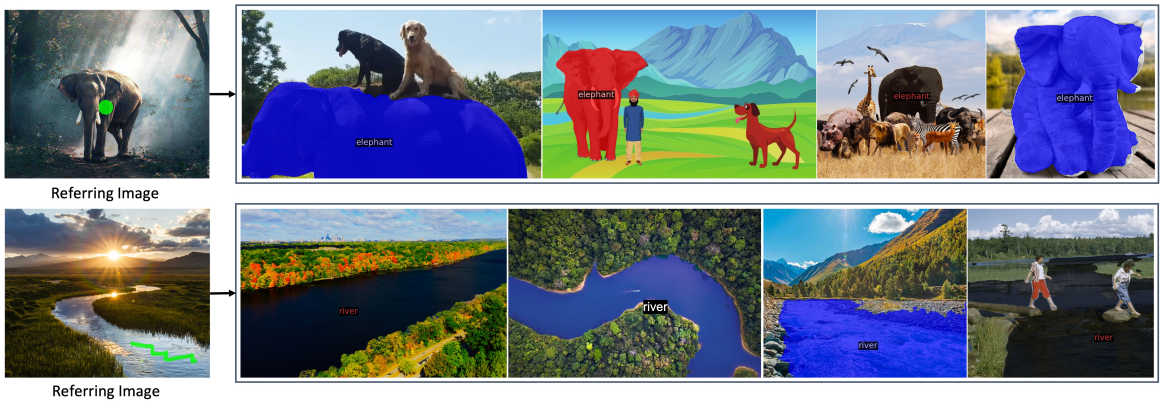


Figure 5.6: **Zero-shot** visual referring segmentation with *SEEM*. Given a referring image with simple spatial hints, *SEEM* can segment the regions which are semantically similar in different target images.

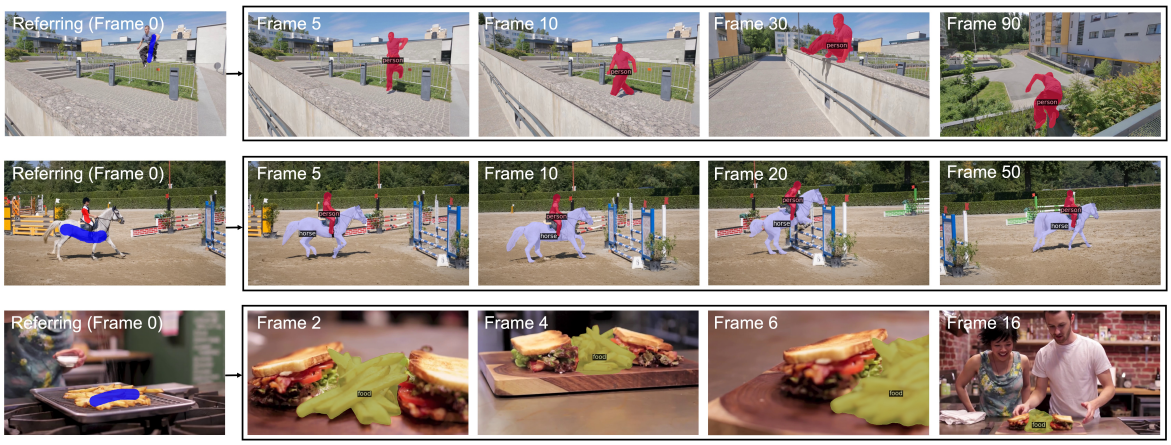


Figure 5.7: **Zero-shot** video object segmentation using the first frame plus one stroke. From top to bottom, the videos are “parkour" and “horsejump-low” from DAVIS [183], and video 101 from YouCook2 [295]. *SEEM* precisely segments referred objects even with significant appearance changes caused by blurring or intensive deformations.

of interest by simply clicking or drawing a scribble. Taking these prompts, *SEEM* can simultaneously produce both masks and semantic labels for the objects. Note that our model is open-vocabulary, which empowers it to label unseen categories when given the candidate vocabulary (i.e., cheetah and butterfly in Fig. 5.4). When no vocabulary is given, *SEEM* can segment in a class-agnostic manner.

Text referring segmentation. We show the text referring to segmentation visualization results in Fig. 5.5. The results demonstrate that our model is semantic-aware of open-vocabulary concepts and attributes to understand language. In addition, *SEEM* is able to generalize to unseen scenarios like cartoons, movies, and games.

Visual referring segmentation. In Fig.5.6, we show *SEEM*’s segmentation results when prompted with referring regions from another image. By simply drawing a click or scribble on one referring image, *SEEM* can take it as input and segment objects with similar semantics on other images. Notably, this referring segmentation has a powerful generalization capability to images of other domains. For example, by referring to the elephant in the forest, another object of the same category can be segmented well under

drastically different scenes like cartoons, plush toys, and grassland.
Video object segmentation. In Fig. 5.7, we further show *SEEM*'s referring segmentation ability on the video object segmentation task in a zero-shot manner. By referring to the objects in the first frame with scribbles, *SEEM* can precisely segment the corresponding objects in the following frames, even when the following objects change in appearance by blurring or intensive deformations.

5.5 Conclusion

We presented *SEEM*, which can segment everything (all semantics) everywhere (all pixels) all at once (all possible prompt compositions). Apart from performing generic open-vocabulary segmentation, *SEEM* can interactively take different types of visual prompts from the user, including click, box, polygon, scribble, text, and referring region from another image. These visual prompts are mapped into a *joint visual-semantic space* with a prompt encoder, which makes our model versatile to various prompts and can flexibly compose different prompts. Extensive experiments indicate that our model yields competitive performance on several open-vocabulary and interactive segmentation benchmarks. Further studies revealed the robust generalization ability of our model in accurately segmenting images based on diverse user intents. We hope our work will serve as a stepping stone toward a universal and interactive interface for image segmentation and beyond.

Chapter 6

Interfacing Foundation Models' Embeddings

Publication Statement. This chapter is joint work with Linjie Li, Jianfeng Wang, Jianwei Yang, Mingyu Ding, Zhengyuan Yang, Feng Li, Hao Zhang, Shilong Liu, Arul Aravinthan, Junyi Wei, Yong Jae Lee, Lijuan Wang. The paper version of this chapter appeared in arXiv 2023 [300].

We present *FIND*, a generalized interface for aligning foundation models' embeddings. As shown in Fig. 6.1, a lightweight transformer interface without tuning any foundation model weights is enough for unified image (segmentation) and dataset-level (retrieval) understanding. The proposed interface has the following favorable attributes: (1) Generalizable. It applies to various tasks spanning retrieval, segmentation, *etc.*, under the same architecture and weights. (2) Prototypable. Different tasks are able to be implemented through prototyping attention masks and embedding types. (3) Extendable. The proposed interface is adaptive to new tasks, and new models. (4) Interleavable. With the benefit of multi-task multi-modal training, the proposed interface creates an interleaved shared embedding space. In light of the interleaved embedding space, we introduce *FIND*-Bench, which introduces new training and evaluation annotations to the COCO dataset for interleaved segmentation and retrieval. Our approach achieves state-of-the-art performance on *FIND*-Bench and competitive performance on standard retrieval and segmentation settings.

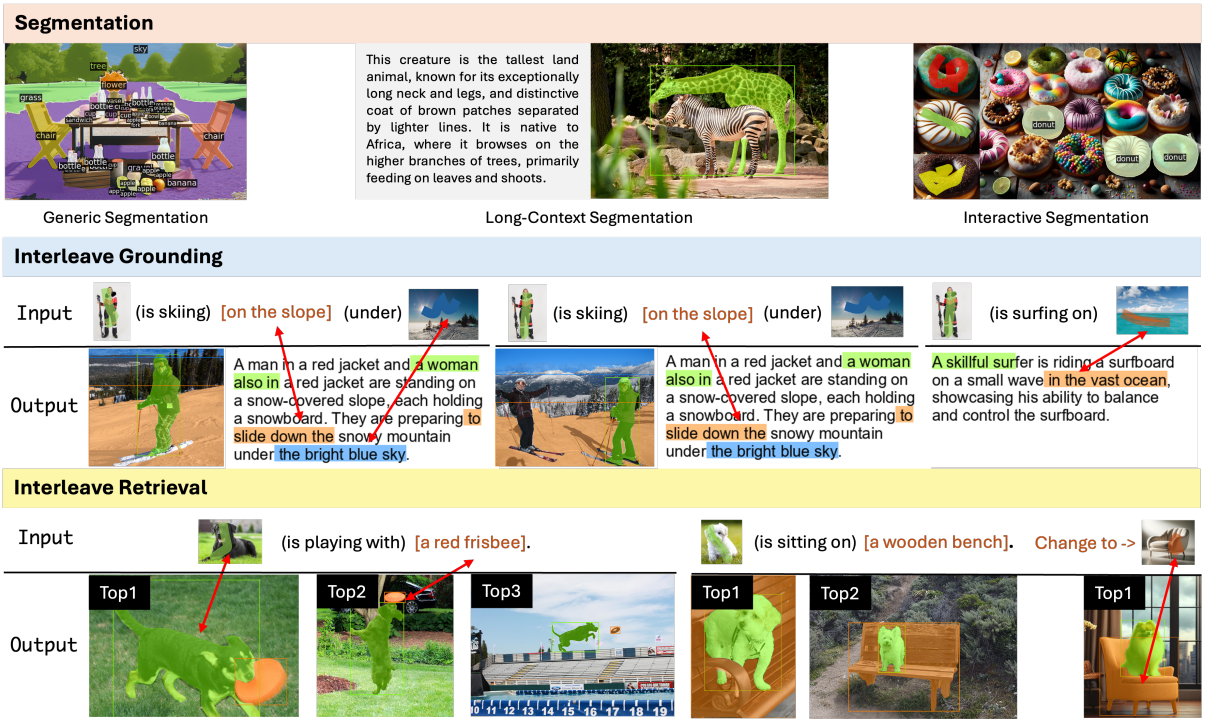


Figure 6.1: The proposed *FIND* interface is generalizable to tasks that span granularity (pixel to image) and modality (vision to language) with an interleaved representation space. Our approach is not only effective for in-domain tasks such as generic segmentation, interactive segmentation, grounded segmentation, etc., but it is also generalizable to downstream zero-shot tasks including cross-image interleave retrieval and text grounding. The search space for the retrieval task here is the COCO validation set.

6.1 Introduction

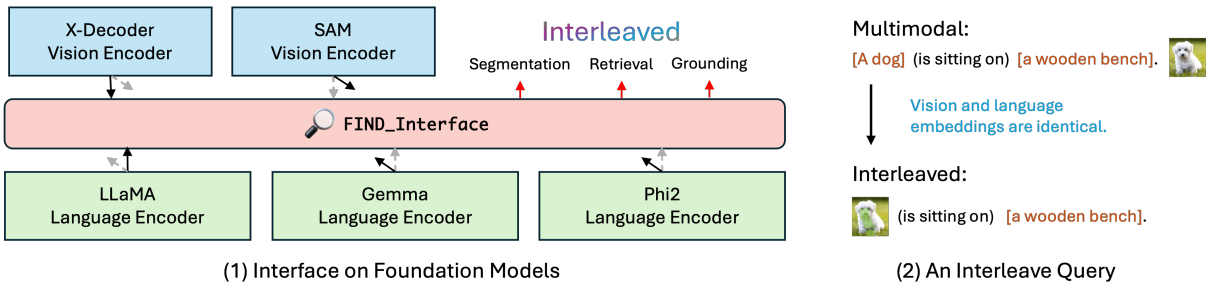


Figure 6.2: (1) The concept of interfacing foundation models embedding, the black arrow means active attached modules and the gray arrow means the option that it can switch to. (2) The concept of Interleaved v.s. Multimodal.

With the exhilarating progress in foundation models across the vision and language domains, such as GPT4(V) [176], DALLE-3 [177], SAM [103], and LLaMA [223], *etc.*, we have reached a stage where deep learning models achieve remarkable performances

on both vision and language domains [14, 122]. Specifically, models like GPT-4(V) [176] have showcased human-level perception and reasoning skills [274].

Recent studies such as LLaVA [149] and BLIP [127] have extensively investigated the use of foundational models for visual Q&A tasks. Both BLIP and LLaVA process features from vision foundational models through a language decoder, treating the visual features purely as text tokens. This approach, however, neglects the potential for foundational model embeddings to interact in a more integrated manner (e.g. interleaved). Additionally, by focusing solely on visual Q&A, these models limit their capability to a specific input-output type, disregarding the inherent flexibility and adaptability of foundational model embeddings.

In this work, we aim to expand the output space (e.g. extend to pixel-level output) of foundational models, unlocking their potential for interleaved understanding and reasoning. To accomplish this, we introduce an INterface for Foundation models' embeDdings (FIND), which utilizes the pre-trained foundational model embeddings to jointly handle downstream tasks of varying granularities (from pixel to image) in an interleaved manner. This interface facilitates task-adaptive prototyping, meaning that only the configuration file needs to be modified - not the model architecture - when adapting to new tasks. As all vision-language tasks are trained uniformly, an interleaved shared embedding space is created where vision and language references can be interchanged and augmented. For instance, as depicted in Fig. 1, we can process queries where text and images jointly describe a scenario. Furthermore, we anticipate that this interface can bridge pre-trained foundational models from various modalities. For example, by interfacing SAM and LLaMA, we can associate SAM features with semantic meanings.

To more effectively align and evaluate the interleaved embedding space, we construct a new dataset named FIND-Bench. This dataset utilizes COCO images and includes new annotations for integrated grounding and segmentation. Interestingly, these annotations are directly generated from GPT-4, which does not take in any visual input. Despite its lack of visual input processing, GPT-4 can directly link specific segments of generated image descriptions with their corresponding COCO annotation IDs. For instance, it can associate the phrase 'the white dog' with the annotation ID of the white dog in the image. This unique capability enables the creation of training and evaluation datasets for retrieval and grounding in an interleaved context.

In summary, we make the following contributions:

- We introduce the *FIND* interface, designed to enhance the potential of foundational model embeddings for more downstream tasks.
- The proposed *FIND* Interface is generalizable, flexible, and extendable to various tasks and foundation models.

- An interleaved shared embedding space is created for foundation models through the unified interface.
- We propose a new Benchmark, *FIND*-Bench, which includes new training and evaluation ground truths for interleave segmentation and retrieval.
- Our model achieves SoTA performance on interleaved image retrieval and segmentation and shows better or comparable performance on generic, interactive, and grounded segmentation and image-text retrieval.

6.2 Related Work

Foundation Models. Recent years have seen a speedy evolution of foundation models in diverse areas such as computer vision [277], natural language processing [226, 46, 13, 176], and their interactions [3, 127, 267]. For example, GPT-3 [13] heralds breakthroughs in natural language understanding and generation tasks, like text completion, translation, summarization, and question answering. As a vision foundation model, Florence [277, 249] can be easily adapted for various computer vision tasks, such as classification, retrieval, object detection, VQA, image captioning, video retrieval, and action recognition. Flamingo [3] bridges powerful pre-trained vision-only and language-only models by token fusion with cross-attention. BLIP-2 [127] proposes an efficient pretraining strategy that bootstraps vision-language pre-training with a lightweight Q-Former in two stages.

Given those different foundation models in various modalities, we believe that LLMs and vision models can be unified in the embedding space. Different from previous multi-modal approaches, such as Flamingo [3], LLaVA [149] and Q-Former (BLIP-2) [127] that feed the vision foundation model output into a language decoder and use the LLM as an interpreter, our goal is to interface foundation model embeddings.

Interleaved Image-Text Understanding. Previous works have explored interleaved visual understanding in the context of visual question answering, visual dialogue, image captioning, and interleaved image retrieval [107, 62, 3]. In addition, recent works [280] explore contextual detection that associates phrases with visual content in a sentence. Although these works reveal interleaved capabilities for image understanding, they lack an evaluation benchmark, as well as a complete training dataset. Though [296, 116, 4] propose a new benchmark on interleaved generation and understanding of image and document level, there is no benchmark available for the interleaved tasks between interactive image parts and phrases.

To this end, we introduce the interleaved segmentation and interleaved retrieval tasks with our carefully designed benchmark *FIND*-Bench. We further propose our approach which uses masked attention to control different tasks, and we prototype the attention

procedure following a protocol with adaptive retrieval, segmentation, and grounding in an interleaved manner. Compared to works that use masked attention [127, 302] to fuse the vision output and LLM input, we care more about interfacing and unifying foundation models' embeddings, i.e., an LLM acts as both the encoder and decoder.

Image Understanding. Vision Transformers [80, 222, 237, 208, 246, 49, 67, 281, 194, 198] have dominated a wide range of key image understanding tasks, such as image retrieval, detection, and segmentation, by applying self-attention to a sequence of image patches. Some multimodal methods [31, 133, 288] have shown good performance for retrieval tasks. However, they are not able to handle pixel-level understanding tasks, like instance segmentation. On the other hand, open-vocabulary segmentation methods have recently drawn much attention, including generic segmentation [23, 302, 48], interactive segmentation [66, 103] that separates objects by actively integrating user inputs, and grounded segmentation [302, 299] that grounds object segments from language descriptions. In this work, we propose *FIND* as a unified interface that can support all the above tasks, while maintaining good performance, and further enabling two new tasks of interleaved segmentation and interleaved retrieval. We unify these tasks by interfacing foundation models' embeddings.

6.3 *FIND* Approach

To bridge the embedding spaces of vision and language foundation models, we develop the *FIND* Interface, which seamlessly assembles multi-modal embeddings at the semantic level. In this section, we motivate and provide details of our *FIND*-Interface. The model pipeline possesses the following valuable properties (see Fig. 6.3.b): (1) It should be able to sample features from the foundation model embeddings for the corresponding task (Embedding Sampler). (2) Those sampled embeddings should be effectively interact with each other for corresponding tasks in a unified manner (*FIND*-Interface). (3) The output embeddings can easily be decoded for the corresponding tasks (Projection and Task Head).

6.3.1 Preliminary

To begin, we first introduce a unified definition (Fig. 6.3.a) for the downstream tasks we explore in this chapter. We believe that segmentation, grounding, and retrieval can all be generalized to a similarity mapping problem of the embeddings, where each embedding is associated with an output (e.g. an image, a sentence, a segmentation mask). Formally, we define:

$$\text{Selected Output} = \arg\max(\mathbf{O} \times \mathbf{O}^T)$$

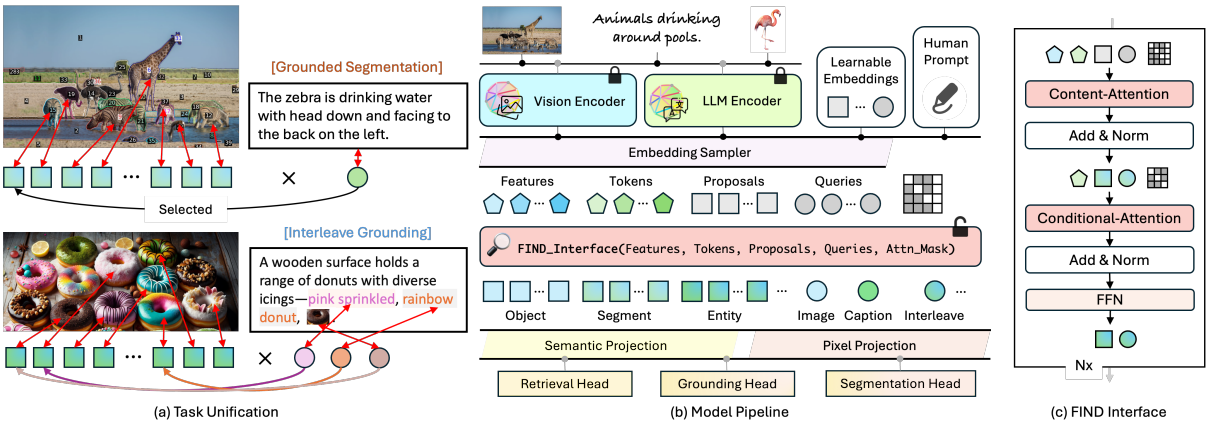


Figure 6.3: (a) Task Unification illustration. (b) *FIND* approach pipeline. The shape of different polygons represents different embedding types, and the color of the polygons represents different input information. (c) Detailed architecture of the *FIND* Interface.

where O is output embedding corresponding to the squares and circles in Fig. 6.3.a The selected output could be the corresponding text sentence for an image query in image retrieval, or selected segment in an image for a grounding sentence, etc.

Having established the task definition, we proceed to define the inputs. For each task supported by *FIND*, we define the inputs as $F = \{f_{.a}, f_{.b}, ..., f_{.z}\}$, $T = \{t_{.a}, t_{.b}, ..., t_{.z}\}$, $P = \{p_{.a}, p_{.b}, ..., p_{.z}\}$, $Q = \{q_{.a}, q_{.b}, ..., q_{.z}\}$, where F, T, P, Q are *features*, *tokens*, *proposals*, and *queries*, respectively. The subscripts $\{a, b, ..., z\}$ denote the embedding used for each task such as grounding proposals $(p.grounding)$, class queries $(q.class)$, etc. as shown in Table 6.1. Each task also contains two attention operations: *content attention* and *conditional attention*. Below, we formally define each embedding type and operation:

• *Features:* Image and text features that are directly predicted by foundation models.

• *Tokens:* Content embeddings that are sampled from the image, and language features using an embedding sampler (e.g. corresponding to a region of an image referred by human click or a portion of a sentence for an entity).

• *Proposals:* Blank embeddings decomposing information from features used as the output proposals, with shape $[n, d]$ where n is the proposal number, and d is the embedding size. (e.g. The squares shown in Fig. 6.3.a.)

• *Queries:* Blank embeddings compress information from features or tokens directly used for output, with shape $[1, d]$ where d is the embedding dimension. (e.g. The circle shown in Fig. 6.3.a.)

– *Content Attention:* Attention operations that allow proposals and queries to gather information from features and tokens given attention mask.

– *Condition Attention:* Attention operations that allow proposals and queries to condition on specific information (e.g. token embeddings) given attention mask.

6.3.2 Pipeline

Our model is designed to interface with a pair of arbitrary vision and language foundation models.

Embeddings Preparation Given the input image I and input text T, the vision encoder **V** and language encoder **L** encode the image and text into features, respectively:

$$f_{\text{.image}} = \mathbf{V}(I), \quad f_{\text{.text}} = \mathbf{L}(T). \tag{6.1}$$

Similar to SEEM [302], we use an embedding sampler to sample customized vision and language tokens from features for a specific task or human prompt given the sampling coordinates $coor$. After sampling, we obtain $t_{\text{.image}}, t_{\text{.text}} = \text{Embedding\_Sampler}(f_{\text{.image}}, f_{\text{.text}}, coor)$. In addition, the embedding sampler is also responsible for sampling learnable queries and proposals from the embedding pool. Given learnable queries $\hat{q}$ (e.g. with shape $[1, 512]$), and proposals $\hat{p}$ (e.g. with shape $[1, 512]$), the task-specific queries and proposals are sampled by the indices idx through $q_{\text{.x}}, p_{\text{.x}} = \text{Embedding\_Sampler}(\hat{q}, \hat{q}, idx)$. The embedding sampler is an interpolation or grid sample layer that samples tokens from features and blank learnable embeddings. We show the specific embeddings that are used for each task in Table 6.1.

***FIND* Interface** After the embeddings are prepared for each task, we define the task-specific attention mask for both content and conditional attention. Given an arbitrary $\text{Input} = \{f_{\cdot a}, ..., f_{\cdot z}, t_{\cdot a}, ..., t_{\cdot z}, p_{\cdot a}, ..., p_{\cdot z}, q_{\cdot a}, ..., q_{\cdot z}\}$, the attention mask is a 2D binary matrix M_{attn} with shape $[len(\text{input}), len(\text{input})]$, where the positive attention region is defined in Table 6.1 in the columns of content attention and condition attention. For example, $q.image : f.image$ means the attention mask is positive in the direction of $q.image \leftarrow f.image$ where image queries can see image features. As shown in Fig. 6.3, the *FIND* interface takes in only task-specific embeddings and attention masks and output queries and proposals for outputs. We illustrate the detailed operation in Sec. 6.3.3.

Projection The outputs of *FIND* interface are a set of queries and proposals: $\text{Output} = \{p_{\cdot a}, ..., p_{\cdot z}, q_{\cdot a}, ..., q_{\cdot z}\}$. We then project the output using linear layers, $\text{L}_{semantic}$ and L_{pixel}, to obtain semantic or pixel outputs: $\text{O}_s = \text{L}_{semantic}(\text{Output})$, $\text{O}_p = \text{L}_{pixel}(\text{Output})$. The semantic outputs are used for retrieval and semantic mappings, while pixel outputs are used for mask prediction.

Task Head With the projected embeddings, each task can be represented as a similarity mapping procedure. The proposal masks ($\text{M}_{proposal}$), proposal similarity scores ($\text{P}^{s,p}_{sim}$), and query similarity scores ($\text{Q}^{s,p}_{sim}$) are defined as:

$$\mathrm{P}_{sim}^{s,p} \cup \mathrm{Q}_{sim}^{s,m} = \mathrm{O}^{\{s,p\}} \times \mathrm{O}^{\{s,p\}} \tag{6.2}$$

$$\mathrm{M}_{proposal} = \mathrm{O}^{p} \times \mathrm{f}_{.image} \tag{6.3}$$

where we abuse the symbol of $\cup$ to note $\mathrm{P}_{sim}^{s,p}$ and $\mathrm{Q}_{sim}^{s,p}$ are computed in the same way. For each task, the output is a combination of $\mathrm{P}_{sim}^{s,p}$, and $\mathrm{Q}_{sim}^{s,p}$, where the similarity scores define the selected proposals in P, or selected queries Q across the dataset. In addition, each proposal is associated with a mask, which gives pixel-level output for $\mathrm{P}_{sim}^{s,m}$.

6.3.3 Interface Operators

The *FIND* Interface contains two main operators which are attention layers. *Content attention* defines the operation that gathers information through a certain search space defined by the attention mask, and *conditional attention* further constrains the search space. Here, when we mention gathering information, we denote the abstract procedure that proposals and queries are attending with corresponding features and tokens through attention layers. After which the proposals and queries are the weighted sum of corresponding features' and tokens' embeddings. We formally define the content and conditional attention as follows, in which the q, k, v are the same for each attention. As shown in Fig. 6.3.b, the interface operators are a stack of modules that contain content and conditional attentions, thus we use $\{\}_{.t}$, and $\{\}_{.t+1}$ to denote the current and next layer variables.

$$\mathrm{T}_t, \mathrm{P}_t, \mathrm{Q}_t = \texttt{Cont\_Attn}([P_t, Q_t, F, T_t], M_a^t) \tag{6.4}$$

$$\mathrm{P}_{t+1}, \mathrm{Q}_{t+1}, \mathrm{T}_{t+1} = \texttt{Cond\_Attn}([P_t, Q_t, T_t], M_a^d) \tag{6.5}$$

where Cont_Attn, and Cond_Attn denote content and conditional attention, while M_a^t, and M_a^d denote attention mask for Cont_Attn and Cond_Attn.

Given the embeddings, pipeline, and operators defined above, we show the prototype for each task in Table 6.1. To be more understandable, we give a case study for interleave segmentation under our prototypable framework.

6.3.4 Case Study: Interleave Segmentation

As shown in Table 6.1, interleave segmentation makes use of both proposals and queries denoted as $p.entity$ and $q.entity$ that are sampled from the learnable embeddings, with shape $[100, 512]$ and $[n, 512]$, where n is the total number of entities queried in the image. In addition, interleave segmentation also uses spatial and interleave tokens denoted by $t.\_spatial$, $t.interleave$.

Table 6.1: Multi-Modal Interface. We define each task under the prototype of the interface that enables a shared embedding space, and a unified and flexible architecture for future tasks. Where p, q, t, f stands for proposals, queries, tokens, memories, and features. The colors red, blue, and yellow mean embeddings of vision, language and interleave modality. "y:x" denotes that in the attention module, the attention mask is visible between y and x in the unidirectional manner which means x is visible to y.

Task	Embeddings			Operators		
	Proposals	Queries	Tokens	Content Attention	Condition Attention	Projection
Generic Segmentation	object	class	class	p.object : f.image q.class : t.class	p:p, q:q, t:t	Pixel, Semantic
Grounded Segmentation	grounding	description	description	p.grounding : f.image q.description : t.description	p:p, q:q, t:t p.grounding : t.description	Pixel, Semantic
Image-Text Retrieval	-	image, caption	caption	q.image : f.image q.caption : t.caption	p:p, q:q, t:t	Semantic
Interactive Segmentation	segment	spatial	spatial	p.segment : f.image q.spatial : t.spatial	p:p, q:q, t:t p.segment : t.spatial	Pixel, Semantic
Interleave Segmentation	entity	entity	interleave, _spatial	p.entity : f.image q.entity : t.interleave t.interleave : t._spatial	p:p, q:q, t:t p.entity : t.interleave	Pixel, Semantic
Interleave Retrieval	-	image, interleave	interleave, _spatial	q.image : f.image q.interleave : t.interleave t.interleave : t._spatial	p:p, q:q, t:t	Semantic

The spatial tokens are sampled from image features $f.image$ with $t.\_spatial = \text{Embedding\_Sampler}(f_{.image}, coor)$, where corr is the spatial mask coordinates. $t.interleave$ is the identity mapping of $f.text$, where the spatial entities are represented with [INTERACTIVE]. We formally define $input = \{q.entity, p.entity, f.image, t.\_spatial, t.interleave\}$. According to the operators' column in Table 6.1, we can represent the attention mask in the following format:

$$M_a^t = \begin{bmatrix} F & \mathbf{T} & F & F & F \\ F & F & F & F & \mathbf{T} \\ F & F & F & F & F \\ F & F & F & F & F \\ F & F & F & \mathbf{T} & F \end{bmatrix} \quad M_a^d = \begin{bmatrix} \mathbf{T} & F & F & \mathbf{T} \\ F & \mathbf{T} & F & F \\ F & F & \mathbf{T} & F \\ F & F & F & \mathbf{T} \end{bmatrix} \tag{6.6}$$

The index of matrix coordinates follows the $input$ order, and there is no $f.image$ involved in M_a^d. The corresponding attention mask for each task is shown in Fig. 6.3.c. We then compute $\text{P}, \text{Q} = \texttt{FIND}(input, M_a^t, M_a^d)$, where $\text{P} = \{p.entity\}$, $\text{Q} = \{q.entity\}$. Then, we project P, Q to the corresponding space with $O_s = \{p_s.entity, q_s.entity\} = L_{semantic}(\{P, Q\})$, and $O_p = \{p_p.entity, q_p.entity\} = L_{pixel}(\{P, Q\})$.

After we get the projected embeddings on semantic and pixel space, we compute the similarity between queries and proposals:

$$P_{sim}^s = p_s.entity \times q_s.entity \tag{6.7}$$

$$M_{proposal} = p_p.entity \times f.image \tag{6.8}$$

where P_{sim}^s is the similarity between proposals and queries on semantic space with shape $[100, n]$, and $M_{proposal}$ is the mask for each proposal. We compute the index for the proposal with $\text{P}_{idx} = argmax(\text{P}_{sim}^s, \text{dim} = 1)$. And the final output mask for interleave segmentation is computed as $\text{M}_{output} = \text{M}_{proposal}[\text{P}_{idx}]$.

(a) COCO-Entity Training (b) COCO-Entity Validation (c) COCO-Paragraph Validation

Figure 6.4: Example ground truth for FIND-Bench. (a) Grounded captions for COCO-Entity training set. (b) Grounded captions together with interleaved visual references for COCO-Entity validation set. (c) Grounded captions for COCO-Paragraph validation set. (The interleaved visual references are omitted due to space limit.)

6.4 *FIND* Bench

In order to evaluate the capability of interleaved visual understanding, we propose a new benchmark called **FIND-Bench**. This includes a fully annotated dataset for both training and evaluation, featuring pairs of images and interleaved entity-based query, as shown in Fig. 6.4. FIND-Bench supports two new tasks, interleaved image retrieval and interleaved grounded segmentation, which we introduce in detail below.

6.4.1 Task Definition

Interleaved Image Retrieval Let I and T denote image and text, respectively. We define a search query as a sequence $Q = \{q_1, q_2, \ldots, q_n\}$, where q_k can be either an instance of I_i or T_j. An example could be "A person with a red shirt is playing with [IMAGE] on the grassland.", where [IMAGE] is an image. The search space, represented as $S = \{I_1, I_2, \ldots, I_m\}$, is an image dataset.

Interleaved Grounded Segmentation Similar to interleaved image retrieval, we again define I as images, S, T as texts, where I, S represents entities (e.g. image reference, "a person"), and T represents text connections. The search query is formulated as a sequence $Q = \{q_1, q_2, \ldots, q_n\}$, $q_i \in \{I, S, T\}$. And the search space is all the segments for an image $\{o_1, o_2, \ldots o_n\}$. The objective is to find the corresponding o_i in the image for each instance of I, S in the query Q.

6.4.2 Dataset

Data Engine We create this new dataset on COCO images with ground truth labels, including captions and panoptic segmentation annotations. In addition, we leverage LLaVA [149] to generate detailed captions as instance-level pseudo-ground truth, and GPT4 as the augmentation engine. We show the pseudo-code of the data engine in Algo. 1, where GPT4 takes in all provided text information to generate entity-associated

Algorithm 2 Pseudo code for Data Engine.

```
# Inputs: llava_cap, coco_cap, coco_pano, lvis_inst; Annotation of single image from multiple
  source.
def data_engine(llava_cap,coco_cap,coco_pano,lvis_inst):
    Content = '''generate image captions with grounded entities and attributes with following:
     pseudo image description: <{}>,
     ground truth image captions: <{}>,
     ground truth bounding boxes (x0,y0,w,h):  (x0,y0) is the coordinate of the top-left corner,
     (w,h) is box size),
     category_info, and entity_id: <{}>,
     entity_proposal:  <{}>,
     an example output format would be:  "(entity_id)<A woman> sitting next to (entity_id)<a
     handsome man>, with their hands holding together under (entity_id)<the blue sky>.",
     where (entity_id) and <xxx> are associated with the ground truth bounding boxes.
     entity_id is the real id number in the ground truth bounding boxes.  generated caption
     constraints: 1. xxx; 2. xxx; '''.
     format(llava_cap, coco_cap, coco_pano, lvis_inst)) ;output = GPT4(content)
```

captions. Finally, we use SEEM to find the most visual-like cross-image instance for the labeled entity in each image. The pseudo-ground truths are verified by humans for quality control.

Statistics As shown in the table below, our proposed dataset contains 118k training images, 5k validation images, together with 300k training captions, and 1000k training entities. Notably, each entity is associated with a bounding box, mask, phrase, or visual reference to support both text and interleaved grounding.

	Training			Evaluation			Entity Association		
	Images	Captions	Entities	Images	Captions	Entities	Mask	Phrase	Visual
COCO-Entity	118189	353219	1104907	4990	4990	15305	✓	✓	✓
COCO-Paragraph	-	-	-	4981	4981	22569	✓	✓	✓

6.5 Experiments

Datasets We use COCO [145] as our main training and evaluation dataset, which spans diverse annotation types. In addition to COCO-panoptic, we make use of the annotations from Ref-COCO [272, 164, 175], COCO-Karpathy [99], and the proposed COCO-Entity dataset. Unless specified otherwise, our model is jointly trained on all tasks listed in Table 6.1.

Settings We benchmark our method on three different sizes: Tiny (FocalNet), Base (Davit-d3), and Large (Davit-d3) models. The vision backbone reuses the X-Decoder pre-trained weights, unless otherwise specified as SAM. The language pre-trained weights are LLaMa, unless specified as UniCL. During training, we fixed the vision and language encoders and only train the FIND-Interface.

Evaluation Metrics We evaluate all the tasks with their standard evaluation metrics. For the newly proposed interleaved image retrieval task, we use IR@5 and IR@10 (Interleave-to-image Retrieval precision at rank 5/10) as the evaluation metrics. For interleaved

Table 6.2: Benchmark on general multi-modal understanding tasks with one model architecture with joint training for all. In the 2nd and 3rd columns, we compare the training dataset for each method, as well as whether the tasks are jointly trained for producing the results. *Unlike X-Decoder and FIND, SEEM is trained with a deformable vision encoder. We report both the ensembled and the decoder retrieval results for X-Decoder (un-ensemble/ensemble), and the finetuned and pre-trained results for blip2 (finetuned/pre-trained). Note that we compute the ITC score for blip2 instead of ITM.

			Generic Segmentation			Grounded Segmentation						Interactive Segmentation			Image-Text Retrieval					
			COCO			RefCOCO-g		COCO-Entity		COCO-Paragraph		Pascal VOC			COCO-Karpathy		COCO-Entity		COCO-Paragraph	
	Data	Joint	PQ	mAP	mIoU	cIoU	mIoU	cIoU	mIoU	cIoU	mIoU	Point	Circle	Box	IR@1	TR@1	IR@1	TR@1	IR@1	TR@1
*Mask2Former (T) [35]	COCO (0.12M)	-	53.2	43.3	63.2		-	-	-	-	-	-	-	-	-	-	-	-	-	-
*Mask2Former (B) [35]	COCO (0.12M)	-	56.4	46.3	67.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
*Mask2Former (L) [35]	COCO (0.12M)	-	**57.8**	48.6	67.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Grounding-SAM (H) [155]	Grounding (5M)	✓	-	-	-	-	-	58.9	57.7	56.1	56.6	-	-	-	-	-	-	-	-	-
SAM (B) [103]	SAM (11M)	-	-	-	-	-	-	-	-	-	-	58.2	-	61.8	-	-	-	-	-	-
SAM (L) [103]	SAM (11M)	-	-	-	-	-	-	-	-	-	-	68.1	-	63.5	-	-	-	-	-	-
*SEEM (T) [302]	COCO+LVIS (0.12M)	✗	50.8	39.7	62.2	60.9	65.7	54.3	56.1	52.6	54.6	83.5	86.0	71.8	-	-	-	-	-	-
*SEEM (B) [302]	COCO+LVIS (0.12M)	✗	56.1	46.4	66.3	65.0	69.6	57.2	58.7	56.1	57.4	87.3	88.8	75.5	-	-	-	-	-	-
*SEEM (L) [302]	COCO+LVIS (0.12M)	✗	57.5	47.7	**67.6**	65.6	70.3	54.8	57.8	53.8	56.7	88.5	**89.6**	76.5	-	-	-	-	-	-
X-Decoder (T) [299]	COCO+ITP (4.12M)	✗	52.6	41.3	62.4	59.8	*	-	-	-	-	-	-	-	40.7 / 49.3	55.0 / 66.7	46.5 / 52.6	48.0 / 55.6	54.8 / 62.3	58.5 / 66.1
X-Decoder (B) [299]	COCO+ITP (4.12M)	✗	56.2	45.8	66.0	64.5	*	-	-	-	-	-	-	-	50.2 / 54.5	66.8 / 71.2	49.2 / 56.9	51.3 / 58.1	58.1 / 67.5	62.5 / 70.1
X-Decoder (L) [299]	COCO+ITP (4.12M)	✗	56.9	46.7	67.5	64.6	*	-	-	-	-	-	-	-	56.4 / 58.6	73.1 / 76.1	58.1 / 60.0	**59.9** / 62.7	58.7 / 71.6	**72.0** / 74.1
CLIP/ImageBind (H) [62, 41]	ITP (400M)	✓	-	-	-	-	-	-	-	-	-	-	-	-	49.4	65.9	53.4	57.6	59.6	64.8
FROMAGe (L) [107]	CC (12M)	✗	-	-	-	-	-	-	-	-	-	-	-	-	27.5	37.8	27.4	33.1	32.8	41.3
BLIP-2 (L) [127]	COCO+IPT (130.1M)	✗	-	-	-	-	-	-	-	-	-	-	-	-	**63.4** / 59.1	**74.4** / 65.2	**59.1** / 58.8	59.8 / 56.4	66.3 / 64.6	65.8 / 60.1
FIND (T)	COCO (0.12M)	✓	51.0	42.3	62.0	61.1	65.3	68.5	62.5	65.0	59.4	84.3	85.8	74.5	40.4	53.0	51.0	51.5	61.2	62.9
FIND (B)	COCO (0.12M)	✓	55.5	49.0	65.7	65.3	69.3	69.5	63.0	**67.2**	60.1	86.3	88.0	75.0	45.8	60.6	56.3	56.7	65.5	69.1
FIND (L)	COCO (0.12M)	✓	56.7	**50.8**	67.4	**65.9**	**70.5**	**69.7**	**64.2**	66.6	**61.2**	**88.5**	89.5	**77.4**	46.3	61.9	57.2	58.2	**67.2**	68.6

Table 6.3: Benchmark on interleaved understanding with the jointly trained model with one set of weights. We conduct solid experiments on baselines approach ImageBind, FROMAGe, and BLIP-2, where we exhaustively try the best settings.

	Interleave Segmentation						Interleave Retrieval				Generic Segmentation								
	COCO-Entity			COCO-Paragraph			COCO-Entity		COCO-Paragraph		Class			Visual Context			Description		
	cIoU	mIoU	AP50	cIoU	mIoU	AP50	IR@5	IR@10	IR@5	TR@5	PQ	mAP	mIoU	PQ	mAP	mIoU	PQ	mAP	mIoU
Mask2Former (L) [35]	-	-	-	-	-	-	-	-	-	-	57.8	48.6	67.4	-	-	-	-	-	-
Grounding-SAM (H) [155]	58.9	57.7	63.2	56.1	56.6	62.5	-	-	-	-	-	-	-	-	-	-	-	-	-
CLIP/ImageBind (H) [62, 41]	-	-	-	-	-	-	51.4	61.3	58.7	68.9	-	-	-	-	-	-	-	-	-
FROMAGe (L) [107]	-	-	-	-	-	-	24.1	34.2	26.0	36.6	-	-	-	-	-	-	-	-	-
BLIP-2 (L) [127]	-	-	-	-	-	-	20.8 / 34.3	25.8 / 47.7	22.1 / 39.3	27.1 / 54.7	-	-	-	-	-	-	-	-	-
X-Decoder (T) [299]	-	-	-	-	-	-	23.6	32.2	25.6	35.5	52.6	41.3	62.4	-	-	-	18.5	15.9	22.5
X-Decoder (B) [299]	-	-	-	-	-	-	26.7	35.8	32.1	42.0	56.2	46.3	67.1	-	-	-	20.8	15.0	24.7
X-Decoder (L) [299]	-	-	-	-	-	-	26.8	36.2	32.2	43.4	57.8	48.6	67.4	-	-	-	23.5	21.1	21.7
SEEM (T) [302]	67.6	67.2	75.8	65.9	65.7	74.4	-	-	-	-	50.8	39.7	62.2	-	-	-	18.6	15.7	16.0
SEEM (B) [302]	69.4	69.2	77.8	69.2	68.6	77.3	-	-	-	-	56.1	46.4	66.3	-	-	-	22.9	21.6	20.0
SEEM (L) [302]	68.3	69.0	77.5	67.7	68.4	77.0	-	-	-	-	**56.9**	46.7	**67.5**	-	-	-	24.0	26.4	18.7
FIND (T)	74.9	68.1	79.5	73.2	66.4	77.7	43.5	57.1	49.4	63.9	51.0	42.3	62.0	41.8	32.3	51.6	19.5	30.2	**35.5**
FIND (B)	76.3	69.7	**81.8**	**75.1**	68.0	79.7	51.4	64.6	60.5	73.4	55.5	49.0	65.7	47.1	36.7	53.6	16.5	26.7	26.7
FIND (L)	**76.3**	**69.7**	81.7	74.7	**68.6**	**79.7**	**53.4**	**66.7**	**62.7**	**75.0**	56.7	**50.8**	67.4	**49.5**	**38.9**	**57.1**	**27.0**	**31.2**	26.8

grounded segmentation, we evaluate based on cIoU (pixel-wise IoU), and mIoU (image-wise IoU).

Baselines We evaluate ImageBind [62], FROMAGe [107], BLIP2 [127] for the interleaved retrieval task, and Grounding-SAM [155] for interleaved (text-only) grounded segmentation on FIND-Bench. We make every effort to design the pipeline to achieve the best possible performance for the baselines.

6.5.1 Main Results

In the main experiments, we evaluate the capability of *FIND* on both general multi-modal settings and interleaved settings.

Comparison on standard multi-modal settings. Table 6.2 compares *FIND* with strong baselines on generic segmentation tasks including panoptic segmentation, instance segmentation, and semantic segmentation. In addition, we demonstrate the segmentation capability in both referring segmentation (RefCOCO-g: one sentence is associated with one instance) and grounded segmentation (COCO-Entity and COCO-Paragraph: one sentence is associated with multiple instances) settings. Moreover, we also benchmark *FIND*'s performance in image-text retrieval on three different ground truth types on

COCO, where the average sentence length for the splits (Karpathy, Entity, and Paragraph) gradually increases. Below are the takeaways:
The instance segmentation result stands out: When compared with models of similar architecture, such as Mask2Former, X-Decoder, and SEEM, our approach with a large vision encoder performs extremely well on instance segmentation. It achieves a performance that is 2.2 points higher than Mask2Fomer (L), which makes use of deformable convolution as well. Note that the segmentation training data is identical between Mask2Former and *FIND*. The performance gain is likely because we have a fully unified segmentation and grounding pipeline so that the semantic ground truth from each domain is mutually beneficial.
Grounded segmentation and referring segmentation are mutually beneficial: In *FIND*, we formulate both grounded segmentation and referring segmentation in a unified way, so that a language description query is attended with language tokens to gather information spanning the language token range. Afterwards, a similarity map is computed between the language description query and segment proposals. The matched proposal is finally used for predicting the mask. As shown in Table 6.2, in addition to state-of-the-art performances on both COCO-Entity and COCO-Paragraph, our model also achieves the best result over strong baselines on Ref-COCOg dataset, including SEEM which is trained with deformable convolution.
Interactive segmentation performance is preserved in the unified settings. Unlike SEEM which is only trained on image-only tasks, *FIND* is trained also on image-text tasks, such as image-text retrieval. With the splitting design of proposals and queries, the training in interactive segmentation and image-text retrieval is independent at the embedding level. Thus, it enables our approach to achieve competitive performances (i.e. *FIND* 88.5/89.5/77.4 vs. SEEM 88.5/89.6/76.5).
The "less fascinating" results on image-text retrieval: The main reason for the sub-optimal solution of our approach on image-text retrieval is caused by the batch size during finetuning. All the models are trained jointly for all the tasks, to create the interleaved shared embedding space between image, object, word, and sentence. Pilot experiments in X-Decoder have shown that training with different resolutions such as 1024 for image and 224 for language does not generalize well across granularities (e.g. 1024x1024 image will perform poorly on image text retrieval, 224x224 image cannot be well segmented). Thus, in *FIND*, we train our model with the same resolution for all tasks. In Table 6.2, all the models are either 384x384 with batch size 384 or 1024x1024 with batch size 192 for all tasks. Note that all other tables show results with the model with 640x640 training resolution and 192 batch size. Our approach achieves competitive performance on COCO-Entity and COCO-Paragraph compared with strong baselines.
Comparisons on interleaved settings. In Table 6.3, we evaluate *FIND* on the interleaved image and pixel level understanding tasks on *FIND*-Bench. Different from COCO-Entity

Table 6.4: Ablation study on different foundation model architectures.

		Generic Segmentation						Grounding	Interactive	Retrieval	
		Class			Description			g-Ref	VOC	COCO-Karpathy	
Vision	Language	PQ	mAP	mIoU	PQ	mAP	mIoU	cIoU	1-IoU	IR@1	TR@1
X-Decoder (T) [299]	UniCL [260]	48.5	39.0	61.4	12.4	20.7	18.9	61.3	82.6	40.4	54.0
X-Decoder (T) [299]	LLaMa [223]	48.5	38.9	61.2	19.5	30.2	35.5	61.6	82.5	40.2	52.2
SAM (B) [103]	UniCL [260]	42.5	37.6	53.6	4.5	17.7	17.9	64.9	81.6	29.1	39.5
SAM (B) [103]	LLaMa [223]	42.5	36.9	53.0	6.1	15.6	16.6	58.9	81.5	27.0	35.5

and COCO-Paragraph in Table 6.2, the entity in the text is randomly replaced with visual content with 0.5 probability.

Interleaved Segmentation: We build an interleaved segmentation baseline using the SEEM model. Instead of formulating the grounding task in a sentence that SEEM doesn't support, we simply separately infer each entity for either interactive or grounding for SEEM. As shown in Table 6.3, *FIND* outperforms SEEM with around +8 points on both COCO-Entity and COCO-Paragraph under cIoU metrics.

Interleaved Retrieval: Apart from interleaved segmentation capability at the image level, we also explore the cross-image interleaved retrieval capability that is a zero-shot task to the current settings. As the interleaved reference objects are also selected from COCO val2017 sets, IR@1 is not meaningful, thus we only report IR@5 and IR@10 results. For ImageBind and BLIP-2, we use the ensemble scores of all the texts, sentences, and images. We follow the original settings of FROMAGe to perform interleaved image-text retrieval. Our performance is substantially higher than the proposed baselines, which again indicates the effectiveness of our interleaved shared embedding space.

Generic Segmentation: In addition to the classic evaluation on generic segmentation with class names or fixed index, here we replace the categories with either class descriptions (a long description without the corresponding class name) or visual prompts (an average feature for the object embeddings in each class). With the benefit of LLMs, *FIND* can perform much better on description-based generic segmentation. This is quite intuitive, that a large language model has a smoother representation when describing the same thing in multiple ways. In addition, an LLM is better at dealing with long context. In addition, we also showcase that *FIND* is usable in the visual context setting.

6.5.2 Ablation Study

We ablate our approach in three perspectives: (1) How well the proposed interface is generalized to different foundation models. (2) What is the effectiveness of each task in the unified pipeline? (3) The effectiveness of using intermediate layers of the LLM representation.

Apply to different foundation model architectures: In the main experiments, we use X-Decoder as the vision encoder, and LLaMA as the language encoder, which shows convening performance on all the benchmarks. X-Decoder has been trained to pair up

Table 6.5: Ablate on each training task and language encoder feature level.

		COCO			g-Ref	Entity	VOC	Karpathy		Entity	
		PQ	mAP	mIoU	cIoU	cIoU	Point	IR@1	TR@1	IR@1	TR@1
Task	All	48.5	39.0	**61.4**	**61.3**	73.0	82.6	**40.4**	**54.0**	**50.8**	**51.9**
	- Retrieval	48.5	39.0	61.1	60.6	**73.2**	**82.8**	-	-	44.3	44.8
	- Grounding	48.6	39.1	61.3	-	40.9	82.8	-	-	45.3	46.2
	- Interactive	48.6	38.8	61.0	-	36.5	-	-	-	31.4	33.4
	- Interleave	**48.9**	**39.3**	61.0	-	-	-	-	-	-	-
Language Level	[-1]	48.3	39.1	61.2	61.3	73.0	82.6	38.9	52.2	50.3	50.8
	[-6]	47.8	38.8	60.4	60.3	72.9	81.3	38.1	49.9	48.1	47.5
	[-12]	**48.5**	**39.0**	**61.4**	61.3	**73.0**	**82.6**	**40.4**	**54.0**	**50.8**	**51.9**
	[-18]	48.2	39.0	61.1	62.2	72.6	82.2	40.1	52.7	50.6	50.5
	[-24]	48.5	38.8	61.5	**61.6**	72.9	82.6	40.2	52.2	50.5	51.3
	[-30]	48.1	39.2	61.1	60.1	73.3	82.4	37.9	49.3	49.4	50.0

vision and language embeddings, however, SAM is only trained on segmentation data without any semantic meaning. Thus, we use SAM as an ablation foundation model, to study how important is vision encoder trained with semantic data. For the language encoder, we adopt UniCL which has the same size as Bert to study the difference between a standard language encoder, and an LLM encoder. As shown in Table 6.4, UniCL and LLaMA usually have very similar performance with X-Decoder as vision encoder, except that LLaMA is extremely effective for language description. Although the performance of SAM is much worse than its counterpart X-Decoder on semantic understanding after aligning the interface, our approach also shows that without any modification to SAM, it applies to semantic understanding tasks on generic, grounded segmentation, and image-text retrieval.

Independent task effectiveness: We explore the independent task effectiveness by gradually removing a task in Table 6.5. Removing image-text retrieval hurts interleave retrieval performance (we actually don't train with cross-image visual prompts) by a great margin. Additionally, further removing the grounding task also decreases the performance of the entity-based grounding task. As interleave grounding is highly related to interactive segmentation, removing interactive segmentation also decreases interleave segmentation performance. Last, when only panoptic segmentation is trained, the performance is very similar to other settings, which indicates that the unified interface is consistent with the procedure of training the basic understanding task.

Varying the feature embeddings layer for LLM: The large language model takes in language tokens as input and outputs the generated text based on the input contents. Thus, it is easy to think that the LLM embeddings would be less semantic near both the input and output layers. However, we believe that the intermediate feature layers would be best aligned with vision embeddings that are highly clustered and semantic. In Table 6.5, we study the performance across generic segmentation, grounding, and image-texture retrieval with features from layer -1 (close to output) to layer -30 (close to input). We can observe that the features at layer -12 have the best performance, and both the top and bottom layers are much worse for image-text retrieval on COCO-Karparthy splits. Throughout the paper, we use -12 layer features for LLaMA.

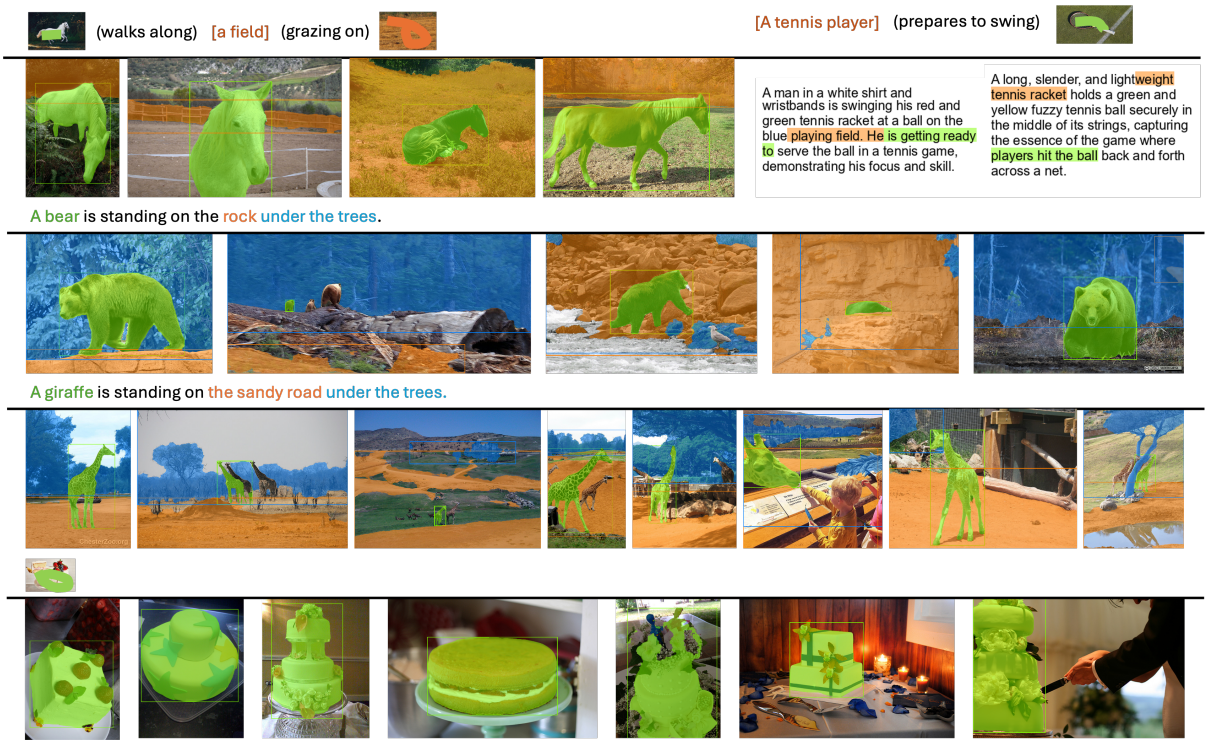


Figure 6.5: Qualitative results on interleave segmentation and retrieval.

6.5.3 Qualitative Results

We qualitatively demonstrate the effectiveness of our approach in Fig. 6.1 and Fig. 6.5. Fig. 6.1 demonstrates that generic segmentation using *FIND* is able to handle complex scenes. Notably, the picnic image for generic segmentation is generated from Dalle-3, which indicates that our approach is working on both in-domain and out-of-domain images. Further, sometimes one may not exactly know the exact word that we want to reference, thus we also show that our model works well with complex descriptions. The model is also able to maintain the interactive segmentation capability.

On the interleaved image retrieval settings, we show that our model is able to find both visually similar and context-relevant images. For example, using a black dog as an example tends to retrieve images that contain black dogs, and the same for white dogs. In addition, the vision and language query is also exchangeable, where changing a bench to a chair image is able to retrieve the result that has the exact matched instance.

In addition, although we never trained with language grounding, our model can generalize to interleaved phrase grounding in that image, and the text can find the exact grounded phrases in the long paragraph.

Conclusions and Future Directions This work proposes the *FIND* Interface, a generalized interface for aligning foundation models' embeddings, and the corresponding *FIND* benchmark for training and evaluation. In the future, we can further extend *FIND* to integrate with novel foundation models, extend to long context, explore more

cross-modal tasks, etc.

Broader Impact. Our proposed approach inherits ethical or social issues (e.g. bias amplification, privacy risks, energy consumption) of foundational models.

Chapter 7

Future Work

Supported by multimodal visual understanding, many future directions could be powered along the direction. The ongoing trend includes integrating LLMs with visual understanding. The potential direction includes visual assist LLM agents, code generation, and robotic applications with better pixel-wise and natural knowledge understanding. In Fig. 7.1, we have summarized the concrete idea associated with each direction.

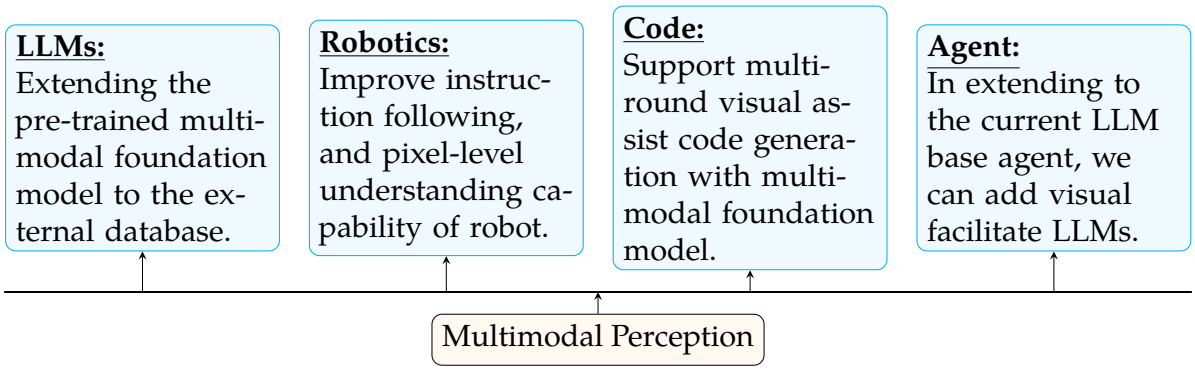


Figure 7.1: Potential directions along LLMs, Robotics, Code, and Agent.

Large Language Models

Large language models (LLMs) have now encompassed multimodal understanding and the capability for multi-round multimodal conversations, as illustrated in our recent work [154, 284]. Furthermore, my study in FIND [216] demonstrates that LLMs can directly operate on unpaired vision foundation models and LLMs. It enables interleave retrieval over the database using LLM features. A current focus in LLM development is in-context learning for integrating external knowledge. An emerging and promising trend is the incorporation of new knowledge into already-trained foundation models, enabling straightforward information updates. While there has been significant research on retrieval-augmented methods, direct expansion of the model's weight presents a more efficient way for enhancing LLMs with additional knowledge.

Robotics

The advancement in robotic technology encompasses significant developments in both hardware and software aspects. As depicted in Fig. 7.2, I give a comprehensive overview of the robotics field. This includes integral components like robot hardware, control systems, and sensors, all of which are crucial for enabling robots to interact effectively with their environment. Deep learning models, nowadays a building block of the control systems, play a pivotal role in bridging sensor APIs with control APIs. These models are involved in various planning stages, such as task planning, which integrates high-level instructions with external knowledge; policy planning, which refines these instructions; and value planning, which prioritizes instructions during execution. Among the emerging technologies, LLM agents or 'robot transformers' are noteworthy for their ability to encapsulate the entire control system. My research on steerable models [284, 154] is particularly significant in this context, as these models facilitate the translation of human instructions into internal robotic functions. Moreover, generalist models [216, 302, 299], capable of processing data from pixels and images to videos, provide robots with a nuanced understanding of their surroundings, enhancing their operational capabilities.

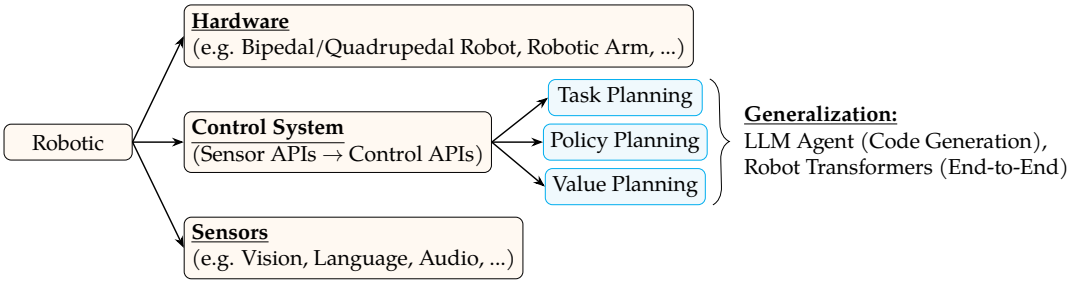


Figure 7.2: Core research area under the scope of robotics.

Code Generation

Beyond large language models, code generation represents another active field of research along the line, brimming with untapped potential. As depicted in Fig. 7.3 (a), previous studies have delved into the corpus of code granularity, ranging from syntax to the use of tools/agents, and modalities encompassing both vision and language. However, current research has yet to fully address two crucial aspects: the ability to perform multi-round, multimodal code generation, and the integration of generated code at the project level. Drawing inspiration from my work on steerable models [284, 154] and my expertise in visual understanding, I propose that multi-round, multimodal conversational abilities could bring new potentials to code generation. Such capabilities would enable more natural and iterative improvements to the codebase, informed by the results of code execution.

Agent

In light of LLMs, people are starting to explore the potential of LLM being the “brain" for various scenarios, especially in the virtual world. Fig. 7.3 illustrates the building blocks of an LLM agent, which typically comprises three key components: Source modality, base model, and application task. For source modalities, currently dominated by language and code is potentially to expand to vision modality. Besides these input sources, the foundational aspect of such models usually relies on pre-trained LLMs, which are tailored to specific modalities through prompting. While recent publications have begun to explore task-specific fine-tuning of these models, there remains a notable gap in research regarding multimodal fine-tuning and understanding. I posit that a multi-modal foundation model could significantly enhance the capabilities of agents in this domain. By integrating diverse modalities, such a model could offer a more robust and versatile framework for agent-based applications, opening new avenues for their deployment and functionality.

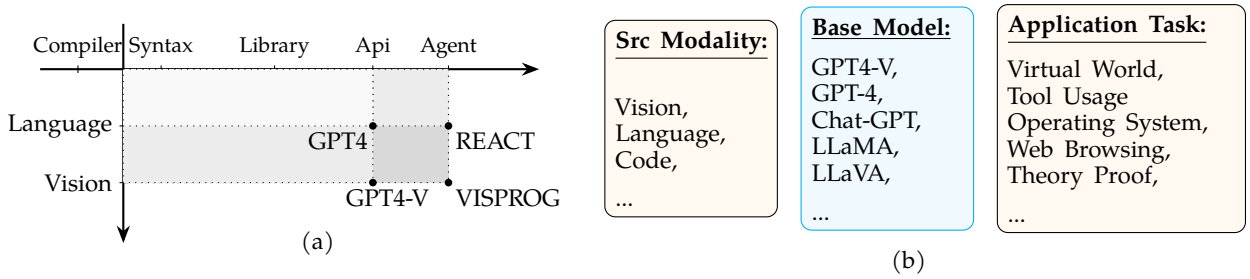


Figure 7.3: (a): The scope of code generation with representative works, with x-axis denoting granularity, and y-axis denoting modality. (b): The overall pipeline of LLM agents.

References

[1] 1994. Self-organization of transformation-invariant neural detectors for constituents which recur within different perceptual patterns. In *Webber, chris js*.

[2] Acuna, David, Huan Ling, Amlan Kar, and Sanja Fidler. 2018. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 859–868.

[3] Alayrac, Jean-Baptiste, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *arXiv preprint*.

[4] An, Jie, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Lijuan Wang, and Jiebo Luo. 2023. Openleaf: Open-domain interleaved image-text generation and evaluation. *arXiv preprint arXiv:2310.07749*.

[5] Anderson, Peter, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 6077–6086.

[6] Antol, Stanislaw, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Iccv*.

[7] Arbelaez, Pablo, Michael Maire, Charless Fowlkes, and Jitendra Malik. 2010. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence* 33(5):898–916.

[8] Azulay, Aharon, and Yair Weiss. 2018. Why do deep convolutional networks generalize so poorly to small image transformations? In *Jmlr*.

[9] Beltagy, Iz, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. In *arxiv:2004.05150*.

[10] Bietti, Alberto, and Julien Mairal. 2017. Invariance and stability of deep convolutional representations. In *Neurips*.

[11] Bloem-Reddy, Benjamin, and Yee Whye Teh. 2020. Probabilistic symmetries and invariant neural networks. In *Jmlr*.

[12] Bolya, Daniel, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. 2019. YOLACT: real-time instance segmentation. In *Iccv*.

[13] Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33:1877–1901.

[14] Bubeck, Sébastien, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

[15] Caelli, Terry M, and Zhi-Qiang Liu. 1988. On the minimum number of templates required for shift, rotation and size invariant pattern recognition. In *Pattern recognition*.

[16] Cai, Zhaowei, Gukyeong Kwon, Avinash Ravichandran, Erhan Bas, Zhuowen Tu, Rahul Bhotika, and Stefano Soatto. 2022. X-detr: A versatile architecture for instance-wise vision-language tasks. *arXiv preprint arXiv:2204.05626*.

[17] Canny, John. 1986. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence* (6):679–698.

[18] Carion, Nicolas, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *Eccv*.

[19] Castrejon, Lluis, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. 2017. Annotating object instances with a polygon-rnn. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 5230–5238.

[20] Chaman, Anadi, and Ivan Dokmanic. 2021. Truly shift-invariant convolutional neural networks. In *Cvpr*.

[21] Chen, Liang-Chieh, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. 2017. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence* 40(4):834–848.

[22] Chen, Liang-Chieh, George Papandreou, Florian Schroff, and Hartwig Adam. 2017. Rethinking atrous convolution for semantic image segmentation. In *Cvpr*.

[23] ———. 2017. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*.

[24] Chen, Liang-Chieh, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Eccv*.

[25] Chen, Ting, Saurabh Saxena, Lala Li, David J Fleet, and Geoffrey Hinton. 2021. Pix2seq: A language modeling framework for object detection. *arXiv preprint arXiv:2109.10852*.

[26] Chen, Ting, Saurabh Saxena, Lala Li, Tsung-Yi Lin, David J Fleet, and Geoffrey Hinton. 2022. A unified sequence interface for vision tasks. *arXiv preprint arXiv:2206.07669*.

[27] Chen, Xi, Zhiyan Zhao, Yilei Zhang, Manni Duan, Donglian Qi, and Hengshuang Zhao. 2022. Focalclick: towards practical interactive image segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 1300–1309.

[28] Chen, Xinlei, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C. Lawrence Zitnick. 2015. Microsoft COCO captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.

[29] Chen, Xinlei, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.

[30] Chen, Yen-Chun, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. UNITER: Universal image-text representation learning. In *Eccv*.

[31] Chen, Yen-Chun, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. UNITER: universal image-text representation learning. In *Eccv*, vol. 12375, 104–120.

[32] Chen, Zhe, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. 2022. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*.

[33] Cheng, Bowen, Ross Girshick, Piotr Dollár, Alexander C Berg, and Alexander Kirillov. 2021. Boundary iou: Improving object-centric image segmentation evaluation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 15334–15342.

[34] Cheng, Bowen, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. 2021. Masked-attention mask transformer for universal image segmentation. *arXiv preprint arXiv:2112.01527*.

[35] ———. 2022. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 1290–1299.

[36] Cheng, Bowen, Omkar Parkhi, and Alexander Kirillov. 2021. Pointly-supervised instance segmentation. *arXiv preprint arXiv:2104.06404*.

[37] Cheng, Bowen, Alexander G Schwing, and Alexander Kirillov. 2021. Per-pixel classification is not all you need for semantic segmentation. *arXiv preprint arXiv:2107.06278*.

[38] Cheng, Ho Kei, and Alexander G. Schwing. 2022. XMem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *Eccv*.

[39] Cheng, Ho Kei, Yu-Wing Tai, and Chi-Keung Tang. 2021. Modular interactive video object segmentation: Interaction-to-mask, propagation and difference-aware fusion. In *Cvpr*.

[40] Cheng, Tianheng, Xinggang Wang, Lichao Huang, and Wenyu Liu. 2020. Boundary-preserving mask r-cnn. In *European conference on computer vision*, 660–676. Springer.

[41] Cherti, Mehdi, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. 2023. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 2818–2829.

[42] Cho, Jaemin, Jie Lei, Hao Tan, and Mohit Bansal. 2021. Unifying vision-and-language tasks via text generation. In *International conference on machine learning*, 1931–1942. PMLR.

[43] Cordts, Marius, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. 2016. The cityscapes dataset for semantic urban scene understanding. In *Cvpr*.

[44] Dai, Jifeng, Kaiming He, and Jian Sun. 2015. Convolutional feature masking for joint object and stuff segmentation. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 3992–4000.

[45] Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Cvpr*.

[46] Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Naacl-hlt (1)*.

[47] Ding, Jian, Nan Xue, Gui-Song Xia, and Dengxin Dai. 2022. Decoupling zero-shot semantic segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 11583–11592.

[48] Ding, Mingyu, Xiaochen Lian, Linjie Yang, Peng Wang, Xiaojie Jin, Zhiwu Lu, and Ping Luo. 2021. Hr-nas: Searching efficient high-resolution neural architectures with lightweight transformers. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 2982–2992.

[49] Ding, Mingyu, Bin Xiao, Noel Codella, Ping Luo, Jingdong Wang, and Lu Yuan. 2022. Davit: Dual attention vision transformers. *arXiv preprint arXiv:2204.03645*.

[50] Ding, Zheng, Jieke Wang, and Zhuowen Tu. 2022. Open-vocabulary panoptic segmentation with maskclip. *arXiv preprint arXiv:2208.08984*.

[51] Dong, Bin, Fangao Zeng, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. 2021. Solq: Segmenting objects by learning queries. *Advances in Neural Information Processing Systems* 34:21898–21909.

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

[53] Dou, Zi-Yi, Aishwarya Kamath, Zhe Gan, Pengchuan Zhang, Jianfeng Wang, Linjie Li, Zicheng Liu, Ce Liu, Yann LeCun, Nanyun Peng, Jianfeng Gao, and Lijuan Wang. 2022. Coarse-to-fine vision-language pre-training with fusion in the backbone. In *Neurips*.

[54] Dou, Zi-Yi, Yichong Xu, Zhe Gan, Jianfeng Wang, Shuohang Wang, Lijuan Wang, Chenguang Zhu, Pengchuan Zhang, Lu Yuan, Nanyun Peng, Zicheng Liu, and Michael Zeng. 2022. An empirical study of training end-to-end vision-and-language transformers. In *Cvpr*.

[55] Everingham, Mark, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2015. The pascal visual object classes challenge: A retrospective. In *Ijcv*.

[56] Felzenszwalb, Pedro F, Ross B Girshick, David McAllester, and Deva Ramanan. 2009. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence* 32(9):1627–1645.

[57] Fu, King-Sun, and JK Mui. 1981. A survey on image segmentation. *Pattern recognition* 13(1):3–16.

[58] Gan, Zhe, Linjie Li, Chunyuan Li, Lijuan Wang, Zicheng Liu, and Jianfeng Gao. 2022. Vision-language pre-training: Basics, recent advances, and future trends. *arXiv preprint arXiv:2210.09263*.

[59] Gao, Peng, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. 2021. Fast convergence of detr with spatially modulated co-attention. *arXiv preprint arXiv:2101.07448*.

[60] Geirhos, Robert, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. 2019. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *Iclr*.

[61] Ghiasi, Golnaz, Xiuye Gu, Yin Cui, and Tsung-Yi Lin. 2021. Open-vocabulary image segmentation. *arXiv preprint arXiv:2112.12143*.

[62] Girdhar, Rohit, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. 2023. Imagebind: One embedding space to bind them all. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 15180–15190.

[63] Girshick, Ross. 2015. Fast r-cnn. In *Proceedings of the ieee international conference on computer vision*, 1440–1448.

[64] Girshick, Ross, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 580–587.

[65] Gonzales, Rafael C, and Richard E Woods. 2002. Digital image processing. Prentice hall New Jersey.

[66] Grady, Leo. 2006. Random walks for image segmentation. *IEEE transactions on pattern analysis and machine intelligence* 28(11):1768–1783.

[67] Graham, Benjamin, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. 2021. Levit: a vision transformer in convnet's clothing for faster inference. In *Iccv*, 12259–12269.

[68] Gu, Xiuye, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. 2021. Open-vocabulary object detection via vision and language knowledge distillation. *arXiv preprint arXiv:2104.13921*.

[69] Gu, Zhihao. 2021. Spatiotemporal inconsistency learning for deepfake video detection. In *arxiv:2109.01860*.

[70] Gupta, Agrim, Piotr Dollar, and Ross Girshick. 2019. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 5356–5364.

[71] Gupta, Tanmay, Amita Kamath, Aniruddha Kembhavi, and Derek Hoiem. 2022. Towards general purpose vision systems: An end-to-end task-agnostic vision-language architecture. In *Cvpr*.

[72] Hafiz, Abdul Mueed, and Ghulam Mohiuddin Bhat. 2020. A survey on instance segmentation: state of the art. *International journal of multimedia information retrieval* 9(3):171–189.

[73] Hariharan, Bharath, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. 2014. Simultaneous detection and segmentation. In *European conference on computer vision*, 297–312. Springer.

[74] ———. 2015. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 447–456.

[75] He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Iccv*.

[76] He, Kaiming, Jian Sun, and Xiaoou Tang. 2010. Guided image filtering. In *Eccv*.

[77] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Iccv*.

[78] ———. 2015. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence* 37(9): 1904–1916.

[79] ———. 2016. Deep residual learning for image recognition. In *Cvpr*.

[80] Heo, Byeongho, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. 2021. Rethinking spatial dimensions of vision transformers. In *Iccv*, 11936–11945.

[81] Hu, Jie, Liujuan Cao, Yao Lu, ShengChuan Zhang, Yan Wang, Ke Li, Feiyue Huang, Ling Shao, and Rongrong Ji. 2021. Istr: End-to-end instance segmentation with transformers. *arXiv preprint arXiv:2105.00637*.

[82] Hu, Ping, Fabian Caba Heilbron, Oliver Wang, Zhe Lin, Stan Sclaroff, and Federico Perazzi. 2020. Temporally distributed networks for fast video semantic segmentation. In *Cvpr*.

[83] Hu, Ping, Federico Perazzi, Fabian Caba Heilbron, Oliver Wang, Zhe Lin, Kate Saenko, and Stan Sclaroff. 2020. Real-time semantic segmentation with fast attention. In *Eccv workshop*.

[84] Hu, Ronghang, Marcus Rohrbach, and Trevor Darrell. 2016. Segmentation from natural language expressions. In *European conference on computer vision*, 108–124. Springer.

[85] Hu, Ronghang, and Amanpreet Singh. 2021. Unit: Multimodal multitask learning with a unified transformer. In *Proceedings of the ieee/cvf international conference on computer vision*, 1439–1449.

[86] Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Cvpr*.

[87] Huang, Shijia, Feng Li, Hao Zhang, Shilong Liu, Lei Zhang, and Liwei Wang. 2022. A unified mutual supervision framework for referring expression segmentation and generation. *arXiv preprint arXiv:2211.07919*.

[88] Huang, Zeyi, Haohan Wang, Eric P Xing, and Dong Huang. 2020. Self-challenging improves cross-domain generalization. In *Eccv*.

[89] Huang, Zhaojin, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. 2019. Mask scoring r-cnn. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 6409–6418.

[90] Huynh, Dat, Jason Kuen, Zhe Lin, Jiuxiang Gu, and Ehsan Elhamifar. 2022. Open-vocabulary instance segmentation via robust cross-modal pseudo-labeling. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 7020–7031.

[91] Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *Cvpr*.

[92] Jain, Jitesh, Jiachen Li, MangTik Chiu, Ali Hassani, Nikita Orlov, and Humphrey Shi. 2022. Oneformer: One transformer to rule universal image segmentation. *arXiv preprint arXiv:2211.06220*.

[93] Jia, Chao, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *Icml*.

[94] Jia, Xu, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. 2016. Dynamic filter networks. In *Neurips*.

[95] Jiang, Yifan, S Chang, and Z Wang. 2021. Transgan: Two pure transformers can make one strong gan, and that can scale up. CVPR.

[96] Johnander, Joakim, Martin Danelljan, Emil Brissman, Fahad Shahbaz Khan, and Michael Felsberg. 2018. A generative appearance model for end-to-end video object segmentation. 1811.11611.

[97] Kamath, Aishwarya, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. 2021. Mdetr-modulated detection for end-to-end multi-modal understanding. In *Proceedings of the ieee/cvf international conference on computer vision*, 1780–1790.

[98] Kannan, Harini, Alexey Kurakin, and Ian Goodfellow. 2018. Adversarial logit pairing. In *arxiv:1803.06373*.

[99] Karpathy, Andrej, and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 3128–3137.

[100] Karras, Tero, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2021. Alias-free generative adversarial networks. In *arxiv:2106.12423*.

[101] Kim, Wonjae, Bokyung Son, and Ildoo Kim. 2021. ViLT: Vision-and-language transformer without convolution or region supervision. In *Icml*.

[102] Kirillov, Alexander, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. 2019. Panoptic segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 9404–9413.

[103] Kirillov, Alexander, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. 2023. Segment anything. 2304.02643.

[104] Kirillov, Alexander, Yuxin Wu, Kaiming He, and Ross Girshick. 2020. Pointrend: Image segmentation as rendering. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 9799–9808.

[105] Kitaev, Nikita, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.

[106] Kittler, Josef. 1983. On the accuracy of the sobel edge detector. *Image and Vision Computing* 1(1):37–42.

[107] Koh, Jing Yu, Ruslan Salakhutdinov, and Daniel Fried. 2023. Grounding language models to images for multimodal inputs and outputs.

[108] Kojima, Takeshi, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

[109] Kolesnikov, Alexander, André Susano Pinto, Lucas Beyer, Xiaohua Zhai, Jeremiah Harmsen, and Neil Houlsby. 2022. UViM: A unified modeling approach for vision with learned guiding codes. *arXiv preprint arXiv:2205.10337*.

[110] Konishi, Scott, Alan L. Yuille, James M. Coughlan, and Song Chun Zhu. 2003. Statistical edge detection: Learning and evaluating edge cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25(1):57–74.

[111] Krishna, Ranjay, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision* 123(1):32–73.

[112] Krizhevsky, Alex, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images.

[113] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton. 2017. Imagenet classification with deep convolutional neural networks. *Communications of the ACM* 60(6):84–90.

[114] Kurakin, Alexey, Ian Goodfellow, and Samy Bengio. 2017. Adversarial examples in the physical world. In *Iclr workshop*.

[115] Lambert, John, Zhuang Liu, Ozan Sener, James Hays, and Vladlen Koltun. 2020. Mseg: A composite dataset for multi-domain semantic segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 2879–2888.

[116] Laurençon, Hugo, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander M Rush, Douwe Kiela, et al. 2023. Obelics: An open web-scale filtered dataset of interleaved image-text documents. In *Thirty-seventh conference on neural information processing systems datasets and benchmarks track*.

[117] Law, Hei, and Jia Deng. 2018. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the european conference on computer vision (eccv)*, 734–750.

[118] Lee, Juho, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. 2019. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Icml*.

[119] Lee, Kimin, Honglak Lee, Kibok Lee, and Jinwoo Shin. 2017. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *Iclr*.

[120] Lee, Kimin, Kibok Lee, Honglak Lee, and Jinwoo Shin. 2018. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Neurips*.

[121] Li, Boyi, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and René Ranftl. 2022. Language-driven semantic segmentation. *arXiv preprint arXiv:2201.03546*.

[122] Li, Chunyuan, Zhe Gan, Zhengyuan Yang, Jianwei Yang, Linjie Li, Lijuan Wang, and Jianfeng Gao. 2023. Multimodal foundation models: From specialists to general-purpose assistants. *arXiv preprint arXiv:2309.10020* 1:2.

[123] Li, Da, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. 2017. Deeper, broader and artier domain generalization. In *Iccv*.

[124] Li, Feng, Qing Jiang, Hao Zhang, Tianhe Ren, Shilong Liu, **Zou, Xueyan**, Huaizhe Xu, Hongyang Li, Chunyuan Li, Jianwei Yang, Lei Zhang, and Jianfeng Gao. 2023. Visual in-context prompting. *arXiv*.

[125] Li, Feng, Hao Zhang, Shilong Liu, Lei Zhang, Lionel M Ni, Heung-Yeung Shum, et al. 2022. Mask dino: Towards a unified transformer-based framework for object detection and segmentation. *arXiv preprint arXiv:2206.02777*.

[126] Li, Feng, Hao Zhang, Peize Sun, Xueyan Zou, Shilong Liu, Jianwei Yang, Chunyuan Li, Lei Zhang, and Jianfeng Gao. 2023. Semantic-sam: Segment and recognize anything at any granularity. *arXiv preprint arXiv:2307.04767*.

[127] Li, Junnan, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*.

[128] Li, Junnan, Ramprasaath R Selvaraju, Akhilesh Deepak Gotmare, Shafiq Joty, Caiming Xiong, and Steven Hoi. 2021. Align before fuse: Vision and language representation learning with momentum distillation. In *Neurips*.

[129] Li, Liunian Harold, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. VisualBERT: A simple and performant baseline for vision and language. *arXiv preprint*.

[130] Li, Liunian Harold, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. 2022. Grounded language-image pre-training. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 10965–10975.

[131] Li, Songnan, Lin Ma, Fan Zhang, and King Ngi Ngan. 2010. Temporal inconsistency measure for video quality assessment. In *28th picture coding symposium*.

[132] Li, Xiang Lisa, and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

[133] Li, Xiujun, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *ECCV*, 121–137.

[134] Li, Xiujun, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Eccv*.

[135] Li, Yajun. 1992. Reforming the theory of invariant moments for pattern recognition. In *Pattern recognition*.

[136] Li, Yanwei, Hengshuang Zhao, Xiaojuan Qi, Yukang Chen, Lu Qi, Liwei Wang, Zeming Li, Jian Sun, and Jiaya Jia. 2021. Fully convolutional networks for panoptic segmentation with point-based supervision. *arXiv preprint arXiv:2108.07682*.

[137] Li, Yin, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. 2004. Lazy snapping. *ACM Transactions on Graphics (ToG)* 23(3):303–308.

[138] Li, Yuheng, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. 2023. Gligen: Open-set grounded text-to-image generation. *arXiv preprint arXiv:2301.07093*.

[139] Li, Zhiqi, Wenhai Wang, Enze Xie, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, Ping Luo, and Tong Lu. 2022. Panoptic segformer: Delving deeper into panoptic segmentation with transformers. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 1280–1289.

[140] Liang, Justin, Namdar Homayounfar, Wei-Chiu Ma, Yuwen Xiong, Rui Hu, and Raquel Urtasun. 2020. Polytransform: Deep polygon transformer for instance segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 9131–9140.

[141] Liao, Fangzhou, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. 2018. Defense against adversarial attacks using high-level representation guided denoiser. In *Cvpr*.

[142] Lin, Huaijia, Xiaojuan Qi, and Jiaya Jia. 2019. Agss-vos: Attention guided single-shot video object segmentation. In *Iccv*.

[143] Lin, Tsung-Yi, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. 2017. Feature pyramid networks for object detection. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 2117–2125.

[144] Lin, Tsung-Yi, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the ieee international conference on computer vision*, 2980–2988.

[145] Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Eccv*.

[146] Lin, Zheng, Zheng-Peng Duan, Zhao Zhang, Chun-Le Guo, and Ming-Ming Cheng. 2022. Focuscut: Diving into a focus view in interactive segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 2637–2646.

[147] Lin, Zhihui, Tianyu Yang, Maomao Li, Ziyu Wang, Chun Yuan, Wenhao Jiang, and Wei Liu. 2022. Swem: Towards real-time video object segmentation with sequential weighted expectation-maximization. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 1362–1372.

[148] Liu, Chenxi, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, and Alan Yuille. 2017. Recurrent multimodal interaction for referring image segmentation. In *Proceedings of the ieee international conference on computer vision*, 1271–1280.

[149] Liu, Haotian, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*.

[150] Liu, Jiang, Hui Ding, Zhaowei Cai, Yuting Zhang, Ravi Kumar Satzoda, Vijay Mahadevan, and R Manmatha. 2023. Polyformer: Referring image segmentation as sequential polygon generation.

[151] Liu, Liyuan, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han. 2020. Understanding the difficulty of training transformers. *arXiv preprint arXiv:2004.08249*.

[152] Liu, Qin, Zhenlin Xu, Gedas Bertasius, and Marc Niethammer. 2022. Simpleclick: Interactive image segmentation with simple vision transformers. *arXiv preprint arXiv:2210.11006*.

[153] Liu, Qin, Meng Zheng, Benjamin Planche, Srikrishna Karanam, Terrence Chen, Marc Niethammer, and Ziyan Wu. 2022. Pseudoclick: Interactive image segmentation with click imitation. In *Computer vision–eccv 2022: 17th european conference, tel aviv, israel, october 23–27, 2022, proceedings, part vi*, 728–745. Springer.

[154] Liu, Shilong, Hao Cheng, Haotian Liu, Hao Zhang, Feng Li, Tianhe Ren, **Zou, Xueyan**, Jianwei Yang, Hang Su, Jun Zhu, and Chunyuan Li Lei Zhang, Jianfeng Gao. 2023. Llava-plus: Learning to use tools for creating multimodal agents. *arXiv*.

[155] Liu, Shilong, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. 2023. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*.

[156] Liu, Yun, Ming-Ming Cheng, Xiaowei Hu, Kai Wang, and Xiang Bai. 2017. Richer convolutional features for edge detection. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 3000–3009.

[157] Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Iccv*.

[158] Long, Jonathan, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *Cvpr*.

[159] Lu, Jiasen, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Neurips*.

[160] Lu, Jiasen, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. 2022. Unified-io: A unified model for vision, language, and multi-modal tasks. *arXiv preprint arXiv:2206.08916*.

[161] Lüddecke, Timo, and Alexander Ecker. 2022. Image segmentation using text and image prompts. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 7086–7096.

[162] Madry, Aleksander, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *Iclr*.

[163] Mairal, Julien, Piotr Koniusz, Zaid Harchaoui, and Cordelia Schmid. 2014. Convolutional kernel networks. In *Neurips*.

[164] Mao, Junhua, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. 2016. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 11–20.

[165] Margffoy-Tuay, Edgar, Juan C Pérez, Emilio Botero, and Pablo Arbeláez. 2018. Dynamic multimodal instance segmentation guided by natural language queries. In *Proceedings of the european conference on computer vision (eccv)*, 630–645.

[166] Marr, David, and Ellen Hildreth. 1980. Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences* 207(1167):187–217.

[167] Martin, David R, Charless C Fowlkes, and Jitendra Malik. 2004. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE transactions on pattern analysis and machine intelligence* 26(5):530–549.

[168] Massa, Francisco, and Ross Girshick. 2018. maskrcnn-benchmark: Fast, modular reference implementation of Instance Segmentation and Object Detection algorithms in PyTorch. https://github.com/facebookresearch/maskrcnn-benchmark. Accessed: [Oct.10 2019].

[169] Meng, Depu, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. 2021. Conditional detr for fast training convergence. In *Proceedings of the ieee/cvf international conference on computer vision*, 3651–3660.

[170] Milletari, Fausto, Nassir Navab, and Seyed-Ahmad Ahmadi. 2016. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3d vision (3dv)*, 565–571. IEEE.

[171] Minaee, Shervin, Yuri Y Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. 2021. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*.

[172] Minderer, Matthias, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, et al. 2022. Simple open-vocabulary object detection with vision transformers. *arXiv preprint arXiv:2205.06230*.

[173] Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. In *Nature*.

[174] Muandet, Krikamol, David Balduzzi, and Bernhard Schölkopf. 2013. Domain generalization via invariant feature representation. In *Icml*.

[175] Nagaraja, Varun K, Vlad I Morariu, and Larry S Davis. 2016. Modeling context between objects for referring expression understanding. In *European conference on computer vision*, 792–807. Springer.

[176] OpenAI. 2023. Gpt-4 technical report. 2303.08774.

[177] ———. 2023. Improving image generation with better captions. Tech. Rep., OpenAI.

[178] Ordonez, Vicente, Girish Kulkarni, and Tamara Berg. 2011. Im2text: Describing images using 1 million captioned photographs. *Advances in neural information processing systems* 24.

[179] Pal, Nikhil R, and Sankar K Pal. 1993. A review on image segmentation techniques. *Pattern recognition* 26(9):1277–1294.

[180] Paris, Sylvain, Pierre Kornprobst, Jack Tumblin, Frédo Durand, et al. 2009. Bilateral filtering: Theory and applications.

[181] Park, Taesung, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. 2019. Semantic image synthesis with spatially-adaptive normalization. In *Cvpr*.

[182] Peng, Sida, Wen Jiang, Huaijin Pi, Xiuli Li, Hujun Bao, and Xiaowei Zhou. 2020. Deep snake for real-time instance segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 8533–8542.

[183] Perazzi, Federico, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. 2016. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 724–732.

[184] Plummer, Bryan A, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the ieee international conference on computer vision*, 2641–2649.

[185] Pont-Tuset, Jordi, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. 2017. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*.

[186] Proakis, John G, and Dimitris G Manolakis. 1992. Digital signal processing. In *Mpc, new york*.

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

[188] Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research* 21(1):5485–5551.

[189] Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Emnlp*.

[190] Rao, Yongming, Wenliang Zhao, Guangyi Chen, Yansong Tang, Zheng Zhu, Guan Huang, Jie Zhou, and Jiwen Lu. 2022. Denseclip: Language-guided dense prediction with context-aware prompting. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 18082–18091.

[191] Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems* 28:91–99.

[192] Rezatofighi, Hamid, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. 2019. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 658–666.

[193] Richardson, Elad, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. 2021. Encoding in style: a stylegan encoder for image-to-image translation. In *Cvpr*.

[194] Riquelme, Carlos, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. 2021. Scaling vision with sparse mixture of experts. *NeurIPS* 34.

[195] Rombach, Robin, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models.

In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 10684–10695.

[196] Rosenberg, D. 1974. Box filter. US Patent 3,815,754.

[197] Rowley, Henry A, Shumeet Baluja, and Takeo Kanade. 1998. Rotation invariant neural network-based face detection. In *Cvpr*.

[198] Ryoo, Michael S., AJ Piergiovanni, Anurag Arnab, Mostafa Dehghani, and Anelia Angelova. 2021. Tokenlearner: What can 8 learned tokens do for images and videos? *arXiv: Computer Vision and Pattern Recognition*.

[199] Schick, Timo, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*.

[200] Shankar, Vaishaal, Achal Dave, Rebecca Roelofs, Deva Ramanan, Benjamin Recht, and Ludwig Schmidt. 2019. A systematic framework for natural perturbations from videos. *arXiv:1906.02168*.

[201] Shannon, Claude Elwood. 1949. Communication in the presence of noise. In *Proceedings of the ire*.

[202] Sharma, Piyush, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: Long papers)*, 2556–2565.

[203] Shi, Jianbo, and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence* 22(8):888–905.

[204] Shin, Taylor, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

[205] Simonyan, Karen, and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. *ICLR*.

[206] Singh, Amanpreet, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. 2022. Flava: A foundational language and vision alignment model. In *Cvpr*.

[207] Sofiiuk, Konstantin, Ilia A. Petrov, and Anton Konushin. 2021. Reviving iterative training with mask guidance for interactive segmentation. 2102.06583.

[208] Srinivas, Aravind, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. 2021. Bottleneck transformers for visual recognition. 16519–16529.

[209] Strudel, Robin, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. 2021. Segmenter: Transformer for semantic segmentation. *arXiv preprint arXiv:2105.05633*.

[210] Su, Hang, Varun Jampani, Deqing Sun, Orazio Gallo, Erik Learned-Miller, and Jan Kautz. 2019. Pixel-adaptive convolutional neural networks. In *Cvpr*.

[211] Su, Weijie, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2019. VL-BERT: Pre-training of generic visual-linguistic representations. In *Iclr*.

[212] Szegedy, Christian, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv:1312.6199*.

[213] Tan, Hao, and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Emnlp*.

[214] Tan, Mingxing, and Quoc V Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Icml*.

[215] Tenney, Ian, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*.

[216] **Zou, Xueyan**, Linjie Li, Jianfeng Wang, Jianwei Yang, Mingyu Ding, Zhengyuan Yang, Feng Li, Hao Zhang, Shilong Liu, Arul Aravinthan, Yong Jae Lee, and Lijuan Wang. 2023. Interfacing foundation models' embeddings. *arXiv*.

[217] **Zou, Xueyan**, Haotian Liu, and Yong Jae Lee. 2022. End-to-end instance edge detection. *arXiv*.

[218] **Zou, Xueyan**, Fanyi Xiao, Zhiding Yu, and Yong Jae Lee. 2020, Best Paper Award. Delving deeper into anti-aliasing in convnets. *BMVC*.

[219] **Zou, Xueyan**, Fanyi Xiao, Zhiding Yu, Yuheng Li, and Yong Jae Lee. 2022, Special Issue. Delving deeper into anti-aliasing in convnets. *IJCV*.

[220] Tian, Zhi, Chunhua Shen, and Hao Chen. 2020. Conditional convolutions for instance segmentation. In *Computer vision–eccv 2020: 16th european conference, glasgow, uk, august 23–28, 2020, proceedings, part i 16*, 282–298. Springer.

[221] Torre, Vincent, and Tomaso A Poggio. 1986. On edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2):147–163.

[222] Touvron, Hugo, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2021. Training data-efficient image transformers & distillation through attention. 10347–10357. PMLR.

[223] Touvron, Hugo, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

[224] Tyleček, Radim, and Radim Šára. 2013. Spatial pattern templates for recognition of objects with regular structure. In *German conference on pattern recognition*, 364–374. Springer.

[225] VainF. 2020. DeepLabv3Plus-Pytorch.

[226] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Neurips*.

[227] Wang, Haohan, Songwei Ge, Eric P Xing, and Zachary C Lipton. 2019. Learning robust global representations by penalizing local predictive power. In *Neurips*.

[228] Wang, Haohan, Zexue He, Zachary C Lipton, and Eric P Xing. 2019. Learning robust representations by projecting superficial statistics out. In *Iclr*.

[229] Wang, Huiyu, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. 2021. Max-deeplab: End-to-end panoptic segmentation with mask transformers. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 5463–5474.

[230] Wang, Jianfeng, Xiaowei Hu, Zhe Gan, Zhengyuan Yang, Xiyang Dai, Zicheng Liu, Yumao Lu, and Lijuan Wang. 2021. Ufo: A unified transformer for vision-language representation learning. *arXiv preprint arXiv:2111.10023*.

[231] Wang, Jianfeng, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. 2022. Git: A generative image-to-text transformer for vision and language. *arXiv preprint arXiv:2205.14100*.

[232] Wang, Jiaqi, Kai Chen, Rui Xu, Ziwei Liu, Chen Change Loy, and Dahua Lin. 2019. Carafe: Content-aware reassembly of features. In *Iccv*.

[233] Wang, Peng, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In *Icml*.

[234] ———. 2022. Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. *arXiv preprint arXiv:2202.03052*.

[235] Wang, Qiang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip HS Torr. 2019. Fast online object tracking and segmentation: A unifying approach. In *Proceedings of the ieee conference on computer vision and pattern recognition*.

[236] Wang, Ting-Chun, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. 2018. High-resolution image synthesis and semantic manipulation with conditional gans. In *Iccv*.

[237] Wang, Wenhai, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. 2021. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*.

[238] Wang, Wenhui, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, et al. 2022. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*.

[239] Wang, Xinlong, Wen Wang, Yue Cao, Chunhua Shen, and Tiejun Huang. 2023. Images speak in images: A generalist painter for in-context visual learning. 2212.02499.

[240] Wang, Xinlong, Xiaosong Zhang, Yue Cao, Wen Wang, Chunhua Shen, and Tiejun Huang. 2023. Seggpt: Segmenting everything in context. *arXiv preprint arXiv:2304.03284*.

[241] Wang, Zirui, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. 2021. Simvlm: Simple visual language model pretraining with weak supervision. *arXiv preprint arXiv:2108.10904*.

[242] Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

[243] Wood, Jeffrey. 1996. Invariant pattern recognition: a review. In *Pattern recognition*.

[244] Wu, Jiannan, Yi Jiang, Peize Sun, Zehuan Yuan, and Ping Luo. 2022. Language as queries for referring video object segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 4974–4984.

[245] ———. 2022. Language as queries for referring video object segmentation. *arXiv preprint arXiv:2201.00487*.

[246] Wu, Kan, Houwen Peng, Minghao Chen, Jianlong Fu, and Hongyang Chao. 2021. Rethinking and improving relative position encoding for vision transformer. In *Iccv*, 10033–10041.

[247] Wu, Yuxin, and Kaiming He. 2018. Group normalization. In *Eccv*.

[248] Wu, Yuxin, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. 2019. Detectron2. https://github.com/facebookresearch/detectron2.

[249] Xiao, Bin, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. 2023. Florence-2: Advancing a unified representation for a variety of vision tasks. *arXiv preprint arXiv:2311.06242*.

[250] Xie, Cihang, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. 2019. Feature denoising for improving adversarial robustness. In *Cvpr*.

[251] Xie, Enze, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. 2020. Polarmask: Single shot instance segmentation with polar representation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 12193–12202.

[252] Xie, Enze, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. 2021. Segformer: Simple and efficient design for semantic segmentation with transformers. In *arxiv:2105.15203*.

[253] Xie, Saining, and Zhuowen Tu. 2015. Holistically-nested edge detection. In *Proceedings of the ieee international conference on computer vision*, 1395–1403.

[254] Xu, Jiarui, Shalini De Mello, Sifei Liu, Wonmin Byeon, Thomas Breuel, Jan Kautz, and Xiaolong Wang. 2022. Groupvit: Semantic segmentation emerges from text supervision. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 18134–18144.

[255] Xu, Mengde, Zheng Zhang, Fangyun Wei, Han Hu, and Xiang Bai. 2023. Side adapter network for open-vocabulary semantic segmentation. *arXiv preprint arXiv:2302.12242*.

[256] Xu, Ning, Brian Price, Scott Cohen, Jimei Yang, and Thomas S Huang. 2016. Deep interactive object selection. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 373–381.

[257] Xu, Ning, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. 2018. Youtube-vos: A large-scale video object segmentation benchmark. *arXiv preprint arXiv:1809.03327*.

[258] Yan, Bin, Yi Jiang, Jiannan Wu, Dong Wang, Ping Luo, Zehuan Yuan, and Huchuan Lu. 2023. Universal instance perception as object discovery and retrieval. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 15325–15336.

[259] Yang, Jianwei, Chunyuan Li, and Jianfeng Gao. 2022. Focal modulation networks. *arXiv preprint arXiv:2203.11926*.

[260] Yang, Jianwei, Chunyuan Li, Pengchuan Zhang, Bin Xiao, Ce Liu, Lu Yuan, and Jianfeng Gao. 2022. Unified contrastive learning in image-text-label space. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 19163–19173.

[261] Yang, Jianwei, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. 2023. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *arXiv preprint arXiv:2310.11441*.

[262] Yang, Jinyu, Mingqi Gao, Zhe Li, Shang Gao, Fangjing Wang, and Feng Zheng. 2023. Track anything: Segment anything meets videos. 2304.11968.

[263] Yang, Linjie, Yuchen Fan, and Ning Xu. 2019. Video instance segmentation. In *Iccv*.

[264] Yang, Zhao, Jiaqi Wang, Yansong Tang, Kai Chen, Hengshuang Zhao, and Philip HS Torr. 2022. Lavt: Language-aware vision transformer for referring image segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 18155–18165.

[265] Yang, Zhengyuan, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Faisal Ahmed, Zicheng Liu, Yumao Lu, and Lijuan Wang. 2022. Unitab: Unifying text and box outputs for grounded vision-language modeling. In *European conference on computer vision*, 521–539. Springer.

[266] Yao, Lewei, Jianhua Han, Youpeng Wen, Xiaodan Liang, Dan Xu, Wei Zhang, Zhenguo Li, Chunjing Xu, and Hang Xu. 2022. Detclip: Dictionary-enriched visual-concept paralleled pre-training for open-world detection. *arXiv preprint arXiv:2209.09407*.

[267] Yao, Lewei, Runhui Huang, Lu Hou, Guansong Lu, Minzhe Niu, Hang Xu, Xiaodan Liang, Zhenguo Li, Xin Jiang, and Chunjing Xu. 2022. FILIP: fine-grained interactive language-image pre-training. In *ICLR*.

[268] Ye, Linwei, Mrigank Rochan, Zhi Liu, and Yang Wang. 2019. Cross-modal self-attention network for referring image segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 10502–10511.

[269] Ye, Mang, Xu Zhang, Pong C Yuen, and Shih-Fu Chang. 2019. Unsupervised embedding learning via invariant and spreading instance feature. In *Cvpr*.

[270] Yin, Xuwang, and Vicente Ordonez. 2017. Obj2text: Generating visually descriptive language from object layouts. *arXiv preprint arXiv:1707.07102*.

[271] Yu, Jiahui, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. 2022. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*.

[272] Yu, Licheng, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. 2016. Modeling context in referring expressions. In *European conference on computer vision*, 69–85. Springer.

[273] Yu, Qihang, Huiyu Wang, Siyuan Qiao, Maxwell Collins, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. 2022. k-means mask transformer. In *European conference on computer vision*, 288–307. Springer.

[274] Yu, Weihao, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. 2023. Mm-vet: Evaluating large multimodal models for integrated capabilities. 2308.02490.

[275] Yu, Zhiding, Chen Feng, Ming-Yu Liu, and Srikumar Ramalingam. 2017. Casenet: Deep category-aware semantic edge detection. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 5964–5973.

[276] Yu, Zhiding, Weiyang Liu, Yang Zou, Chen Feng, Srikumar Ramalingam, BVK Kumar, and Jan Kautz. 2018. Simultaneous edge alignment and learning. In *Proceedings of the european conference on computer vision (eccv)*, 388–404.

[277] Yuan, Lu, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, Ce Liu, Mengchen Liu, Zicheng Liu, Yumao Lu, Yu Shi, Lijuan Wang, Jianfeng Wang, Bin Xiao, Zhen Xiao, Jianwei Yang, Michael Zeng, Luowei Zhou, and Pengchuan Zhang. 2021. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*.

[278] Yun, Sangdoo, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Iccv*.

[279] Zaheer, Manzil, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. In *Neurips*.

[280] Zang, Yuhang, Wei Li, Jun Han, Kaiyang Zhou, and Chen Change Loy. 2023. Contextual object detection with multimodal large language models. *arXiv preprint arXiv:2305.18279*.

[281] Zhai, Xiaohua, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. 2021. Scaling vision transformers. *arXiv: Computer Vision and Pattern Recognition*.

[282] Zhang, Hao, Feng Li, Huaizhe Xu, Shijia Huang, Shilong Liu, Lionel M Ni, and Lei Zhang. 2023. Mp-former: Mask-piloted transformer for image segmentation. *arXiv preprint arXiv:2303.07336*.

[283] Zhang, Hao, Feng Li, Xueyan Zou, Shilong Liu, Chunyuan Li, Jianwei Yang, and Lei Zhang. 2023. A simple framework for open-vocabulary segmentation and detection. In *Proceedings of the ieee/cvf international conference on computer vision*, 1020–1031.

[284] Zhang, Hao, Hongyang Li, Feng Li, Tianhe Ren, **Zou, Xueyan**, Shilong Liu, Shijia Huang, Jianfeng Gao, Lei Zhang, Chunyuan Li, and Jianwei Yang. 2023. Llava-grounding: Grounded visual chat with large multimodal models. *arXiv*.

[285] Zhang, Haotian, Pengchuan Zhang, Xiaowei Hu, Yen-Chun Chen, Liunian Harold Li, Xiyang Dai, Lijuan Wang, Lu Yuan, Jenq-Neng Hwang, and Jianfeng Gao. 2022. Glipv2: Unifying localization and vision-language understanding. *arXiv preprint arXiv:2206.05836*.

[286] Zhang, Hongyi, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2017. mixup: Beyond empirical risk minimization. In *Iclr*.

[287] Zhang, Pengchuan, Xiyang Dai, Jianwei Yang, Bin Xiao, Lu Yuan, Lei Zhang, and Jianfeng Gao. 2021. Multi-scale vision longformer: A new vision transformer for high-resolution image encoding. *arXiv preprint arXiv:2103.15358*.

[288] Zhang, Pengchuan, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. 2021. Vinvl: Making visual representations matter in vision-language models. *arXiv preprint arXiv:2101.00529*.

[289] ———. 2021. VinVL: Revisiting visual representations in vision-language models. In *Cvpr*.

[290] Zhang, Renrui, Zhengkai Jiang, Ziyu Guo, Shilin Yan, Junting Pan, Hao Dong, Peng Gao, and Hongsheng Li. 2023. Personalize segment anything model with one shot. 2305.03048.

[291] Zhang, Richard. 2020. Making convolutional networks shift-invariant again. In *Icml*.

[292] Zhang, Zhiyuan, Binh-Son Hua, David W Rosen, and Sai-Kit Yeung. 2019. Rotation invariant convolutions for 3d point clouds deep learning. In *3dv*.

[293] Zhao, Zihao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International conference on machine learning*, 12697–12706. PMLR.

[294] Zhong, Yiwu, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luowei Zhou, Xiyang Dai, Lu Yuan, Yin Li, et al. 2022. Regionclip: Region-based language-image pretraining. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 16793–16803.

[295] Zhou, Luowei, Chenliang Xu, and Jason Corso. 2018. Towards automatic learning of procedures from web instructional videos. In *Proceedings of the aaai conference on artificial intelligence*, vol. 32.

[296] Zhu, Wanrong, Jack Hessel, Anas Awadalla, Samir Yitzhak Gadre, Jesse Dodge, Alex Fang, Youngjae Yu, Ludwig Schmidt, William Yang Wang, and Yejin Choi. 2023. Multimodal c4: An open, billion-scale corpus of images interleaved with text. *arXiv preprint arXiv:2304.06939*.

[297] Zhu, Xizhou, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. 2020. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*.

[298] Zou, Xueyan, Zi-Yi Dou, Jianwei Yang, Zhe Gan, Linjie Li, Chunyuan Li, Xiyang Dai, Harkirat Behl, Jianfeng Wang, Lu Yuan, et al. 2022. Generalized decoding for pixel, image, and language. *arXiv preprint arXiv:2212.11270*.

[299] ———. 2023. Generalized decoding for pixel, image, and language. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 15116–15127.

[300] Zou, Xueyan, Linjie Li, Jianfeng Wang, Jianwei Yang, Mingyu Ding, Feng Li, Hao Zhang, Shilong Liu, Arul Aravinthan, Yong Jae Lee, and Lijuan Wang. Interfacing foundation models' embeddings. *in submission to CVPR2024*.

[301] Zou, Xueyan, Fanyi Xiao, Zhiding Yu, and Yong Jae Lee. 2020. Delving deeper into anti-aliasing in convnets. In *Bmvc*.

[302] Zou, Xueyan, Jianwei Yang, Hao Zhang, Feng Li, Linjie Li, Jianfeng Gao, and Yong Jae Lee. 2023. Segment everything everywhere all at once. *arXiv preprint arXiv:2304.06718*.

[303] Zou, Xueyan, Linjie Yang, Ding Liu, and Yong Jae Lee. 2021. Progressive temporal feature alignment network for video inpainting. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 16448–16457.

[304] Zou, Zhengxia, Zhenwei Shi, Yuhong Guo, and Jieping Ye. 2019. Object detection in 20 years: A survey. *arXiv preprint arXiv:1905.05055*.