Towards Unified Visual Perception

Shuyang Sun

St Cross College

University of Oxford

A thesis presented for the degree of

Doctor of Philosophy

Hilary, 2024

Abstract

This thesis explores the frontier of visual perception in computer vision by leveraging the capabilities of Vision Transformers (ViTs) to create a unified framework that addresses cross-task and cross-granularity challenges. Drawing inspiration from the human visual system's ability to process visual information at varying levels of detail and the success of Transformers in Natural Language Processing (NLP), we aim to bridge the gap between broad visual concepts and their fine-grained counterparts. Our investigation is structured into three parts.

First, we delve into a range of training methods and architectures for ViTs, with the goal of gathering valuable insights. These insights are intended to guide the optimization of ViTs in the subsequent phase of our research, ensuring we build a strong foundation for enhancing their performance in complex visual tasks.

Second, our focus shifts towards the recognition of fine-grained visual concepts, employing precise annotations to delve deeper into the intricate details of visual scenes. Here, we tackle the challenge of discerning and classifying objects and pixels with remarkable accuracy, leveraging the foundational insights gained from our initial explorations of ViTs.

In the final part of our thesis, we demonstrate how language can serve as a bridge, enabling vision-language models, which are only trained to recognize images, to navigate countless visual concepts on fine-grained entities like objects and pixels without the need for fine-tuning.

Acknowledgements

First and foremost, I express my deepest gratitude to my supervisor, Philip Torr, for his unwavering support and guidance throughout my DPhil studies. I am truly grateful for the opportunity to pursue my PhD under his mentorship. Phil's vibrant enthusiasm and energy have been incredibly motivating and encouraging.

I also owe a great deal of thanks to my mentors and hosts, including Victor Prisacariu, Siyang Li, Xiuye Gu, Weijun Wang, Liang-Chieh Chen, Song Bai, Vladlen Koltun, Philipp Krähenbühl, Liang Chen and Greg Slabaugh, and also my examiners, Prof. Andrew Zisserman and Prof. Roberto Cipolla, who have played a pivotal role in shaping me into the researcher I am today.

My time in the office, surrounded by Francesco, Csabi, Ameya, Fabio, Ashkan, Jindong, Jiawang, Guangrun, Yansong, Will, Jimmy and all other TVG members, has been invaluable. The warmth we shared made for a nurturing environment. A special thanks to Xiaojuan, who has been a role model in honing my research skills.

A heartfelt thank you to my external collaborators, Xiaoyu Yue, Jie-Neng Chen, and Ruifei He, for their dedication and enriching discussions. My interns, Runjia, Yuxi, Zhongrui, Jianhao and Zhaochong, deserve special mention for their hard work and the joy they brought to our team.

I am grateful for the natural serenity provided by the horses and cattle at Port Meadow and the geese and ducks at Iffley Lock, which are therapeutic during my darkest times. The food and lovely community at St Cross College also offered great comfort to me.

A special acknowledgement goes to Dr. Zhanghui Kuang and Prof. Wanli Ouyang for introducing me to the fascinating world of computer vision and deep learning. I am also grateful to Five AI for fully funding my DPhil study.

Last but certainly not least, my heartfelt appreciation goes to my family and my partner for their endless love, support and understanding.

Declaration

This thesis is submitted to the Department of Engineering Science, The University of Oxford, in fulfilment of the requirements for the degree of Doctor of Philosophy. I declare that this thesis is entirely my own work, and except where otherwise stated, describes my own research.

Shuyang Sun

St Cross College

April 2024

Contents

Chapter 1

Introduction

1.1 Background

The human visual system has the remarkable ability to process visual information at various levels of detail. For instance, upon observing another person, one might initially perceive the individual in a general sense before choosing to concentrate on finer aspects such as the face, eyes, or even eyebrows. This capacity demonstrates the system's aptitude for distilling visual input into hierarchical representations, ranging from broad overviews to intricate details and from the fundamental building blocks of pixels to complete objects.

Parallel to this, in the domain of computer vision, the methodology for understanding visual concepts varies, encompassing a spectrum of frameworks and procedures. These concepts are categorized into three primary entities: images, objects, and pixels. To facilitate the recognition of these diverse visual elements, data is curated with annotations at corresponding levels of granularity, and distinct task-specific decoders are employed for analysis.

In this thesis, we address the methodologies for cross-task and cross-granularity visual representation learning from a modern point of view. We aim to build a visual

system that can simultaneously perform multiple tasks with different granularities. To achieve this goal, we divide this problem into three parts:

1. The training techniques and architectural structure of Vision Transformers.
2. Learning to recognize fine-grained concepts (objects and pixels) with fine-grained annotations.
3. Learning to recognize cross-granularity concepts in an open domain.

The introduction is arranged as the following: Section 1.2 introduces the motivations of the thesis. Section 1.3 illustrates the outline and organization of the thesis. Section 1.4 lists all research papers included in this thesis.

1.2 Motivation

1.2.1 Motivation from psychology.

Extensive psychological research has provided compelling evidence that human vision possesses the remarkable ability to analyze and interpret complex scenes through the construction of compositional representations. This process involves discerning part-whole relationships that span the spectrum from the granular elements, like pixels, to more abstract constructs such as parts, objects, and entire scenes [115, 139]. This ability underscores the human visual system's integrative approach to perception, seamlessly blending various levels of granularity into a cohesive understanding of the visual world. On the contrary, in the domain of computer vision, tasks are segregated according to the granularity of the elements to be recognized, such as images, objects, and pixels, each employing distinct architectures and weight parameters. This approach stands in contrast to the integrative mechanism of human vision.

1.2.2 Motivation from the field of NLP.

The Transformer [273] architecture has demonstrated its efficacy in processing sequence-based data within the realm of Natural Language Processing (NLP), particularly when trained on a large scale. By expanding the capacity of the model through the utilization of extensive datasets, it is possible to construct foundational models [15, 74, 318] capable of addressing various tasks with different granularities in a "zero-shot" fashion. Motivated by this, we aim to investigate the potential applicability and effectiveness of Vision Transformers (ViTs) within the domain of computer vision. Meanwhile, the foundation model in the field of NLP is highly dependent on prompt engineering, which is rarely explored in the visual domain.

1.2.3 Motivation from multi-modal understanding.

Despite the variability in the granularity of visual entities within the visual domain, ranging from broad scenes to specific objects and even to individual pixels, these entities often converge within a unified labeling framework in the linguistic domain. For instance, the description of a person can remain consistent whether referring to their presence in a full image or within a more confined area, such as a bounding box. This observation suggests a promising avenue for recognizing fine-grained visual concepts by anchoring them within the linguistic domain. Taking advantage of the commonalities in language descriptions across different levels of visual detail, it becomes possible to create a more unified and efficient framework for visual recognition.

1.3 Thesis Outline and Contributions

This section illustrates the basic outline of the thesis and summarizes the key contributions of all papers included. We group the thesis into three parts: explorations of Vision Transformers, cross-task and cross-granularity visual perception and open-vocabulary cross-granularity visual perception.

Part I: Explorations of Vision Transformers

By investigating the structures of Vision Transformers (ViTs) [78], our goal is to adapt the transformative capabilities of the Transformer architecture [273] in natural language processing to the realm of computer vision. This endeavor aims at developing a foundational model for visual tasks, paralleling the success seen in natural language processing.

Within this context, we demonstrate that incorporating inductive biases tailored for 2D imagery can significantly enhance the efficiency and effectiveness of Vision Transformers' learning processes [78]. For instance, in Chapter 2, we experiment with integrating a 2D window-based attention mechanism within the architecture. Although such attention mechanisms typically incur high computational costs, our innovative approach of separating local and global attentions allows us to maintain detailed information while enlarging the model's receptive field.

Additionally, our exploration reveals limitations in the conventional patch-wise division approach employed by Vision Transformers for processing 2D images, mainly due to the uneven distribution of information across patches. To address this, we provide solutions in two aspects. In Chapter 3, we introduce a progressive sampling module into the architecture, enabling a more focused analysis on areas within images that are informative. Furthermore, Chapter 4 explores how discrepancies between the image inputs and the corresponding label spaces can be mitigated. We achieve this by applying attention weights to the label assignment process during the CutMix technique [332], thereby ensuring a more precise alignment between the input data and its annotations. This adaptation not only improves the performance of the model, but also aligns more closely with the nuanced nature of visual perception.

Part II: Cross-task and Cross-granularity Visual Perception

Following our exploration of learning visual representations using Vision Transformers in Part I, we advance our research by leveraging this knowledge towards crafting a unified framework capable of recognizing a variety of visual entities. Our objective has been to develop a model proficient in cross-granularity visual perception, utilizing both in-domain data and distinct training datasets. Specifically, in Chapter 5, we introduce a model architecture based on a part-whole hierarchy, designed to identify concepts at the image, object, and pixel levels. However, this approach initially resulted in suboptimal performance, attributed to the disparate nature of the data sources and the differing objectives of the training processes.

To overcome these challenges, we delve into the potential of harnessing in-domain data to harmonize the training goals for disparate tasks. In Chapter 6, our investigation focuses on aligning the training objectives of two distinct tasks, *i.e.* instance segmentation and semantic segmentation. We discover that the training goal of the simpler task can serve as a preliminary step, or a 'relaxed' version, for the more complex, composite task. This insight leads to a novel training strategy, whereby starting with the simpler task accelerates and enhances the learning process for the composite task. Through this methodology, we not only streamline the training process but also achieve superior performance, thereby contributing to the advancement of models that can navigate the intricacies of visual perception across different levels of detail.

Part III: Open-vocabulary Cross-granularity Visual Perception

Language serves as a comprehensive labelling framework encompassing a wide array of visual concepts. However, the creation of fine-grained annotations, such as bounding boxes and segmentation masks, is often constrained by high labelling costs. As a result, the lexicons associated with most fine-grained visual datasets are relatively narrow. Concurrently, vision-language foundation models like CLIP, which are trained on pairs of images and text, typically lack the capacity for fine-grained visual analysis due to their training on more generalized data.

In Chapter 7, we address this gap by demonstrating the feasibility of augmenting a pre-trained CLIP model with fine-grained visual perception capabilities without any training efforts, specifically for tasks such as image segmentation. This advancement underscores the potential of leveraging language as a versatile and universal descriptor to bridge the divide between broad visual concepts and their more detailed, fine-grained counterparts.

1.4 Publications

The content spanning from Chapter 2 through to Chapter 7 is composed of an individual research paper, each making a distinct contribution to the field. The majority of these papers have undergone rigorous peer review and have been accepted for publication at leading conferences and journals, attesting to their quality and the relevance of their findings. Chapter 5 presents a paper in pre-print status. Aside from necessary formatting adjustments to ensure consistency across the dissertation, no changes have been made to the original texts of these published works. We provide a detailed listing of these papers below.

Chapter 2: Shuyang Sun, Xiaoyu Yue, Hengshuang Zhao, Philip HS Torr, and Song Bai. Patch-based separable transformer for visual recognition. IEEE Transactions

on Pattern Analysis and Machine Intelligence (T-PAMI), 2022.

Chapter 3: Xiaoyu Yue*, Shuyang Sun*, Zhanghui Kuang, Meng Wei, Philip HS Torr, Wayne Zhang, and Dahua Lin. Vision transformer with progressive sampling. In International Conference on Computer Vision (ICCV), 2021.

Chapter 4: Jie-Neng Chen*, Shuyang Sun*, Ju He, Philip HS Torr, Alan Yuille, and Song Bai. Transmix: Attend to mix for vision transformers. In Conference on Computer Vision and Pattern Recognition (CVPR), 2022.

Chapter 5: Shuyang Sun*, Xiaoyu Yue*, Song Bai, and Philip Torr. Visual parser: Representing part-whole hierarchies with transformers. arXiv:2107.05790 (2021).

Chapter 6: Shuyang Sun, Weijun Wang, Andrew Howard, Qihang Yu, Philip Torr, and Liang-Chieh Chen. ReMaX: Relaxing for better training on efficient panoptic segmentation. In Advances in Neural Information Processing Systems (NeurIPS), 2023.

Chapter 7: Shuyang Sun*, Runjia Li*, Philip Torr, Xiuye Gu, and Siyang Li. CLIP as RNN: Segment Countless Visual Concepts without Training Endeavor. In Conference on Computer Vision and Pattern Recognition (CVPR), 2024.

Part I: Explorations of Vision Transformers

Chapter 2

Patch-based Separable Transformer for Visual Recognition

The paper has been accepted for publication at the IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022

Patch-based Separable Transformer for Visual Recognition

Shuyang Sun[1] Xiaoyu Yue[2] Hengshuang Zhao[3] Philip H.S. Torr[1] Song Bai[4]

[1]University of Oxford [2]The University of Sydney [3]Hong Kong University

[4]Bytedance AI Research

kevinsun@robots.ox.ac.uk

Abstract

The computational complexity of transformers limits it to be widely deployed onto frameworks for visual recognition. Recent work [78] significantly accelerates the network processing speed by reducing the resolution at the beginning of the network, however, it is still hard to be directly generalized onto other downstream tasks *e.g.* object detection and segmentation like CNN. In this paper, we present a transformer-based architecture retaining both the local and global interactions within the network, and can be transferable to other downstream tasks. The proposed architecture reforms the original full spatial self-attention into pixel-wise local attention and patch-wise global attention. Such factorization saves the computational cost while retaining the information of different granularities, which helps generate multi-scale features required by different tasks. By exploiting the factorized attention, we construct a Separable Transformer (SeT) for visual modeling. Experimental results show that SeT outperforms the previous state-of-the-art transformer-based approaches and its CNN counterparts on three major tasks including image classification, object detection and instance segmentation.

*The first two authors contribute equally to this work.

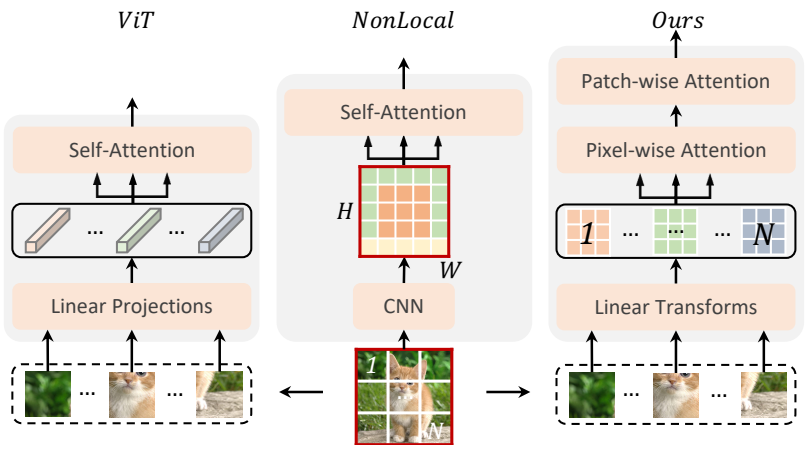


Figure 1. Left: Vision Transformer (ViT) [78] directly projects all pixels of each patch into global tokens and applies self-attention afterwards. Middle: the full spatial attention (*a.k.a.* Non-local [288] block) takes all pixels of an image as inputs so that it will have huge memory and computational cost when the resolution is high. Right: our SeT factorizes the full spatial attention into (1) a pixel-wise attention that only interacts with local pixels of each patch and (2) a patch-wise attention that reasons the global relationship between patches.

1 Introduction

Convolutional Neural Networks (CNNs) [106, 159, 244] have been dominating nearly all vision tasks since the emergence of AlexNet [159]. However, CNN has been proven to be less effective when applied on some other sequential data like natural language, where the attention-based Transformers [273] serve as the most popular feature extractor. The attention model like Transformer [273], due to its data-adaptive nature, conceptually have a larger capacity and preserve the translational equivariance [224, 346]. These properties imply the potential of attention-based models as a better engine for vision tasks.

Existing attention-based networks [78, 128, 224, 346] cannot be practically transferred onto downstream tasks due to the computational and memory inefficiency [128, 346] or the lack of multi-scale features [78, 265]. These attention-based models can be roughly divided into two kinds, (1) token-based global attention [78, 265] and (2) convolution-like local attention [128, 224, 346]. To save the computational and memory cost, Vision Transformer (ViT) [78] as a typical representation of the token-based approach, proposes to summarize all pixels of independent patches into

global visual tokens before feeding them into the Transformer encoder. However, as human vision knows when and how to capture the information at the highest resolution, an ideal visual processor should be capable to analyze both the local and global information at the same time. Visual modeling without interacting with the local information is counter-intuitive. As a consequence, the lack of local information modeling in ViT results in data-inefficiency and limited transferability to downstream tasks that call for multi-scale features.

As for the local attention, although existing convolution-free networks [128, 224, 346] based on the pyramid architecture are data-efficient and could (conceptually) be transferable to downstream tasks, all these works only propose local attention within their building blocks, which limits the model in capturing long-range global visual patterns. Besides, they are still memory-expensive, or too slow due to the incompatibility with the modern hardware accelerator, or both. Such inefficiency also prevents them to be widely deployed for other downstream vision tasks that may need to process images of way larger resolutions.

The aim of this work is to construct a practical transformer-based network with strong performance and transferability to down-stream tasks so that it can be used as an alternative to conventional CNN. To this end, the network needs to take advantages of both local and global interactions. As shown in Figure 1, similar to ViT (Figure 1 left), we divide the entire image into N patches. However, instead of projecting all pixels of each patch into global tokens, we first query the pixels of each patch locally with their close neighbors using the *pixel-wise* multi-head attention. After the local pixel-wise interaction, the spatial information of each patch will be gathered, so that the *patch-wise* attention can use them to infer the correlation among patches. The whole process can be also regarded as a spatial factorization of the original full spatial attention (*a.k.a.* Non-local [288] block in Figure 1 middle). Therefore we name it as Separable Transformer (SeT). Such factorization largely reduces the memory cost compared to the full spatial attention so that both local

and global attention can be practically applied even on large feature maps. In summary, the contribution of this paper is three-fold:

1. By factorizing the full spatial attention into local pixel-wise attention and patch-wise attention, SeT is the first transformer-based architecture retaining both local and global interactions throughout the network.

2. SeT achieves state-of-the-art performance compared with existing transformer-based architectures. Besides, SeT can be well transferred onto other downstream tasks like object detection and instance segmentation.

3. We also reveal that the it is the global interaction that makes the token-based transformers to be unfriendly towards convergence. By enhancing the locality of the transformer, SeT can be more data-efficient compared to the token-based approaches *e.g.* ViT [78] and DeiT [265].

2 Related Work

Convolutional neural networks. Conventional CNNs have dominated the field of computer vision. After the great success of AlexNet [159] for image classification, lots of following powerful convolutional architectures are developed for image recognition [106, 132, 222, 262]. They serve as the basic backbones for various downstream computer vision tasks, such as semantic segmentation [7, 347] and object detection [93, 95, 105, 230].

Transformers and self-attention mechanisms for visual recognition. With the success of Memory Networks [250] and Transformers [273] for natural language modeling, lots of works in the field of computer vision attempted to migrate similar self-attention mechanism as an independent block into CNNs for image classification [12, 46, 130, 296], object detection [24, 127, 246] and video prediction [138, 288] *etc.*

Recent works tried to replace all convolutional layers in neural networks with local

attention layers to build up self-attention based networks [58, 119, 127, 224, 343, 346]. Though these works are proven to be successful in improving performance while reducing the network complexity in terms of FLOPS and the number of parameters. The memory cost and runtime latency of these models are still very higher than the conventional CNNs, which prevents them to be widely deployed for practical use. To mitigate this problem, the Vision Transformer (ViT) [78] chose to largely reduce the image resolution and only retain global information while processing. The success of such radical change on image modeling is surprising but the downside is also obvious. Compared with other existing models, ViT requires much more training data for generalization and the results are still worse than the state-of-the-art CNN counterparts *e.g.* [222, 262] when both models are well-tuned on ImageNet. Some concurrent works like Swin Transformer [187], PVT [286] and MViT [86] use either local attention or global attention for feature extraction. Another concurrent work ViL [343] propose both global and local interactions within its structure, which is the most similar work to ours. However, its local attention is a convolution-like self-attention with dense sliding window like [128, 224, 346], which will lead to very high computational latency as we discussed above. Unlike the listed works above, as a transformer-based network, our model can do both local and global attention simultaneously as we cut down the computational cost via spatial grouping.

Task-specific transformers. In addition to the transformer based architectures designed for image recognition, there are also task-specific transformer-based architectures as the transferability of the current transformer is still limited. Specifically, DETR [24] and its variant Deformable DETR [359], UP-DETR [66], Sparse RCNN [252] are designed specifically for object detection and instance segmentation. What proposed in this paper is independent to these works as we intend to build up a versatile transformer-based network that can be directly transferred onto other down-stream tasks as the engine for extracting features.

Kernel factorization methods. Recent works design different efficient CNN ar-

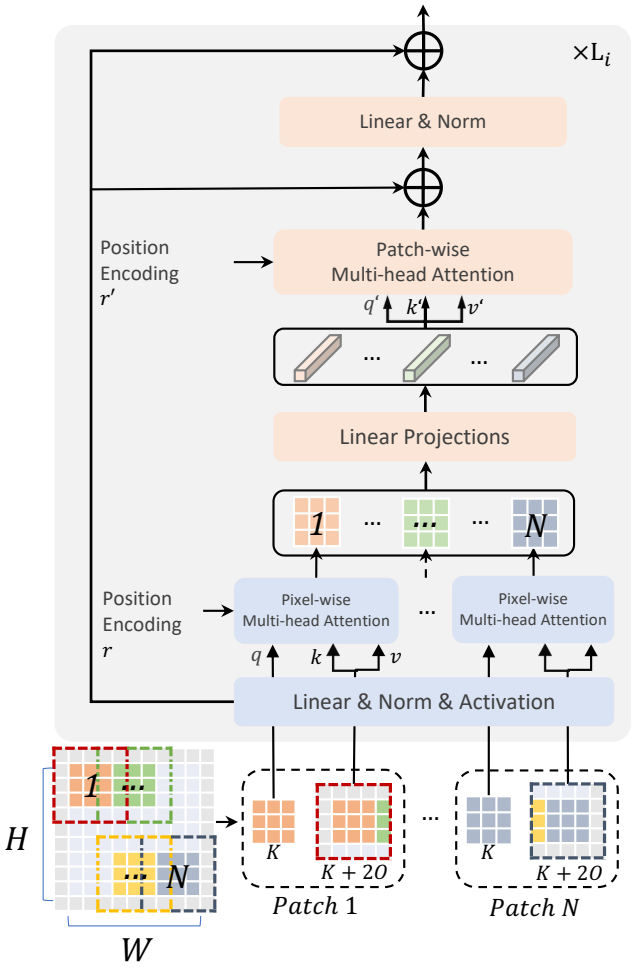


Figure 2. Overview of SeT. L_i is the number of blocks of stage i. Here we show an example when $K = 3, O = 1$.

chitectures by factorizing the convolutions on the channel [126, 236, 306, 345], or spatiotemporal dimension [308]. Our patch-based separable attention also takes advantage of the grouping concept, but it is applied to spatial dimension instead. Going beyond the spatial grouping, we also introduce to enlarge the reference scope of the key and value so that pixels within the pixel-wise attention can have a larger field of view and learn to model long-range interactions.

3 Patch-base Separable Transformer

The patch-based Separable Transformer (SeT) is primarily composed of two sub-modules, the pixel-wise attention for extracting local features, and the patch-wise

attention for reasoning the global interactions. **Acknowledgement.** This work is supported by the UKRI grant: Turing AI Fellowship EP/W002981/1 and EPSRC/-MURI grant: EP/N019474/1. We would also like to thank the Royal Academy of Engineering and FiveAI.

3.1 Overview

SeT can be regarded as a variant of the traditional Transformer Encoder. The basic building block of SeT is exhibited in Figure 2. Given an input image $x \in \mathbb{R}^{C\times H\times W}$, the whole image is first divided into N ***query patches*** with no overlapping in between. The patch size of each patch is $K \times K$, so that $N = \frac{HW}{K^2}$. Note that if HW cannot be exactly divided by K^2, then we will apply zero padding to the input and mask it out like [24, 224] in the lateral self-attention modules. Apart from these non-overlapping patches, we further crop N ***reference patches*** with a larger patch size $(K+2O)\times(K+2O)$ and step size K, so that each pixel within the query patch, especially those at the patch boundary, can refer to a wider neighborhood like convolution. The prepared query patches and reference patches will be parallelly sent into SeT for processing.

Before being sent to the self-attention modules, all pixels on the feature map will be first transformed by a linear operation. The linear transformation will not change the spatial size of each patch, instead, it maps the C channels of each pixel into $D = C/B$, which can be viewed as a bottleneck design with $B\times$ channel reduction like [106]. The transformed features will be then fed into the pixel-wise separable attention. Following the *query-key-value* formulation of the original multi-head self-attention, here in the pixel-wise attention, features of the query patch serve as the query and those of the reference patch serve as the key and value. We adopt the grouping concept in [273, 306] which applies independent groups of weights to generate attention maps. The output of the pixel-wise attention are still patches with local features corresponding to a specific part of input. Therefore a linear

projection is needed to squeeze the spatial dimension of each patch into 1 so that each patch is gathered as a visual token. These visual tokens will serve as the ingredients of the patch-wise separable attention for global reasoning. In this way, both the local and global information are modelled by SeT.

Following [273], skip connections are applied from the input to the output of the separable attention module. We simplify the original feed-forward layer into a single linear transformation followed with normalization after the identity mapping. The transformed feature will be added to the input of the block to generate the final output of the SeT building block.

3.2 Pixel-wise Separable Attention

Pixel-wise separable attention make pixels of each patch interact with their close neighbors. All pixels $x^p \in \mathbb{R}^{K^2 \times D}$ of a specific patch p share a same neighborhood $x_n^p \in \mathbb{R}^{(K+2O)^2 \times D}$. Given three learnable linear transformation with weights W_q, W_k, W_v, we obtain three variables q, k, v representing the query, key and value of the self-attention. Note that $q \in \mathbb{R}^{K^2 \times D}$ is generated using contents in the query patch x^p while the $k, v \in \mathbb{R}^{(K+2O)^2 \times D}$ are linear transformations of the content of the reference patch x_n^p. Following [12, 224, 246], a learnable relative positional embedding $r \in \mathbb{R}^{(K+2O)^2 \times D}$ is applied to capture the location information, in this way, the attention affinity matrix M can be calculated as:

$$\begin{aligned} q &= x^p W_q, \\ k &= x_n^p W_k, \\ v &= x_n^p W_v, \\ M &= q \cdot k^T + q \cdot r^T, \end{aligned} \tag{2.1}$$

where $M \in \mathbb{R}^{K^2 \times (K+2O)^2}$ represents the correspondence between every pixel in the query patch to pixels in the reference patch. Following the common practice, we

further apply a softmax normalization on the matrix by:

$$M_{i,j} = \frac{1}{\sqrt{D}} \frac{e^{M_{i,j}}}{\sum_l e^{M_{i,l}}}. \tag{2.2}$$

A normalization factor $\frac{1}{\sqrt{D}}$ is set here to prevent the output distribution of softmax to be one-hot. The matrix M represents the affinity matrix between pixels of the query patch and the reference patch. We can aggregate the values in v using M by:

$$y^p = M \cdot v, \tag{2.3}$$

where $y^p \in \mathbb{R}^{K^2 \times D}$ represents the output of each head of the pixel-wise separable attention. Following [273], the above process can run in multiple heads in parallel. The output of each head will be stacked together as the final output.

3.3 Patch-wise Separable Attention

Patch-wise separable attention intends to update all pixels within each patch with the inferred correlation with other patches. Given $y = \{y^1, ..., y^N\} \in \mathbb{R}^{N \times K^2 \times D}$, we first normalize y with a BatchNorm [135] layer, then apply a linear transformation to all K^2 pixels of the patch that maps K^2 dimension into 1, which is identical to a weighted average pooling operation. Denote the down-sampled output as $y_d \in \mathbb{R}^{N \times D}$, and reshape y to $NK^2 \times D$. Similar to the formulation of the pixel-wise separable attention, we denote the linear transformations and their weights for the patch-wise separable attention as $q' \in \mathbb{R}^{NK^2 \times D}$ and $k', v' \in \mathbb{R}^{N \times D}$ and $W_{q'}, W_{k'}, W_{v'}$, then the attention matrix can be formulated as:

$$\begin{aligned} q' &= yW_{q'} \\ k' &= y_d W_{k'} \\ v' &= y_d W_{v'} \\ M' &= q' \cdot (k')^T + q' \cdot (r')^T, \end{aligned} \tag{2.4}$$

where $r' \in \mathbb{R}^{N \times D}$ represents the absolute positional embedding for all patches, and $M' \in \mathbb{R}^{NK^2 \times N}$ denotes the affinity matrix between patches. Then the output of the patch-wise separable attention can be calculated as:

$$\begin{aligned} M'_{i,j} &= \frac{1}{\sqrt{D}} \frac{e^{M'_{i,j}}}{\sum_l e^{M'_{i,l}}} \\ z &= M' \cdot v', \end{aligned} \tag{2.5}$$

where $z \in \mathbb{R}^{NK^2 \times D}$ is the final output of the patch-wise attention. Following [273], we apply skip connections from the input x^p to the output of the attention. Note that the pixel-wise and patch-wise separable attention can be binded together or applied independently. When both attention are applied together, the final output of the entire attention module is z. If the global separable attention is not applied, the final output will turn to be y. In Section 4, we will compare both effect of applying the two separable attention modules.

3.4 Computational Cost of Separable Attention

The factorization above reduces the computational cost compared with the conventional full spatial attention [246, 288]. Given the same input to the standard full spatial attention, which takes the *all2all* interaction between every two pixels on a feature map, its computational cost $\mathcal{C}_{full}$ can be calculated as:

$$\mathcal{C}_{full} = \mathcal{O}(CHWHW), \tag{2.6}$$

where the square of $H \times W$ will result in huge memory and computing cost when the spatial size is high.

As for the computational cost of SeT, the computational complexity $\mathcal{C}_{SeT}$ is:

$$\begin{aligned} \mathcal{C}_{SeT} &= \mathcal{O}(CK^2(K+2O)^2 + CK^2N^2) \\ &= \mathcal{O}(CK^2(K+2O)^2 + C\frac{H^2W^2}{K^2}) \end{aligned} \tag{2.7}$$

Table 1. General specification for SeT family. Building blocks of SeT are shown in brackets.

Stage	Output Resolution	Block Settings
	$\frac{H}{4} \times \frac{W}{4}$	$7 \times 7, 64$, Conv, stride 2 3×3 Max Pool, stride 2
Stage_1	$\frac{H}{4} \times \frac{W}{4}$	$\begin{bmatrix} C = 64 \times B \\ D = 64 \\ G = 4 \end{bmatrix} \times L_1$
Stage_2	$\frac{H}{8} \times \frac{W}{8}$	2×2 Avg Pool, stride 2 $\begin{bmatrix} C = 128 \times B \\ D = 128 \\ G = 8 \end{bmatrix} \times L_2$
Stage_3	$\frac{H}{16} \times \frac{W}{16}$	2×2 Avg Pool, stride 2 $\begin{bmatrix} C = 256 \times B \\ D = 256 \\ G = 8 \end{bmatrix} \times L_3$
Stage_4	$\frac{H}{32} \times \frac{W}{32}$	2×2 Avg Pool, stride 2 $\begin{bmatrix} C = 512 \times B \\ D = 512 \\ G = 8 \end{bmatrix} \times L_4$
	1×1	1000, Linear

To make it clear, we calculate $\frac{\mathcal{C}_{SeT}}{\mathcal{C}_{full}}$ for better comparison:

$$\begin{aligned} \frac{\mathcal{C}_{SeT}}{\mathcal{C}_{full}} &= \frac{CK^2(K+2O)^2 + C\frac{H^2W^2}{K^2}}{CH^2W^2} \\ \frac{\mathcal{C}_{SeT}}{\mathcal{C}_{full}} &= \frac{1}{K^2} + \frac{K^2(K+2O)^2}{H^2W^2} \\ \frac{\mathcal{C}_{SeT}}{\mathcal{C}_{full}} &\approx \frac{K^2(K+2O)^2}{H^2W^2}. \end{aligned} \tag{2.8}$$

For simplicity, we ignore the fractional $\frac{1}{K^2}$ as we set $K > 3$ in practice. Since normally a patch is just a small portion of the entire feature map, *s.t.* $(K+2O)^2 < HW$, $\mathcal{C}_{SeT} < \mathcal{C}_{full}$. SeT will save more computational cost when the feature map resolution is high. The huge reduction in computation and memory cost makes it possible to transfer attention onto larger feature maps.

Table 2. Hyper-parameters for models of SeT family.

Model	Number of Blocks $[L_1, L_2, L_3, L_4]$	B	K	O
SeT-8	$[2, 2, 2, 2]$	2	8	2
SeT-16	$[3, 4, 6, 3]$	4	8	2
SeT-33	$[3, 4, 23, 3]$	4	8	2

3.5 Network Structure

To construct an entire network using the above basic block, we first divide the whole network into four stages according to the resolution of their feature maps. Note that the network is constructed with the following hyper-parameters:

- B is the bottleneck factor
- G represents the number of heads (groups) of the multi-head attention.
- C represents the number of channels for the input and output of each block.
- L_1, L_2, L_3, L_4 are the numbers of blocks for stage $1, 2, 3, 4$ respectively.
- $K, K+2O$ are the patch sizes of query patch and reference patch as illustrated above.

We build up three models named SeT-8, set-16 and SeT-33 according to the number of the building blocks. Based on the experimental results in Section 4, for the first two stages of SeT, we only use pixel-wise attention in the building blocks, while for the last two stages, both pixel-wise local attention and patch-wise global attention are applied.

Apart from the attention model, we also construct a hybrid model by replacing all separable attention components of the building block with a 3×3 convolutions at the first two stages of the network.

4 Experiments for Image Classification

4.1 Implementation Details on ImageNet

We first conduct experiments for image classification on the ImageNet 2012 classification dataset [233] that includes 1000 classes. There are 1.2 million images for training and 50 thousands images for validation. Unlike ViT [78], which is pre-trained on other datasets *e.g.* ImageNet-21k, we only train our network on the training set of ImageNet. We train the network using a batch size of 2048. The learning rate is first warmed up for 5 epochs from 0 to 0.6 and will be decreased to 0 at the end of the training process following a cosine annealing schedule [195]. Previous transformer-based backbones like [78, 265] heavily rely on the data augmentation and regularization, as the learnt weights are data-dependant. However, we find our network can be well-tuned with much fewer bells and whistles. Details about this will be discussed in the following sections.

4.2 Experimental results on ImageNet

We show the experimental results on ImageNet in Table 3, which includes results for both CNN architectures *e.g.* ResNet [106], SEResNet[130] and RegNet [222] and transformer-based architectures *e.g.* ViT [78], DeiT [265], Local Relation Networks (LRNet) [128] and Stand-Alone Networks (SAResNet) [224]. Note that we categorize the attention models like [128, 224] as transformer-based architectures because the whole backbone of attention model is identical to a transformer encoder.

SeT *vs.* Token-based Transformers. We first compare SeT to the token-based Vision Transformer (ViT), which radically reduces the image resolution at the beginning of the network. When both networks are trained from scratch only training on the training set of ImageNet, SeT-33 can outperform ViT-B by 4.9% with only about half of its number of parameters and FLOPS. The remarkable improvement on ImageNet demonstrates the importance of retaining the local features for image

Table 3. Experimental results on ImageNet. [1] represents our enhanced re-implementation using training tricks *e.g.* Mixup and AutoAug shown in the second last row of Table 5.

Model	Params (M)	FLOPS (G)	throughput (img/s)	Top-1 Acc (%)
CNN Architectures				
ResNet-50 [106]	25.5	4.1	1226	76.2
ResNet-101 [106]	44.4	7.8	753	77.4
ResNet-152 [106]	60.0	11.6	526	78.3
ResNet-50++[1]	25.5	4.1	1226	78.6
ResNet-101++[1]	44.4	7.8	753	80.4
RegNetY-4G++ [265]	20.6	4.0	1156.7	80.0
RegNetY-8G++ [265]	39.2	8.0	591.6	81.7
Transformer Architectures				
ViT-B↑384 [78]	86.4	55.4	85.9	77.9
DeiT-Ti [265]	5.7	1.6	2536.5	72.2
DeiT-S [265]	22.1	4.6	940.4	79.8
DeiT-B [265]	86.6	17.6	292.3	81.8
LRNet-18 [128]	14.4	2.5	-	74.6
LRNet-50 [128]	23.3	4.3	-	77.3
LRNet-101 [128]	42.0	8.0	-	78.5
Swin-Ti [187]	29	4.5	755	81.3
Swin-S [187]	50	8.7	437	83.0
MViT-B-maxpool [86]	37	7.8	-	82.5
MViT-B [86]	37	7.8	-	83.0
PVT-T [286]	13.2	1.9	-	75.1
PVT-S [286]	24.5	3.8	-	79.8
PVT-M [286]	44.2	6.7	-	81.2
SAResNet-26 [224]	13.7	2.7	-	74.8
SAResNet-50 [224]	18.0	3.6	-	77.6
SeT-8	9.7	1.8	1336	78.1
SeT-16	24.7	4.4	683	81.7
SeT-33	40.3	7.7	401	83.3

recognition.

Another token-based vision transformer, DeiT, is also listed in Table 3 for comparison. We note that DeiT is *concurrent* to our work. The basic structure of DeiT is identical to what proposed in ViT. The biggest difference between them is that DeiT can achieve competitive performance on ImageNet without using any additional datasets for pre-training. DeiT manages to do this by successfully finding extensive data augmentations and regularization techniques. When compare SeT with DeiT, we observe that SeT-8 can surpass DeiT-Tiny by a significant 5.9% in terms of top-1 accuracy. As for small models like set-16, it can outperform DeiT-S

Table 4. Experimental results on ImageNet when applying factorization on DeiT.

Model	Params (M)	FLOPS (G)	Top-1 Acc (%)
DeiT-S [265]	22.1	4.6	79.8
SeT-ViT-S	23.9	4.2	80.4

by 1.7% with much fewer data augmentations and training tricks, which indicates that SeT is much more data-efficient than the DeiT. SeT-33 as a larger model in the SeT family can achieve comparable performance to DeiT with 56.3% fewer FLOPS and 53.1% fewer parameters. As for some other concurrent Transformers like PVT [286] and Swin Transformer [187], SeT can also show its superior performance. For example, SeT-8 can outperform PVT-T by a significant 3%, and SeT-16 can outperform PVT-S and Swin-T by 1.9% and 0.4% respectively.

SeT *vs.* Local-aware Transformers. SeT is better than existing local-aware models. Apart from token-based methods, we further compare SeT with other local-aware attention backbones *e.g.* LRNet[128] and SAResNet[224]. LRNet [128] is designed for inferring local relationship between pixels with attentive mechanism. As shown in Table 3, SeT outperforms LRNet when compared under similar model capacity. As for the comparison with SAResNet, SeT-8 can surpass SAResNet26 by 3.5% with about 33.3% fewer FLOPS. The comparison with local-aware attention-based models demonstrates the significance of the global attention in SeT.

SeT *vs.* CNN architectures. In this part, we observe that SeT can be comparable with the state-of-the-art CNN architectures when FLOPS $\leq$ 8G. We compare SeT with CNN baselines including ResNet [106], Squeeze-Excite (SE) ResNet [130] and RegNetY (*w/* SE). In addition to the results of their original paper, we also compare with their better-tuned versions (followed by ++ in Table 3) reported in [246, 265] for fair comparison. Note that the tricks like data augmentation and regularization applied on these baselines can lead to significant improvement. For example, the top-1 accuracy of SEResNet50++ is 2.5% higher than its original version. The

Table 5. Training tricks used for improving DeiT and SeT. SeT can outperform DeiT with much fewer data augmentations and regularization techniques.

Model	Optimizer	RandAug	AutoAug	MixUp	CutMix	Color Jitter	RandErase	DropPath	Repeated Aug	Label Smooth	Act. Func.	Epochs	Top-1
DeiT-S	AdamW	✓	✗	✓	✓	✓	✓	✓	✓	✓	GELU	300	79.8
ViT-B	SGD	✗	✓	✓	✗	✗	✗	✓	✗	✓	GELU	300	71.5
SeT-16	SGD	✗	✗	✗	✗	✗	✗	✓	✗	✓	ReLU	350	79.2
		✗	✗	✗	✗	✗	✗	✓	✗	✓	SiLU	350	79.6
		✗	✓	✓	✗	✗	✗	✓	✗	✓	SiLU	350	80.5
	AdamW	✓	✗	✓	✓	✓	✓	✓	✗	✓	SiLU	300	81.7

Table 6. ✓and ✗in the brackets represent the existence of the module in stage [1, 2, 3, 4]. If both the pixel-wise and patch-wise attentions are not applied, then a 3×3 convolution is placed as alternative, making it to be a hybrid model. Models without postfix are trained for 120 epochs without bells and whistles. ++ indicates that the model is trained under the training regime shown in the last row in Table 5.

Model	Params (M)	FLOPs (G)	Pixel-wise	Patch-wise	Top-1 Acc (%)
SeT-16	25.5	4.1	[✗,✗,✗,✗]	[✗,✗,✗,✗]	76.9
	24.8	4.2	[✗,✗,✓,✓]	[✗,✗,✓,✓]	78.1 (**+1.2**)
	24.2	4.1	[✓,✓,✓,✓]	[✗,✗,✓,✓]	78.2 (**+1.3**)
	19.2	3.8	[✓,✓,✓,✓]	[✗,✗,✗,✗]	77.5 (**+0.6**)
	24.7	4.4	[✓,✓,✓,✓]	[✓,✓,✓,✓]	75.6 *
SeT-16++	24.2	4.1	[✓,✓,✓,✓]	[✗,✗,✓,✓]	81.3
	24.7	4.4	[✓,✓,✓,✓]	[✓,✓,✓,✓]	81.7

proposed SeT achieves $1.4\% - 1.9\%$ superior to SEResNet. As for the comparison with RegNetY, the top-1 accuracy between these two models are very close.

4.3 SeT is data-efficient compared to ViT/DeiT

As discussed above, SeT is better than ViT in term of accuracy. Here we further show that SeT is more data-efficient compared with the token-based transformers *e.g.* ViT and DeiT. Table 5 exhibits the data augmentations and regularization techniques we used compared to ViT and DeiT. Unlike ViT that pre-trains itself on other extra-large datasets like ImageNet-21k and JFT. SeT is only trained with the data in the training set of ImageNet. Instead of first training on other datasets,

Table 7. Relative positional encoding is significantly better than the absolute positional encoding.

Positional Encoding	Params	FLOPS	Top-1 Acc (%)
Relative	24.2	4.1	80.5
Absolute	24.2	4.0	78.1

DeiT proposes to train their network only using the data of ImageNet and achieves a competitive result by adopting heavy data augmentations and regularization techniques. As shown in Table 5, when compared with DeiT, SeT-16 can achieve even more competitive results on ImageNet trained with much fewer data augmentations, and techniques for regularization. The last row of Table 5 show that the performance of SeT can be further improved using those bells and whistles like Mixup [340] and Auto-augmentation [61] and SiLU [226]. Concretely, tricks including repeated augmentation [121], random augmentation [62], CutMix [332], random erase [351] and color jittering [159] are used for improving DeiT but are not applied to SeT. We also notice that SeT can be well-tuned by SGD while DeiT or ViT can only be well-optimized using Adam [146]. This demonstrates that SeT is more data-efficient than the token-based transformers, which further implies the importance of retaining the local features in visual modeling.

As shown in Table 4, we also found that when applying the factorization proposed in this paper to DeiT-S, our method can further boost the baseline performance with lower computational cost. For all ablation studies in this paper we propose the training regime in the fourth row of Table 5 if not specified.

4.4 Effects of local and global separable attention

Table 6 shows the effects of applying local and global attention into SeT. As all images we feed into the network are of size 224×224, the size of the feature map (7×7) at the last stage will be smaller than the query patch size we set (8×8). Therefore, the pixel-wise local attentions here at the last stage are equivalent to the

Table 8. Average Precision (%) of instance segmentation and object detection with Mask RCNN on MS COCO *val-2017* . AP^m_* and AP^b_* denote the average precision for masks and bounding boxes respectively, and AP^*_S, AP^*_M, AP^*_L denote the AP for small, medium and large objects respectively. + denotes better training regime using Adam optimizer and multi-scale training proposed in [286].

			Instance Segmentation			Object Detection		
Backbone	Params (M)	FLOPs (G)	AP^m	$AP^m_S/AP^m_M/AP^m_L$	AP^m_{50}/AP^m_{75}	AP^b	$AP^b_S/AP^b_M/AP^b_L$	AP^b_{50}/AP^b_{75}
ResNet-50	44.2	180	34.7	18.3/37.4/47.2	55.7/37.2	38.2	21.9/40.9/49.5	58.8/41.4
SeT-16	41.3	196	37.2 (+2.5)	21.6/40.3/49.9	60.1/39.0	40.6 (+2.4)	25.6/44.0/52.6	63.4/43.7
PVT-S+	44.1	230	37.8 (+3.1)	20.4/40.3/53.6	60.1/40.3	40.4 (+2.2)	22.9/43.0/55.4	62.9/43.8
Swin-T+	48	267	39.8 (+5.1)	24.4/43.3/54.3	63.3/42.7	43.7 (+5.5)	28.5/47.0/57.3	66.6/47.7
SeT-16+	41.3	196	39.9 (+5.2)	24.4/43.3/54.3	63.3/42.7	44.0 (+5.8)	29.7/47.8/57.2	66.8/48.0

Table 9. Experimental results for object detection when embedding into RetinaNet. * SAResNet-50 is trained with a longer training schedule for 150 epochs.

Backbone	Params	FLOPs	AP^b	$AP^b_S/AP^b_M/AP^b_L$	AP^b_{50}/AP^b_{75}
ResNet-50	37.7M	239G	36.5	20.4/40.3/48.1	55.4/39.1
SAResNet-50*	-	-	36.6(+0.1)	18.5/40.6/51.6	54.5/39.2
SeT-16	36.9M	255G	39.2(+2.7)	24.4/42.9/50.6	60.3/41.3

global attention as it can capture all the content of the image. However, in this case, we can still apply patch-wise here so that the weights within the patch-wise attention module can be pre-trained for the down-stream tasks *e.g.* object detection and instance segmentation that need to process images at higher resolutions. We note that all results are trained with 120 epoch using the tricks shown in the second row of Table 5. As shown in 6, when compare the third row with the fourth row, the application of the **patch-wise attention in stage** $3, 4$ **leads to** 0.7% **gain on ImageNet**. However, when applying global attention to the shallow stages like Stage 1 and Stage 2 (last row in Table 6), we observe that the network suffers from an under-fitting problem and cannot be well-tuned using the same training regime. When using the Adam optimizer and trained under the training regime shown in the last row of Table 5, our SeT-16 can be well-tuned and achieve its best performance 81.7%. The comparison between the third and the fifth row also reveals that applying global interactions in the early stages can be the reason why

transformer-based method can be hard to be tuned under SGD optimizer. We also observe that SeT can be successfully combined with CNN architectures (the second row in Table 6). If we replace all convolutions within the stage 3, 4 with the separable attention, the network can have an obvious 1.2% gain compared with the baseline.

4.5 Relative vs. Absolute positional encoding

Recent works [12, 224] propose relative positional encoding to incorporate location information into contents. The relative positional encoding Table 7 demonstrates the significance of the relative positional encoding r we proposed in the pixel-wise attention. With just a fractional increase in FLOPS, the relative positional encoding performs 2.4% higher than the absolute one.

5 Experiments on MS COCO

Thanks to the pyramid-style design, SeT can be transferred onto other down-stream tasks that require multi-scale features *e.g.* object detection and instance segmentation. We evaluate SeT on the challenging MS COCO dataset, which consists of 115k images for training (*train-2017*) and 5k images (*val-2017*) for validation. We train models on *train-2017* and report the results on *val-2017*. All reported results follow the official definition of Average Precision (AP) metrics by MS COCO, which includes AP_{50} and AP_{75} (averaged over IoU thresholds) and AP_S, AP_M, AP_L (AP at different scales). Annotations of MS COCO include both bounding boxes and polygon masks for object detection and instance segmentation respectively.

We embed SeT into two popular frameworks. The one is RetinaNet, which is an one-stage detector for object detection. The other is Mask-RCNN, which is a two-stage detector that can do object detection and instance segmentation simultaneously. All modules of SeT except for the positional encoding of the patch-wise attention are first pre-trained on ImageNet before conducting experiment on MS COCO. Note that the activation function we used here is ReLU instead of SiLU.

Integrate SeT into RetinaNet. We first integrate SeT into the popular one-stage detector RetinaNet [177] for object detection. We use SeT as the backbone followed by a Feature Pyramid Network (FPN) [176] to refine the multi-scale features of the network. We take 16 images in one batch for 12 epochs (1×) for training. The learning rate is initially warmed up for 500 steps from 0 to 0.02 and then decayed after 8 and 11 epochs by a factor of 0.1.

Table 9 compare set-16 with the other two baselines ResNet-50 and SAResNet-50. Despite the longer training schedule of SAResNet (150 epochs), SeT still outperforms SAResNet by a clear 2.6%. When comparing SeT with ResNet-50, SeT achieves a significant 1.8%. We also notice that SeT is remarkably better in detecting small objects. To be specific, in terms of the AP of small objects (AP^b_S), SeT performs 4.0% better than ResNet-50, and 5.9% better than SAResNet. The improvement compared with these two baselines validate the efficacy of the global attention.

Integrate SeT into Mask-RCNN. We further embed SeT into the two-stage framework Mask-RCNN for object detection and instance segmentation. Similar to what proposed in RetinaNet, we also integrate the features of SeT into FPN for multi-scale interactions. All models reported in Table 8 are trained under a 1× scheme. As shown in Table 8, SeT is 2.5% higher than ResNet-50 for instance segmentation and 2.4% higher for object detection. This means that the multi-scale features extracted from SeT can be generalized onto the down-stream tasks that require precise modeling of the input images. We also train our network under a better training regime for transformers proposed in [286] to further validate the efficacy of our network. As shown in Table 8, under the same training regime (denoted with a + behind), our network SeT-16 can outperform PVT by a significant 2.1% for instance segmentation and 3.6% for object detection. It can also slightly outperform Swin-T as shown in Table 8. This validates the importance of preserving the local features for the input image.

In conclusion, the success on object detection and instance segmentation demon-

strates the transferability of SeT to down-stream tasks. We thereby draw the conclusion that SeT can be a promising approach to be used as an alternative to the conventional CNN.

6 Conclusion

In this paper, we present SeT that factorizes the original full spatial self-attention into pixel-wise local attention and patch-wise global attention. Such factorization largely reduces the computational cost while retaining the information of different granularity, which helps generate multi-scale features required by different tasks. Extensive experiments demonstrate that SeT performs better than the state-of-the-art transformer-based approaches on ImageNet, and can be well transferred to other down-stream tasks *e.g.* object detection and instance segmentation.

Acknowledgement. This work is supported by the UKRI grant: Turing AI Fellowship EP/W002981/1 and EPSRC/MURI grant: EP/N019474/1. We would also like to thank the Royal Academy of Engineering and FiveAI.

Chapter 3

Vision Transformer with Progressive Sampling

The paper has been accepted for publication at the International Conference on Computer Vision (ICCV), 2021

Vision Transformer with Progressive Sampling

Xiaoyu Yue*[1] Shuyang Sun*[2] Zhanghui Kuang[3] Meng Wei[4]

Philip Torr[2] Wayne Zhang[3,6] Dahua Lin[1,5]

[1]Centre for Perceptual and Interactive Intelligence [2]University of Oxford

[3]SenseTime Research [4]Tsinghua University [5]The Chinese University of Hong Kong

[6]Shanghai Jiao Tong University

xyyue@cpii.hk kevinsun@robots.ox.ac.uk

https://github.com/yuexy/PS-ViT

Abstract

Transformers with powerful global relation modeling abilities have been introduced to fundamental computer vision tasks recently. As a typical example, the Vision Transformer (ViT) directly applies a pure transformer architecture on image classification, by simply splitting images into tokens with a fixed length, and employing transformers to learn relations between these tokens. However, such naive tokenization could destruct object structures, assign grids to uninterested regions such as background, and introduce interference signals. To mitigate the above issues, in this paper, we propose an iterative and progressive sampling strategy to locate discriminative regions. At each iteration, embeddings of the current sampling step are fed into a transformer encoder layer, and a group of sampling offsets is predicted to update the sampling locations for the next step. The progressive sampling is differentiable. When combined with the Vision Transformer, the obtained PS-ViT network can adaptively learn where to look. The proposed PS-ViT is both effective and efficient. When trained from scratch on ImageNet, PS-ViT performs 3.8% higher than the vanilla ViT in terms of top-1 accuracy with about 4× fewer parameters and 10× fewer FLOPs.

*The first two authors contribute equally to this work.

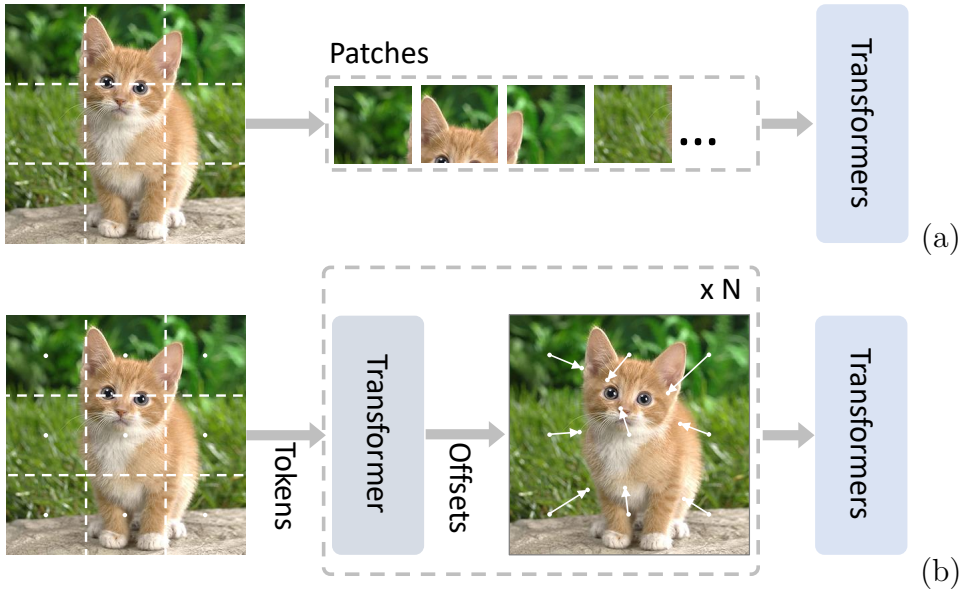


Figure 1. Comparison between the naive tokenization scheme in ViT [79] and the progressive sampling in our proposed PS-ViT. (a) The naive tokenization scheme generates a sequence of image patches which are embedded and then fed into a stack of transformers. (b) Our PS-ViT iteratively samples discriminative locations. $\times N$ indicates N sampling iterations.

1 Introduction

Transformers [75, 274] have become the de-facto standard architecture for natural language processing tasks. Thanks to their powerful global relation modeling abilities, researchers attempt to introduce them to fundamental computer vision tasks such as image classification [42, 79, 225, 266, 297], object detection [25, 67, 259, 349, 360] and image segmentation [281] recently. However, transformers are initially tailored for processing mid-size sequences, and of quadratic computational complexity w.r.t. the sequence length. Thus, they cannot directly be used to process images with massive pixels.

To overcome the computational complexity issue, the pioneer Vision Transformer (ViT) [79] adopts a naive tokenization scheme that partitions one image into a sequence of regularly spaced patches, which are linearly projected into tokens. In this way, the image is converted into hundreds of visual tokens, which are fed into a stack of transformer encoder layers for classification. ViT attains excellent results, espe-

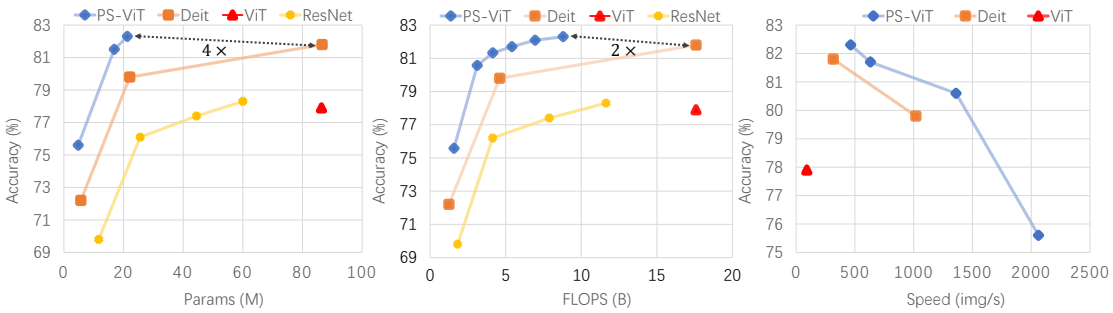


Figure 2. Comparisons between PS-ViT with state-of-the-art networks in terms of top-1 accuracy on ImageNet, parameter number, FLOPs, and speed. The chart on the left, middle and right show top-1 accuracy *vs.* parameter numbers, FLOPs and speed respectively. The speed is tested on the same V100 with a batch size of 128 for fair comparison.

cially when pre-trained on large-scale datasets, which proves that full-transformer architecture is a promising alternative for vision tasks. However, the limitations of such a naive tokenization scheme are obvious. First, the hard splitting might separate some highly semantically correlated regions that should be modeled with the same group of parameters, which destructs inherent object structures and makes the input patches to be less informative. Figure 1 (a) shows that the cat head is divided into several parts, resulting in recognition challenges based on one part only. Second, tokens are placed on regular grids irrespective of the underlying image content. Figure 1 (a) shows that most grids focus on the uninterested background, which might lead to the interesting foreground object is submerged in interference signals.

The human vision system organizes visual information in a completely different way than indiscriminately processing a whole scene at once. Instead, it progressively and selectively focuses attention on interesting parts of the visual space when and where it is needed while ignoring uninterested parts, combining information from different fixations over time to understand the scene [231].

Inspired by the procedure above, we propose a novel transformer-based progressive sampling module, which is able to learn where to look in images, to mitigate the issues caused by the naive tokenization scheme in ViT [79]. Instead of sampling from fixed locations, our proposed module updates the sampling locations in an iterative manner. As shown in Figure 1 (b), at each iteration, tokens of the current

sampling step are fed into a transformer encoder layer, and a group of sampling offsets is predicted to update the sampling locations for the next step. This mechanism utilizes the capabilities of the transformer to capture global information to estimate offsets towards regions of interest, by combining with the local contexts and the positions of current tokens. In this way, attention progressively converges to discriminative regions of images step by step as what human vision does. Our proposed progressive sampling is differentiable, and readily plugged into ViT instead of the hard splitting, to construct end-to-end Vision Transforms with Progressive Sampling Networks dubbed as PS-ViT. Thanks to task-driven training, PS-ViT tends to sample object regions correlated with semantic structures. Moreover, it pays more attention to foreground objects while less to ambiguous background compared with simple tokenization.

The proposed PS-ViT outperforms the current state-of-the-art transformer-based approaches when trained from scratch on ImageNet. Concretely, it achieves 82.3% top-1 accuracy on ImageNet which is higher than that of the recent ViT's variant DeiT [266] with only about 4× fewer parameters and 2× fewer FLOPs. As shown in Figure 2, we observe that PS-ViT is remarkably better, faster, and more parameter-efficient compared to state-of-the-art transformer-based networks ViT and DeiT.

2 Related Work

Transformers are first proposed for sequence models such as machine translation [274]. Benefiting from their powerful global relation modeling abilities, and highly efficient training, transformers achieve significant improvements and become the de-facto standard in many Natural Language Processing (NLP) tasks [16, 75, 217, 221].

Transformers in Computer Vision. Inspired by the success of transformers in NLP tasks, many researchers attempt to apply transformers, or attention mechanism in computer vision tasks, such as image classification [13, 42, 79, 128, 225, 256, 266, 297], object detection [25, 67, 259, 330, 349, 360], image segmentation [281], low-

level image processing [29, 215, 315], video understanding [291] generation [294], multi-modality understanding [44, 167, 251], and Optical Character Recognition (OCR) [240, 285, 329]. Transformers' powerful modelling capacity comes at the cost of computational complexity. Their consumed memory and computation grow quadratically w.r.t. the token length, which prevents them to being directly applied to images with massive pixels as tokens. Axial attention [120] applied attention along a single axis of the tensor without flattening to reduce the computational resource requirement. iGPT [42] simply down-sampled images to one low resolution, trained a sequence of transformers to auto-regressively predict pixels and achieved promising performance with a linear probe. ViT [79] regularly partitioned one high-resolution image into 16×16 patches, which were fed into one pure transformer architecture for classification, and attained excellent results even compared to state-of-the-art convolutional networks for the first time. However, ViT needs pretraining on large-scale datasets, thereby limiting their adoption. DeiT [266] proposed a data-efficient training strategy and a teacher-student distillation mechanism [117], and improved ViT's performance greatly. Moreover, it is trained on ImageNet only, and thus considerably simplifies the overall pipeline of ViT. Our proposed PS-ViT also starts from ViT. Instead of splitting pixels into a small number of visual tokens, we propose a novel progressive sampling strategy to avoid structure destruction and focus more attention on interesting regions.

Hard Visual Attention. PS-ViT as a series of glimpses akin to hard visual attention [5, 82, 205, 313], makes decisions based on a subset of locations only in the input image. However, PS-ViT is differentiable and can be easily trained in an end-to-end fashion while previous hard visual attention approaches are non-differentiable and trained with Reinforcement Learning (RL) methods. These RL-based methods have proven to be less effective when scaled onto more complex datasets [82]. Moreover, our PS-ViT targets at progressively sampling discriminative tokens for Vision Transformers while previous approaches locate interested regions for convolutional neural networks [5, 82, 205] or sequence decoders [313]. Our work is also related

Name	Description
$\mathbf{F} \in \mathbb{R}^{C\times H\times W}$	the feature map extracted by the feature extractor module
$\mathbf{p}_t \in \mathbb{R}^{2\times(n\times n)}$	the sampling points at the iteration t
$\mathbf{P}_t \in \mathbb{R}^{C\times(n\times n)}$	the position embeddings at the iteration t
$\mathbf{o}_t \in \mathbb{R}^{2\times(n\times n)}$	the sampling offsets at the iteration t
$\mathbf{T}'_t \in \mathbb{R}^{C\times(n\times n)}$	tokens sampled from $\mathbf{F}$ at the iteration t
$\mathbf{T}_t \in \mathbb{R}^{C\times(n\times n)}$	tokens predicted by the progressive sampling module at the iteration t

Table 1. The list of symbols and notations used in this paper.

to the deformable convolution [65, 358] and deformable attention [359] mechanism, however, the motivation and the way of pixel sampling in this work are different from what proposed in the deformable convolution and attention mechanism.

3 Methodology

In this section, we first introduce our progressive sampling strategy and then describe the overall architecture of our proposed PS-ViT network. Finally, we will elaborate on the details of PS-ViT. Symbols and notations of our method are presented in Table 1.

3.1 Progressive Sampling

ViT [79] regularly partitions one image into 16×16 patches, which are linearly projected into a set of tokens, regardless of the content importance of image regions and the integral structure of objects. To pay more attention to interesting regions of images and mitigate the problem of structure destruction, we propose one novel progressive sampling module. As it is differentiable, it is adaptively driven via the following vision transformer based image classification task.

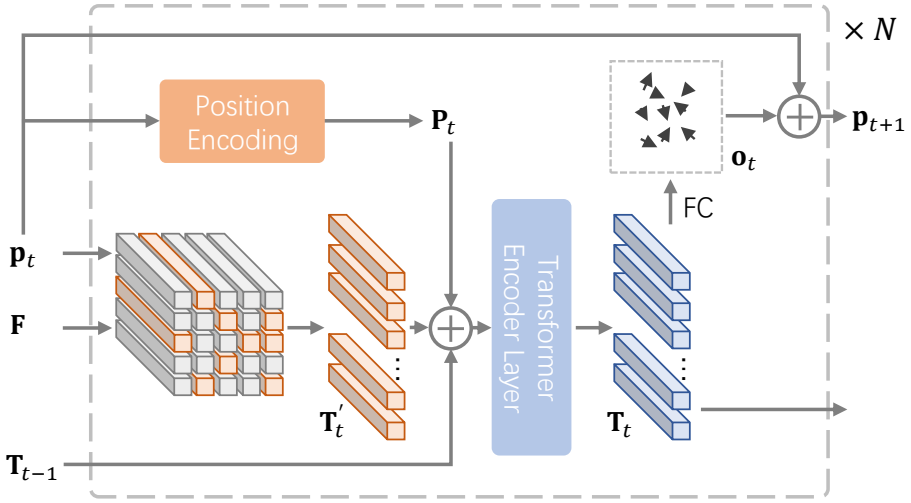


Figure 3. The architecture of the progressive sampling module. At each iteration, given the sampling location $\mathbf{p}_t$ and the feature map $\mathbf{F}$, we sample the initial tokens $\mathbf{T}_t^{'}$ at $\mathbf{p}_t$ over $\mathbf{F}$, which are element-wisely added with the positional encodings $\mathbf{P}_t$ generated based on $\mathbf{p}_t$, and the output tokens $\mathbf{T}_{t-1}$ of the last iteration, and then fed into one transformation encoder layer to predict the tokens $\mathbf{T}_t$ of the current iteration. The offset matrix $\mathbf{o}_t$ is predicted via one fully-connected layer based on $\mathbf{T}_t$, which is added with $\mathbf{p}_t$ to obtain the sampling positions $\mathbf{p}_{t+1}$ for the next iteration. The above procedure is iterated N times.

Our progressive sampling module is an iterative framework. Given the input feature map $\mathbf{F} \in \mathbb{R}^{C\times H\times W}$ with C, H, and W being the feature channel dimension, height and width respectively, it finally outputs a sequence of tokens $\mathbf{T}_N \in \mathbb{R}^{C\times(n\times n)}$, where $(n \times n)$ indicates the number of samples over one image and N is the total iterative number in the progressive sampling module.

As shown in Figure 3, at each iteration, the sampling locations are updated via adding them with the offset vectors of the last iteration. Formally,

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \mathbf{o}_t,\ t \in \{1, \ldots, N-1\}, \tag{3.1}$$

where $\mathbf{p}_t \in \mathbb{R}^{2\times(n\times n)}$, and $\mathbf{o}_t \in \mathbb{R}^{2\times(n\times n)}$ indicate the sampling location matrix and the offset matrix predicted at the iteration t. For the first iteration, we initialize the $\mathbf{p}_1$ to be the regularly-spaced locations as done in ViT [79]. Concretely, the i-th

location $\mathbf{p}_1^i$ is given by

$$\begin{aligned}
\mathbf{p}_1^i &= [\pi_i^y s_h + s_h/2, \pi_i^x s_w + s_w/2] \\
\pi_i^y &= \lfloor i/n \rfloor \\
\pi_i^x &= i - \pi_i^y * n \\
s_h &= H/n \\
s_w &= W/n,
\end{aligned} \tag{3.2}$$

where π_i^y and π_i^x map the location index i to the row index and the column one respectively. $\lfloor \cdot \rfloor$ indicates the floor operation. s_h and s_w are the step size in the y and x axial direction respectively. Initial tokens are then sampled over the input feature map at the sampled locations as follows:

$$\mathbf{T}_t^{'} = \mathbf{F}(\mathbf{p}_t),\ t \in \{1, \ldots, N\}, \tag{3.3}$$

where $\mathbf{T}_t^{'} \in \mathbb{R}^{C \times (n \times n)}$ is the initial sampled tokens at the iteration t. As elements of $\mathbf{p}_t$ are fractional, the sampling is implemented via the bilinear interpolation operation, which is differentiable w.r.t. both the input feature map $\mathbf{F}$ and the sampling locations $\mathbf{p}_t$. The initial sampled tokens, the output tokens of the last iteration, and the positional encodings of the current sampling locations are further element-wisely added before being fed into one transformer encoder layer to get the output tokens of the current iteration. Formally, we have

$$\begin{aligned}
\mathbf{P}_t &= \mathbf{W}_t \mathbf{p}_t \\
\mathbf{X}_t &= \mathbf{T}_t^{'} \oplus \mathbf{P}_t \oplus \mathbf{T}_{t-1} \\
\mathbf{T}_t &= \text{Transformer}(\mathbf{X}_t),\ t \in \{1, \ldots, N\},
\end{aligned} \tag{3.4}$$

where $\mathbf{W}_t \in \mathbb{R}^{C \times 2}$ is the linear transformation that projects the sampled locations $\mathbf{p}_t$ to the positional encoding matrix $\mathbf{P}_t$ of size $C \times (n \times n)$, all iterations share the same $\mathbf{W}_t$. $\oplus$ indicates the element-wise addition while Transformer$(\cdot)$ is the mulit-

head self-attention based transformer encoder layer, which will be elaborated in Section 3.3. Note that $\mathbf{T}_0$ is a zero matrix in Equation (3.4). ViT [79] uses the 2-D sinusoidal positional embeddings of patch indices. Since their patches are regularly spaced, patch indices can exactly encode the relative coordinates of patch centers in one image. However, it does not hold true in our case as our sampled locations are non-isometric as shown in Figure 1. We project the normalized absolute coordinates of sampled locations to one embedding space as the positional embeddings instead. Finally, the sampling location offsets are predicted for the next iteration except at the last iteration as follows:

$$\mathbf{o}_t = \mathbf{M}_t \mathbf{T}_t, \ t \in \{1, \ldots, N-1\}, \tag{3.5}$$

where $\mathbf{M}_t \in \mathbb{R}^{2 \times C}$ is the learnable linear transformation for predicting the sampling offset matrix.

With the progressive sampling strategy, the sampled locations progressively converge to interesting regions of images. Therefore, we name it by progressive sampling.

3.2 Overall Architecture

As shown in Figure 4, the architecture of the PS-ViT consists of four main components: 1) a feature extractor module to predict dense tokens; 2) a progressive sampling module to sample discriminative locations; 3) a vision transformer module that follows the similar configuration of ViT [79] and DeiT [266]; 4) a classification module.

The feature extractor module aims at extracting the dense feature map $\mathbf{F}$, where the progressive sampling module can simple tokens $\mathbf{T}_t$. Each pixel of the dense feature map $\mathbf{F}$ can be treated as a token associated with a patch of the image. We employ the convolutional stem and the first two residual blocks in the first stage of the ResNet50 [102] as our feature extractor module because the convolution operator is

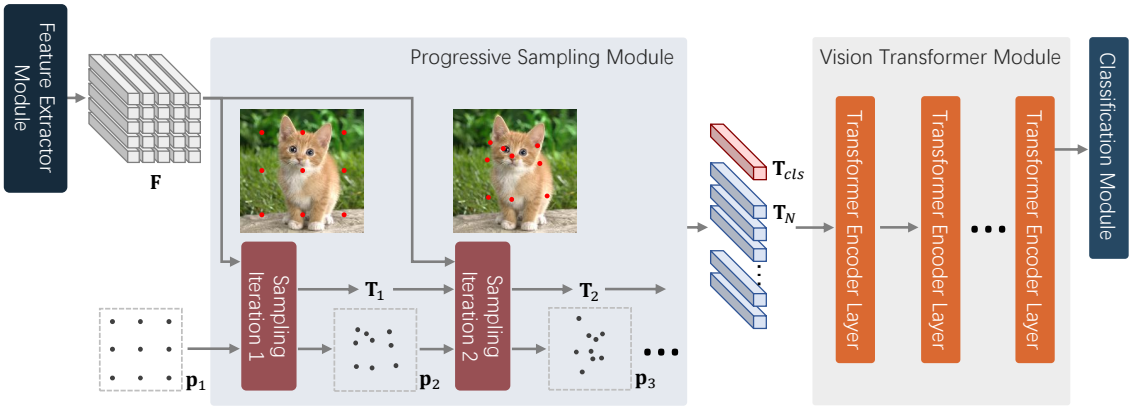


Figure 4. Overall architecture of the proposed Progressive Sampling Vision Transformer (PS-ViT). Given an input image, its feature map $\mathbf{F}$ is first extracted by the feature extractor module. Tokens $\mathbf{T}_i$ are then sampled progressively and iteratively at adaptive locations $\mathbf{p}_i$ over $\mathbf{F}$ in the progressive sampling module. The final output tokens $\mathbf{T}_N$ of the progressive sampling module are padded with the classification token $\mathbf{T}_{cls}$ and further fed into the vision transformer module to refine $\mathbf{T}_{cls}$, which is finally classified in the classification module.

especially effective at modeling spatially local contexts.

The vision transformer module follows the architecture adopted in ViT [79] and DeiT [266]. We pad an extra token named by the classification token $\mathbf{T}_{cls} \in \mathbb{R}^{C\times 1}$ to the output tokens $\mathbf{T}_N$ of the last iteration in the progressive sampling module, and fed them into the vision transformer module. Formally,

$$\overline{\mathbf{T}} = \mathrm{VTM}([\mathbf{T}_{cls}, \mathbf{T}_N]), \tag{3.6}$$

where VTM indicates the vision transformer module function which is a stack of transformer encoder layers, and $\overline{\mathbf{T}} \in \mathbb{R}^{C\times(n\times n+1)}$ is the output. Note that, the positional information has been fused into $\mathbf{T}_N$ in the progressive sampling module, so we do not need to add positional embedding here. The classification token refined through the vision transformer module is finally used to predict the image classes. We use the cross entropy loss to end-to-end train the proposed PS-ViT network.

Networks	N	N_v	C	M	#params	#FLOPs
PS-ViT-Ti	4	8	192	3	4.7 M	1.6 B
PS-ViT-Ti†	4	8	192	3	3.6 M	1.6 B
PS-ViT-B	4	10	384	6	21.3 M	5.4 B
PS-ViT-B†	4	10	384	6	16.9 M	5.4 B

Table 2. PS-ViT configurations. † indicates weight sharing between different iterations in the Progressive Sampling Module (PSM). N, N_v, C and M are the iteration number in PSM, the transformer encoder layer number in the vision transformer module, the dimension of tokens, and the head number in each transformer respectively.

3.3 Implementation

Transformer Encoder Layer. The transformer encoder layer serves as the basic building block for the progressive sampling module and the vision transformer module. Each transformer encoder layer has a *multi-head self-attention* and a *feed-forward unit*.

Given queries $\mathbf{Q} \in \mathbb{R}^{D\times L}$, keys $\mathbf{K} \in \mathbb{R}^{D\times L}$ and values $\mathbf{V} \in \mathbb{R}^{D\times L}$ with D being the dimension and L being the sequence length, the scaled dot-product self attention can be calculated as:

$$\text{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\mathbf{Q}^T\mathbf{K}/\sqrt{D})\mathbf{V}^T, \tag{3.7}$$

where $\mathbf{Q}^T$ indicates the transpose of $\mathbf{Q}$, and softmax$(\cdot)$ is the softmax operation applied over each row of the input matrix. For Multi-Head self-Attention (MHA), the queries, keys and values are generated via linear transformations on the inputs for M times with one individual learned weight for each head. Then attention function is applied in parallel on queries, keys and values of each head. Formally,

$$\begin{aligned} \text{MHA}(\mathbf{Z}) &= \mathbf{W}^o[\mathbf{H}_1, \ldots, \mathbf{H}_M]^T, \\ \mathbf{H}_i &= \text{Attn}(\mathbf{W}_i^Q\mathbf{Z}, \mathbf{W}_i^K\mathbf{Z}, \mathbf{W}_i^V\mathbf{Z}), \end{aligned} \tag{3.8}$$

Model	Image size	Params (M)	FLOPs (B)	Top-1 (%)	Top-5 (%)
CNN-based					
R-18 [102]	224^2	11.7	1.8	69.8	89.1
R-50 [102]	224^2	25.6	4.1	76.1	92.9
R-101 [102]	224^2	44.5	7.9	77.4	93.5
X-50-32×4d [307]	224^2	25.0	4.3	79.3	94.5
X-101-32×4d [307]	224^2	44.2	8.0	80.3	95.1
RegNetY-4GF [223]	224^2	20.6	4.0	79.4	-
RegNetY-6.4GF [223]	224^2	30.6	6.4	79.9	-
RegNetY-16GF [223]	224^2	83.6	15.9	80.4	-
Transformer-based					
ViT-B/16 [79]	384^2	86.4	55.5	77.9	-
DeiT-Ti [266]	224^2	5.7	1.3	72.2	-
DeiT-S [266]	224^2	22.1	4.6	79.8	-
DeiT-B [266]	224^2	86.4	17.6	81.8	-
PS-ViT-Ti/14	224^2	4.8	1.6	75.6	92.9
PS-ViT-B/10	224^2	21.3	3.1	80.6	95.2
PS-ViT-B/14	224^2	21.3	5.4	81.7	95.8
PS-ViT-B/18	224^2	21.3	8.8	82.3	96.1

Table 3. Comparison with state-of-the-art networks on ImageNet with single center crop testing. The number after "/" is the sampling number in each axial direction. *e.g.*, PS-ViT-Ti/14 indicates PS-ViT-Ti with 14×14 sampling locations.

where $\mathbf{W}^o \in \mathbb{R}^{\frac{C}{M} \times C}$ is a learnable linear projection. $\mathbf{W}_i^Q \in \mathbb{R}^{\frac{C}{M} \times C}$, $\mathbf{W}_i^K \in \mathbb{R}^{\frac{C}{M} \times C}$ and $\mathbf{W}_i^V \in \mathbb{R}^{\frac{C}{M} \times C}$ are the linear projections for the queries, keys and values of the i-th head respectively.

The feed-forward unit of the transformer encoder layer consists of two fully connected layers with one GELU non-linear activation [109] between them and the latent variable dimension being $3C$. For simplicity, the transformer encoder layers in both the progressive sampling module and the vision transformer module keep the same settings.

Progressive Sampling Back-propagation. The back-propagation of the progressive sampling is straightforward. According to Equation (3.1) and Equation (3.3), for each sampling location i, the gradient w.r.t. the sampling offsets $\mathbf{o}_t^i$ at the iteration t is computed as:

$$\begin{aligned}\frac{\partial \mathbf{T}_t'^i}{\partial \mathbf{o}_{t-1}^i} &= \frac{\partial \mathbf{F}(\mathbf{p}_{t-1}^i + \mathbf{o}_{t-1}^i)}{\partial \mathbf{o}_{t-1}^i} \\ &= \sum_{\mathbf{q}} \frac{\partial K(\mathbf{q}, \mathbf{p}_{t-1}^i + \mathbf{o}_{t-1}^i)}{\partial \mathbf{o}_{t-1}^i} \mathbf{F}(\mathbf{q}),\end{aligned} \tag{3.9}$$

where $K(\cdot, \cdot)$ is the kernel for bilinear interpolation to calculate weights for each integral spatial location $\mathbf{q}$.

Network Configuration. The feature dimension C, the iteration number N in the progressive sampling module, the vision transformer layer number N_v in the vision transformer module, and the head number M in each transformer layer affect the model size, FLOPs, and performances. In this paper, we configure them with different speed-performance tradeoffs in Table 2 so that the proposed PS-ViT can be used in different application scenarios. The number of sampling points along each spatial dimension n is set as 14 by default.

Considering the sampling in each iteration is conducted over the same feature map $\mathbf{F}$ in the progressive sampling module, we try to share weights between those iterations to further reduce the number of trainable parameters. As shown in Table 2, about 25% parameters can be saved in this setting.

4 Experiments

4.1 Experimental Details on ImageNet

All the experiments for image classification are conducted on the ImageNet 2012 dataset [160] that includes 1k classes, 1.2 million images for training, and 50 thousand images for validation. We train our proposed PS-ViT on ImageNet without pretraining on large-scale datasets. We train all the models of PS-ViT using PyTorch [216] with 8 GPUs. Inspired by the data-efficient training as done in [266], we use the AdamW [194] as the optimizer. The total training epoch number and the batch size are set to 300 and 512 respectively. The learning rate is initialized with

Epochs	300
Optimizer	AdamW
Batch size	512
Learning rate	0.0005
Learning rate decay	cosine
Weight decay	0.05
Warmup epochs	5
Label smooth	0.1
Dropout	0.1
Rand Augment	(9, 0.5)
Mixup probability	0.8
CutMix probability	1.0

Table 4. Training strategy and hyper-parameter settings.

0.0005, and decays with the cosine annealing schedule [195]. We regularize the loss via the smoothing label with $\epsilon = 0.1$. We use random crop, Rand-Augment [63], Mixup[341], and CutMix [333] to augment images during training. Images are resized to 256×256, and cropped at the center with 224×224 size when testing. Training strategy and its hyper-parameter settings are summarized in Table 4.

4.2 Results on ImageNet

We compare our proposed PS-ViT with state-of-the-art networks on the standard image classification benchmark ImageNet in terms of parameter numbers, FLOPS, and top-1 and top-5 accuracies in Table 3.

Comparison with CNN based networks. Our PS-ViTs considerably outperform ResNets [102] while with much fewer parameters and FLOPs. Specifically, Compared with ResNet-18, PS-ViT-Ti/14 absolutely improves the top-1 accuracy by 5.8% while reducing 6.9 M parameters and 0.2 B FLOPs. We can observe a similar trend when comparing PS-ViT-B/10 (PS-ViT-B/14) and ResNet-50 (ResNet-101). Our proposed PS-ViT achieves superior performance and computational efficiency when compared with the state-of-the-art CNN based network RegNet [223]. Particularly, when compared with RegNetY-16GF, PS-ViT-B/18 improves the top-1 accuracy by

n	Params (M)	FLOPs (B)	Top-1 (%)	Top-5 (%)
10	21.3	3.1	80.6	95.2
12		4.2	81.3	95.5
14		5.4	81.7	95.8
16		7.0	82.1	95.8
18		8.8	82.3	96.1

Table 5. Effect of the sampling number n in each axial direction.

1.9% with about a quarter of parameters and a half of FLOPS.

Comparison with transformer based networks. Table 3 shows that our proposed PS-ViT outperforms ViT [79] and its recent variant DeiT [266]. In particular, PS-ViT-B/18 achieves 82.3% top-1 accuracy which is 0.5% higher than the baseline model DeiT-B while with 21 M parameters and 8.8 B FLOPs only. Our performance gain attributes to two parts. First, PS-ViT samples CNN-based tokens which is more efficient than raw image patches used in ViT [79] and DeiT [266]. Second, our progressive sampling module can adaptively focus on regions of interest and produce more semantically correlated tokens than the naive tokenization used in [79, 266].

4.3 Ablation Studies

The PS-ViT models predict on the class token in all the ablation studies.

A larger sampling number n leads to better performance. We first evaluate how the sampling number parameter n affects the PS-ViT performance. The sequence length of sampled tokens which is fed into the vision transformer module is n^2. The more the sampled tokens, the more information PS-ViT can extract. However, sampling more tokens would increase the computation and memory usage. Table 5 reports the FLOPs, and top-1 and top-5 accuracies with different n. It has been shown that the FlOPs increases as n becomes larger, and the accuracy increases when $n \leq 16$ and plateaus when $n > 16$. Considering the speed-accuracy trade-off, we set $n = 14$ by default except as otherwise noted.

Model	N	Top-1 (%)	Top-5 (%)
PS-ViT-B/14	1	80.6	95.3
	2	81.5	95.6
	4	81.7	95.8
	6	81.8	95.7
	8	81.9	95.7
	9	81.7	95.8
	10	81.6	95.8

Table 6. Effect of the iteration number N in the progressive sampling module.

The performance can be further improved with more iterations of progressive sampling. We then evaluate the effect of the iteration number N of the progressive sampling module in Table 6. To keep the computational complexity unchanged, all models in Table 6 have $14 - N$ transformer layers in the vision transformer module, and totally 14 transformer layers in the entire network. $N = 1$ indicates the sampling points will not be updated. It has been shown that PS-ViT performs the best when $N = 8$ and the accuracy begins to decline when $N > 8$. As we keep the total number of transformer layers unchanged, increasing N will result in the decrease of transformer layers in lateral modeling, which might damage the performance. Considering the accuracy improvement is negligible from $N = 4$ to $N = 8$, we set $N = 4$ by default except as otherwise noted.

Fair comparison with ViT. The network hyper-parameters in the transformer encoder of PS-ViT are different from the original setting of ViT. For a fair comparison, we further study how ViT performs when the network hyper-parameters are set to be the same as ours. We set the number of layers, channels, heads, and the number of tokens to be the same as what was proposed in PS-ViT-B/14, and train the network under the same training regime. As shown in Table 7, ViT achieves 78.4% top-1 accuracy, which is greatly inferior to its PS-ViT counterpart. We thereby conclude that the progressive sampling module can fairly boost the performance of ViT.

	Top-1 (%)	Top-5 (%)
ViT∗	78.4	94.1
PS-ViT-B/14	81.7	95.8

Table 7. Comparison between our PS-ViT with ViT. ∗ means the model with the same model configuration and training strategy.

Figure 5. Visualization of sampled locations in the proposed progressive sampling module. The start points of arrows are initial sampled locations ($\mathbf{p}_1$) while the end points of arrows are the final sampled locations ($\mathbf{p}_4$).

Sharing weights between sampling iterations. Model size (parameter number) is one of the key factors when deploying deep models on terminal devices. Our proposed PS-ViT is very terminal device friendly as it can share weights in the progressive sampling module with a negligible performance drop. Table 8 compares PS-ViT with and without weight sharing in the progressive sampling module. It has been shown that weight sharing can reduce the parameter number by about 21%∼23% while with a slight performance drop, especially for PS-ViT-B/12 and PS-ViT-B/14.

4.4 Speed Comparison

Our proposed PS-ViT is efficient not only in theory but also in practice. Table 9 compare the efficiency of state-of-the-arts networks in terms of FLOPs and speed (images per second). For fair comparison, we measure the speed of all of the models on a server with one 32GB V100 GPU. The batch size is fixed to 128 and the number of images that can be inferred per second is reported averaged over 50 runs. It has

Model	Params (M)	Top-1 (%)	Top-5 (%)
PS-ViT-Ti/14	4.8	75.6	92.9
PS-ViT-Ti†/14	3.7	74.1	92.3
PS-ViT-B/10	21.3	80.6	95.2
PS-ViT-B†/10	16.9	80.0	94.8
PS-ViT-B/12	21.3	81.3	95.5
PS-ViT-B†/12	16.9	80.9	95.3
PS-ViT-B/14	21.3	81.7	95.8
PS-ViT-B†/14	16.9	81.5	95.6

Table 8. Comparison PS-ViT with and without weight sharing in the progressive sampling module. † indicates weight sharing.

Model	FLOPs (B)	Speed (img/s)	Top-1
RegNetY-4.0GF	4.0	1097.6	79.4
RegNetY-6.4GF	6.4	487.0	79.9
RegNetY-16GF	15.9	351.0	80.4
ViT-B/16	55.5	92.4	77.9
DeiT-S	4.6	1018.2	79.8
DeiT-B	17.6	316.1	81.8
PS-ViT-Ti/14	1.6	1955.3	75.6
PS-ViT-B/10	3.1	1348.0	80.6
PS-ViT-B/14	5.4	765.6	81.7
PS-ViT-B/18	8.8	463.8	82.3

Table 9. Comparison the efficiency of PS-ViT, and that of state-of-the-art networks in terms of FLOPs and speed.

been shown that PS-ViT is much more efficient than ViT and DeiT when their top-1 accuracies are comparable. Specifically, PS-ViT-B/14 and DeiT-B have similar accuracy around 81.7%. However, PS-ViT-B/14 achieves about 2.4 times and 3.3 times as fast as DeiT-B in terms of speed and FLOPs respectively. PS-ViT-B/10 speeds up ViT-B/16 by about 14.6 times and 17.9 times in terms of speed and FLOPs while improving 2.7% top-1 accuracy.

Model	IM	C10	C100	Flowers	Cars
ViT-B/16	77.9	98.1	87.1	89.5	-
ViT-L/16	76.5	97.9	86.4	89.7	-
DeiT-B	81.8	99.1	90.8	98.4	92.1
PS-ViT-B/14	81.7	99.0	90.8	98.8	92.9

Table 10. Top-1 accuracy on other datasets. ImageNet and CIFAR are abbreviated to "IM" and "C".

4.5 Visualization

In order to explore the mechanism of the learnable sampling locations in our method, we visualize the predicted offsets of our proposed progressive sampling module in Figure 5. We can observe that the sampling locations are adaptively adjusted according to the content of the images. Sampling points around objects tend to move to the foreground area and converge to the key parts of objects. With this mechanism, discriminative regions such as the chicken head are sampled densely, retaining the intrinsic structure information of highly semantically correlated regions.

4.6 Transfer Learning

In addition to ImageNet, we also transfer PS-ViT to downstream tasks to demonstrate its generalization ability. We follow the practice done in DeiT [266] for fair comparison. Table 10 shows results for models that have been pre-trained on ImageNet and finetuned for other datasets including CIFAR-10 [158], CIFAR-100 [158], Flowers-102 [214] and Stanford Cars [157]. PS-ViT-B/14 can perform on-par-with or even better than DeiT-B with about 4× fewer FLOPS and parameters on all these datasets.

5 Conclusions

In this paper, we propose an efficient Vision Transformers with Progressive Sampling (PS-ViT). PS-ViT first extracts feature maps via a feature extractor, and then progressively selects discriminative tokens with one progressive sampling module. The sampled tokens are fed into a vision transformer module and the classification module for image classification. PS-ViT mitigates the structure destruction issue in the ViT and adaptively focuses on interesting regions of objects. It achieves considerable improvement on ImageNet compared with ViT and its recent variant DeiT. We also provide a deeper analysis of the experimental results to investigate the effectiveness of each component. Moreover, PS-ViT is more efficient than its transformer based competitors both in theory and in practice.

Chapter 4

TransMix: Attend to Mix for Vision Transformers

The paper has been accepted for publication at the IEEE Conference on Computer Vision and Pattern Recognition, 2022

Patch-based Separable Transformer for Visual Recognition

Jie-Neng Chen*[1] Shuyang Sun*[2] Ju He[1] Philip Torr[2] Alan Yuille[1] Song Bai[3]

[1]Johns Hopkins University [2]University of Oxford [3]ByteDance Inc.

Abstract

Mixup-based augmentation has been found to be effective for generalizing models during training, especially for Vision Transformers (ViTs) since they can easily overfit. However, previous mixup-based methods have an underlying prior knowledge that the linearly interpolated ratio of targets should be kept the same as the ratio proposed in input interpolation. This may lead to a strange phenomenon that sometimes there is no valid object in the mixed image due to the random process in augmentation but there is still response in the label space. To bridge such gap between the input and label spaces, we propose TransMix, which mixes labels based on the attention maps of Vision Transformers. The confidence of the label will be larger if the corresponding input image is weighted higher by the attention map. TransMix is embarrassingly simple and can be implemented in just a few lines of code without introducing any extra parameters and FLOPs to ViT-based models. Experimental results show that our method can consistently improve various ViT-based models at scales on ImageNet classification. After pre-trained with TransMix on ImageNet, the ViT-based models also demonstrate better transferability to semantic segmentation, object detection and instance segmentation. TransMix also exhibits to be more robust when evaluating on 4 different benchmarks. Code is publicly available at https://github.com/Beckschen/TransMix.

*The first two authors contribute equally to this work.

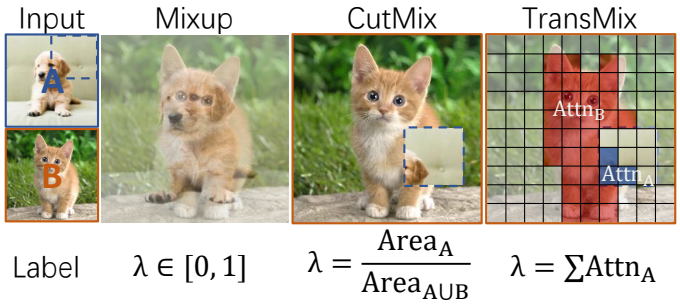


Figure 1. Mixup [340] and CutMix [334] samples λ (proportion of label $\mathbf{y}_A$) randomly from a Beta distribution, while our TransMix calculates λ with the sum of the values within the attention map that intersects with A (denoted as Attn_A, rendered in blue).

1 Introduction

Transformers [273] have been dominant in nearly all tasks in natural language processing. Recently, transformer-based architectures like Vision Transformer (ViT) [80] have been introduced into the field of computer vision and show great promise on tasks like image classification [80, 81, 187, 268], object detection [88, 187, 287] and image segmentation [187, 247, 287]. However, recent works have found that ViT-based networks are hard to optimize and can easily overfit if the training data is not sufficient. A quick solution to this problem is to apply data augmentation and regularization techniques during training. Among them, the mixup-based methods like Mixup [340] and CutMix [334] are proven to be particularly helpful for generalizing the ViT-based network [267].

Mixup takes a pair of inputs $\mathbf{x}_A, \mathbf{x}_B$ and their corresponding labels $\mathbf{y}_A, \mathbf{y}_B$, then creates an artificial training example $\lambda\mathbf{x}_A + (1-\lambda)\mathbf{x}_B$ with $\lambda\mathbf{y}_A + (1-\lambda)\mathbf{y}_B$ as its ground truth. Here $\lambda \in [0, 1]$ is the random mixing proportion sampled from a Beta distribution. This pre-assumes that linear interpolations of feature vectors should lead to linear interpolations of the associated targets.

However, we argue that the above pre-assumption does not always stay true since **not all pixels are created equal**. As shown in Figure 1, pixels in the background will not contribute to the label space as equally as those in the salient area. Some existing works [145, 271, 277] also find this problem and solve it by means of only

mixing the most descriptive parts on the input level. Nevertheless, manipulating on inputs with the above methods may narrow the space of augmentation since they tend to less consider to put the background image into the mixture. Meanwhile, the above methods cost more number of parameters and/or training throughput to extract the salient region of input. For example, Puzzle-Mix [145] requires model to forward and backward twice in an iteration and Attentive-Cutmix [277] introduce a 24M external CNN to extract salient features.

Instead of investigating how to better mix images on the input level, in this paper, we focus more on how to mild the gap between the input and the label space through the learning of label assignment. We find that the attention maps that are naturally generated in Vision Transformers can be well suited for this job. As shown in Figure 1, we simply set λ (weight of $\mathbf{y}_A$) as the sum of weights of attention map lying in A. In this way, the labels are re-weighted by the significance of each pixel instead of linearly interpolated with the same ratio as the mixed inputs. Since the attention map is naturally generated in ViT-based models, our method can be merged into the their training pipeline with no extra parameters and minimal computation overhead.

We show that such frustratingly simple idea can lead to consistent and remarkable improvement for a wide range of tasks and models. As exhibited in Fiugre 2, TransMix can steadily boost all the listed ViT-based models. Notably, TransMix can further lift the top-1 accuracy on ImageNet by 0.9% for both DeiT-S and a large variant XCiT-L. Interestingly, the largest model XCiT-L gains the most among all XCiT model scales.

Moreover, we demonstrate that if the model is first pre-trained with TransMix on ImageNet, the superiority can be further transferred onto downstream tasks including object detection, instance segmentation, semantic segmentation and weakly-supervised object segmentation/localization. We also observe that TransMix can help the model to be more rubust after evaluating it on 4 different benchmarks.

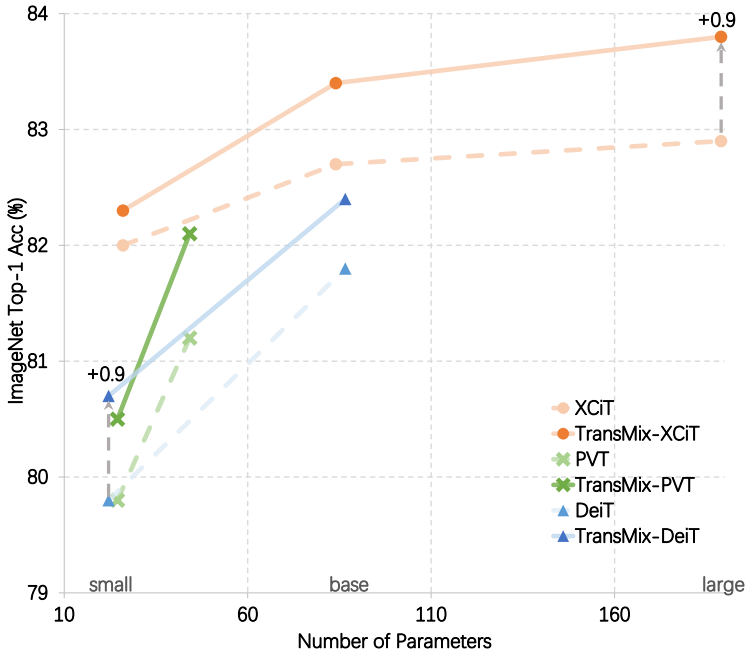


Figure 2. TransMix can steadily improve a wide range of state-of-the-art ViT-based models on ImageNet with no parameter and minimal computation overhead. See results for more model variants in Table 1.

2 Related Work

Vision Transformers (ViTs). Recently, Vision Transformer (ViT) [80] was proposed to adapt the Transformer for image recognition by tokenizing and flattening images into a sequence of tokens. ViT is based on a sequence of Transformer blocks consisting of multi-head self attention layers and feed-forward networks. DeiT [267] strengthens ViT by introducing a powerful training recipe and adopting knowledge distillation. Built upon the success of ViT, many efforts have been devoted to improving ViT and adapting it into various vision tasks including image classification [81, 103, 113, 187, 267, 268, 325], object localization/detection [88, 90, 187, 287] and image segmentation [31, 187, 247, 287].

Mixup and its variants. Data augmentation has been widely studied to prevent DeepNets from over-fitting to the training data. To train and improve vision Transformer stably, Mixup and CutMix are two of the most helpful augmentation methods [267]. Mixup [340] is a successful image mixture technique that obtains an

augmented image by pixel-wisely weighted combination of two global images. The following Mixup variants [97, 111, 145, 239, 271, 276, 277, 334] can be categorized into global image mixture (e.g. Manifold-Mixup [276], Un-Mix [239]) and regional image mixture (e.g. CutMix [334], Puzzle-Mix [145], Attentive-CutMix [277] and SaliencyMix [271]). Among all Mixup variants, the saliency-based methods including the attentive-CutMix, puzzle-Mix and saliency-CutMix are the most similar ones to our approach. However, TransMix has two fundamental differences with them: (1) Previous saliency-based methods *e.g.* [145, 271, 277] enforce the image patch cropped in a salient region of the input image. Instead of manipulating in the input space, our TransMix focuses on how to more accurately assigning labels in the label space. (2) Previous saliency-based methods like [277] may use extra parameters to extract the saliency region. TransMix naturally exploits the Transformer's attention mechanism without any extra parameters. Experimental results also show that TransMix can lead to better results on ImageNet compared with these methods.

Data-adaptive loss weight assignment. TransMix re-assigns the ground truth labels with attentional guidance, which is related to data-adaptive loss weight assignment. Some existing works have found that the attention-like information can help to alleviate the long-tail problems for tasks like point cloud analyzing [197], instance segmentation [283], image demosaicing [254] *etc.*

3 TransMix

3.1 Setup and Background

CutMix data augmentation CutMix is a simple data augmentation technique combining two input-label pairs $(\mathbf{x}_A, \mathbf{y}_A)$ and $(\mathbf{x}_B, \mathbf{y}_B)$ to augment a new training

sample $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})$. Formulaically,

$$\tilde{\mathbf{x}} = \mathbf{M} \odot \mathbf{x}_A + (\mathbf{1} - \mathbf{M}) \odot \mathbf{x}_B, \tag{4.1}$$

$$\tilde{\mathbf{y}} = \lambda \mathbf{y}_A + (1 - \lambda)\mathbf{y}_B, \tag{4.2}$$

where $\mathbf{M} \in \{0, 1\}^{HW}$ denotes a binary mask indicating where to drop out and fill in from two images, $\mathbf{1}$ is a binary mask filled with ones, and $\odot$ is element-wise multiplication. λ is the proportion of $\mathbf{y}_A$ in the mixed label.

During augmentation, a randomly sampled region in $\mathbf{x}_B$ is removed and filled in with the patch cropped from A of $\mathbf{x}_A$, where the patch's bounding box coordinates are uniformly sampled as (r_x, r_y, r_w, r_h). The mixed-target assignment factor λ is equal to the cropped area ratio $\frac{r_w r_h}{WH}$.

Self-attention Self-attention, as introduced by [275], operates on an input matrix $\mathbf{x} \in \mathbb{R}^{N \times d}$, where N is the number of tokens, each of dimensionality d. The input $\mathbf{x}$ is linearly projected to queries, keys and values, using the weight matrices $\mathbf{w}_q \in \mathbb{R}^{d \times d_q}$, $\mathbf{w}_k \in \mathbb{R}^{d \times d_k}$ and $\mathbf{w}_v \in \mathbb{R}^{d \times d_v}$, such that $\mathbf{q}{=}\mathbf{x}\mathbf{w}_q$, $\mathbf{k}{=}\mathbf{x}\mathbf{w}_k$ and $\mathbf{v}{=}\mathbf{x}\mathbf{w}_v$, where $d_q = d_k$. Queries and keys are used to compute an attention map $\mathcal{A}(\mathbf{q}, \mathbf{k}) = \mathit{Softmax}(\mathbf{q}\mathbf{k}^\top / \sqrt{d_k}) \in \mathbb{R}^{N \times N}$, and the output of the self-attention operation is defined as the weighted sum of N token features in $\mathbf{v}$ with the weights corresponding to the attention map: $\text{Attention}(\mathbf{q}, \mathbf{k}, \mathbf{v}) = \mathcal{A}(\mathbf{q}, \mathbf{k})\mathbf{v}$. Single-head self-attention can be extended to multi-head self-attention by linearly projecting the queries, keys and values g times with different, learned linear projections to d_k, d_k and d_v dimensions, respectively.

3.2 TransMix

We propose TransMix to assign mixup labels with the guidance of attention map, where the attention map is defined specifically as the **multi-head class attention** $\mathbf{A}$, which is calculated as a part of self-attention. In the classification task, a class

token is a query $\mathbf{q}$ whose corresponding keys $\mathbf{k}$ are the all input tokens, and class attention $\mathbf{A}$ is the attention map from the class token to the input tokens, summarizing which input tokens are the most useful to the final classifier. We then propose to use the class attention $\mathbf{A}$ to mix labels.

Multi-head Class Attention Vision Transformers (ViTs)[80] divide and embed an image $\mathbf{x} \in \mathbb{R}^{3\times H\times W}$ to p patch tokens $\mathbf{x}_{patches} \in \mathbb{R}^{p\times d}$, and aggregate the global information by a class token $\mathbf{x}_{cls} \in \mathbb{R}^{1\times d}$, where d is the dimension of embedding. ViTs operate on the patch embedding $\mathbf{z} = [\mathbf{x}_{cls}, \mathbf{x}_{patches}] \in \mathbb{R}^{(1+p)\times d}$.

Given a Transformer with g attention heads and input patch embedding $\mathbf{z}$, we parametrize the multi-head class-attention with projection matrices $\mathbf{w}_q$, $\mathbf{w}_k \in \mathbb{R}^{d\times d}$. The class attention for each head can be formulated as:

$$\mathbf{q} = \mathbf{x}_{cls} \cdot \mathbf{w}_q, \tag{4.3}$$

$$\mathbf{k} = \mathbf{z} \cdot \mathbf{w}_k, \tag{4.4}$$

$$\mathbf{A}' = \mathit{Softmax}(\mathbf{q} \cdot \mathbf{k}^\top / \sqrt{d/g}), \tag{4.5}$$

$$\mathbf{A} = \{\mathbf{A}'_{0,i}, \mid i \in [1, p]\}, \tag{4.6}$$

where $\mathbf{q} \cdot \mathbf{k}^\top \in \mathbb{R}^{1\times(1+p)}$ indicates the class token is a query whose corresponding keys are the all input tokens, and $\mathbf{A} \in [0, 1]^p$ is the attention map from the class token to the image patch tokens, summarizing which patches are the most useful to the final classifier. When there are multiple heads in the attention, we simply average across all attention heads to obtain $\mathbf{A} \in [0, 1]^p$. In implementation, $\mathbf{A}$ in Eqn. (4.6) is available as an intermediate output from the last Transformer block without architecture modification.

Mixing labels with the attention map A We follow the process of input mixture proposed in CutMix, which is defined in Eqn. (4.1), then we re-calculate λ (the proportion of $\mathbf{y}_A$ in Eqn. (4.2)) with the guidance of the attention map $\mathbf{A}$:

Algorithm 1 Pseudocode of TransMix in a PyTorch-like style.

```
# H, W: the height and width of the input image
# p: number of patches
# M: 0-initialized mask with shape (H,W)
# downsample: downsample from length (H*W) to (p)
# (bx1, bx2, by1, by2): bounding box coordinate

for (x, y) in loader: # load a minibatch with N pairs
   # CutMix image in a minibatch
   M[bx1:bx2, by1:by2] = 1
   x[:,:,M==1] = x.flip(0)[:,:,M==1]
   M = downsample(M.view(-1))

   # attention matrix A: (N, p)
   logits, A = model(x)

   # Mix labels with the attention map
   lam = matmul(A, M)
   y = (1-lam) * y + lam * y.flip(0)

   CrossEntropyLoss(logits, y).backward()
```

$$\lambda = \mathbf{A} \cdot \downarrow (\mathbf{M}). \tag{4.7}$$

Here $\downarrow (\cdot)$ denotes the nearest-neighbor interpolation down-sampling that can transform the original $\mathbf{M}$ from HW into p pixels . Note that we omit the dimension unsqueezing in Eqn. (4.7) for simplicity. In this way, the network can learn to reassign the weight of labels for each data point dynamically based on their responses in the attention map. The input that is better focused by the attention map will be assigned with a higher value in the mixed label.

3.3 Pseudo-code

Algorithm 1 provides the pseudo-code of TransMix in a pytorch-like style. The clean pseudo-code shows that simply few lines of code can boost the performance in the plug-and-play manner.

4 Experiments

In this section, we mainly demonstrate the effectiveness, transferability, robustness, and generalizability of TransMix. We verify the effectiveness of TransMix on ImageNet-1k classification in Section 4.1 and the transferability onto downstream

tasks including semantic segmentation, object detection, and instance segmentation in Section 4.2. The robustness of TransMix is examined on 4 benchmarks in Section 4.3. Interestingly, we discover the mutual effects of TransMix and attention in Section 4.4. We validate the generalizability to Swin Transformer which is lacking class-token in Section 4.5. Lastly, TransMix is compared with the state-of-the-art Mixup augmentation variants in Section 4.6.

4.1 ImageNet Classification

Implementation Details We use ImageNet-1k [73] to train and evaluate our methods for image classification. ImageNet-1k consists of 1.28M training images and 50k validation images, labeled across 1000 semantic categories. The implementation is based on the Timm [295] library. Unless specified otherwise, we make minimal changes to hyperparameters compared to the DeiT [267] training recipe. We examined various baseline vision Transformer models including DeiT [267], PVT [287], CaiT [268], and XCiT [81], and the training schemes will be slightly adjusted to the official papers' implementations.

All Transformers are trained for 300 epochs expect that [81] and [268] report 400 epochs for XCiT and CaiT respectively. As deploying DeiT [267] training scheme, all baselines have already contained the carefully tuned regularization methods including RandAug [64], Stochastic Depth [133], Mixup [340] and CutMix [334]. To ease implementation, TransMix shares the same cropped region with CutMix for the input, whereas the label assignment is the mean of both methods. We throw away repeated augment [122] due to its negative effects examined in [113]. We set warmup epoch to 20 expect DeiT-B keeping 5. The accuracy of our baseline implementation fluctuates only by $\pm 0.1\%$ compared with results reported in DeiT [267]. The attention map $\mathbf{A}$ in Eqn. 4.6 can be obtained as an intermediate output from the multi-head self-attention layer of the last Transformer block.

Results As shown in Table 1, TransMix can steadily boost the top-1 accuracy on

ImageNet for all the listed models. No matter how complex the model is, TransMix can always help to boost the baseline performance. Note that these models are with a wide range of model complexities, and the baselines are all carefully tuned with various data augmentation techniques *e.g.* RandAug [64], Mixup [340] and CutMix [334]. To be specific, TransMix can promote the top-1 accuracy of the small variant DeiT-S by 0.9%. Benefit from the higher attention quality, TransMix can also lift the top-1 accuracy of the large model XCiT-L by a remarkable 0.9%. We emphasize that these systematic improvement with just a tiny tweak on data augmentation is significant when compared with the structural modification on models. For example, CrossViT-B [28] only lifts the DeiT-B baseline result by 0.4% with 20.9% parameters overhead while TransMix leads to more improvement in a parameter-free style. Particularly, TransMix consistently boosts the base/large variants in the range of 0.6% to 0.9%, which is more striking than engineering new architectures such as PiT-B [113], T2T-24 [327], CrossViT-B [28] with the gains of 0.2%, 0.5%, 0.4% respectively.

4.2 Transfer to Downstream Tasks

ImageNet pre-training is the de-facto standard practice for many visual recognition tasks [104]. Before training for downstream tasks, the weights pre-trained on ImageNet is used to initialize the Transformer backbone. We demonstrate the transferability of our TransMix-based pre-trained models on the downstream task including semantic segmentation, object detection and instance segmentation, on which we observe the improvements over the vanilla pre-trained baselines.

Semantic Segmentation In our experiments, the sequence of patch encoding $z_{patches} \in \mathbb{R}^{p \times d}$ is decoded to a segmentation map $s \in \mathbb{R}^{H \times W \times K}$ where K is the number of semantic classes. We adopt two convolution-free decoders: (1) Linear decoder (2) Segmenter decoder. The reason for adopting the Linear decoder is to preserve the pre-trained information to the greatest extent. For linear decoder, a

Models	Params	FLOPs	Top-1 Acc (%)	+TransMix Top-1 Acc (%)
DeiT-T [267]	5.7M	1.6G	72.2	**72.6**
PVT-T [287]	13.2M	1.9G	75.1	**75.5**
XCiT-T [81]	12M	2.3G	79.4	**80.1**
CaiT-XXS[268]	17.3M	3.8G	79.1	**79.8**
DeiT-S [267]	22.1M	4.7G	79.8	**80.7**
PVT-S [287]	24.5M	3.8G	79.8	**80.5**
XCiT-S [81]	26M	4.8G	82.0	**82.3**
Swin-T [187]	28.3M	4.5G	81.3	**81.8**
PVT-M [287]	44.2M	6.7G	81.2	**82.1**
Swin-S [187]	49.6M	8.8G	83.0	**83.2**
PVT-L [287]	61.4M	9.8G	81.7	**82.4**
XCiT-M [81]	84M	16.2G	82.7	**83.4**
DeiT-B [267]	86.6M	17.6G	81.8	**82.4**
XCiT-L	189M	36.1G	82.9	**83.8**

Table 1. TransMix can steadily boost the a wide range of model variants *e.g.* DeiT, PVT, CaiT, and XCiT on ImageNet-1k classification. Note that all the baselines have been already carefully tuned with extensive augmentation and regularization techniques *e.g.* Mixup [340], CutMix[334], RandAug[64], DropPath[134] *etc.*

point-wise linear layer on DeiT patch encoding $z_{patches} \in \mathbb{R}^{p \times d}$ is used to produce patch-level logits $z_{lin} \in \mathbb{R}^{p \times K}$, which are reshaped and bilinearly upsampled to segmentation map s. The Segmenter [247] decoder is a Transformer-based decoder namely Mask Transformer introduced in [247, 279].

We train and evaluate the models on the Pascal Context [207] dataset and report Intersection over Union (mIoU) averaged over all classes as the main metric. The training set contains 4998 images with 59 semantic classes plus a background class. The validation set contains 5105 images. The training scheme follows [207] which is built on MMSegmentation [57]. As a reference, the result of ResNet101-Deeplabv3+ [37, 40] is reported in MMSegmentation [57].

According to Table 2, TransMix pre-trained DeiT-S-Linear and DeiT-S-Segmenter improve over the vanilla pre-trained baselines by 0.6% and 0.9% mIoU respectively.

Backbone	Decoder	TransMix-pretrained	mAcc	mIoU	mIoU (MS)
ResNet101 [102]	Deeplabv3+ [40]		57.4	47.3	48.5
DeiT-S [267]	Linear		59.4	49.1	49.6
		✓	**60.2**	**49.7**	**50.3**
	Segmenter [247]		60.4	49.7	50.5
		✓	**61.4**	**50.6**	**51.2**

Table 2. Overhead-free impact of TransMix on transferring to downstream **semantic segmentation** task on the Pascal Context [207] dataset. (MS) denotes multi-scale testing.

There are consistent improvements on multi-scale testing.

Object Detection and Instance Segmentation Object detection and instance segmentation experiments are conducted on COCO 2017. All models are trained on 118K images and evaluated 5K validation images. We study on PVT [287] as the detection backbone since its pyramid features make it favorable to object detection. The weights pre-trained on ImageNet is used to initialize the PVT backbone. We train and evaluate Mask R-CNN detector with the PVT backbone initialized with either vanilla (CutMix) or TransMix pre-trained weights for both object detection and instance segmentation. Following PVT [287], we adopt 1× training schedule (*i.e.*, 12 epochs) to train the detector on mmDetection [33] framework. Results for Mask R-CNN with ResNet backbone are reported in mmDetection [33] as references. As showed in Table 3, we find that without introducing extra parameter, the detector initialized with TransMix-pretrained backbone improves over CutMix-pretrained backbone by 0.5% box AP and 0.6% mask AP. Note that regularization-based pre-training for backbone has limited capability on improving downstream object detection. For instance, the recent Mixup variant SaliencyMix [271] only improved 0.16% box AP over CutMix-pretrained model on a smaller detection dataset.

Backbone	Params	Object detection			Instance segmentation		
		AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{mk}	AP^{mk}_{50}	AP^{mk}_{75}
ResNet50 [105]	44.2M	38.0	58.6	41.4	34.4	57.1	36.7
ResNet101 [105]	63.2M	40.4	61.1	44.2	36.4	57.7	38.8
PVT-S [287]	44.1M	40.4	62.9	43.8	37.8	60.1	40.3
TransMix-PVT-S	44.1M	**40.9**	**63.8**	**44.0**	**38.4**	**60.7**	**41.3**

Table 3. Overhead-free impact of TransMix on transferring to downstream **object detection and instance segmentation** using Mask R-CNN [105] with PVT [287] backbone on COCO val2017. AP^{bb} denotes bounding box AP for object detection and AP^{mk} denotes mask AP for instance segmentation.

4.3 Robustness Analysis

Recently the discussions regarding the robustness of vision Transformer are emerging [9, 201, 211]. To verify if TranMix can improve ViT-based models' robustness and out-of-distribution performance, we evaluated our TransMix pre-trained models on four robustness scenarios including occlusion, spatial structure shuffling, natural adversarial example, and out-of-distribution detection.

Robustness to Occlusion [211] studies whether ViTs perform robustly in occluded scenarios, where some or most of the image content is missing. To be specific, vision Transformers divide an image into M=196 patches belonging to a 14x14 spatial grid; *i.e.* an image of size 224×224×3 is split into 196 patches, each of size 16×16×3. Patch Dropping means replacing original image patches with blank 0-value patches. As an example, dropping 100 such patches from the input is equivalent to losing 51% of the image content. Following [211], we showcase the classification accuracy on ImageNet-1k validation set with three dropping settings. (1) *Random Patch Dropping*: A subset of M patches is randomly selected and dropped. (2) *Salient (foreground) Patch Dropping*: This studies the robustness of ViTs against occlusions of highly salient regions. [211] thresholds DINO's attention map to obtain salient patches, which are dropped by ratios. (3) *Non-salient (background) Patch Dropping*: The least salient regions of an image are selected and dropped following the same approach as above.

As shown in Figure. 3, DeiT-S with TransMix outperform vanilla DeiT-S on all occlusion levels especially for extreme occlusion (information loss ratio >0.7).

Sensitivity to Spatial Structure Shuffling We study the model's sensitivity to the spatial structure by shuffling on input image patches. Specifically, we randomly shuffle the image patches with different grid sizes following [211]. Note that a shuffle grid size of 1 means no shuffle, and a shuffle grid size of 196 means all patch tokens are shuffled. Figure 4 shows the consistent improvements over baseline, and the accuracy averaged on all shuffled grid sizes for TransMix-DeiT-S and DeiT-S are 62.8% and 58.4% respectively. The superior 4.2% gain indicates that TransMix enables Transformers rely less on positional embedding to preserve the most informative context for classification.

Natural Adversarial Example The ImageNet-A dataset [112] adversarially collects 7500 unmodified, natural but "hard" real-world images, which are drawn from some challenging scenarios (e.g., fog scene and occlusion). The metric for assessing classifiers' robustness to adversarially filtered examples includes the top-1 accuracy, Calibration Error (CalibError) [112, 161], and Area Under the Response Rate Accuracy Curve (AURRA). CalibError judges how classifiers can reliably forecast their accuracy. AURRA is an uncertainty estimation metric introduced in [112]. As shown in Table 4, TransMix-trained DeiT-S is superior to vanilla DeiT-S on all metrics.

Out-of-distribution Detection The ImageNet-O [112] is an adversarial out-of-distribution detection dataset, which adversarially collects 2000 images from outside ImageNet-1K. The anomalies of unforeseen classes should result in low-confidence predictions. The metric is the area under the precision-recall curve (AUPR) [112]. Table 4 indicates that TransMix-trained DeiT-S outperform DeiT-S by 1% AUPR.

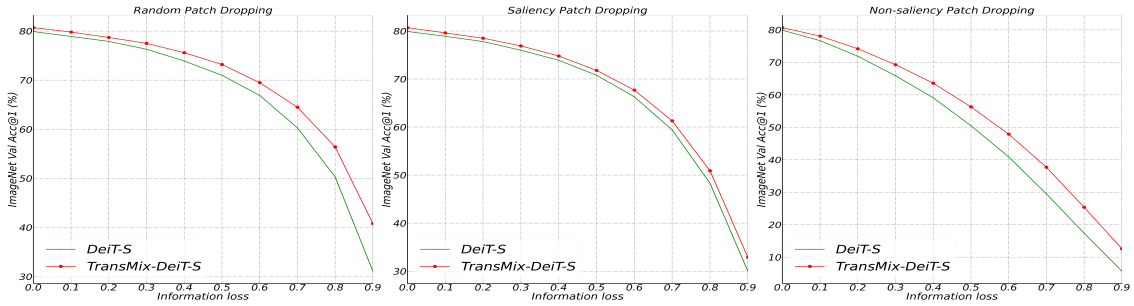


Figure 3. Robustness against occlusion. Model's robustness against occlusion with different information loss ratios is studied. 3 patch dropping settings: Random Patch Dropping (left), Salient Patch Dropping (middle), and Non-Salient Patch Dropping (right) are considered.

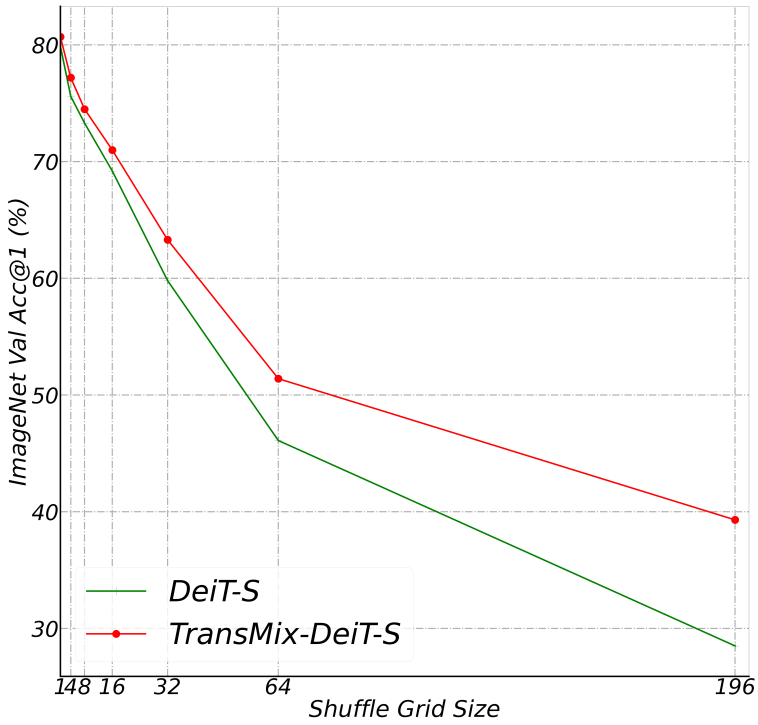


Figure 4. Robustness against shuffle. Model's robustness against shuffle with different grid shuffle sizes is studied.

	Nat. Adversarial Example			Out-of-Dist
Models	Top1-Acc	Calib-Error↓	AURRA	AUPR
DeiT-S	19.1	32.0	23.8	20.9
TransMix-DeiT-S	**21.1**	**31.2**	**28.8**	**21.9**

Table 4. Model's robustness against natural adversarial examples on ImageNet-A and out-of-distribution examples on ImageNet-O.

4.4 Mutual Effect of TransMix and Attention

Will TransMix Benefit Attention? To evaluate the quality of attention matrix, we directly threshold the class-token attention **A** from DeiT-S to obtain a binary attention mask (the same with [26, 211]] with threshold 0.9) and then conduct two tasks including (1) ***Weakly Supervised Automatic Segmentation*** on Pascal VOC 2012 benchmark [83]. (2) ***Weakly Supervised Object Localization (WOSL)*** on ImageNet-1k validation set [234] where the bounding box annotations are only available for evaluation. For task (1), we compute the Jaccard similarity between ground truth and binary attention masks over the PASCAL-VOC12 validation set. For task (2), different from CAM-based methods for CNNs, we directly generate one tight bounding box from the binary attention masks, which is compared with ground-truth bounding box on ImageNet-1k. Both tasks are weakly-supervised since only the class-level ImageNet labels are used for training models (*i.e.* neither bounding box supervision for object localization nor per-pixel supervision for segmentation). The attention masks generated from TransMix-DeiT-S or vanilla DeiT-S are compared with ground-truth on these two benchmarks. The evaluated scores can quantitatively help us to understand if TransMix has a positive effect on the quality of attention map.

Can Better Attention Nurture TransMix? The experiments above prove that TransMix can benefit attention map, and it's natural to ask that can better attention map nurture TransMix in return? We hypothesize that the better attention map is used, the more accurate TransMix adjusts the mixed-target assignment. For

	Segmentation JI (%)	Localization mIoU (%)
DeiT-S	29.2	34.9
TransMix-DeiT-S	**29.9**	**44.4**

Table 5. Quantitative evaluation of the attention map. Segmentation JI denotes the Jaccard index for weakly supervised segmentation on Pascal VOC and Localization mIoU denotes the bounding box mIoU for weakly supervised object localization on ImageNet-1k.

example, Dino [26] confirm that the attention maps obtained from the model via self-supervised training [10, 26] retain greater quality. To validate if a better attention map helps TransMix, we design an experiment that replaces the attention map with that generated from a parameter-frozen external model. The external parameter-frozen model can be (1) Dino self-supervised pre-trained DeiT-S (2) Deit-S that is fully-supervised trained on ImageNet-1k. (3) Deit-S that is fully-supervised trained with a knowledge distillation setting on ImageNet-1k. However, the results shown in Table 6 contradict the hypothesis.

Intriguing Dynamic Property With pre-trained Dino as the attention provider, the performance is slightly worse than that of self-serving. Training with attention guidance from a external fully-supervised parameter-frozen DeiT-S, TransMix suffers from a significant drop from 80.7% to 80.4% top-1 accuracy, though it is still better than vanilla model's 79.8%. This phenomenon can ascribe to the dynamic property of TransMix, meaning that the per-iteration parameter update will dynamically diversify the self-attention for the same input image. In contrast, the parameter-frozen external models statically produce the same self-attention for an image, and thus undermine the regularization capability.

4.5 Generalizability Study

One might be wondering if TransMix can be generalized to those models without the class token such as Swin-Transformer (Swin) [187]. Such models directly apply average pooling onto patch tokens to obtain logits, and therefore how much each patch token contributes to the final prediction is a black-box procedure without class

Attn Provider	Self	Dino	DeiT-pretrained	DeiT-distilled
top-1 Acc	80.7	80.6	80.4	80.4

Table 6. Using external (parameter-frozen) models to generate ttention map as the alternative to original attention map **A** used for TransMix.

Models	Params	FLOPs	top-1 Acc (%)
Swin-T [187]	28.3M	4.5G	81.3
CA-Swin-T [187, 268]	28.3M	4.2G	81.6
TransMix-CA-Swin-T	28.3M	4.2G	**81.8**
Swin-S [187]	49.6M	8.8G	83.0
CA-Swin-S [187, 268]	49.6M	8.5G	82.8
TransMix-CA-Swin-S	49.6M	8.5G	**83.2**

Table 7. Generalization to Swin Transformer [187] which lacks the class-token. CA denote the class attention block [268]. CA-Swin replaces Swin's last block with a CA block with fewer FLOPs.

attention **A**.

To tackle the aforementioned issue, we develop a Swin variant named as CA-Swin that replaces the last Swin block with a classification attention (CA) block without parameter overhead, which makes it possible to generalize TransMix onto Swin. Inspired by CaiT [268], the classification attention block aims at inserting the class token in a plug-and-play manner to those Transformers originally with only patch tokens, and make the classification attention **A** accessible. We then compare the Swin-T, CA-Swin-T, TransMix-CA-Swin-T on ImageNet-1k with the same experimental setup in Sec. 4.1. All three models are at the same 28.3M parameters. TransMix-CA-Swin-T and CA-Swin-T have 7% fewer FLOPs than the baseline Swin-T. The top1 validation accuracy are 81.3%, 81.6% and 81.8% for Swin-T, CA-Swin-T and TransMix-CA-Swin-T, respectively. TransMix on Swin-S improves performance with fewer FLOPs as well. This preliminary study empirically proves the generalizability of TransMix.

4.6 Comparison with State-of-the-art Mixup Variants

In this section, we provide the comprehensive comparison with many state-of-the-art mixup vairiants on ImageNet-1k. This is the first time that compare these variants on vision Transformer in a fair setting. The implementation details for Mixup variants on top of DeiT-S are provided in the supplementary material. All mentioned models are built upon DeiT training recipe towards a fair comparison. Baseline in Table 8 is chosen to be the default DeiT-S framework excluding CutMix in training. Measured on image per second (im/sec), training speed (*i.e.* training throughput) is performed in average of five runs for images at resolution 224×224 under 128 batch size with a Tesla-V100 graphic card, and takes account of data mixup, model forward and backward in train-time.

Table 8 shows TransMix significantly outperforms all other Mixup variants. The saliency-based methods (*e.g.* SaliencyMix and Puzzle-Mix) reveal no advantages to vision Transformer, compared to the vanilla CutMix. We analyze that these methods are cumbersomely tuned and face difficulty in transferring to new architecture. For example, Attentive-CutMix bring not only extra time but also parameter overhead as it introduces an external model to extract saliency map. Puzzle-Mix performs the lowest speed as it forward and backward twice during one training iteration. By contrast, TransMix yields a striking 2.1% performance advancement with the highest training throughput and no parameter-overhead.

Ablation Study Unlike suprisingly 8 hyper-parameters in PuzzleMix, our proposed TransMix exists very clean and introduces almost no hyper-parameter. Still, we conduct ablation study for TransMix regarding the attention map generation in the supplementary material, which shows that the default is the best.

Visualization We provide the visualization of TransMix as shown in Figure 5. For instance, the first row illustrate that the old area-based label assignment is counter-intuitive as image A's foreground is occluded by image B's patch, and TransMix

Method	Backbone	Params	Speed (im/sec)	top-1 Acc (%)
Baseline	DeiT-S	22M	322	78.6
CutMix [334]		22M	322	79.8 **+1.2**
Attentive-CutMix [277]		46M	239	77.5 (-1.1)
SaliencyMix [271]		22M	314	79.2 **+0.6**
Puzzle-Mix [145]		22M	139	79.8 **+1.2**
TransMix		22M	322	**80.7** **+2.1**

Table 8. Top1-accuracy, training speed (im/sec) and number of parameters comparison with state-of-the-art Mixup variants on ImageNet-1k. All listed models are built upon DeiT training recipe for fair comparison. Training speed (im/sec) takes account of data mixup, model forward and backward in train-time.

corrects the label assignment via Transformer attention. TransMix is able to lift the label weight if the discriminative fine-grained attribute appears (*e.g.* Pomeranian dog's cheek and eyes in the second row).

5 Conclusion

In this paper, we present TransMix, a simple yet effective data augmentation technique that assigns Mixup labels with attentional guidance for Vision Transformers. TransMix naturally exploits Transformer's attention map to assign the confidence for the mixed-target, and lifts the top-1 accuracy on ImageNet by 0.9% for both DeiT-S and a large variant XCiT-L. Extensive experiments are conducted to verify the effectiveness, transferability, robustness and generalizability of TransMix on totally 10 benchmarks.

Limitations Since we are the first work that pushes an extra mile for the Mixup-based methods towards augmenting vision Transformers, we indeed have limitations as follows:

(1) TransMix can not handle well with those backbones without class token, as it strongly relies on the class attention. This limitation can be mitigated in Section 4.5

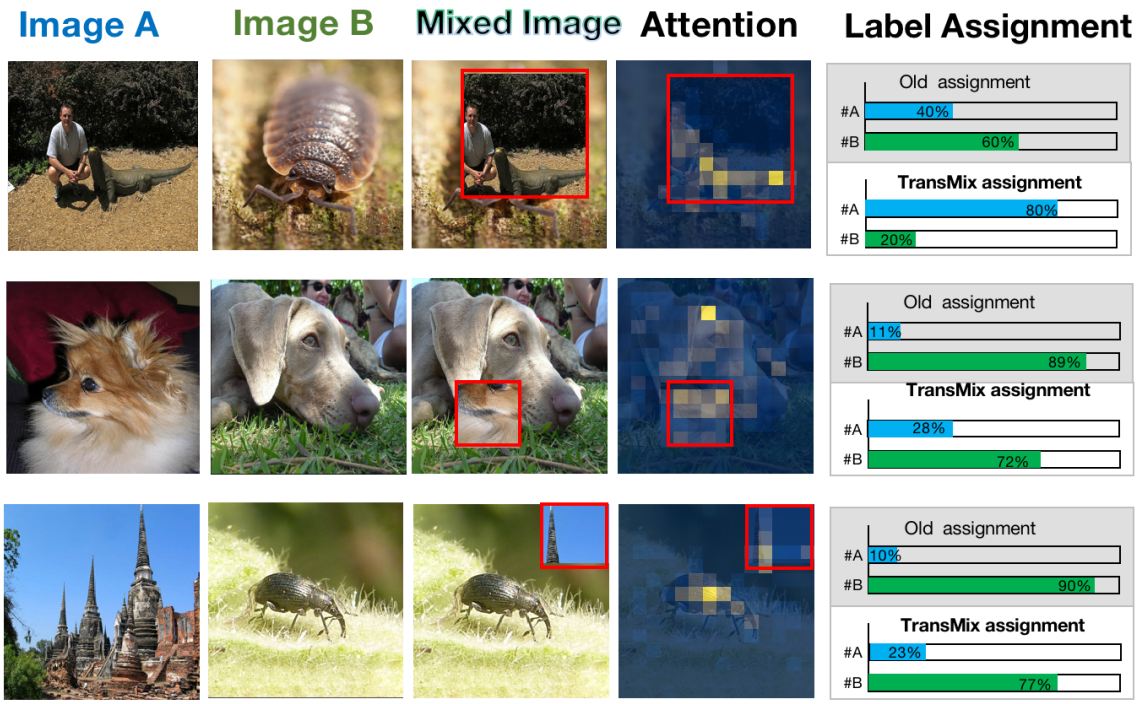


Figure 5. The visualization including image A, image B, mixed image, attention map obtained from XCiT-L when input mixed image, and corresponding label assignments. The label assignments include both the old area-ratio assignment and new TransMix assignment.

at the cost of architecture modification. (2) TransMix requires the attention map to be spatially aligned with the input, resulting in poor compatibility with deformable-based Transformer (e.g. PS-ViT [331], DeformDETR [361]). This can be potentially solved by calibrating attention map to the input spatial location by leveraging deformed offset grid.

Acknowledgement We would like to thank Xiaoyu Yue, Huiyu Wang, Qihang Yu and Chen Wei for their feedbacks. This work is done while the first two authors intern at Bytedance Inc. Jie-Neng Chen and Alan Yuille are supported by ONR N00014-21-1-2812, and the Lustgarten Foundation for Pancreatic Cancer Research. Shuyang Sun and Philip Torr are supported by the Turing AI Fellowship EP/W002981/1, ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1 and EPSRC/MURI grant EP/N019474/1. We would also like to thank the Royal Academy of Engineering and FiveAI.

Part II: Cross-task and Cross-granularity Visual Perception

Chapter 5

Visual Parser: Unifying Visual Frameworks with Part-whole Transformers

This paper is released as a pre-print on Arxiv at https://arxiv.org/abs/2107.05790.

Visual Parser: Unifying Visual Frameworks with Part-whole Transformers

Shuyang Sun[1] Xiaoyu Yue[2] Song Bai[1] Philip Torr[1]

[1]University of Oxford [2]The University of Sydney

kevinsun@robots.ox.ac.uk

https://github.com/kevin-ssy/ViP

Abstract

Existing frameworks exploit various task-specific structures to recognize the visual properties *e.g.* parts and instances since they cannot capture the compositional properties directly from the backbone encoder. This paper presents Visual Parser (ViP), **a unified framework** for visual tasks *e.g.* image classification, detection and segmentation **without task-specific structures** but just compact linear projections overhead, which brings great simplicity to visual perception. Such unification is carried out by the explicit construction of the compositional "part" and "whole" visual representations. Concretely, the part is a set of learnable embeddings representing discriminative entities like objects, and the whole is a batch of feature maps for dense prediction. The two-level features are iteratively composed with an encoder-decoder interaction, during which the part is first updated from and then decoded into the whole. We demonstrate that a stand-alone ViP can be directly applied to image classification, object detection and panoptic segmentation and achieve competitive performance on the challenging ImageNet and MS COCO. The code will be made publicly available for reproduction.

*The first two authors contribute equally to this work.

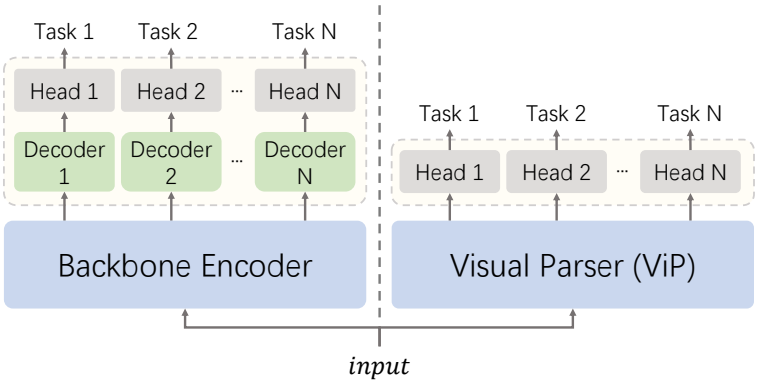


Figure 1. *Left*: Previous backbone encoders need lateral task-specific decoders when applied to down-stream tasks. *Right*: Our Visual Parser (ViP) can extract compositional representations directly as a backbone encoder, which means it is unnecessary for us to further apply task-specific down-stream decoders on top of the ViP. Heads are defined as compact modules like linear projections or MLP layers.

1 Introduction

Strong evidence has been found in psychology that human vision is able to parse a complex scene into part-whole compositional representations that include surrogates from the lowest pixels to the high-level properties (*e.g.* parts, objects, scenes) [115, 139]. Constructing such a part-whole hierarchy [116] enables neural networks to capture compositional representations [270] directly from images, which can promisingly help to detect properties of different levels directly from the backbone representations.

To the best of our knowledge, most current backbone networks do not model such part-whole hierarchy explicitly. Instead, these backbone networks describe everything in the whole image only with dense feature maps but cannot find some visual compositions *e.g.* parts and instances directly in the backbone representations. Therefore, as shown in Figure 1, when transferring the backbone architecture to some downstream tasks *e.g.* object detection, we still need to apply task-specific structures (*a.k.a.* decoders) on top of the backbone encoder to represent the parts and instances, which diverges frameworks for different visual tasks and also forms a cumbersome transfer process.

In one word, ideal modeling of the visual representation should be able to model such

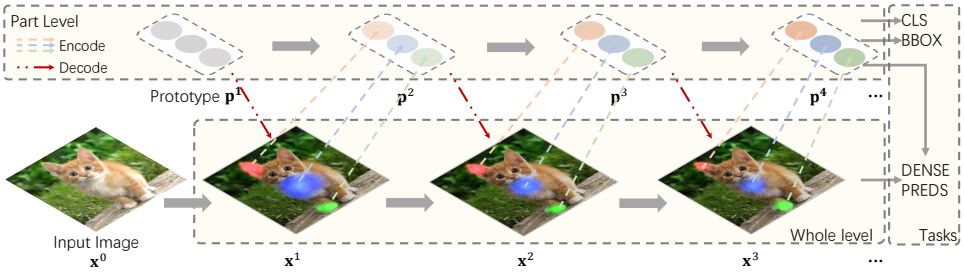


Figure 2. The overall pipeline of the Visual Parser (ViP). Given an input image and the prototype parameters $\mathbf{p}^1$ as input, we iteratively refine the features of the two levels with the proposed encoder-decoder process. The stand-alone ViP can be directly used for image classification, object detection and dense prediction like panoptic segmentation.

part-whole hierarchy so that we can get rid of the task-variant decoders as shown in Figure 1. With such a part-whole hierarchy, we can learn to extract a set of unified compositional representations for all visual perception tasks. To this end, we present Visual Parser (ViP), which constructs the simplest part-whole hierarchy to model the compositional visual representation. The hierarchy of ViP has two levels. One represents the *part*, which only contains the most essential high-level information describing the visual input, and the other is for the *whole*, which describes the visual input in a regular spatial coordinate frame system. Normally a part vector (*e.g.* representing a cat) can be dynamically associated with several vectors of the whole (*e.g.* pixels at the body, tail), forming a one-to-many mapping between the two levels. To obtain information for the part, we first apply an encoder between the two levels to compose the part features with the features of the whole. Then the encoded part feature will be mapped back to the whole by a decoder, as shown in Figure 2. Such cross-level interaction is iteratively applied throughout the network, constituting a bi-directional pathway between the two levels.

Benefiting from the part-whole hierarchy, ViP can generate unified compositional representations for image-level, instance-level and pixel-level predictions. These representations can be directly applied for different tasks *e.g.* image classification, object detection and panoptic segmentation without task-specific structures overhead, which greatly simplifies the overall pipeline for unified visual perception. Such unification creates new possibilities for tasks with fine-grained annotations (*e.g.* detection

and segmentation) to be directly pre-trained on datasets with only coarse-grained labels (*e.g.* classification) in almost an end-to-end way. This further paves the way towards the unified visual perception like BERT [74] in the field of NLP.

Our main contributions are as follows: (1) ViP is the first to provide a unified solution for image categorization, detection and panoptic segmentation. Such unification is an essential step towards unified perception, yet can help to achieve competitive performance compared to other task-specific frameworks. (2) The proposed encoder-decoder interaction can effectively communicate the information between part and whole level, and generate strong visual representations in both global and local granularity. (3) Experimental results also demonstrate that the stand-alone ViP can achieve very competitive results compared to the current state-of-the-art backbones. As a bonus, the whole representation of ViP can be also exploited by frameworks specifically designed for downstream tasks.

2 Method

2.1 Overview

The overall pipeline is shown in Figure 2. There are two inputs of ViP, including an input image and a set of learnable parameters. These parameters represent the prototype of the parts, and will be used as initial clues indicating the region that each part should be associated with. The entire network consists of several basic blocks (iterations). For block $i \in \{2, 3, ..., B\}$, there are two kinds of features describing the two hierarchical levels. One is the part representation $\mathbf{p}^i \in \mathbb{R}^{N \times C_p}$ and the other is the whole feature maps $\mathbf{x}^i \in \mathbb{R}^{L \times C}$. Here N is a pre-defined constant number indicating the number of parts within the input image, and L denotes the number of pixels of the feature map (whole), which equals to $width \times height$. B, C are the numbers of blocks and channels respectively. The representation of parts for each block is dynamically encoded from the corresponding whole feature maps

through an attention-based approach. Given the representation of the part $\mathbf{p}^{i-1}$ and the whole $\mathbf{x}^{i-1}$ from the previous block $i-1$, an attention-based encoder is applied to fill the information of the whole into the part $\mathbf{p}^{i}$ of the current block. Since the attention mechanism assigns each pixel on the feature map with a weight indicating the affinity between the pixel and the corresponding part, only the spatial information in $\mathbf{x}^{i-1}$ that is semantically close to the input part $\mathbf{p}^{i-1}$ can be updated into $\mathbf{p}^{i}$.

Information within the encoded parts will be then transferred back into the feature maps, so each pixel on the feature map can interact with the information in a wider range. Since the information within the parts is highly condensed, the computational cost between the pixels and the parts is much lower than the original pixel-wise global attention [246, 288]. This encoder-decoder process constitutes the basic building block of ViP. By stacking the building block iteratively, the network can learn to construct a two-level part-whole hierarchy explicitly.

2.2 Part Encoder

The part encoder is responsible for extracting the part information based on the previous part-whole input. The encoder is implemented with an attention mechanism. Given the part representation $\mathbf{p}^{i-1} \in \mathbb{R}^{N \times C_p}$ of the last block $i-1$, we first normalize it with Layer Normalization [6], then use it as the query of the attention block. The whole feature map from the last block $\mathbf{x}^{i-1} \in \mathbb{R}^{L \times C}$ serves as the key and value after the normalization. The information of the whole will be condensed into the part representations via attention, which can be formulated as:

$$\begin{aligned} Q_e^{i-1} &= [\mathrm{LN}(\mathbf{p}^{i-1}), \mathbf{d}_e], \\ K_e^{i-1} &= [\mathrm{LN}(\mathbf{x}^{i-1}), \mathbf{d}_w], \\ V_e^{i-1} &= \mathrm{LN}(\mathbf{x}^{i-1}), \\ \hat{\mathbf{p}}^{i-1} &= \mathbf{p}^{i-1} + \mathrm{MHA}(Q_e^{i-1}, K_e^{i-1}, V_e^{i-1}), \end{aligned} \tag{5.1}$$

where $\hat{\mathbf{p}}^{i-1} \in \mathbb{R}^{N \times C_p}$ is the output of the attention block, and $LN(\cdot)$ represents the Layer Normalization [6], $[\cdot,\cdot]$ is the concatenation. $\mathbf{d}_w \in \mathbb{R}^{L \times C_d}, \mathbf{d}_e \in \mathbb{R}^{N \times C_d}$ are sinusoidal positional encodings for the whole and the part respectively. $\mathrm{MHA}(\cdot,\cdot,\cdot)$ denotes the multi-head attention mechanism [273] which can be formulated as

$$\begin{aligned} \mathrm{MHA}(Q,K,V) &= [\mathrm{head}_1, ..., \mathrm{head}_g]W_O, \\ \mathrm{head}_i &= \mathrm{softmax}(\frac{QW_i^Q(KW_i^K)^\top}{\sqrt{C_g}})VW_i^V, \end{aligned} \tag{5.2}$$

where the $W_O \in \mathbb{R}^{(gC_g^V) \times C}$, $W_i^Q \in \mathbb{R}^{C_g^Q \times C_g}$, $W_i^K \in \mathbb{R}^{C_g^K \times C_g}, W_i^V \in \mathbb{R}^{C_g^V \times C_g^V}$ denote the matrices of parameters for the linear mappings of the multi-head attention mechanism. The product of the query and the key is normalized by a temperature factor $\frac{1}{\sqrt{C_g}}$ to avoid it being one-hot after the lateral softmax calculation. Here C_g^Q, C_g^K, C_g^V, C_g are numbers of channels in each head of Q, K, V and the hidden dimension.

Reasoning across different parts. After each part is filled with the information from the feature maps, we apply a part-wise reasoning module to enable information communication between parts. In order to save computational cost, we just apply a single linear projection with learnable weights $W_p \in \mathbb{R}^{N \times N}$. An identity mapping and the normalization are also applied here as a residual block. The process of the part-wise reasoning can be formulated as:

$$\hat{\mathbf{p}}_r^{i-1} = \hat{\mathbf{p}}^{i-1} + W_p \mathrm{LN}(\hat{\mathbf{p}}^{i-1}), \tag{5.3}$$

where $\hat{\mathbf{p}}_r^{i-1}$ represents the output for the part-wise reasoning. This process can be also implemented by a self-attention module. We compare the performance of the two implementations in Section 4.5.

Activating the part representations. The part representation learned above may not be all meaningful since different objects may have different numbers of parts describing themselves. We thereby further apply a Multi-Layer Perceptron

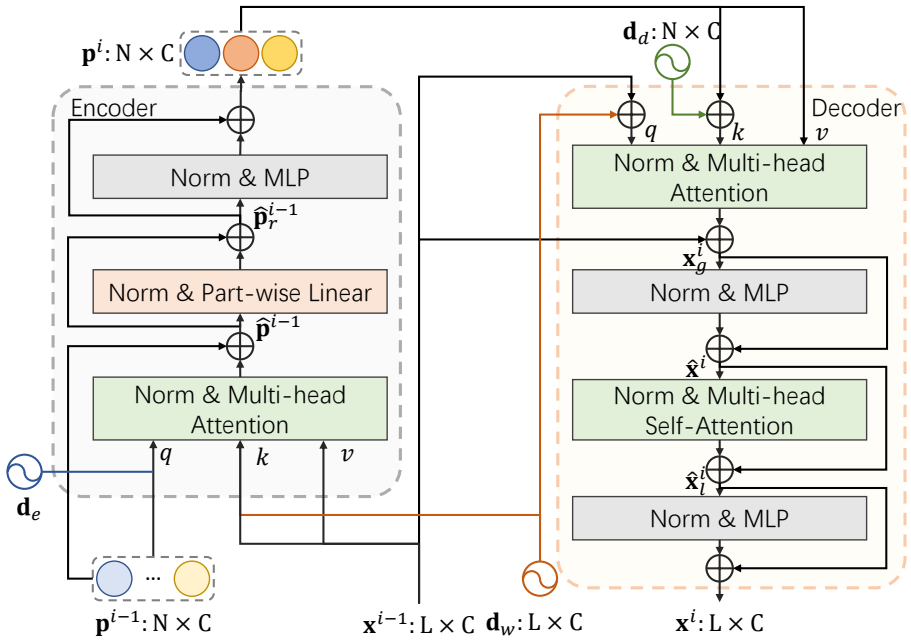


Figure 3. The basic building block of ViP. The symbol $\bigoplus$ denotes element-wise summation. Here we omit the relative positional embedding $\mathbf{r}^i$ for clear demonstration.

(MLP) that has two linear mappings with weights W_1, W_2 and biases b_1, b_2 and an activation function (GELU [110]) $\sigma(\cdot)$ in its module. The activation function will only keep the useful parts to be active, while those identified to be less helpful will be squashed. In this way, we obtain the part representation $\mathbf{p}^i$ for block i by:

$$\begin{aligned} \mathbf{p}^i &= \hat{\mathbf{p}}_r^{i-1} + \mathrm{MLP}(\hat{\mathbf{p}}_r^{i-1}), \\ \mathrm{MLP}(a) &= \sigma(\mathrm{LN}(a)W_1 + b_1)W_2 + b_2. \end{aligned} \tag{5.4}$$

The above process demonstrates that the part representation generated by the previous block will be used to initialize the parts of the next iteration. Thus the randomly initialized part representations will be gradually refined with the information from the whole within each block.

2.3 Whole Decoder

As shown in Figure 3, there are two inputs for the decoder, the part representation $\mathbf{p}^i$ and the whole representation $\mathbf{x}^{i-1}$. Interactions within the decoder can be divided into a part-whole global interaction between the parts and the feature maps, and a

patch-based local attention between pixels in a local window of the whole feature maps.

Part-whole global interaction. We first apply the part-whole global attention to fill each pixel on the whole representations $\mathbf{x}^{i-1}$ with the global information encapsulated in the parts $\mathbf{p}^i$. The part-whole global attention completely follows the classic attention paradigm [273], which takes $\mathbf{x}^{i-1}$ as the query input, and $\mathbf{p}^i$ as inputs of the key and value. Therefore each pixel of the whole representation can have a long-range interaction with the encoded parts. Before feeding into the attention, both part and whole representations will be normalized by Layer Normalization. An identity mapping and a MLP are also applied as what does in common practice [78]. The process of the part-whole interaction in the decoder can be written as:

$$\begin{aligned} Q_d^i &= \mathrm{LN}(\mathbf{x}^{i-1} + \mathbf{d}_w), \\ K_d^i &= \mathrm{LN}(\mathbf{p}^i + \mathbf{d}_d), \\ V_d^i &= \mathrm{LN}(\mathbf{p}^i), \\ \mathbf{x}_g^i &= \mathbf{x}^{i-1} + \mathrm{MHA}(Q_d^i, K_d^i, V_d^i), \\ \hat{\mathbf{x}}^i &= \mathbf{x}_g^i + \mathrm{MLP}(\mathbf{x}_g^i), \end{aligned} \tag{5.5}$$

where $\mathbf{d}_d \in \mathbb{R}^{N \times C}$ is the positional encoding for parts in decoders, the definitions of the attention mechanism and MLP are identical to those defined in Eq. (5.1) (5.4). Note that $\mathbf{d}_d$ is shared across all blocks of each stage. The axis that the softmax function normalizes on is the last dimension (specifically, the part dimension with N inputs).

Reasoning across different partitions of the whole. The above process, in both the encoder and the decoder, has completed the cross-level interactions for the i^{th} iteration. In addition to the long-range modeling that the cross-level interaction provided, we also apply a local attention for fine-grained feature modeling. We divide the spatial feature maps into non-overlapping patches with size $k \times k$, then apply a multi-head self-attention module for all pixels within each patch. We denote

the pixels of patch t as $\mathbf{x}_t^i \in \mathbb{R}^{k^2 \times C}$, then the process of the local attention can be written as:

$$\begin{aligned} \hat{\mathbf{x}}_t^i &= \mathbf{x}_t^i + \mathrm{MHA}(\mathrm{LN}(\mathbf{x}_t^i), \mathrm{LN}(\mathbf{x}_t^i + \mathbf{r}^i), \mathrm{LN}(\mathbf{x}_t^i)), \\ \hat{\mathbf{x}}_l^i &= \{\hat{\mathbf{x}}_1^i, ..., \hat{\mathbf{x}}_t^i, ..., \hat{\mathbf{x}}_{N_p}^i\}, \\ \mathbf{x}^i &= \hat{\mathbf{x}}_l^i + \mathrm{MLP}(\hat{\mathbf{x}}_l^i), \end{aligned} \tag{5.6}$$

where $N_p = \frac{L}{k^2}$ denotes the total number of patches, $\mathbf{r}^i \in \mathbb{R}^{k^2 \times C}$ is the relative positional embedding. The implementation of the relative positional embedding follows the design in [12, 246]. To save the computational cost, $\mathbf{r}^i$ is factorized into two embeddings $\mathbf{r}_h^i \in \mathbb{R}^{(2k-1)\times\frac{C}{2}}, \mathbf{r}_w^i \in \mathbb{R}^{(2k-1)\times\frac{C}{2}}$ for the dimension of height and width respectively. Here another MLP layer is applied to further activate the feature of the whole.

2.4 Stand-alone ViP for Downstream Tasks

Unlike previous backbone that can only perform as a feature extractor in the downstream frameworks, ViP can work as a stand-alone framework for downstream tasks without task-specific decoders (*e.g.* detectors, segmentors) since it can generate compositional representations as shown in Figure 1. We evaluate the stand-alone ViP on two downstream tasks including object detection and panoptic segmentation.

Object detection. When applying ViP for object detection, we first generate L parts then select those with the lowest cost via bipartite matching. The selected part representations will be transformed by a bounding box head and another classification head to generate the bounding boxes' coordinates and the classification score. Concretely, the bounding box head is implemented by a simple MLP layer, and the classification head is a linear projection. The loss formulation can be found in the supplementary material.

Panoptic segmentation. ViP can be also applied for panoptic segmentation in-

dependently due to its compositional property. Basically, panoptic segmentation predicts a set of prediction maps that include both instance-level and semantic-level classes, which means that the output should contain the information of both levels. To this end, we employ the part and whole as instance-level and semantic-level descriptors respectively, and merge them together to generate the panoptic output. Given the part representation and whole representation obtained from the end of the network, we first upsample the whole representation to get high-resolution feature maps, and then directly use the part representation as kernels to perform convolution over the upsampled feature maps of the whole to generate instance masks (*i.e.* 'things'). We also apply a 1×1 convolution layer over the upsampled feature maps to get semantic masks of 'stuff' classes. Finally, we merge instance masks and semantic masks for the panoptic segmentation results.

2.5 Architecture Specification

In this paper, we first design four kinds of different variants called **ViP-Tiny (Ti), ViP-Small (S), ViP-Medium (M), ViP-Base (B)** respectively. Note that these variants are of feature pyramid. We also construct two other variants without such pyramid, namely **ViP-Ti-Plain** and **ViP-S-Plain**. These variants have some common features in the design. (1) Architectures of all these models except for ViP-Ti-Plain and ViP-S-Plain are divided into four stages according to the spatial resolution L of the feature map. Given an input with spatial size $H \times W$, the output spatial sizes of the feature maps for the four stages are $\frac{H}{4} \times \frac{W}{4}$, $\frac{H}{8} \times \frac{W}{8}$, $\frac{H}{16} \times \frac{W}{16}$ and $\frac{H}{32} \times \frac{W}{32}$. (2) The expansion rates of the MLP within the encoder and the decoder, which indicates the ratio between the number of channels of the hidden output and the input, are set to be 1 and 3 separately. (3) The patch size for the self-attention module of the decoder is set to $\{8, 7, 7, 7\}$ for four different stages. (4) At the beginning of each stage, there is a patch embedding responsible for down-sampling and channel-wise expansion. We employ a separable convolution with normalization here with kernel size 3×3 to perform the down-sampling operation for the whole

representation. Since the number of channels of the part representation for each stage may vary, another linear operation is applied to align the number of channels between parts in different stages. The detailed specification of the ViP family can be found in the supplementary material due to the limitation of space.

3 Related Work

CNNs and frameworks that predict using dense pixels. Conventional CNNs [106, 132, 222, 244, 260, 262] are prevalent in nearly all fields of computer vision *e.g.* image classification [11, 14, 106, 222, 256, 262], object detection [21, 93, 95, 105, 177, 186, 230] and semantic segmentation [7, 35, 191, 319, 347] since AlexNet [159] demonstrates its power for image recognition on ImageNet [72]. Although CNN is first proven to be effective in the downstream tasks *e.g.* object detection with extra task-specific structures (*a.k.a.* decoders) [94, 96], some recent works have proven that the dense pixels produced by the backbone can be directly used for describing objects [164, 264, 356], human poses [23, 293] and also semantic maps [192] in an end-to-end approach. However, since the CNNs can only produce non-compositional dense feature maps, it is hard for these frameworks to be directly applied to tasks that require compositional predictions *e.g.* panoptic segmentation and scene parsing. An existing work [172] that aims to solve this problem still needs to first generate kernels (decoders) based on the dense pixels then decode the instances and semantic information with these kernels.

Part-whole compositional visual representations. Tu et al. [270] first devise a Bayesian framework to parse the image into a part-whole hierarchy for unifying all the major vision tasks. Then some research follow the first step building compositional framework with graphical models [71, 301]. Some recent works can also learn to group and associate compositional representations from the deep backbone for scene parsing [173] and occluded object detection [153]. Capsule Networks (CapsNets) [235] were first proposed to use a dynamic routing algorithm to allocate neu-

rons to represent a small portion of the visual input. Some other extensive works based on CapsNets [118, 154, 269] show remarkable performance on some small datasets, however, these works cannot be well when scaled onto large datasets. The recent proposal of GLOM [116] gives an idea to build up a hierarchical representation with attention, but it gives no practical experiments. This paper borrows some ideas from these works to build a rather simple hierarchy with two levels for modeling basic visual representation. For example, the iterative attention mechanism in our model is similar to the dynamic routing designed in CapsNet [235] or iterative attention mechanism [24, 190, 269].

Transformers and self-attention mechanism. With the success of Memory Networks [250] and Transformers [273] for natural language modeling [69, 76, 298, 318], lots of works in the field of computer vision attempted to migrate similar self-attention mechanism as an independent block into CNNs for image classification [12, 46, 130, 296], object detection [24, 127, 246] and video action recognition [87, 138, 288].

Recent works tried to replace all convolutional layers in neural networks with local attention layers to build up self-attention-based networks [58, 119, 127, 224, 272, 346]. To resolve the inefficiency problem, Vision Transformer (ViT) [78, 265] chose to largely reduce the image resolution and only retain the global visual tokens. To aid the global token-based ViT with local inductive biases, several papers incorporate convolution-like design into ViT [56, 70, 114, 299, 328]. Apart from the token-based approach, some works [188, 286] that retain the spatial pyramids have also been proven to be effective. Different from the above existing works that extract either tokens or spatial feature maps for final prediction, ViP extracts both the token-based representations (the part) and spatial feature maps (the whole).

Set-based Transformers for object detection. A range of recent methods explore object detection as set predictions [24, 88, 202, 359]. The object queries proposed in these methods are similar to the parts defined in ViP. To be clear, we

Model	Input Size	Params (M)	FLOPS (G)	Top-1 Acc (%)
CNN Architectures				
BoTNet-T3 [246]	224^2	33.5	7.3	81.7
BoTNet-T4 [246]	224^2	54.7	10.9	82.8
BoTNet-T5 [246]	256^2	75.1	19.3	83.5
RegNetY-4G [222]	224^2	20.6	4.0	80.0
RegNetY-8G [222]	224^2	39.2	8.0	81.7
RegNetY-16G [222]	224^2	83.6	15.9	82.9
Transformer Architectures				
DeiT-Ti [265]	224^2	5.7	1.6	72.2
DeiT-S [265]	224^2	22.1	4.6	79.8
DeiT-B [265]	224^2	86.6	17.6	81.8
DeiT-B↑384 [265]	384^2	86.6	55.4	83.1
T2T-ViT-14 [328]	224^2	21.5	5.2	81.5
T2T-ViT-19 [328]	224^2	39.2	8.9	81.9
T2T-ViT-24 [328]	224^2	64.1	14.1	82.3
TNT-S [100]	224^2	23.8	5.2	81.5
TNT-B [100]	224^2	65.6	14.1	82.9
Swin-T [188]	224^2	29	4.5	81.3
Swin-S [188]	224^2	50	8.7	83.0
Swin-B [188]	224^2	88	15.4	83.3
ViP-Ti	224^2	12.8	1.7	79.0
PS-ViP-Ti	224^2	13.9	1.9	80.9 (+**1.9**)
ViP-S	224^2	34.9	4.8	81.8
PS-ViP-S	224^2	36.8	5.3	82.9 (+**1.1**)

Table 1. Results on ImageNet-1K.

emphasize that there are two key differences between our method and theirs: (1) ViP is essentially an encoder backbone that can be widely transferred onto different tasks while these methods focus on building up decoders for object detection. (2) There is only one direction (whole → part) within the interaction between the two levels of these methods while such interaction in ViP is bi-directional. **Token-based global attention mechanism.** The interaction between the part and the whole is related to the token-based global attention mechanism. Recent works including [11, 45, 46, 296] propose to tokenize the input feature map generated by the convolution block. Our work is different from theirs in three aspects: (1) The part representations are explicitly and iteratively refined in ViP, while in these works, the tokens are latent bi-product of each block. (2) We intend to design a hierarchy that can be used

for final prediction while these works focus on designing a module then plugging it into limited blocks of the network. (3) In detail, the token extraction and the bi-directional pathway designed in ViP are quite different from their pipelines.

4 Experiments

	N	FLOPS (G)	Top-1 (%)
ViP-Ti	16	1.6	78.1
	32	1.7	79.0
	64	1.8	79.1

Table 2. Effect of number of parts for ViP-Ti.

	w/ shifted window	Top-1 (%)
ViP-S		81.8
	✓	**82.2**

Table 3. ViP can be also compatible with shifted window proposed in [188].

4.1 Image Classification on ImageNet-1K

Experimental settings. For image classification, we evaluate our models on ImageNet-1K [72], which consists of 1.28M training images and 50K validation images categorized into 1,000 classes. The network is trained for 300 epochs using AdamW [146] and a half-cosine annealing learning rate scheduler. The learning rate is warmed up for 20 epochs to reach the initial 1×10^{-3}. Weight decays for ViP-Ti are set to be 0.03, while those for ViP-S, ViP-M are 0.05. The drop ratios of Stochastic Depth (*a.k.a* DropPath) [134] are linearly scaled from 0.1, 0.1, 0.2, 0.3 along the layer depth for each layer of ViP-Ti, ViP-S, ViP-M and ViP-B respectively. The total training batch size is set to be 1024 for all model variants. We primarily follow the settings of the data augmentation adopted in [265], except for the repeat augmentation [121]. Note that for all results for image classification on ImageNet reported in this paper, we do not use any external dataset for pre-training.

ViP *vs.* CNNs. Table 1 compares ViP family with some CNN models on ImageNet. The RegNet is also better tuned using training tricks in [265]. ViP is both cost-efficient and parameter-efficient compared to these models. For example, ViP-M can achieve a competitive 83.3% with only 49.6M parameters and 8.0G FLOPS.

The counterpart BOTNet-T5 needs 25.5M more parameters and 11.3G FLOPS to achieve similar performance. When scaling the input to resolution 384^2, ViP-B is able to further improve its top-1 accuracy to 84.2%.

ViP *vs.* other Transformer-based methods. As shown in Table 1, ViP consistently outperforms previous state-of-the-art Transformer-based models in terms of accuracy and model size. Especially, ViP-B achieves 83.8% ImageNet top-1 accuracy, which is 0.5% higher than Swin-B [188] with fewer parameters and FLOPS. A similar trend can also be observed when scaled onto larger models, *e.g.* ViP-M achieves 83.3% top-1 accuracy, outperforming TNT-B [100], T2T-ViT-24 [328], PVT-Large [286] by 0.4%, 1.0%, 1.6% respectively.

4.2 Stand-alone ViP for Object Detection

Experimental settings. We evaluate ViP on COCO dataset [178] when applying ViP as an independent framework for object detection. The dataset contains 115k images for training (*train-2017*) and 5k images (*val-2017*) for validation. We train models on *train-2017* and report the results on *val-2017*. We measure our results following the official definition of Average Precision (AP) metrics given by MS COCO, which includes AP_{50} and AP_{75} (averaged over IoU thresholds 50 and 75) and AP_S, AP_M, AP_L (AP at scale Small, Medium and Large). The whole network is first pre-trained on ImageNet-1k, then trained on COCO for 150/300 epochs with the initial learning rate and weight decay set to be 0.0001 and 0.05. For both schedules, the learning rate drops by 0.1 at 100 epoch. We follow the multi-scale training proposed in [24]. We embed a MLP layer as the bounding box head and a fully-connected layer on top of the part representation to predict the bounding box coordinates and the classification score. The shorter sides of the input size for ViP-Ti and ViP-Ti-plain are set to be 640 and 512 respectively, while those for other ViP variants are set to be 800.

Results. We investigate two different model variants ViP-Ti and ViP-S on COCO

Method	Backbone	Epochs	Params (M)	FLOPs (G)	AP
YOLOS [88]	DeiT-Ti	300	7	19	28.6
YOLOS [88]	DeiT-Ti⚗	300	7	21	30.0
ViP-Ti	-	300	15	20	30.5
ViP-Ti-Plain	-	150	9	20	31.3
DETR[24]	R18-DC5	150	29	129	36.9
YOLOS[88]	DeiT-S	150	31	194	36.1
YOLOS[88]	DeiT-S⚗	150	31	194	37.2
ViP-S	-	150	35	**82**	**37.3**
ViP-S-Plain	-	150	23	**109**	**39.5**

Table 4. Stand-alone ViP for object detection on COCO *val*. ⚗ denotes the use of knowledge distillation during the backbone pre-training.

for object detection. As shown in Table 4. ViP can achieve a better computation-AP trade-off when compared with other transformer-based frameworks *e.g.* DETR and YOLOS for object detection. Specifically, our ViP-S-plain can outperform YOLOS-DeiT-S by a significant 3.4% with about 44% fewer FLOPs (109G *vs.* 194G). Note that we do not apply data distillation on ViP when pre-training it on ImageNet-1k. The success of data distillation on YOLOS [88] indicates that the performance of ViP can be further lifted with better pre-training.

4.3 Stand-alone ViP for Panoptic Segmentation

Experimental settings. When applying ViP as an independent framework for panoptic segmentation, we conduct experiments on the challenging COCO dataset [178]. The panoptic segmentation results are evaluated by the PQ metric proposed in [149]. The performance of thing and stuff are also reported, noted as PQ^{Th}, PQ^{St}, respectively. Similar to the settings we proposed for object detection, we first pre-train the whole network on ImageNet-1k, then trained on COCO with the initial learning rate and weight decay set to be 0.0001 and 0.05 for 50 epochs. The learning rate is decreased by 0.1 at the 33th and 43th epoch. We use multi-scale training with the short edge of images randomly sampled from [640, 800]. For semantic segmenta-

Method	Backbone	Params (M)	FLOPs (G)	Epochs	PQ/PQTh/PQSt
Panoptic FPN [148]	R50-FPN	45.8	238	36	41.5/48.5/31.1
SOLOv2 [290]	R50-FPN	-	-	36	42.1/49.6/30.7
DETR [24]	R50+6Enc	42.8	137	175*	43.4/48.2/36.3
Panoptic FCN [172]	R50-FPN	37.0	244	36	43.6/49.3/35.0
MaskFormer [51]	R50+6Enc	45.0	181	300	46.5/51.0/39.8
MaskFormer [51]	Swin-T	42.0	179	300	47.6/52.5/40.3
ViP-S	-	38.0	**100**	50	45.0/48.7/39.5
ViP-S	+FPN	41.9	183	50	46.9/51.1/40.5
ViP-S	+FPN	41.9	183	300	**48.2**/53.1/40.8
ViP-S-Plain	-	24.8	125	50	45.5/48.9/40.4

Table 5. Stand-alone ViP for Panoptic Segmentation on COCO *val*. * means that 150 epochs-training for object detection and 25 epochs finetuing for panoptic segmentation.

tion, we add another convolution layer on top of the whole representations to predict masks of stuff defined in COCO. We also embed the panoptic FPN [148] on top of the multi-scale whole feature generated by ViP as an alternative.

The stand-alone ViP can outperform the state-of-the-art counterparts for panoptic segmentation on COCO. We compare ViP with state-of-the-art panoptic segmentation counterparts in Table 5. ViP surpasses previous box-based methods and box/NMS-free methods even with much lower computational cost. Specifically, compared with transformer-based method DETR, our ViP-S absolutely improves the PQ by 1.6% while reducing 4.8M parameters and 37G FLOPs. When equipped with FPN, ViP-S achieves 46.9% PQ with just 50 epochs training, which is higher than MaskFormer with ResNet-50 backbone with 6 extra transformer encoders. When trained with a longer schedule (300 epochs), ViP-S can obtain a competitive 48.2% PQ.

4.4 ViP as Backbones of Existing Detectors

Experimental settings. For object detection and instance segmentation, we evaluate ViP on the challenging MS COCO dataset [178]. Experiments are implemented

Backbone	AP^b	AP^b_{50}	AP^b_{75}	AP^b_S	AP^b_M	AP^b_L	Params (M)	FLOPS (G)
R18 [177]	31.8	49.6	33.6	16.3	34.3	43.2	11.0 (21.3)	37 (189)
PVT-T [286]	36.7(+4.9)	56.9	38.9	22.6	38.8	50.0	12.3 (23.0)	70 (221)
ViP-Ti	**39.7**+7.9	60.6	42.2	23.9	42.9	53.0	10.0 (20.2)	**30** (182)
R50 [177]	36.5	55.4	39.1	20.4	40.3	48.1	23.3 (37.7)	84 (239)
PVT-S [286]	40.4(+3.9)	61.3	43.0	25.0	42.9	55.7	23.6 (34.2)	134 (286)
Swin-T [188]	41.5	62.1	44.2	25.1	44.9	55.5	27.5 (38.5)	118 (270)
ViP-S	**43.0**+6.5	64.0	45.9	28.9	46.7	56.3	24.6 (35.6)	**78** (230)
R101 [177]	38.5	57.8	41.2	21.4	42.6	51.1	42.3 (56.7)	160 (315)
X101-32 [177]	39.9(+1.4)	59.6	42.7	22.3	44.2	52.5	41.9 (56.4)	164 (319)
PVT-M [286]	41.9(+3.4)	63.1	44.3	25.0	44.9	57.6	43.7 (54.3)	222 (374)
Swin-S [188]	44.5(+6.0)	65.7	47.5	27.4	48.0	59.9	48.8 (59.8)	162 (314)
ViP-M	**44.5**+6.0	65.9	47.4	30.7	48.1	58.1	48.8 (59.8)	**135** (287)
X101-64 [177]	41.0	60.9	44.0	23.9	45.2	54.0	81.0 (95.5)	317 (473)
PVT-L [286]	42.6	63.7	45.4	25.8	46.0	58.4	60.9 (71.5)	324 (476)

Table 6. Various backbones with RetinaNet. Here R and X are abbreviations for ResNet and ResNeXt. Parameters and FLOPS in black are for backbones, while those in (gray) are for the entire frameworks.

based on the open source mmdetection [34] platform. All models are trained under two different training schedules 1× (12 epochs) using the AdamW [146] optimizer with the same weight decay set for image classification. After a 500 iteration warming-up, the learning rate is initialized at 1×10^{-4} then decayed by 0.1 after [8, 11] epochs. For data augmentation, we only apply random flipping with a probability of 0.5 and scale jittering from 640 to 800. The batch size for each GPU is 2 and we use 8 GPUs to train the network for all experiments. Stochastic Depth [134] is also applied here as what was proposed on ImageNet. We embed ViP into two popular frameworks for object detection and instance segmentation, RetinaNet [177] and Cascade Mask-RCNN [21]. When incorporating ViP into these frameworks, ViP serves as the backbone followed by a Feature Pyramid Network (FPN) [176]

Backbone	AP^b	AP^b_{50}	AP^b_{75}	AP^b_S	AP^b_M	AP^b_L	AP^m	AP^m_{50}	AP^m_{75}	AP^m_S	AP^m_M	AP^m_L	Params (M)	FLOPS (G)
R18	38.7	56.2	41.3	21.3	40.8	52.9	34.0	53.5	36.4	17.4	36.0	48.1	11.0 (69.0)	37 (686)
ViP-Ti	**45.4**+6.7	64.6	48.9	29.1	48.8	60.1	**39.9**+5.9	61.7	42.7	24.1	43.0	54.3	10.0 (67.0)	**30** (678)
R50	41.2	59.4	45.0	23.9	44.2	54.4	35.9	56.6	38.4	19.4	38.5	49.3	23.3 (82.0)	84 (739)
S50	45.4	64.1	49.2	28.3	49.1	58.8	39.5	61.4	42.5	23.1	43.0	52.8	25.1 (82.9)	110 (763)
Swin-T	48.1	67.1	52.2	30.4	51.5	63.1	41.7	64.4	45	24	45.2	56.9	27.5 (85.6)	97 (745)
ViP-S	**48.5**+7.3	67.5	52.5	31.9	51.8	63.2	**42.2**+6.3	64.8	45.7	25.9	45.5	56.7	24.6 (81.8)	**78** (726)
R101	42.9	61.0	46.6	24.4	46.5	57.0	37.3	58.2	40.1	19.7	40.6	51.5	42.3 (101.0)	160 (815)
X101-32	44.3	62.7	48.4	25.4	48.4	58.1	38.3	59.7	41.2	20.6	42.0	52.3	41.9 (100.6)	164 (819)
S101	47.7	66.4	51.9	30.1	51.8	61.4	41.4	63.7	45.1	24.7	45.2	54.9	45.7 (104)	209 (862)
ViP-M	**49.9**+7.0	69.5	54.2	33.1	53.4	65.1	**43.5**+6.2	66.4	47.2	26.8	46.9	59.1	48.8 (107.0)	**135** (785)
X101-64	45.3	63.9	49.6	26.7	49.4	59.9	39.2	61.1	42.2	21.6	42.8	53.7	81.0 (139.7)	317 (972)

Table 7. Various backbones with Cascade Mask R-CNN. All results are trained under 1× schedule. S denotes ResNeSt [337].

refining the multi-scale whole representations. All weights within the backbone are first pre-trained on ImageNet-1K.

Integrating ViP into RetinaNet. Table 6 exhibits the experimental results when embedding different backbones into RetinaNet. When trained under the 1× schedule, our ViP-Ti can outperform its counterpart ResNet-18 by 7.9, which is a large margin since it is even higher than the performance obtained by ResNet-101 (4× larger than ViP-Ti in terms of FLOPS and parameters). The ViP-S, which is just about the size of ResNet-50, can even outperforms the largest model ResNeXt-101-64×4d listed in Table 6 by a clear 2.0. For larger variant ViP-M, it can further boost the performance to a higher level 44.5. ViP can also outperform the counterparts of Swin Transformer.

Integrating ViP into Cascade Mask RCNN. Table 7 shows the results when incorporating different backbones into Cascade Mask RCNN [21]. As shown in Table 7, all variants of the ViP family can achieve better performance compared to

their counterparts. Notably, as a tiny model with only 10M parameters and 30G FLOPS, ViP-Ti can achieve comparable performance obtained by the largest variant in ResNe(X)t family ResNeXt-101-64×4d which contains 81M parameters and 317G FLOPS. Meanwhile, ViP-S can perform slightly better than Swin-T with about 20% fewer computational cost. ViP also scales well with larger models. ViP-M further lifts the performance to 49.9 for object detection and 43.5 for instance segmentation.

4.5 Ablation studies

Some of ablation studies and visualization results are added to the supplementary material due to the limit of space.

Number of parts. As shown in Table 2, the number of parts N is crucial when the model is small. Concretely, for ViP-Ti, N=32 can lead to a remarkable 0.8% improvement on ImageNet compared to N=16. However, the improvement comes to be saturated when adding more parts to ViP-Ti.

Effects of the part-wise linear. Different from the original design in Transformer [273] that uses a self-attention module for part-wise communication, we replace it with a simple linear operation to save the computational cost. Introducing such a simple linear operation into ViP-S can lead to a 0.4% gain on ImageNet with only a fractional increase in parameters (0.03M) and FLOPS (0.02G).

Compatibility with shifted windows. ViP can be also compatible with shifted windows like what was proposed in [188]. As shown in Table 3, the shifted window can further boost ViP-S by 0.4%. Since the model structure is not the major focus of this paper, we do not apply this to all variants of the ViP family for simplicity.

5 Conclusion

In this work, we construct a unified framework for visual tasks *e.g.* image classification, object detection and panoptic segmentation. ViP can generate compositional

visual representation by dividing it into the part level and the whole level with a novel encoder-decoder interaction. Unlike previous frameworks that require task-specific structures, our ViP can work as a stand-alone framework for downstream tasks. Extensive experiments demonstrate that the stand-alone ViP without task-specific decoders can achieve competitive results on the three vision tasks.

Limitation and broader impact. (1) Though the stand-alone ViP can outperform previous pure transformer-based methods like YOLOS [88], it is not dominant when compared to the state-of-the-art task-specific methods for object detection. This may be due to the lack of multi-scale features when applying ViP for object detection, and can be further optimized by adding top-up structures like FPN.

(2) Some of the model structures in ViP *e.g.* feature pyramid, window partitions and convolution in Transformers can be found in some of the existing works. Instead of optimizing the model in terms of structures, the major focus of this paper is to find a uniform solution for most visual tasks.

(3) This paper presents how to unify pipelines of different tasks in terms of model frameworks, however in this project, we do not train ViP jointly on datasets for simultaneous multi-task learning though it is applicable. Since ViP can produce compositional representations that required by the downstream tasks, all these features can also be pre-trained with data with higher volume and variety. Due to the limitation of computing resources, we leave this to future work.

(4) This paper only pre-trains the ViP in a fully supervised way. In future works, we expect ViP to be pre-trained in a self-supervised or a semi-supervised approach so that we can construct a stronger backbone encoder like BERT in the field of NLP for most downstream tasks.

Chapter 6

ReMaX: Relaxing for Better Training on Efficient Panoptic Segmentation

The paper has been accepted for publication at the Thirty-seventh Annual Conference on Neural Information Processing Systems (NeurIPS) 2023.

ReMaX: Relaxing for Better Training on Efficient Panoptic Segmentation

Shuyang Sun[1] Weijun Wang[2] Qihang Yu[2] Andrew Howard[2] Philip Torr[1] Liang-Chieh Chen[2]

[1]University of Oxford [2]Google Research

kevinsun@robots.ox.ac.uk

Abstract

This paper presents a new mechanism to facilitate the training of mask transformers for efficient panoptic segmentation, democratizing its deployment. We observe that due to its high complexity, the training objective of panoptic segmentation will inevitably lead to much higher false positive penalization. Such unbalanced loss makes the training process of the end-to-end mask-transformer based architectures difficult, especially for efficient models. In this paper, we present ReMaX that adds relaxation to mask predictions and class predictions during training for panoptic segmentation. We demonstrate that via these simple relaxation techniques during training, our model can be consistently improved by a clear margin **without** any extra computational cost on inference. By combining our method with efficient backbones like MobileNetV3-Small, our method achieves new state-of-the-art results for efficient panoptic segmentation on COCO, ADE20K and Cityscapes. Code and pre-trained checkpoints will be available at https://github.com/google-research/deeplab2.

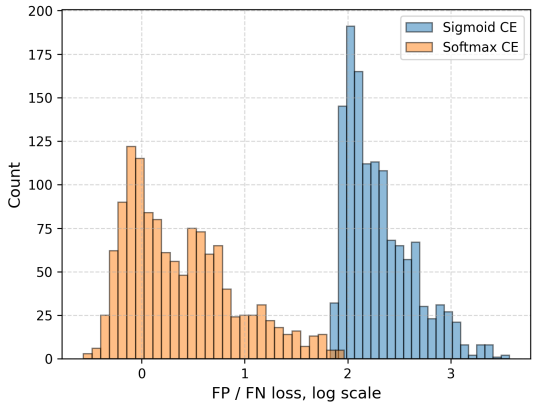


Figure 1. The histogram shows the ratio of false positives to false negatives for the cross-entropy loss, on a logarithmic scale. When using sigmoid as the activation function, the false positive loss is always over 100× greater than the false negative, making the total loss to be extremely unbalanced.

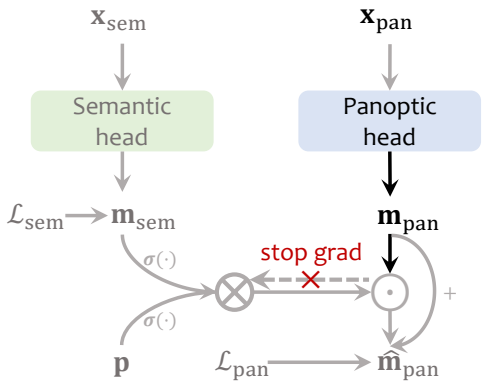


Figure 2. ReMask Operation. Modules, representations and operations rendered in gray are ***not*** used in testing. ⊗ and ⊙ represent the matrix multiplication and Hadamard multiplication and + means element-wise sum. The × symbol and "stop grad" mean that there is no gradient flown to $\mathbf{m}_{\texttt{sem}}$ from $\mathcal{L}_{\texttt{pan}}$ during training.

1 Introduction

Panoptic segmentation [150] aims to provide a holistic scene understanding [270] by unifying instance segmentation [101] and semantic segmentation [108]. The comprehensive understanding of the scene is obtained by assigning each pixel a label, encoding both semantic class and instance identity. Prior works adopt separate segmentation modules, specific to instance and semantic segmentation, followed by another fusion module to resolve the discrepancy [49, 149, 171, 218, 309, 316]. More recently, thanks to the transformer architecture [24, 273], mask transformers [50, 52, 174, 280, 323, 324, 344] are proposed for end-to-end panoptic segmentation by directly predicting class-labeled masks.

Although the definition of panoptic segmentation only permits each pixel to be associated with just one mask entity, some recent mask transformer-based works [48, 52, 166, 344] apply sigmoid cross-entropy loss (*i.e.*, not enforcing a single prediction via softmax cross-entropy loss) for mask supervision. This allows each pixel to be associated with multiple mask predictions, leading to an extremely unbalanced loss during training. As shown in Figure 1, when using the sigmoid cross-entropy

loss to supervise the mask branch, the false-positive (FP) loss can be even $10^3\times$ larger than the false-negative (FN) loss. Surprisingly, such unbalanced loss leads to better results than using softmax cross-entropy, which indicates that the gradients produced by the FP loss are still helpful for better performance.

However, the radical imbalance in the losses makes it difficult for the network to produce confident predictions, especially for efficient backbones [125, 126, 237], as they tend to make more mistakes given the smaller model size. Meanwhile, the training process will also become unstable due to the large scale loss fluctuation. To address this issue, recent approaches [24, 48, 52, 166] need to carefully clip the training gradients to a very small value like 0.01; otherwise, the loss would explode and the training would collapse. In this way, the convergence of the network will also be slower. A natural question thus emerges: *Is there a way to keep those positive gradients, while better stabilizing the training of the network?*

To deal with the aforementioned conflicts in the learning objectives, one naïve solution is to apply *weighted* sigmoid cross entropy loss during training. However, simply applying the hand-crafted weights would equivalently scale the losses for all data points, which means those positive and helpful gradients will be also scaled down. Therefore, in this paper, we present a way that can adaptively adjust the loss weights by only adding training-time relaxation to mask-transformers [50, 52, 174, 280, 324, 344]. In particular, we propose two types of relaxation: Relaxation on Masks (ReMask) and Relaxation on Classes (ReClass).

The proposed ReMask is motivated by the observation that semantic segmentation is a relatively easier task than panoptic segmentation, where only the predicted semantic class is required for each pixel without distinguishing between multiple instances of the same class. As a result, semantic segmentation prediction could serve as a coarse-grained task and guide the semantic learning of panoptic segmentation. Specifically, instead of directly learning to predict the panoptic masks, we add another auxiliary branch during training to predict the semantic segmentation

outputs for the corresponding image. The panoptic prediction is then calibrated by the semantic segmentation outputs to avoid producing too many false positive predictions. In this way, the network can be penalized less by false positive losses.

The proposed ReClass is motivated by the observation that each predicted mask may potentially contain regions involving multiple classes, especially during the early training stage, although each ground-truth mask and final predicted mask should only contain one target in the mask transformer framework [280]. To account for this discrepancy, we replace the original one-hot class label for each mask with a softened label, allowing the ground-truth labels to have multiple classes. The weights of each class is determined by the overlap of each predicted mask with all ground-truth masks.

By applying such simple techniques for relaxation to the state-of-the-art kMaX-DeepLab [324], our method, called ReMaX, can train the network stably without any gradient-clipping operation with a over 10× greater learning rate than the baseline. Experimental results have shown that our method not only speeds up the training by 3×, but also leads to much better results for panoptic segmentation. Overall, ReMaX sets a new state-of-the-art record for efficient panoptic segmentation. Notably, for efficient backbones like MobileNetV3-Small and MobileNetV3-Large [125], our method can outperform the strong baseline by 4.9 and 5.2 in PQ on COCO panoptic for short schedule training; while achieves 2.9 and 2.1 improvement in PQ for the final results (*i.e.*, long schedules). Meanwhile, our model with a Axial-ResNet50 (MaX-S) [282] backbone outperforms all state-of-the-art methods with 3× larger backbones like ConvNeXt-L [189] on Cityscapes [59]. Our model can also achieve the state-of-the-art performance when compared with the other state-of-the-art efficient panoptic segmentation architectures like YOSO [129] and MaskConver [129] on COCO [179], ADE20K [352] and Cityscapes [59] for efficient panoptic segmentation.

2 Related Work

Mask Transformers for image segmentation. Recent advancements in image segmentation has proven that Mask Transformers [280], which predict class-labeled object masks through the Hungarian matching of predicted and ground truth masks using Transformers as task decoders [24, 273], outperform box-based methods [149, 219, 309] that decompose panoptic segmentation into multiple surrogate tasks, such as predicting masks for detected object bounding boxes [105] and fusing instance and semantic segmentation [38, 193] with merging modules [49, 169, 171, 182, 218, 316]. The Mask Transformer based methods rely on converting object queries to mask embedding vectors [138, 263, 290], which are then multiplied with pixel features to generate predicted masks. Other approaches such as Segmenter [248] and MaskFormer [50] have also used mask transformers for semantic segmentation. K-Net [344] proposes dynamic kernels for generating masks. CMT-DeepLab [323] suggests an additional clustering update term to improve transformer's cross-attention. Panoptic Segformer [174] enhances mask transformers with deformable attention [361]. Mask2Former [50] adopts masked-attention, along with other technical improvements such as cascaded transformer decoders [24], deformable attention [361], and uncertainty-based point supervision [152], while kMaX-DeepLab [324] employs k-means cross-attention. OneFormer [136] extends Mask2Former with a multi-task train-once design. Our work builds on top of the modern mask transformer, kMaX-DeepLab [324], and adopts novel relaxation methods to improve model capacity.

The proposed Relaxation on Masks (ReMask) is similar to the masked-attention in Mask2Former [50] and the k-means attention in kMaX-DeepLab [324] in the sense that we also apply pixel-filtering operations to the predicted masks. However, our ReMask operation is fundamentally distinct from theirs in several ways: (1) we learn the threshold used to filter pixels in panoptic mask predictions through a semantic head during training, while both masked-attention [50] and k-means attention [324]

use either hard thresholding or argmax operation on pixel-wise confidence for filtering; (2) our approach relaxes the training objective by applying a pixel-wise semantic loss on the semantic mask for ReMask, while they do not have explicit supervision for that purpose; and (3) we demonstrate that ReMask can complement k-means attention in Section 4.

Acceleration for Mask Transformers for efficient panoptic segmentation. DETR [24] successfully proves that Transformer-based approaches can be used as decoders for panoptic segmentation, however, it still suffer from the slow training problem which requires over 300 epochs for just one go. Recent works [50, 203, 324, 361] have found that applying locality-enhanced attention mechanism can help to boost the speed of training for instance and panoptic segmentation. Meanwhile, some other works [144, 174, 344] found that by removing the bi-partite matching for *stuff* classes and applying a separate group of mask queries for *stuff* classes can also help to speed up the convergence. Unlike them, which apply architectural level changes to the network, our method only applies training-time relaxation to the framework, which do not introduce any extra cost during testing. Apart from the training acceleration, recent works [49, 124, 129, 206, 228] focus on how to make the system for panoptic segmentation more efficient. However, all these works focus on the modulated architecutural design while our approach focus on the training pipeline, which should be two orthogonal directions.

Coarse-to-fine refinement for image segmentation. In the field of computer vision, it is a common practice to learn representations from coarse to fine, particularly in image segmentation. For instance, DeepLab [36, 38] proposes a graph-based approach [39, 155] that gradually refines segmentation results. Recently, transformer-based methods for image segmentation such as [50, 98, 174, 280, 304, 344] have also adopted a multi-stage strategy to iteratively improve predicted segmentation outcomes in transformer decoders. The concept of using coarse-grained features (*e.g.*, semantic segmentation) to adjust fine-grained predictions (*e.g.*, in-

stance segmentation) is present in certain existing works, including [3, 4, 47]. However, these approaches can lead to a substantial increase in model size and number of parameters during both training and inference. By contrast, our ReMaX focuses solely on utilizing the coarse-fine hierarchy for relaxation *without* introducing any additional parameters or computational costs during inference.

Regularization and relaxation techniques. The proposed Relaxation on Classes (ReClass) involves adjusting label weights based on the prior knowledge of mask overlaps, which is analogous to the re-labeling strategy employed in CutMix-based methods such as [30, 334], as well as label smoothing [261] used in image classification. However, the problem that we are tackling is substantially different from the above label smoothing related methods in image classification. In image classification, especially for large-scale single-class image recognition benchmarks like ImageNet [234], it is unavoidable for images to cover some of the content for other similar classes, and label smoothing is proposed to alleviate such labelling noise into the training process. However, since our approach is designed for Mask Transformers [50, 52, 280, 323, 324] for panoptic segmentation, each image is precisely labelled to pixel-level, there is no such label noise in our dataset. We observe that other than the class prediction, the Mask Transformer approaches also introduce a primary class identification task for the class head. The proposal of ReClass operation reduces the complexity for the classification task in Mask Transformers. Prior to the emergence of Mask Transformers, earlier approaches did not encounter this issue as they predicted class labels directly on pixels instead of on masks.

3 Method

Before delving into the details of our method, we briefly recap the framework of mask transformers [280] for end-to-end panoptic segmentation. Mask Transformers like [50, 174, 280, 304, 344] perform both semantic and instance segmentation on the entire image using a single Transformer-based model. These approaches basically

divide the entire framework into 3 parts: a backbone for feature extraction, a pixel decoder with feature pyramid that fuses the feature generated by the backbone, and a transformer mask decoder that translates features from the pixel decoder into panoptic masks and their corresponding class categories.

In the transformer decoder, a set of mask queries is learnt to segment the image into a set of masks by a mask head and their corresponding categories by a classification head. These queries are updated within each transformer decoder (typically, there are at least 6 transformer decoders) by the cross-attention mechanism [273] so that the mask and class predictions are gradually refined. The set of predictions are matched with the ground truth via bipartite matching during training; while these queries will be filtered with different thresholds as post-processing during inference.

3.1 Relaxation on Masks (ReMask)

The proposed Relaxation on Masks (ReMask) aims to ease the training of panoptic segmentation models. Panoptic segmentation is commonly viewed as a more intricate task than semantic segmentation, since it requires the model to undertake two types of segmentation (namely, instance segmentation and semantic segmentation). In semantic segmentation, all pixels in an image are labeled with their respective class, without distinguishing between multiple instances (*things*) of the same class. As a result, semantic segmentation is regarded as a more coarse-grained task when compared to panoptic segmentation. Current trend in panoptic segmentation is to model *things* and *stuff* in a unified framework and resorts to train both the coarse-grained segmentation task on *stuff* and the more fine-grained segmentation task on *things* together using a stricter composite objective on *things*, which makes the model training more difficult. We thus propose ReMask to exploit an auxiliary semantic segmentation branch to facilitate the training.

Definition. As shown in Figure 2, given a mask representation $\mathbf{x}_{\texttt{pan}} \in \mathbb{R}^{HW \times N_Q}$, we apply a panoptic mask head to generate panoptic mask logits $\mathbf{m}_{\texttt{pan}} \in \mathbb{R}^{HW \times N_Q}$.

A mask classification head to generate the corresponding classification result $\mathbf{p} \in \mathbb{R}^{N_Q \times N_C}$ is applied for each query representation $\mathbf{q} \in \mathbb{R}^{N_Q \times d_q}$. A semantic head is applied after the semantic feature $\mathbf{x}_{\texttt{sem}} \in \mathbb{R}^{HW \times d_{\texttt{sem}}}$ from the pixel decoder to produces a pixel-wise semantic segmentation map $\mathbf{m}_{\texttt{sem}} \in \mathbb{R}^{HW \times N_C}$ assigning a class label to each pixel. Here H, W represent the height and width of the feature, N_Q is the number of mask queries, N_C denotes the number of semantic classes for the target dataset, d_q is the number of channels for the query representation, and $d_{\texttt{sem}}$ is the number of channels for the input of semantic head. As for the structure for semantic head, we apply an ASPP module [38] and a 1×1 convolution layer afterwards to transform $d_{\texttt{sem}}$ channels into N_C channels as the semantic prediction. Note that the whole auxiliary semantic branch will be skipped during inference as shown in Figure 2. Since the channel dimensionality between $\mathbf{m}_{\texttt{sem}}$ and $\mathbf{m}_{\texttt{pan}}$ is different, we map the semantic masks into the panoptic space by:

$$\widehat{\mathbf{m}}_{\texttt{sem}} = \sigma(\mathbf{m}_{\texttt{sem}})\sigma(\mathbf{p}^{\intercal}), \tag{6.1}$$

where $\sigma(\cdot)$ function represents the sigmoid function that normalizes the logits into interval $[0, 1]$. Then we can generate the relaxed panoptic outputs $\widehat{\mathbf{m}}_{\texttt{pan}}$ in the semantic masking process as follows:

$$\widehat{\mathbf{m}}_{\texttt{pan}} = \mathbf{m}_{\texttt{pan}} + (\widehat{\mathbf{m}}_{\texttt{sem}} \odot \mathbf{m}_{\texttt{pan}}), \tag{6.2}$$

where the $\odot$ represents the Hadamard product operation. Through the ReMask operation, the false positive predictions in $\mathbf{m}_{\texttt{pan}}$ can be suppressed by $\widehat{\mathbf{m}}_{\texttt{sem}}$, so that during training each relaxed mask query can quickly focus on areas of their corresponding classes. Here we apply identity mapping to keep the original magnitude of $\mathbf{m}_{\texttt{pan}}$ so that we can remove the semantic branch during testing. This makes ReMask as a complete relaxation technique that does not incur any overhead cost during testing. The re-scaled panoptic outputs $\widehat{\mathbf{m}}_{\texttt{pan}}$ will be supervised by the losses $\mathcal{L}_{\texttt{pan}}$.

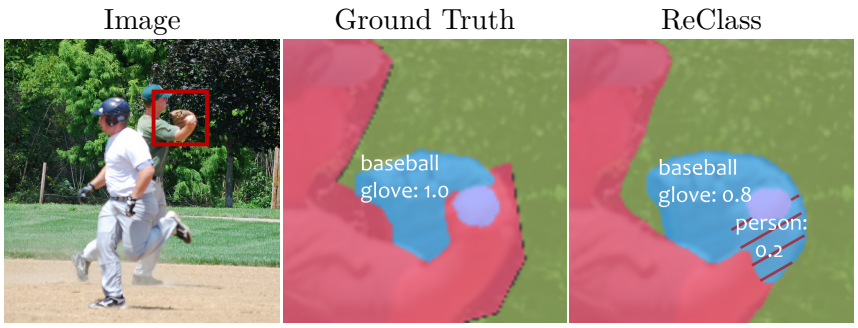


Figure 3. Demonstration on How ReClass works. We utilize the mask rendered in blue as an example. Our ReClass operation aims to soften the class-wise ground truth by considering the degree of overlap between the prediction mask and the ground truth mask. The blue mask intersects with both masks of "baseball glove" and "person", so the final class weights contain both and the activation of "person" in the prediction will no longer be regarded as a false positive case during training.

Stop gradient for a simpler objective to $\widehat{\mathbf{m}}_{\texttt{sem}}$. In order to prevent the losses designed for panoptic segmentation from affecting the parameters in the semantic head, we halt the gradient flow to $\mathbf{m}_{\texttt{sem}}$, as illustrated in Figure 2. This means that the semantic head is solely supervised by a semantic loss $\mathcal{L}_{\texttt{sem}}$, so that it can focus on the objective of semantic segmentation, which is a less complex task.

How does ReMask work? As defined above, there are two factors that ReMask operation helps training, (1) the Hadamard product operation between the semantic outputs and the panoptic outputs that helps to suppress the false positive loss; and (2) the *relaxation* on training objectives that trains the entire network simultaneously with consistent (*coarse-grained*) semantic predictions. Since the semantic masking can also enhance the locality of the transformer decoder like [50, 324], we conducted experiments by replacing $\mathbf{m}_{\texttt{sem}}$ with ground truth semantic masks to determine whether it is the training relaxation or the local enhancement that improves the training. When $\mathbf{m}_{\texttt{sem}}$ is assigned with ground truth, there will be no $\mathcal{L}_{\texttt{sem}}$ applied to each stage, so that $\mathbf{m}_{\texttt{pan}}$ is applied with the most accurate local enhancement. In this way, there are large amount of false positive predictions masked by the ground truth semantic masks, so that the false positive gradient will be greatly reduced. The results will be reported in Section 4.

3.2 Relaxation on Classes (ReClass)

Mask Transformers [50, 174, 280, 324] operate under the assumption that each mask prediction corresponds to a single class, and therefore, the ground truth for the classification head are one-hot vectors. However, in practice, each imperfect mask predicted by the model during the training process may intersect with multiple ground truth masks, especially during the early stage of training. As shown in Figure 3, the blue mask, which is the mask prediction, actually covers two classes ("baseball glove" and "person") defined in the ground truth. If the class-wise ground truth only contains the class "baseball glove", the prediction for "person" will be regarded as a false positive case. However, the existence of features of other entities would bring over-penalization that makes the network predictions to be under-confident.

To resolve the above problem, we introduce another relaxation strategy on class logits, namely Class-wise Relaxation (ReClass), that re-assigns the class confidence for the label of each predicted mask according to the overlap between the predicted and ground truth semantic masks. We denote the one-hot class labels as $\mathbf{y}$, the ground truth binary semantic masks as $\mathcal{S} = [\mathbf{s}_0, ..., \mathbf{s}_{HW}] \in \{0, 1\}^{HW \times N_C}$, the supplement class weights is calculated by:

$$\mathbf{y}_m = \frac{\sigma(\mathbf{m}_{\text{pan}})^\intercal \mathcal{S}}{\sum_i^{HW} \mathbf{s}_i}, \tag{6.3}$$

where $\mathbf{y}_m$ denotes the label weighted by the normalized intersections between the predicted and the ground truth masks. With $\mathbf{y}_m$, we further define the final class weight $\widehat{\mathbf{y}} \in [0, 1]^{N_C}$ as follows:

$$\widehat{\mathbf{y}} = \eta \mathbf{y}_m + (1 - \eta \mathbf{y}_m)\mathbf{y}, \tag{6.4}$$

where the η denotes the smooth factor for ReClass that controls the degree of the relaxation applying to the classification head.

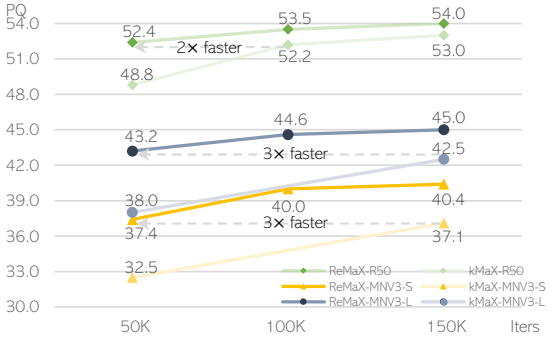


Figure 4. Performance on COCO *val* compared to the baseline kMaX-DeepLab [324]. ReMaX can lead to 3× faster convergence compared to the baseline, and can improve the baselines by a clear margin. The performance of ResNet-50 can be further improved to 54.2 PQ when the model is trained for 200K iterations.

Method	Backbone	Resolution	FPS	PQ
Pan-DeepLab [49]	MNV3-L [125]	641×641	26.3	30.0
Pan-DeepLab [49]	R50 [102]	641×641	20.0	35.1
Real-time [124]	R50 [102]	800×1333	15.9	37.1
MaskConver [228]	MN-MH [54]	640×640	40.2	37.2
MaskFormer [50]	R50 [102]	800×1333	17.6	46.5
YOSO [129]	R50 [102]	800×1333	23.6	48.4
YOSO [129]	R50 [102]	512×800	45.6	46.4
kMaX [324]	R50 [102]	1281×1281	16.3	53.0
ReMaX-T†	MNV3-S [125]	641×641	108.7	**40.4**
ReMaX-S†	MNV3-L [125]	641×641	80.9	**44.6**
ReMaX-M‡	R50 [102]	641×641	51.9	**49.1**
ReMaX-B	R50 [102]	1281×1281	16.3	**54.2**

Table 1. Comparison with other state-of-the-art efficient models (≥ 15 FPS) on COCO *val* set. The Pareto curve is shown in Figure 5 (b). The FPS of all models are evaluated on a NVIDIA V100 GPU with batch size 1. †‡ represent the application of efficient pixel and transformer decoders. Please check the appendix for details.

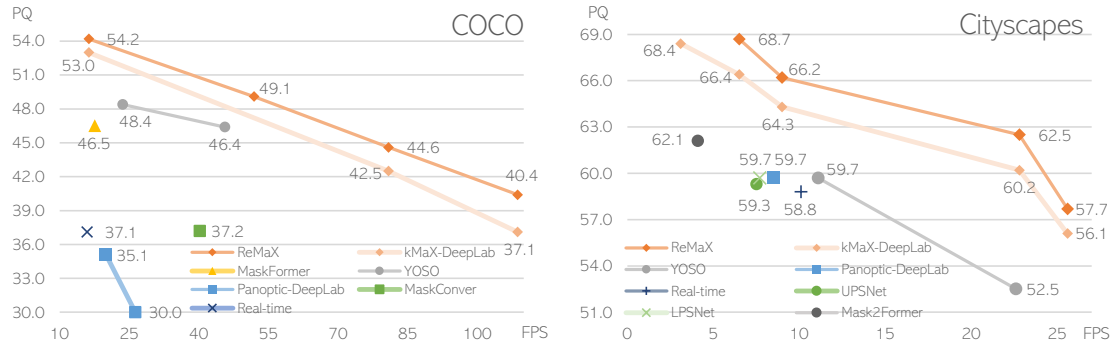
4 Experimental Results

4.1 Datasets and Evaluation Metric.

Our study of ReMaX involves analyzing its performance on three commonly used image segmentation datasets. **COCO** [179] supports semantic, instance, and panoptic segmentation with 80 "things" and 53 "stuff" categories; **Cityscapes** [59] consists of 8 "things" and 11 "stuff" categories; and **ADE20K** [352] contains 100 "things" and 50 "stuff" categories. We evaluate our method using the Panoptic Quality (PQ) metric defined in [150] (for panoptic segmentation), the Average Precision defined in [179] (for instance segmentation), and the mIoU [84] metric (for semantic segmentation).

4.2 Results on COCO Panoptic

Implementation details. The macro-architecture of ReMaX basically follows kMaX-DeepLab [324], while we incorporate our modules introduced in Section 3 into the corresponding heads. Concretely, we use the *key* in each k-means cross-attention operation as $\mathbf{x}_{\texttt{sem}}$ defined in Figure 2. The semantic head introduced during training



(a) (b)

Figure 5. FPS-PQ Pareto curve on (a) **COCO Panoptic *val* set** and (b) **Cityscapes *val* set**. Details of the corresponding data points can be found in Table 1 and 8. We compare our method with other state-of-the-art efficient pipelines for panoptic segmentation including kMaX-DeepLab [324], Mask2Former [50], YOSO [129], Panoptic-DeepLab [49], Real-time Panoptic Segmentation [124], UPSNet [309], LPSNet [123], MaskFormer [52], and MaskConver [228].

consists of an ASPP module [38] and a 1×1 convolution that outputs N_C number of channels. The specification of models with different size is introduced in the appendix.

Training details. We basically follow the training recipe proposed in kMaX-DeepLab [324] but make some changes to the hyper-parameters since we add more relaxation to the network. Here we high-light the necessary and the full training details and specification of our models can be also found in the appendix. The learning rate for the ImageNet-pretrained [234] backbone is multiplied with a smaller learning rate factor 0.1. For training augmentations, we adopt multi-scale training by randomly scaling the input images with a scaling ratio from 0.3 to 1.7 and then cropping it into resolution 1281×1281. Following [280, 323, 324], we further apply random color jittering [60], and panoptic copy-paste augmentation [144, 242] to train the network. DropPath [133, 163] is applied to the backbone, the transformer decoder. AdamW [147, 196] optimizer is used with weight decay 0.005 for short schedule 50K and 100K with a batch size 64. For long schedule, we set the weight decay to 0.02. The initial learning rate is set to 0.006, which is multiplied by a decay factor of 0.1 when the training reaches 85% and 95% of the total iterations. The

entire framework is implemented with DeepLab2 [292] in TensorFlow [1]. Following [280], we apply a PQ-style loss, a Mask-ID cross-entropy loss, and the instance discrimination loss to better learn the feature extracted from the backbone.

For all experiments if not specified, we default to use ResNet-50 as the backbone and apply ReMask to the first 4 stages of transformer decoder. The η for ReClass operation is set to 0.1. All models are trained for 27 epochs (*i.e.*, 50K iterations). The loss weight for the semantic loss applied to each stage in the transformer decoder is set to 0.5.

ReMaX significantly improves the training convergence and outperforms the baseline by a large margin. As shown in Figure 4, we can see that when training the model under different training schedules 50K, 100K and 150K, our method outperform the baselines by a clear margin for all different schedules. Concretely, ReMaX can outperform the state-of-the-art baseline kMaX-DeepLab by a significant 3.6 PQ when trained under a short-term schedule 50K iterations (27 epochs) for backbone ResNet-50. Notably, our model trained with only 50K iterations performs even better than kMaX-DeepLab [324] trained for the 100K iterations (54 epochs), which means that our model can speed up the training process by approximately 2×. We kindly note that the performance of ResNet-50 can be further improved to 54.2 PQ for 200K iterations. ReMaX works very well with efficient backbones including MobileNetV3-Small [125] and MobileNetV3-Large [125], which surpass the baseline performance by 4.9 and 5.2 PQ for 50K iterations, and 3.3 and 2.5 PQ respectively for 150K iterations. These results demonstrate that the proposed relaxation can significantly boost the convergence speed, yet can lead to better results when the network is trained under a longer schedule.

ReMaX *vs.* other state-of-the-art models for efficient panoptic segmentation. Table 1 and Figure 5 (a) compares our method with other state-of-the-art methods for efficient panoptic segmentation on COCO Panoptic. We present 4 models with different resolution and model capacity, namely ReMaX-Tiny (T), ReMaX-

Activation	*w/* ReMaX?	*w/* grad-clip?	PQ
`softmax`	×	×	48.8
`softmax`	✓	×	49.5
`sigmoid`	×	×	50.4
`sigmoid`	×	✓	51.2
`sigmoid`	✓	×	**52.4**

Table 2. The impact of activation function and gradient clipping.

#ReMasks	0	2	4	6
PQ	50.4	51.9	**52.4**	51.5

Table 3. The effect of number of ReMask applied. ReMaX performs the best when ReMask is applied to the first 4 stages of the transformer decoder.

η	0	0.01	0.05	0.1	0.2
PQ	51.7	51.7	51.9	**52.4**	51.5

Table 4. The impact of differnt η defined in Eq. 6.4 for ReClass. Here we observe that the result reaches its peak when $\eta = 0.1$.

w/ identity mapping?	*w/* ReMask in test?	PQ
✓	×	**52.4**
✓	✓	**52.4**
×	✓	52.1
×	×	51.9

Table 5. Effect of applying identity mapping and auxiliary head for ReMask during testing. Removing the auxiliary semantic head will not lead to performance drop when $\widehat{\mathbf{m}}_{\texttt{pan}}$ is applied with identity mapping.

Method	Backbone	FPS	PQ
MaskFormer [52]	R50 [102]	17.6	46.5
K-Net [344]		-	47.1
PanSegFormer [174]		7.8	49.6
Mask2Former [50]		8.6	51.9
kMaX [324]		26.3	53.0
MaskDINO [166]		16.8‡	53.0
ReMaX		26.3†	**54.2**

Table 6. Comparison on COCO *val* with other models using ResNet-50 as the backbone. †The FPS here is evaluated under resolution 1200 × 800 on V100 and the model is trained for 200K iterations. ‡ is evaluated using a A100 GPU.

w/ stop-grad?	*w/* gt?	PQ
✓	×	**52.4**
N/A	✓	45.1
×	×	36.6*

Table 7. The effect of stop gradient and gt-masking. The denotation ***w/* gt?** means whether we use ground-truth semantic masks for $\mathbf{m}_{\texttt{sem}}$. * The result without the stop-gradient operation does not well converge in training.

Small (S), ReMaX-Medium (M) and ReMaX-Base (B). Due to the limit of space, the detailed specification of these models is included in the appendix. According to the Pareto curve shown in Figure 5 (a), our approach outperforms the previous state-of-the-art efficient models by a clear margin. Specifically, on COCO Panoptic *val* set, our models achieve 40.4, 44.6, 49.1 and 54.2 PQ with 109, 81, 52 and 16 FPS for ReMaX-T, ReMaX-S, ReMaX-M and ReMaX-B respectively. The speed of these models is evaluated under the resolution 641 × 641 except for ReMaX-Base, which is evaluated under resolution 1281 × 1281. Meanwhile, as shown in Table 6, our largest model with the backbone ResNet-50 also achieves better performance than the other non-efficient state-of-the-art methods with the same backbone.

Effect of different activation, and the use of gradient clipping. Table 2 presents the effect of using different activation function (`sigmoid` *vs.* `softmax`) for

the Mask-ID cross-entropy loss and the $\sigma(\cdot)$ defined in Eq (6.1). From the table we observe that ReMask performs better when using `sigmoid` as the activation function, but our method can get rid of gradient clipping and still get a better result.

Why does ReMask work due to relaxation instead of enhancing the locality? As discussed in Section 3, to figure out whether it is the relaxation or the pixel filtering that improves the training, we propose experiments replacing $\mathbf{m}_{\texttt{sem}}$ with the ground truth semantic masks during training. When $\mathbf{m}_{\texttt{sem}}$ is changed into the ground truth, all positive predictions outside the ground-truth masks will be removed, which means that the false positive loss would be significantly scaled down. The huge drop (52.4 *vs.* 45.1 PQ in Table 7) indicates that the gradients of false positive losses can benefit the final performance. Table 7 also shows that when enabling the gradient flow from the panoptic loss to the semantic predictions, the whole framework cannot converge well and lead to a drastically drop in performance (36.6 PQ). The semantic masks $\mathbf{m}_{\texttt{sem}}$ faces a simpler objective (*i.e.* only semantic segmentation) if the gradient flow is halted.

The number of mask relaxation. Table 3 shows the effect of the number of ReMask applied to each stage, from which we can observe that the performance gradually increases and reaches its peak at 52.4 PQ when the number of ReMask is 4, which is also our final setting for all other ablation studies. Using too many ReMask (> 4) operations in the network may add too many relaxation to the framework, so that it cannot fit well to the final complex goal for panoptic segmentation.

ReClass can also help improve the performance for ReMaX. We investigate ReClass and its hyper-parameter η in this part and report the results in Table 4. In Table 4, we ablate 5 different η from 0 to 0.2 and find that ReClass performs the best when $\eta = 0.1$, leading to a 0.5 gain compared to the strong baseline. The efficacy of ReClass validates our assumption that each mask may cover regions of multiple classes.

Method	Backbone	FPS	PQ
Mask2Former [50]	R50 [102]	4.1	62.1
Pan-DeepLab [49]	X71 [53]	5.7	63.0
LPSNet [123]	R50 [102]	7.7	59.7
Pan-DeepLab [49]	R50 [102]	8.5	59.7
kMaX-DeepLab [324]	R50 [102]	9.0	64.3
Real-time [124]	R50 [102]	10.1	58.8
YOSO [129]	R50 [102]	11.1	59.7
kMaX-DeepLab [324]	MNV3-L [125]	22.8	60.2
ReMaX	R50 [102]	9.0	**65.4**
ReMaX	MNV3-L [125]	22.8	**62.5**
ReMaX	MNV3-S [125]	25.6	**57.7**

Table 8. Cityscapes *val* set results for lightweight backbones. We consider methods without pre-training on extra data like COCO [179] and Mapillary Vistas [212] and test-time augmentation for fair comparison. We evaluate our FPS with resolution 1025×2049 and a V100 GPU. The FPS for other methods are evaluated using the resolution reported in their original papers.

Method	Backbone	FPS	#params	PQ
Mask2Former [324]	Swin-L†	-	216M	66.6
kMaX-DeepLab [324]	MaX-S†	6.5	74M	66.4
kMaX-DeepLab [324]	ConvNeXt-L†	3.1	232M	68.4
OneFormer [136]	ConvNeXt-L†	-	220M	68.5
ReMaX	MaX-S† [125]	6.5	74M	**68.7**

Table 9. Cityscapes *val* set results for larger backbones. †Pre-trained on ImageNet-22k.

Method	Backbone	Resolution	FPS	PQ	mIoU
MaskFormer [52]	R50	640-2560	-	34.7	-
Mask2Former [50]		640-2560	-	39.7	46.1
YOSO [129]		640-2560	35.4	38.0	-
kMaX [324]		641×641	38.7	41.5	45.0
kMaX [324]		1281×1281	14.4	42.3	45.3
ReMaX	R50	641×641	38.7	**41.9**	**45.7**
ReMaX		1281×1281	14.4	**43.4**	**46.9**

Table 10. ADE20K *val* set results. Our FPS is evaluated on a NVIDIA V100 GPU under the corresponding resolution reported in the table.

Effect of the removing auxiliary semantic head for ReMask during testing. The ReMask operation can be both applied and removed during testing. In Table 5, it shows that the models perform comparably under the two settings. In Table 5 we also show the necessity of applying identity mapping to $\mathbf{m}_{\texttt{pan}}$ during training in order to remove the auxiliary semantic head during testing. Without the identity mapping at training, removing semantic head during testing would lead to 0.5 drop from 52.4 (the first row in Table 5) to 51.9.

4.3 Results on Cityscapes

Implementation details. Our models are trained using a batch size of 32 on 32 TPU cores, with a total of 60K iterations. The first 5K iterations constitute the warm-up stage, where the learning rate gradually increases from 0 to 3×10^{-3}. During training, the input images are padded to 1025×2049 pixels. In addition, we employ a multi-task loss function that includes four loss components with different weights. Specifically, the weights for the PQ-style loss, auxiliary semantic loss,

mask-id cross-entropy loss, and instance discrimination loss are set to 3.0, 1.0, 0.3 and 1.0, respectively. To generate feature representations for our model, we use 256 cluster centers and incorporate an extra bottleneck block in the pixel decoder, which produces features with an output stride of 2. These design are basically proposed in kMaX-DeepLab [324] and we simply follow here for fair comparison.

Results on Cityscapes. As shown in Table 8 and Figure 5 (b), it shows that our method can achieve even better performance when using a smaller backbone MobileNetV3-Large (62.5 PQ) while the other methods are based on ResNet-50. Meanwhile, our model with Axial-ResNet-50 (*i.e.*, MaX-S, 74M parameters) as the backbone can outperform the state-of-the-art models [136, 324] with a ConvNeXt-L backbone ($>$ 220M parameters). The Pareto curve in Figure 5 (b) clearly demonstrates the efficacy of our method in terms of speed-accuracy trade-off.

4.4 Results on ADE20K

Implementation details. We basically follow the same experimental setup as the COCO dataset, with the exception that we train our model for 100K iterations (54 epochs). In addition, we conduct experiments using input resolutions of 1281×1281 pixels and 641×641 respectively. During inference, we process the entire input image as a whole and resize longer side to target size then pad the shorter side. Previous approaches use a sliding window approach, which may require more computational resources, but it is expected to yield better performance in terms of accuracy and detection quality. As for the hyper-parameter for ReMask and ReClass, we used the same setting as what we propose on COCO.

Results on ADE20K. In Table 10, we compared the performance of ReMaX with other methods, using ResNet-50 as the backbone, and found that our model outperforms the baseline model by 1.6 in terms of mIOU, which is a clear margin compared to the baseline, since we do not require any additional computational cost but only the relaxation during training. We also find that our model can surpass

the baseline model kMaX-DeepLab by 1.1 in terms of PQ. When comparing with other frameworks that also incorporate ResNet-50 as the backbone, we show that our model is significantly better than Mask2Former and MaskFormer by 3.7 and 8.7 PQ respectively.

5 Conclusion

The paper presents a novel approach called ReMaX, comprising two components, ReMask and ReClass, that leads to better training for panoptic segmentation with Mask Transformers. The proposed method is shown to have a significant impact on training speed and final performance, especially for efficient models. We hope that our work will inspire further investigation in this direction, leading to more efficient and accurate panoptic segmentation models.

Acknowledgement. We would like to thank Xuan Yang at Google Research for her kind help and discussion. Shuyang Sun and Philip Torr are supported by the UKRI grant: Turing AI Fellowship EP/W002981/1 and EPSRC/MURI grant: EP/N019474/1. We would also like to thank the Royal Academy of Engineering and FiveAI.

Appendix

A Loss Visualization of ReMaX

We visualize the loss applied with ReMask and the loss applied without ReMask in Figure A, from which we can observe that ReMask can effectively reduce extremely high false positive losses; therefore, our method can stabilize the training of the framework.

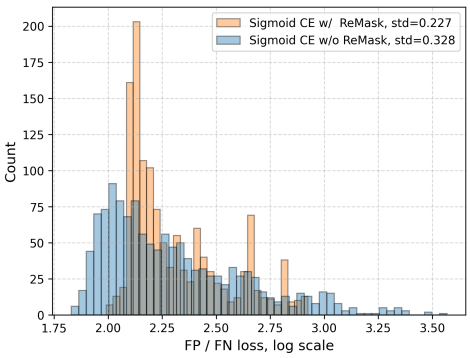


Figure A. The histogram shows the ratio of false positives to false negatives for the cross-entropy loss, on a logarithmic scale.

Method	Backbone	#params	FLOPs	FPS	PQ
kMaX	ConvNeXt-T†	61M	172G	21.8	55.3
ReMaX	ConvNeXt-T†	61M	172G	21.8	**55.9**
Mask2Former	Swin-B†	107M	466G	-	56.4
kMaX	ConvNeXt-S†	83M	251G	16.5	56.3
ReMaX	ConvNeXt-S†	83M	251G	16.5	**56.6**

Table A. Results for larger models on COCO *val* set. FLOPs and FPS are evaluated with the input size 1200×800 and a V100 GPU. †: ImageNet-22K pretraining.

Model	Backbone	Resolution	#Pixel Decoders	#Transformer Decoders	#FLOPs	#Params	FPS
ReMaX-T	MNV3-S [125]	641×641	[1, 1, 1, 1]	[1, 1, 1]	18.8G	18.6M	109
ReMaX-S	MNV3-L [125]	641×641	[1, 1, 1, 1]	[1, 1, 1]	20.9G	22.0M	81
ReMaX-M	R50 [102]	641×641	[1, 5, 1, 1]	[1, 1, 1]	67.8G	50.8M	52
ReMaX-B	R50 [102]	1281×1281	[1, 5, 1, 1]	[2, 2, 2]	294.7G	56.6M	26

Table B. Specification of different models in ReMaX family.

B Model Specification

We provide the specification of our models and their corresponding number of parameters and FLOPs in Table B. We kindly note that the numbers of pixel decoders with the format $[\cdot, \cdot, \cdot, \cdot]$ represent the numbers for features with $[\frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}]$ times of the input size. We use Axial attention [282] for all feature maps with resolution $\frac{1}{32}, \frac{1}{16}$ of the input size, and regular bottleneck residual blocks [102] for the rest. The denotation $[\cdot, \cdot, \cdot]$ for the transformer decoders represents the numbers for resolution of $[\frac{1}{16}, \frac{1}{8}, \frac{1}{4}]$ times of the input size.

C Performance for Larger Models

We also validate the performance of ReMaX for larger models *e.g.* ConvNeXt-Tiny (T) and ConvNeXt-Small (S). From Table A we can find that ReMaX can achieve better results compared to the baseline kMaX-DeepLab [324] and Mask2Former [50]. However, the improvement of ReMaX gets saturated when the numbers become high.

Notably, when using ConvNeXt-T backbone, ReMaX can lead to 0.6 PQ increase over kMaX-DeepLab, while incurring ***no extra*** computational cost during inference. The improvement is noticeable, as kMaX-DeepLab only further improves 1.0 PQ by using ConvNeXt-S backbone, at the cost of extra 36% more parameters (22M) and 46% more FLOPs (79G).

D Limitations

Since we implement our method in TensorFlow, the baselines we can build upon is limited. We will validate our approach on other baselines like Mask2Former [50] in PyTorch for future works. Meanwhile, ReClass measures the weight of each class according to the size of each mask, which may not be accurate and can be further improved in the future.

E Boarder Impact

Our method can help better train models for efficient panoptic segmentation. It can also be used to develop new applications in areas such as autonomous driving, robotics, and augmented reality. For example, in autonomous driving, efficient panoptic segmentation can be used to identify and track other vehicles, pedestrians, and obstacles on the road. This information can be used to help the car navigate safely. In robotics, efficient panoptic segmentation can be used to help robots understand their surroundings and avoid obstacles. This information can be used to help robots perform tasks such as picking and placing objects or navigating through cluttered environments. In augmented reality, efficient panoptic segmentation can be used to overlay digital information on top of the real world. This information can be used to provide users with information about their surroundings or to help them with tasks such as finding their way around a new city. Overall, our method can be used to boost a variety of applications in the field of computer vision and robotics.

Part III: Open-vocabulary Cross-granularity Visual Perception

Chapter 7

CLIP as RNN: Segment Countless Visual Concepts without Training Endeavor

The paper has been accepted for publication, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2024

CLIP as RNN: Segment Countless Visual Concepts without Training Endeavor

Shuyang Sun[*1,2] Runjia Li[*1] Philip Torr[1] Xiuye Gu[2] Siyang Li[2]

[1]University of Oxford [2]Google Research

{kevinsun, runjia, phst}@robots.ox.ac.uk {siyang, xiuyegu}google.com

https://torrvision.com/clip_as_rnn/

Abstract

Existing open-vocabulary image segmentation methods require a fine-tuning step on mask labels and/or image-text datasets. Mask labels are labor-intensive, which limits the number of categories in segmentation datasets. Consequently, the vocabulary capacity of pre-trained VLMs is severely reduced after fine-tuning. However, without fine-tuning, VLMs trained under weak image-text supervision tend to make suboptimal mask predictions. To alleviate these issues, we introduce a novel recurrent framework that progressively filters out irrelevant texts and enhances mask quality without training efforts. The recurrent unit is a two-stage segmenter built upon a frozen VLM. Thus, our model retains the VLM's broad vocabulary space and equips it with segmentation ability. Experiments show that our method outperforms not only the training-free counterparts, but also those fine-tuned with millions of data samples, and sets the new state-of-the-art records for both zero-shot semantic and referring segmentation. Concretely, we improve the current record by 28.8, 16.0, and 6.9 mIoU on Pascal VOC, COCO Object, and Pascal Context.

[*]The first two authors contribute equally to this work.

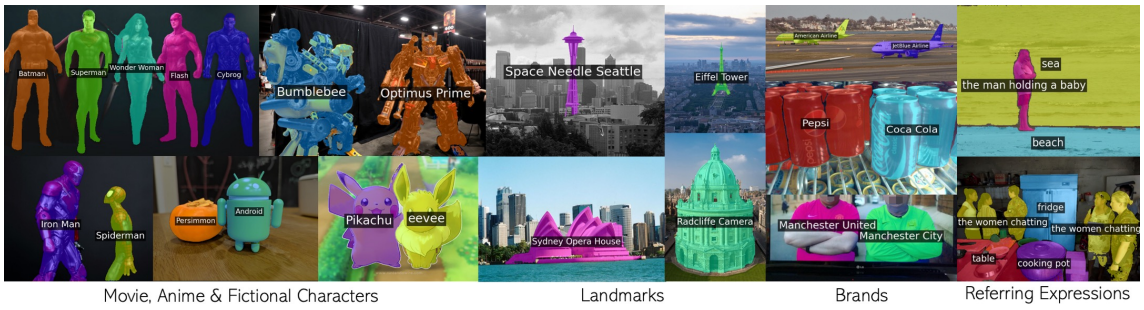


Figure 1. We propose *CLIP as RNN* (*CaR*) to segment concepts in a vast vocabulary, including fictional characters, landmarks, brands, everyday objects, and referring expressions. This figure shows our qualitative results. More visualizations are included in the supplementary material. Best viewed in color and with zoom-in.

1 Introduction

Natural language serves as a bridge to connect visual elements with human-communicable ideas by transforming colors, shapes, and objects *etc.* into descriptive language. On the other hand, human can use natural language to easily instruct computers and robotics to perform their desired tasks. Built upon the revolutionary vision-language model trained on Internet-scale image-text pairs, *e.g.*, CLIP [220], a variaty of studies [27, 175, 184, 199, 227, 241, 310, 322, 354] have explored using pre-trained VLMs for *open-vocabulary image segmentation* — to segment any concepts in the image described by arbitrary text queries.

Among these advances, several works [175, 185, 322] have integrated pre-trained VLMs with segmenters fine-tuned on bounding boxes and masks. While these methods exhibit superior performances on segmentation benchmarks with common categories, their ability to handle a broader vocabulary is hampered by the small category lists in the segmentation datasets used for fine-tuning. As depicted in Figure 2, even though all three methods incorporate CLIP [220], those relying on fine-tuning with mask annotations [175, 185] fail to recognize the concepts like *Pepsi* and *Coca Cola*.

Since box and mask annotations are expensive, another line of works [27, 184, 199, 227, 229, 310] seek to fine-tune the VLM and/or auxiliary segmentation modules with image-level annotations only, *e.g.*, paired image-text data obtained from the

Figure 2. Our method CaR can fully inherit the vast vocabulary space of CLIP, by directly using features from a pre-trained VLM, *i.e.*, CLIP, without any fine-tuning. Although the scene in the image is simple, state-of-the-art methods fine-tuned on segmentation datasets [175, 185] fail to segment and recognize *Pepsi* and *Coca Cola* correctly.

Internet. This would lead to a complicated fine-tuning pipeline. Besides, these segmentation models often have suboptimal mask qualities, as image-level labels cannot directly supervise pixel grouping.

In this paper, we eliminate the fine-tuning on mask annotations or additional image-text pairs to fully preserve the extensive vocabulary space of the pre-trained VLM. However, the pre-training objectives of VLMs are not specifically designed for dense predictions. As a result, existing approaches [43, 180, 354] that do not fine-tune the VLMs struggle to generate accurate visual masks corresponding to the text queries, particularly when some of the text queries refer to non-existing objects in the image. To address this issue, we repeatedly assess the degree of alignment between mask proposals and text queries, and progressively remove text queries with low confidence. As the text queries become cleaner, better mask proposals are consequently obtained. To facilitate this iterative refinement, we propose a novel recurrent architecture with a two-stage segmenter as the recurrent unit, maintaining the same set of weights across all time steps. The two-stage segmenter consists of a mask proposal generator and a mask classifier to assess the mask proposals. Both are built upon a pre-trained CLIP model without modifications. Given an input image and multiple text queries, our model recurrently aligns the visual and textual spaces and generates refined masks as the final output, continuing until a stable state is achieved. Owing to its recurrent nature, we name our entire framework as

CLIP as RNN (*CaR*).

Experimental results demonstrate our approach is remarkably effective. In comparison with methods that do not use additional training data, *i.e.*, zero-shot open-vocabulary semantic segmentation, our approach outperforms the prior art by 28.8, 16.0, and 6.9 mIoU on Pascal VOC [85], COCO Object [178], and Pascal Context [208], respectively. Impressively, even when pitted against models fine-tuned on extensive additional data, our strategy surpasses the best record by 12.6, 4.6, and 0.1 on the three aforementioned datasets, respectively. To assess our model's capacity to handle more complex text queries, we evaluate on the referring image segmentation benchmarks, Ref-COCO, RefCOCO+ and RefCOCOg. CaR outperforms the zero-shot counterparts by a large margin. Moreover, we extend our method to the video domain, and establish a zero-shot baseline for the video referring segmentation on Ref-DAVIS 2017 [143]. As showcased in Figure 1, our proposed approach CaR exhibits remarkable success across a broad vocabulary spectrum, effectively processing diverse queries from celebrities and landmarks to referring expressions and general objects.

Our contributions can be summarized as follows: 1. By constructing a recurrent architecture, our method CaR performs visual segmentation with arbitrary text queries in a vast vocabulary space without the need of fine-tuning. 2. When compared with previous methods on zero-shot open-vocabulary semantic segmentation and referring segmentation, our method CaR outperforms the prior state of the art by a large margin.

2 Related Work

Open-vocabulary segmentation with mask annotations. The success of VLMs [137, 170, 220, 253, 320, 335, 336] has motivated researchers to push the boundaries of traditional image segmentation tasks, moving them beyond fixed label sets and into an open vocabulary by fine-tuning or training VLMs on segmentation

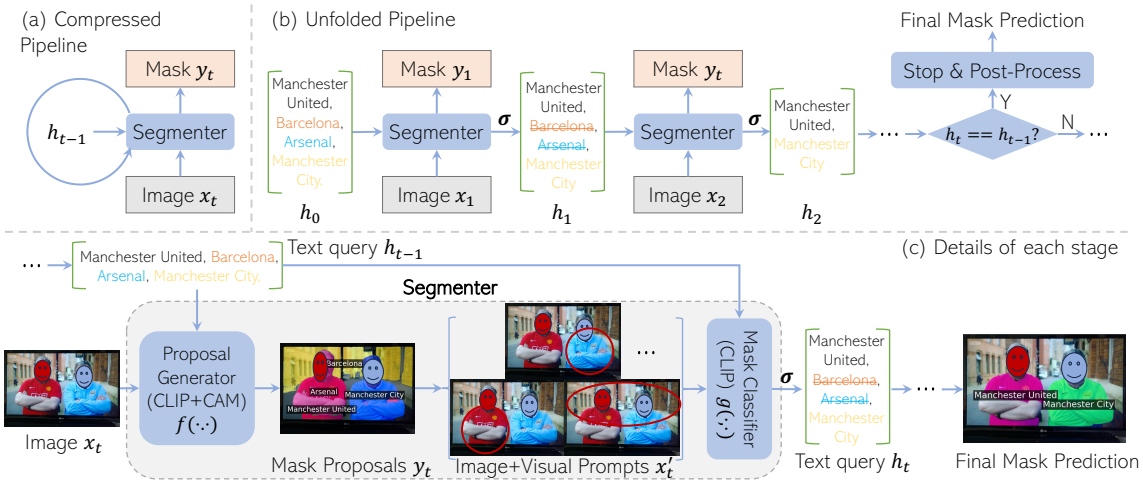


Figure 3. The overall framework of our method CaR. (a), (b): given an image, the user provides a set of text queries that they are interested to segment. This initial set, denoted by h_0, may refer to non-existing concepts in the image, *e.g.*, `Barcelona` and `Arsenal`. In the t-th time step, the frozen segmenter evaluates the degree of alignment between each mask and text query from the previous time step, h_{t-1}, and then low-confidence queries are eliminated by the function σ. **(c)** depicts the detailed architecture of our two-stage segmenter. It consists a mask proposal generator $f(\cdot,\cdot)$, and a mask classifier $g(\cdot,\cdot)$ that assesses the alignment of each mask-text pairs.

datasets [92, 99, 140, 165, 175, 185, 204, 311, 322, 338, 339, 355]. However, as collecting mask annotations for a vast range of fine-grained labels is prohibitively expensive, existing segmentation datasets, *e.g.* [18, 85, 178, 208, 353] have limited vocabularies. Methods fine-tuned on these mask annotations reduce the open-vocabulary capacity inherited from the pre-trained VLMs. In this work, we attempt to preserve the completeness of the vocabulary space in pre-trained VLMs.

Open-vocabulary segmentation without mask supervision. Several works [20, 27, 32, 107, 199, 209, 227, 229, 241, 310, 312, 354] avoid the aforementioned vocabulary reduction issue by not fine-tuning on any mask annotations. Instead, researchers allow semantic grouping to emerge automatically without any mask supervision. GroupViT [310] learns to progressively group semantic regions with weak supervision, using only image-text datasets. Furthermore, it is possible to use a pre-trained VLM for open-vocabulary segmentation without any additional training [141, 241, 354]. For example, MaskCLIP [354] enables CLIP to perform open-vocabulary segmentation by only modifying its image encoder. However, these

methods often suffer from inferior segmentation performance due to the lack of mask supervision, and the modification of the pre-trained VLMs. CaR is closely related to these approaches, we are both in a zero-shot manner without training. CaR stands out by proposing a recurrent framework on a VLM with fixed weights and no alternation on its architecture. Note that our zero-shot is different from the zero-shot semantic segmentation [8, 17, 77, 131, 168, 303, 354] that mirrors the seen/unseen class separation from zero-shot classification in earlier ages.

Segmentation with VLM-generated pseudo-labels. As an alternative direction, recent works have exploited pre-trained VLMs to generate pseudo-masks in a fixed label space, requiring only image-level labels or captions for training [2, 180, 198, 232, 241, 305, 314, 354]. Once pseudo mask labels are obtained, a segmenter with a fixed vocabulary (*e.g.*, DeepLab [36, 41]) can be trained in a fully supervised manner. Among these, CLIP-ES [180] is particularly relevant as it directly uses CLIP for pseudo-mask generation given the class names in ground-truth. However, CLIP-ES [180] requires pseudo-label training while we do not.

Progressive refinement for image segmentation. Progressive refinement in image segmentation has seen significant advancements through various approaches. Recent works [24, 50, 52, 257, 280, 324] such as Cascade R-CNN [22], DETR [24] and CRF-RNN [350] combine a detector (R-CNN [96]), a transformer [273] or a segmenter (denseCRF [156]) repeatedly for iterative refinement. We kindly note that all these works are designed for supervised image instance or semantic segmentation in a closed-set vocabulary. Our method does not require any training effort, yet our way of progressive refinement is fundamentally different from these methods.

3 CLIP as Recurrent Neural Networks

3.1 A Recap on Recurrent Neural Networks

We begin with a concise overview of recurrent neural networks (RNN). RNNs are specifically designed to process sequential data, such as text, speech, and time series. A basic RNN, commonly known as a vanilla RNN, uses the *same* set of weights to process data at all time steps. At each time step t, the process can be expressed as follows:

$$h_t = \sigma(W_{hh}h_{t-1} + W_{xh}x_t + b_h), \tag{7.1}$$

$$y_t = W_{hy}h_t + b_y. \tag{7.2}$$

x_t represents the input, and h_t represents the hidden state which serves as the "memory" to store information from previous steps. y_t denotes the output. W_{hh}, W_{xh}, and W_{hy} are weight matrices, b_h and b_y refer to bias terms, and σ denotes a thresholding function to introduce non-linearity.

An RNN's core lies in its hidden state, h_t, which captures information from past time steps. This empowers RNNs to exploit temporal dynamics within sequences. In our approach CaR, we design a similar process - iteratively aligning the textual and visual domains by assessing the accuracy of each text query through a segmenter, using the *same* set of weights as well. The text queries at each step act like the RNN's hidden state, representing the entities identified in the image at each specific time step.

3.2 Overview

As depicted in Figure 3(a) and (b), our training-free framework operates in a recurrent manner, with a fixed-weight segmenter shared across all time steps. In the t-th time step, the segmenter receives an image $x_t \in \mathbb{R}^{3\times H\times W}$ and a set of text queries h_{t-1} from the preceding step as the input. It then produces two outputs:

Algorithm 2 Pseudo-code of CLIPasRNN in PyTorch style.

```
# img: the input image with shape (3, H, W)
# h_0: a list of the initial N_0 text queries.
# clip: the CLIP model encoding the image and texts.
# cam: the gradient-based CAM model for mask proposal generation.
# eta: a threshold to binarize the masks for visual prompting.
# theta: a threshold defined in Eq. 6.

h_{t-1} = h_0
while len(h_{t-1}) > 0:
    # logits: [1, len(h_{t-1})]
    logits = clip(img, h_{t-1})
    scores = softmax(logits, dim=-1)
    # proposals: [len(h_{t-1}), H, W]
    proposals = cam(clip, img, scores)
    # prompted_img: [len(h_{t-1}), H, W]
    prompted_imgs = apply_visual_prompts(img, proposals, eta)
    # mask_logits: [len(h_{t-1}), len(h_{t-1})]
    mask_logits = clip(prompted_imgs, h_{t-1})
    mask_scores = softmax(mask_logits, dim=-1)
    # diag_scores: [len(h_{t-1})]
    diag_scores = diagonal(mask_scores)
    h_t = []
    for score, label in zip(diag_scores, h_{t-1}):
        if score > theta:
            h_t.append(label)
    if len(h_t) == len(h_{t-1}):
        break
    h_{t-1} = h_t
final_masks = post_process(proposals)
```

a set of masks $y_t \in [0,1]^{N_{t-1}\times H\times W}$ corresponding to N_{t-1} input text queries, and the updated text queries h_t for the subsequent step. For image segmentation, all different time steps share the same x_t.

To delve deeper into the design of our framework, we formulate its operations through Eq. (7.3) to Eq. (7.5).

$$y_t = f(x_t, h_{t-1}; W_f). \tag{7.3}$$

Here the function $f(\cdot,\cdot)$ represents the mask proposal generator and W_f denotes its pre-trained weights. The mask proposal generator processes the input image x_t and the text queries at previous step h_{t-1} to generate candidate mask proposals y_t. Given the mask proposal generator is not pre-trained for dense prediction, the mask proposals y_t from $f(\cdot,\cdot)$ are inaccurate. To assess these mask proposals, we draw visual prompts *e.g.*, red circles or background blur, to the input x_t, based on mask proposals to highlight the masked area on the image. The visual prompting function

$v(\cdot, \cdot)$ is defined as:

$$x'_t = v(x_t, y_t). \tag{7.4}$$

Here x'_t represent N_{t-1} images with the visual prompts. The prompted images x'_t are then passed to the mask classifier $g(\cdot, \cdot)$ with the pre-trained weights W_g, along with the text queries h_{t-1}, to compute a similarity matrix P_t. The entire process of the mask classifier can be defined as:

$$P_t = g(x'_t, h_{t-1}; W_g). \tag{7.5}$$

Finally, after going through a thresholding function $\sigma(\cdot)$, text queries with similarity scores lower than the threshold θ will be removed so that the text queries $h_t = \sigma(P_t)$ for the next step t are obtained. h_t is a potentially reduced set of h_{t-1}. Details of the thresholding function will be given in Section 3.3. This recurrent process continues until the text queries remain unchanged between consecutive steps, *i.e.*, $h_t == h_{t-1}$. We use T to denote this terminal time step. Finally, we apply post-processing described in Section 3.4 to the mask proposals y_T generated in the final time step.

The pseudo-code in PyTorch-style is given in Algorithm 2. Note that users provide the initial text queries h_0, which are unrestricted and can include general object classes ("cat"), proper nouns ("Space Needle"), referring phrases ("the man in red jacket"), *etc.*

3.3 The Two-stage Segmenter

In this section, we explain the two components of our segmenter, *i.e.* a mask proposal generator and a mask classifier, which serve as the recurrent unit. As illustrated in Figure 3(c), the mask proposal generator first predicts a mask for each text query and then the mask classifier filters out irrelevant text queries based on the degree of alignment with their associated masks. We use the frozen pre-trained CLIP model

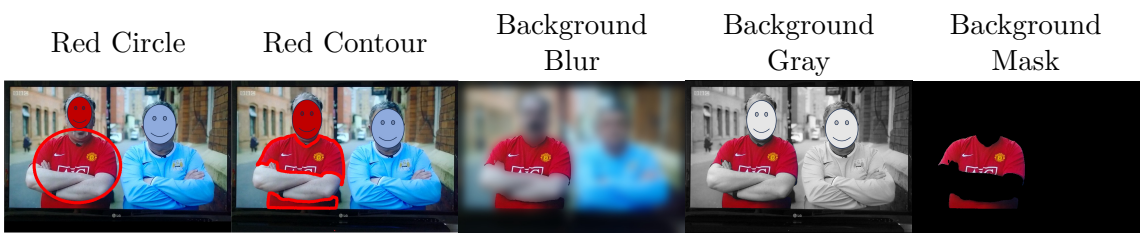


Figure 4. Examples of visual prompts given a mask on the man wearing the jersey of Manchester United.

weights for both the proposal generator and classifier to fully preserve the knowledge encapsulated in CLIP.

Mask proposal generator. To predict the mask proposal y_t, a gradient-based Class-Activation Map (gradCAM) [180, 238] is applied to the pre-trained CLIP. More specifically, the image x_t and text queries h_{t-1} are first fed into CLIP to get a score between the image and each text. We then back-propagate the gradients of the score of each text query (*i.e.*, class) from the feature maps of the CLIP image encoder to obtain a heatmap. Unless otherwise specified, we use the Class Activation Map (CAM) and class affinity (CAA) module from CLIP-ES [180] as our mask proposal generator, with no further training of the CLIP model required. Apart from the text queries at the current step, we explicitly add a set of background queries describing categories that do not exist in the user text queries and calculate their gradients. This helps to suppress the activation from irrelevant texts (*e.g.*, `Barcelona` and `Arsenal` in Figure 3) in the subsequent mask classification process. More details of how CLIP works with gradCAM are provided in the appendix.

Mask classifier. The masks from the proposal generator may be noisy because the input texts are from an unrestricted vocabulary and may refer to non-existing objects in the input image. To remove this type of proposals, we apply another CLIP model to compute a similarity score between each query and its associated mask proposal. A straightforward approach is blacking out all pixels outside the mask region, as shown in the rightmost image in Figure 4, and then computing the visual embedding for the foreground only. However, recent works [198, 243] have found

several more effective *visual prompts* which can highlight the foreground as well as preserve the context in the background. Inspired by this, we apply a variety of visual prompts, *e.g.*, red circles, bounding boxes, background blur and gray background to guide the CLIP model to focus on the foreground region. A threshold η is set to first binarize the mask proposals y_t before applying these visual prompts to the images. Please refer to the supplementary material for more implementation details. After applying visual prompts, we obtain N_{t-1} different prompted images, corresponding to N_{t-1} text queries (h_{t-1}). We feed these images and text queries into the CLIP classifier $g(\cdot,\cdot)$ followed by a softmax operation along the text query dimension to get the similarity matrix $P_t \in \mathbb{R}^{N_{t-1}\times N_{t-1}}$ given the image and text embeddings. We only keep the diagonal elements of P_t as the matching score between the i-th mask and the i-th query. If the score is lower than a threshold θ, the query and its mask are filtered out. Mathematically, the thresholding function $\sigma(\cdot)$ is defined as follows:

$$h_t^i = \sigma(P_t^{ii}) = \begin{cases} h_{t-1}^i, & \text{if } P_t^{ii} \geq \theta \\ \texttt{NULL}, & \text{if } P_t^{ii} < \theta \end{cases} \tag{7.6}$$

where P_t^{ii} is the i-th element of the diagonal of the normalized similarity matrix, and θ is a manually set threshold. `NULL` represents that the i-th text query is filtered out and will not be input to next steps.

3.4 Post-Processing

Once the recurrent process stops, we start to post-process y_T, the masks from the final step T. We employ dense conditional random field (CRF) [156] to refine mask boundaries. When constructing the CRF, the unary potentials are calculated based on the mask proposals of the last step. All hyper-parameters are set to the defaults in [156]. Finally, an argmax operation is applied to the mask output of denseCRF along the dimension of text queries. Thus, for each spatial location of the mask we only keep the class (text query) with the highest response.

Models	Is *VLM pre-trained?*	*w/ aux trainable module?*	#Added Images	Additional Supervision	VOC-20‡	VOC-21	COCO Obj	PC-59‡	PC-60	A-150	A-847	PC-459
zero-shot methods fine-tuned with additional data												
ViL-Seg [184]	✓	✓	12M	text+self	-	34.4	16.4	-	16.3	-	-	-
GroupViT [310]	×	✓	26M	text	74.1	52.3	24.3	23.4	22.4	10.6	4.3	4.9
GroupViT [310]	×	✓	24M	text	79.7	50.8	27.5	-	23.7	-	-	-
SegCLIP [199]	×	✓	3.4M	text+self	-	33.3	15.2	-	19.1	-	-	-
SegCLIP [199]	✓	✓	3.4M	text+self	-	52.6	26.5	-	24.7	8.7	-	-
ZeroSeg [32]	✓	✓	1.3M	self	-	40.8	20.2	-	20.4	-	-	-
ViewCo [229]	✓	✓	26M	text+self	-	52.4	23.5	-	23.0	-	-	-
MixReorg [20]	✓	✓	12M	text	-	47.9	-	-	23.9	-	-	-
CLIPpy [227]	✓	×	134M	text+self	-	52.2	32.0	-	-	13.5	-	-
OVSegmenter [312]	✓	✓	4M	text	-	53.8	25.1	-	20.4	-	-	-
TCL [27]	✓	✓	15M	text+self	77.5	51.2	30.4	24.3	30.3	14.9	-	-
TCL+PAMR [27]	✓	✓	15M	text+self	83.2	55.0	31.6	33.9	30.4	17.1	-	-
zero-shot methods with SAM												
SAMCLIP [278] (*w/* [151])	✓	✓	41M	text+self	-	60.6	-	-	29.2	17.1	-	-
CaR+SAM (Ours, *w/* [142])	✓	-	-	-	-	**70.2**	**37.6**	-	**31.1**	-	-	-
zero-shot methods without fine-tuning on CLIP												
ReCo† [241]	✓	×	-	-	57.7	25.1	15.7	22.3	19.9	11.2	-	-
MaskCLIP† [354]	✓	×	-	-	74.9	38.8	20.6	26.4	23.6	9.8	-	-
CaR (Ours, ***w/o* SAM**)	✓	×	-	-	**91.4**	**67.6**	**36.6**	**39.5**	**30.5**	**17.7**	**8.1**	**13.9**
Δ *w/ the state-of-the-art w/o additional data*					**+16.5**	**+28.8**	**+16.0**	**+13.1**	**+6.9**	**+6.5**	-	-
Δ *w/ the state-of-the-art w/ additional data*					**+8.2**	**+12.6**	**+4.6**	**+5.6**	**+0.1**	**+0.6**	**+3.8**	**+9.0**

Table 1. Comparison to state-of-the-art zero-shot semantic segmentation approaches. Results annotated with a † are reported by [27]. A ✓ is placed if either the visual or text encoder of the VLM is pre-trained. The table shows that our method outperforms not only counterparts without fine-tuning by a large margin, but also those fine-tuned on millions of data samples. For fair comparison, we compare with methods using CLIP [220] as the backbone. ‡VOC-20 and PC-59 represent Pascal VOC and Pascal Context **without** background. VOC-21 and PC-60 represent Pascal VOC and Pascal Context with an additional background class. A-150, A-847 and PC-459 represent ADE-150, ADE-847, and Pascal Context 459 datasets, respectively.

Additionally, we propose to ensemble the CRF-refined masks with SAM [151], as an **optional** post-processing module. This begins with generating a set of mask proposals from SAM using the `automask` mode, without entering any prompts into SAM. To match these SAM proposals with the masks processed by denseCRF, we introduce a novel metric: the *Intersection over the Minimum-mask (IoM)*. If the IoM between a mask from SAM and a CRF-refined mask surpasses a threshold ϕ_{iom}, we consider them matched. Then all SAM proposals matched to the same CRF-refined mask are combined into one single mask. Finally, we compute the IoU between the combined mask and the original CRF-refined mask. If the IoU is greater than a threshold ϕ_{iou}, we adopt the combined mask to replace the original mask, otherwise, we keep the CRF-refined mask. The detailed post-processing steps are explained in the supplementary material.

4 Experiments

4.1 Zero-shot Semantic Segmentation

Datasets. Since our method does not require training, below we introduce only the datasets utilized for evaluation purposes. We conduct assessments for semantic segmentation using the validation (`val`) splits of Pascal VOC, Pascal Context, and COCO Object. Specifically, **Pascal VOC** [84] encompasses 21 categories: 20 object classes (VOC-20) alongside one background class (VOC-21). For **Pascal Context** [208], our evaluation employs the prevalent version comprising 59 classes including both "things" and "stuff" categories (PC-59), and one background ("other") class for the concepts outside of the 59 classes (PC-60). Following [310], we construct the **COCO Object** dataset as a derivative of COCO Stuff [19]. We kindly emphasize that the COCO Object dataset is **not** COCO Stuff since it merges all "stuff" classes into one background class and thus has 81 classes (80 "things" + 1 background) in total. We use the standard mean Intersection-over-Union (mIoU) metric to evaluate our method's segmentation performance. Besides, we report the performance of CaR on ADE-150 (A-150), ADE-847 (A-847) and Pascal Context 459 (PC-459) that consist of 150, 847 and 459 classes respectively.

Implementation details. Our proposed method CaR utilizes the foundational pre-trained CLIP models as the backbone. More precisely, we harness the CLIP model with ViT-B/16 to serve as the underlying framework for the mask proposal generator $f(\cdot,\cdot)$. Concurrently, for the mask classifier $g(\cdot,\cdot)$, we adopt a larger ViT-L/14 version for higher precision based on our ablation study. Unless otherwise specified, the reported quantitative results are post-processed solely with a denseCRF, with no SAM masks involved. When setting the threshold hyper-parameters, we assign $\eta = 0.4$, $\theta = 0.6$, and $\lambda = 0.4$ for Pascal VOC, and $\eta = 0.5$, $\theta = 0.3$, $\lambda = 0.5$ for COCO and $\eta = 0.6$, $\theta = 0.2$, $\lambda = 0.4$ for Pascal context. The specific background queries used for the mask generator $f(\cdot,\cdot)$ are ablated in Section 4.2 and detailed in the supplementary material. For Pascal Context, we use separate groups of background

Dataset	*w/ recurrence?*	CAM	mIoU
Pascal VOC		CLIP-ES [180]	15.2
	✓	CLIP-ES [180]	67.6
	✓	gradCAM [238]	41.1

Table 2. Effect of applying our recurrent architecure and different CAM methods. The recurrence plays a vital role in improving the performance.

queries for "thing" and "stuff". For "thing" categories, all "stuff" categories are added as background queries and vice versa for "stuff" categories. As an optional strategy, we utilize a matching algorithm and perform an ensemble with SAM masks. We set both thresholds, ϕ_{iom} and ϕ_{iou}, to 0.7 for all three datasets. We enable half-precision floating point for CLIP. Since CaR is just a framework designed for inference, all experiments in this paper are conducted on just **one** NVIDIA V100 GPU.

Efficiency. CaR **without SAM** operates at 950 ms with CPU-based CRF (200ms) for 500×500 images. GPU CRF is $\sim 5\times$ faster. CaR is memory-efficient, consuming only 3.6GB GPU memory on Pascal VOC.

CaR significantly outperforms methods without additional training. We also compare CaR with training-free methods like MaskCLIP [354] and ReCo [241]. Across the benchmarks, our model consistently demonstrates an impressive performance uplift. Under a similar setting with no additional training data, CaR surpasses previous state-of-the-art method by 28.8, 16.0 and 6.9 mIoU on Pascal VOC, COCO Object and Pascal Context, respectively.

Training-free CaR even outperforms several methods with additional fine-tuning. As shown in Table 1, we compare our method with previous state-of-the-art methods including ViL-Seg [184], GroupViT [310], SegCLIP [199], ZeroSeg [32], ViewCo [229], CLIPpy [227], and TCL [27], which are augmented with additional data. The prior best results of different datasets are achieved by different methods. Specifically, TCL [27], employing a fully pre-trained CLIP model and fine-tuned on 15M additional data, achieves the highest mIoU (55.0 and 30.4) on Pascal VOC and

Mask Proposal Generator $f(\cdot,\cdot)$	Mask Classifier $g(\cdot,\cdot)$	Pascal VOC	COCO Object
ViT-B/16	ViT-B/16	54.1	15.9
	ViT-L/14	**67.6**	**36.6**
ViT-L/14	ViT-B/16	50.6	14.1
	ViT-L/14	57.6	32.5

Table 3. Effect of different CLIP backbones. We compare various CLIP backbones on Pascal VOC and COCO Object. Results show that we can improve the performance by scaling up the mask classifier.

Pascal Context. CLIPpy [227] sets the previous highest record on COCO Object but also requires extensive data for fine-tuning. Concretely, it first utilizes a ViT-based image encoder pre-trained with DINO [26] and a pre-trained T5 text encoder [213], and then fine-tunes both encoders with 134M additional data. Our method, incurring no cost for fine-tuning, still outperforms these methods by 12.6, 4.5, and 0.1 mIoU on the Pascal VOC, COCO Object, and Pascal Context datasets, respectively. Since "stuff" appears less frequently in the pre-training image-text data for CLIP, CaR also exhibits less sensitivity to "stuff" on Pascal Context.

CaR+SAM further boosts the performance. When integrated with SAM [142, 151], we compare CaR with a concurrent method SAMCLIP [278] and outperform it by 9.6 and 1.9 on Pascal VOC and Pascal Context. We use the recent variant HQ-SAM [142] with **no prompt given** (`automask` mode), and then match the generated masks with metrics designed in Section 3.4. In other words, SAM is only used as a post-processor to refine the boundary of results from CaR. By applying SAM into our framework, our results can be further boosted by 2.6, 1.1 and 0.6 mIoU on Pascal VOC, COCO Object and Pascal Context, respectively.

4.2 Ablation Studies.

Effect of Recurrence. As illustrated in Table 2, the incorporation of the recurrent architecture is crucial to our method. Without recurrence ($a.k.a\, T = 1$), our method functions similarly to CLIP-ES [180] with an additional CLIP classifier, and achieves only 15.2% in mIoU. The recurrent framework can lead to a 52.4% improvement,

Dataset	Visual Prompts					mIoU
	circle	contour	blur	gray	mask	
Pascal VOC	✓					66.9
		✓				66.0
			✓			66.4
				✓		66.1
					✓	61.8
	✓		✓			**67.6**
	✓			✓		67.1
		✓	✓			66.5
			✓	✓		66.3
	✓		✓	✓		66.8

Table 4. Effect of different visual prompts. When multiple visual prompts are checked, we will apply all checked visual prompts simultaneously on one image. The experiments are conducted on Pascal VOC. Results for COCO and Pascal Context are shown in supplementary materials.

reaching an mIoU of 67.6%. The significant improvement validates the effectiveness of the recurrent design of our framework. For VOC and COCO, most images require two steps, and a small portion of images goes beyond two.

Effect of different CAM methods. Table 2 exhibits that our framework is compatible with different CAM methods and could be potentially integrated with other CAM-related designs. When integrated with CLIP-ES [180], our method is 26.5 mIoU higher than that with gradCAM [238]. We kindly note that we do not carefully search the hyper-parameters on gradCAM so the performance could be further improved.

Effect of different CLIP Backbones. We experiment with different settings of CLIP backbones used in the mask proposal generator f and mask classifier g, on Pascal VOC and COCO Object datasets. Results are displayed in Table 3. As for the mask proposal generator, ViT-B/16 outperforms the ViT-L/14 by over 10 mIoU on both Pascal VOC and COCO Object. There is significant mIoU gains when employing the larger ViT-L/14 for the mask classifier over ViT-B/16. Similar observations have been found by [243] that a larger backbone can better understand the visual prompts, which indicates that the performance of our method can be potentially improved by employing large backbones for the mask classifier.

Pascal VOC				COCO Object			
η	θ	λ	mIoU	η	θ	λ	mIoU
0.3	0.6	0.4	67.0	0.5	0.3	0.6	35.4
0.4	0.6	0.4	**67.6**	0.5	0.3	0.4	36.1
0.5	0.6	0.4	67.0	0.4	0.3	0.5	35.8
0.4	0.5	0.4	67.4	0.5	0.3	0.5	**36.6**
0.4	0.7	0.4	67.5	0.6	0.3	0.5	35.9
0.4	0.6	0.3	67.3	0.5	0.4	0.5	36.3
0.4	0.6	0.5	67.0	0.5	0.5	0.5	36.0

Table 5. Effect of different hyper-parameters: the threshold to binarize mask proposals (η), the threshold to remove text queries (θ), and parameter of CLIP-ES's[180] (λ). Experiments are conducted on Pascal VOC and COCO Object.

Effect of different visual prompts. There are various forms of visual prompts, including circle, contour, background blur (`blur`), background gray (`gray`), and background mask (`mask`), *etc.* We study the effects of different visual prompts on the Pascal VOC dataset and Table 4 summarizes the results when applying one or a combination of two of the aforementioned visual prompt types. The highest mIoU score is achieved with the combination of `circle` and `blur`, yielding a mIoU of 67.6. Notably, using `mask` alone results in the lowest mIoU of 61.8, which is a common practice for most previous open-vocabulary segmentation approaches, *e.g.*, [175, 322]. We also evaluate the effect of different visual prompts on COCO Object and Pascal Context, and show the results in the supplementary material.

Effect of hyper-parameters. We perform an ablation study on the performance impact of various hyper-parameter configurations on Pascal VOC, and present the results in Table 5. Hyper-parameters include the mask binarization threshold, η, defined in Section 3.3, the threshold θ employed in the thresholding function defined in Eq. (7.6), and the parameter λ defined in CLIP-ES [180]. The peak performance is recorded at an mIoU of 67.6 for $\eta = 0.4$, $\theta = 0.6$, and $\lambda = 0.4$ on Pascal VOC and 36.6 for $\eta = 0.5$, $\theta = 0.3$, and $\lambda = 0.5$ on COCO Object. Different parameter combinations result in mIoU scores that range from 67.0 to 67.6 on Pascal VOC and from 35.4 to 36.6 on COCO Object.

Effect of background queries. In Table 6, we explore how different background

Dataset	Background queries			mIoU
	`Terrestrial`	`Aquatic` `Atmospheric`	`Man-Made`	
Pascal VOC	×	×	×	64.3
	✓	×	×	65.6
	×	✓	×	64.9
	×	×	✓	66.4
	✓	✓	×	65.8
	×	✓	✓	66.4
	✓	×	✓	65.8
	✓	✓	✓	**67.6**

Table 6. Effect of background queries on Pascal VOC. We divide background queries into: `Terrestrial`, `Aquatic`, `Atmospheric`, and `Man-Made`. We use "None" as the background query for the result in the first row. Specific background queries of each category are shown in the supplementary material.

queries (classes not existing in the input queries) can affect CaR's performance. We find that the segmentation quality improves as we include more diverse background queries: The combination of all three types of background queries delivers the highest mIoU of 67.6. For more details about the background queries of each class, please refer to the supplementary material.

4.3 Referring Segmentation

We evaluate CaR on the referring segmentation task for both images and videos. Again, our method is an inference-only pipeline built upon pre-trained CLIP models, and does not need training/fine-tuning on any types of annotations. For referring segmentation we only use denseCRF [156] for post-processing, and SAM is **not** involved for all experiments in this section for fair comparison. Please refer to the supplementary material for the implementation details.

Datasets. Following [317, 326], we evaluate on **RefCOCO** [321], **RefCOCO+** [321], **RefCOCOg** [200, 210] and **GRES** [181] for the referring image segmentation task. Images used in all three datasets are sourced from the MS COCO [178] dataset and the masks are paired with descriptive language annotations. In RefCOCO+, descriptions about location are prohibited, making the task more challenging. There

Models	RefCOCO			RefCOCO+			RefCOCOg			GRES
	val	testA	testB	val	testA	testB	val	test(U)	val(G)	
weakly-supervised										
TSEG [249]	25.95	-	-	22.62	-	-	23.41	-	-	-
zero-shot										
GL CLIP [326]	26.20	24.94	26.56	27.80	25.64	27.84	33.52	33.67	33.61	-
CaR(Ours)	33.57	35.36	30.51	34.22	36.03	31.02	36.67	36.57	36.63	16.8

Table 7. Comparison to state-of-the-art methods on referring image segmentation in mIoU. CaR is better than all comparison methods in all benchmarks.

are two separate splits of the RefCOCOg dataset, one by UMD (U) [210] and another by Google (G) [300]. Following previous work, we use the standard mIoU metric. Apart from referring image segmentation, we also set up a new baseline for **zero-shot referring video segmentation** on **Ref-DAVIS 2017** [143]. Following [143], we adopt region similarity $\mathcal{J}$, contour accuracy $\mathcal{F}$, and the averaged score $\mathcal{J}\&\mathcal{F}$ as the metrics for evaluation.

Experimental results. Table 7 compares the performance of CaR with other methods on the referring image segmentation tasks across RefCOCO, RefCOCO+, and RefCOCOg. Comparing with other zero-shot methods, our method CaR outperforms Global-Local CLIP (GL CLIP) on all splits of these benchmarks. The performance gap is most pronounced on RefCOCO's `testA` split, where CaR outperforms 10.42 mIoU, and similarly on RefCOCO+'s `testA` split, with a lead of 10.72 mIoU.

$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$
30.34	28.15	32.53

Table 8. Results on Ref-DAVIS 2017.

We also note that GL CLIP [326] uses a pre-trained segmenter Free-SOLO [289] for mask extraction, while CaR is built **without** any pre-trained segmenter. As the first zero-shot method on GRES [181], CaR achieves 16.8 mIoU with no training effort incurred.

For referring video segmentation, we demonstrate in Table 8 that our method achieves 30.34, 28.15 and 32.53 for $\mathcal{J}\&\mathcal{F}$, $\mathcal{J}$ and $\mathcal{F}$ on Ref-DAVIS 2017 [143]. Considering that our method CaR requires neither fine-tuning nor annotations and operates in a zero-shot manner, this performance establishes a strong baseline.

5 Conclusion

We introduce CLIP as RNN (CaR), which preserves the entire large vocabulary space of pre-trained VLMs, by eliminating the fine-tuning process. By constructing a recurrent pipeline with a shared segmenter in the loop, CaR can perform zero-shot semantic and referring segmentation without any additional training efforts. Experiments show that CaR outperforms previous state-of-the-art counterparts by a large margin on eight different benchmarks, *i.e.* Pascal VOC, COCO Object, Pascal Context, ADE-150, ADE-847 and Pascal Context 459 on zero-shot semantic segmentation. We also demonstrate that CaR can handle referring expressions and segment fine-grained concepts like anime characters and landmarks, and also achieves state-of-the-art performance on RefCOCO, RefCOCO+, RefCOCOg and GRES for zero-shot referring segmentation. We hope our work sheds light on future research in open-vocabulary segmentation aiming to further expand the vocabulary space.

Acknowledgement. The majority of this work was done during Shuyang's internship at Google Research. We would like to thank Jindong Gu and Jianhao Yuan at Oxford University, Anurag Arnab, Xingyi Zhou, Huizhong Chen and Neil Alldrin at Google Research for their insightful discussion, Zhongli Ding for image donation. Shuyang Sun and Philip Torr are supported by UKRI grants: Turing AI Fellowship EP/W002981/1. We would also like to thank the Royal Academy of Engineering and FiveAI.

Appendix

A More Experimental Results

A.1 Quantitative Analysis on Vocabulary Space.

We demonstrate that our method CaR has a larger vocabulary space compared to the methods fine-tuned with mask annotations. Here we compare our method with OVSeg [175], which is fine-tuned on ImageNet [234] and COCO [178] with a pre-trained CLIP backbone for the task of referring image segmentation. We believe that referring expressions (*e.g.*, "the person in the red shirt" or "the cat in the mirror") refers to a specific segment using a broad vocabulary. We conduct a comparative analysis between a robust open-vocabulary segmentation benchmark, OVSeg [175], and CaR, utilizing standard referring image segmentation benchmarks [200, 210, 321]. We note that RefCOCO and COCO share the same set of images so OVSeg fine-tuning on COCO may not be counted as zero-shot on RefCOCO. The results, as detailed in Table A, demonstrate that CaR significantly surpasses OVSeg in performance. This disparity in performance suggests that CaR encompasses a more expansive vocabulary space than OVSeg.

A.2 Evaluation without Background

Following [180], our methodology benefits from using background queries in CLIP [220] classification to suppress false positives (predictions not belonging to the input text queries), enhancing segmentation results. Nevertheless, for more comprehensive comparison, we also assess our approach using an alternate evaluation setting, previously established, which omits the background class. Consequently, less emphasis is placed on object boundaries in this setting. We test our method on two datasets: Pascal VOC [85] without background (termed *VOC-20*) and Pascal Context [208] without background (termed *Context-59*). This setting tests the ability of various methods to discriminate between different classes. Our method CaR significantly

Models	RefCOCO			RefCOCO+			RefCOCOg		
	val	testA	testB	val	testA	testB	val	test(U)	val(G)
OVSeg [175]	22.58	19.38	25.63	19.13	15.74	25.30	27.87	29.09	28.31
CaR(Ours)	33.57	35.36	30.51	34.22	36.03	31.02	36.67	36.57	36.63

Table A. Comparison to mask-supervised open-vocabulary methods on referring image segmentation in mIoU. CaR is better than the comparison method, OVSeg, in all splits of the three benchmarks.

Model	Is VLM pre-trained?	#Additional Images	*w/o* Background	
			VOC-20	Context-59
GroupViT† [310]	×	26M	79.7	23.4
PACL [209]	✓	40M	72.3	50.1
TCL [27]	✓	15M	77.5	30.3
MaskCLIP† [354]	✓	-	74.9	26.4
ReCo† [241]	✓	-	57.5	22.3
CaR (Ours)	✓	-	**91.4**	**39.5**

Table B. Comparison with methods under the setting where background is ignored. We compare CaR with prior work on VOC-20, Context-59 in a setting that considers only the foreground pixels (decided by ground truth). Our method shows comparable performance to prior works despite only relying on pretrained feature extractors. †: numbers are from [27].

outperforms previous methods on VOC-20 and Context-59, where all methods use the same setting that ignores the background class. We reached out to the PACL authors to confirm that they did not evaluate background.

B Implementation Details of CAM

In this paper, we integrate two kinds of gradient-based CAM, *i.e.*, Grad-CAM [238] and CLIP-ES [180], respectively.

Integration with Grad-CAM. When integrating Grad-CAM [238] into our framework, we first extract the image and text feature vectors $v_x = f_I(x), v_h = f_T(h)$ from the image and text encoder $f_I(\cdot), f_T(\cdot)$ given an image x and text queries h. We compute a similarity score between the image and text features using the dot product $s = \text{softmax}(v_x \cdot v_h^\intercal)$, where softmax is applied along the dimension of v_h. This score s quantifies the alignment (*a.k.a* similarity) between the image x and the text

h as perceived by the CLIP model. Here h contains multiple queries. To integrate Grad-CAM into our framework, we first compute the gradients of the similarity score with respect to the feature maps of the image encoder by:

$$g = \frac{\partial s}{\partial A^k},$$

where A^k represents the feature maps and g denotes the gradients. Then we compute the neuron importance weights by average-pooling the gradients:

$$\alpha_k = \frac{1}{Z} \sum_i \sum_j g_{ij}^k.$$

Here, α_k is the neuron importance weights and Z is the number of pixels in each feature map. We then calculate a weighted combination of the feature maps A^k and the neuron importance α_k:

$$L = \text{ReLU}\left(\sum_k \alpha_k A^k\right),$$

an activation function ReLU is applied to filter out all negative activations. Specifically, we use the feature map after the first normalization layer of the last residual block to compute the gradients for CAM.

Integration with CLIP-ES. In summary, the CLIP-ES [180] we adopted is composed of a Grad-CAM and a class-aware attention-based affinity (CAA) module. The CAA module is introduced to enhance the vanilla multi-head self-attention (MHSA) for the Vision Transformer in CLIP. Given an image, class-wise CAM maps $M_c \in R^{h \times w}$ for each target class c and the attention weight $W^{attn} \in R^{hw \times hw}$ are obtained from MHSA. For the attention weight, which is made asymmetric due to the use of different projection layers by the query and key, Sinkhorn normalization [245] is applied (alternately applying row-normalization and column-normalization) to convert it into a doubly stochastic matrix D, and the symmetric affinity matrix A

can be derived as follows:

$$A = \frac{D + D^T}{2}, \text{ where } D = \text{Sinkhorn}(W^{attn}). \tag{7.7}$$

For the CAM map $M_c \in R^{h \times w}$, a mask map for each target class c can be obtained by thresholding the CAM with λ. Then a set of bounding boxes can be generated based on the thresholded masks. These boxes are used to mask the affinity weight matrix A, and then each pixel can be refined based on the masked affinity weight and its semantically similar pixels. This refinement process can be formalized as follows:

$$M_c^{aff} = B_c \odot A^t \cdot \text{vec}(M_c), \tag{7.8}$$

where $B_c \in R^{1 \times hw}$ represents the box mask obtained from the CAM of class c, $\odot$ denotes the Hadamard product, t indicates the number of refining iterations, and $\text{vec}(\cdot)$ denotes the vectorization of a matrix. It should be noted that the attention map and CAM are extracted in the same forward pass. Therefore, CAA refinement is performed in real time and does not need an additional stage. Our implementation uses the attention maps from the last 8 layers of Vision Transformer for CAA.

C Implementation Details of Visual Prompts

The Python code of visual prompts is shown in Algorithm C, which is at the end of the supplementary material.

D Breakdown of Background Tokens

We break down the background tokens into 3 sub-categories for ablation study (experiment results are shown in the main manuscript in Table 6):

- `Terrestrial`: [‘ground’, ‘land’, ‘grass’, ‘tree’, ‘mountain’, ‘rock’, ‘valley’, ‘earth’,‘terrain’, ‘forest’, ‘bush’, ‘hill’, ‘field’, ‘pasture’, ‘meadow’, ‘plateau’,

'cliff', 'canyon', 'ridge', 'peak', 'plain', 'prairie', 'tundra', 'savanna', 'steppe', 'crag', 'knoll', 'dune', 'glen', 'dale', 'copse', 'thicket']

- `Aquatic-Atmospheric`: ['sea', 'ocean', 'lake', 'water', 'river', 'sky', 'cloud', 'pond', 'stream', 'lagoon', 'bay', 'gulf', 'fjord', 'estuary', 'creek', 'brook', 'reservoir', 'pool', 'spring', 'marsh', 'swamp', 'wetland', 'glacier', 'iceberg', 'atmosphere', 'stratosphere', 'mist', 'fog', 'rain', 'drizzle', 'hail', 'sleet', 'snow', 'thunderstorm', 'breeze', 'wind', 'gust', 'hurricane', 'tornado', 'monsoon', 'cumulus', 'cirrus', 'stratus', 'nimbus']

- `Man-Made`: ['building', 'house', 'wall', 'road', 'street', 'railway', 'railroad', 'bridge', 'edifice', 'structure', 'apartment', 'condominium', 'skyscraper', 'highway', 'boulevard', 'lane', 'alley', 'byway', 'avenue', 'expressway', 'freeway', 'path', 'overpass', 'underpass', 'viaduct', 'tunnel', 'footbridge', 'crosswalk', 'culvert', 'dam', 'archway', 'causeway', 'plaza', 'square', 'station', 'terminal']

E Implementation Details of Mutual Background for Pascal Context

Our approach involves creating a list of background queries to minimize false positive predictions in mask proposals. However, in the Pascal Context dataset [208], many "stuff" categories (*e.g.* sky, ground, sea) serve as background queries for "object" categories (*e.g.* bird, car, boat). Directly removing these 'stuff' categories from the background query list and generating object and stuff masks using CAM leads to noisy results due to the lack of false positive background suppression. To address this issue, we adopt a mutual background strategy. In this method, object and stuff masks are produced separately, using object categories as the background queries for stuff masks and vice versa. This technique not only maintains the benefit of reducing false positives but also significantly enhances performance in the Pascal Context dataset.

F Implementation Details of Referring Image Segmentation.

We use ViT-B/16 as the backbone of the visual encoder for both the mask proposal generator and mask classifier, and use `circle` and `background blur` as the visual prompts for the inputs of mask classifier. The η, θ, λ were set to (0.5, 0.3, 0.5), (0.2, 0.1, 0.5), (0.5, 0.1, 0.6) for refCOCO, refCOCO+ and refCOCOg, respectively. All splits of these three datasets share the same set of hyper-parameters. We note that we **do not** apply SAM for referring image segmentation.

G More Visualization Results

G.1 Visualization results on different post-processors

Figures A and B present a comparative visualization of the post-processing techniques Conditional Random Field (CRF) and Segment Anything Model (SAM) [142], applied to randomly chosen samples from the VOC [85] and COCO Object datasets [19]. Initial observations reveal that the application of CRF in CaR facilitates the generation of high-quality masks, albeit with notable limitations in delineating boundaries between distinct semantic masks. The integration of SAM enhances the precision of these masks, yielding clearer and more well-defined boundaries. Nevertheless, the implementation of SAM is not without drawbacks; it occasionally leads to false negative predictions, stemming from mismatches between CaR raw masks and SAM candidate masks (the matching algorithm is introduced in the main manuscript), or false positive predictions due to the overly coarse nature of SAM masks. Meanwhile, we find SAM is not very sensitive to stuff classes, so combining SAM on Pascal Context will not lead to much increase in mIoU.

G.2 Visualization comparison for different open-vocabulary segmentation methods.

Figure C presents a qualitative comparison of open-vocabulary segmentation results for a variety of non-standard subjects, including unique characters, brands, and landmarks. These subjects are notably distinct from common objects. The Grounded SAM [185] method demonstrates proficiency in segmenting prominent objects with precision, yet it often misclassify these segments. The OVSeg [175] approach also generates low-quality segmentation masks and inaccurate class predictions. In contrast, our methodology CaR excels by creating high-quality masks with accurate semantic class predictions, showcasing its superior capability in the realm of open-vocabulary segmentation.

H Limitation

The primary limitation of our method is that its performance is bounded by the pre-trained VLM. For example, since the CLIP model utilizes horizontal flipping augmentation during training, it becomes challenging for our model to successfully distinguish between the concepts "left" and "right". However, we believe that this issue can be easily resolved through adjustments, such as incorporating better data augmentation techniques during the pre-training phase.

I Future Potentials and Broader Impact

CaR is simple, straightforward yet highly efficient. To enhance its performance further, we provide two ways to explore. First, incorporating additional trainable modules such as Feature-Pyramid Networks can significantly improve its capability in handling small objects. Second, since our method is fundamentally compatible with various Vision-Language Models (VLMs), it presents an intriguing opportunity to investigate integration with other VLMs. Moreover, CaR can serve the purpose

of generating pseudo-labels for other open-vocabulary segmenters.

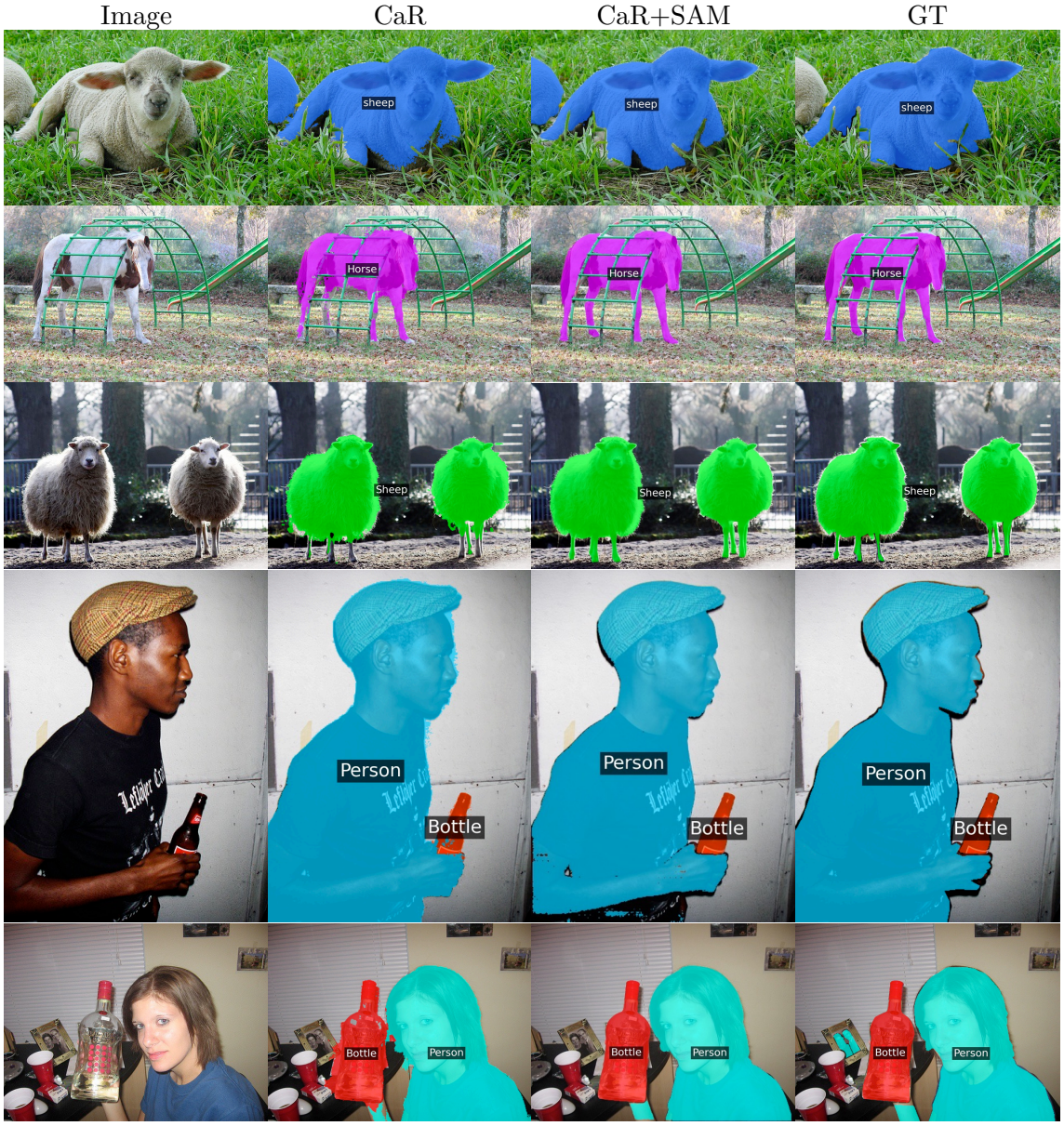


Figure A. Comparison of different post-processors on randomly selected images from PASCAL VOC.

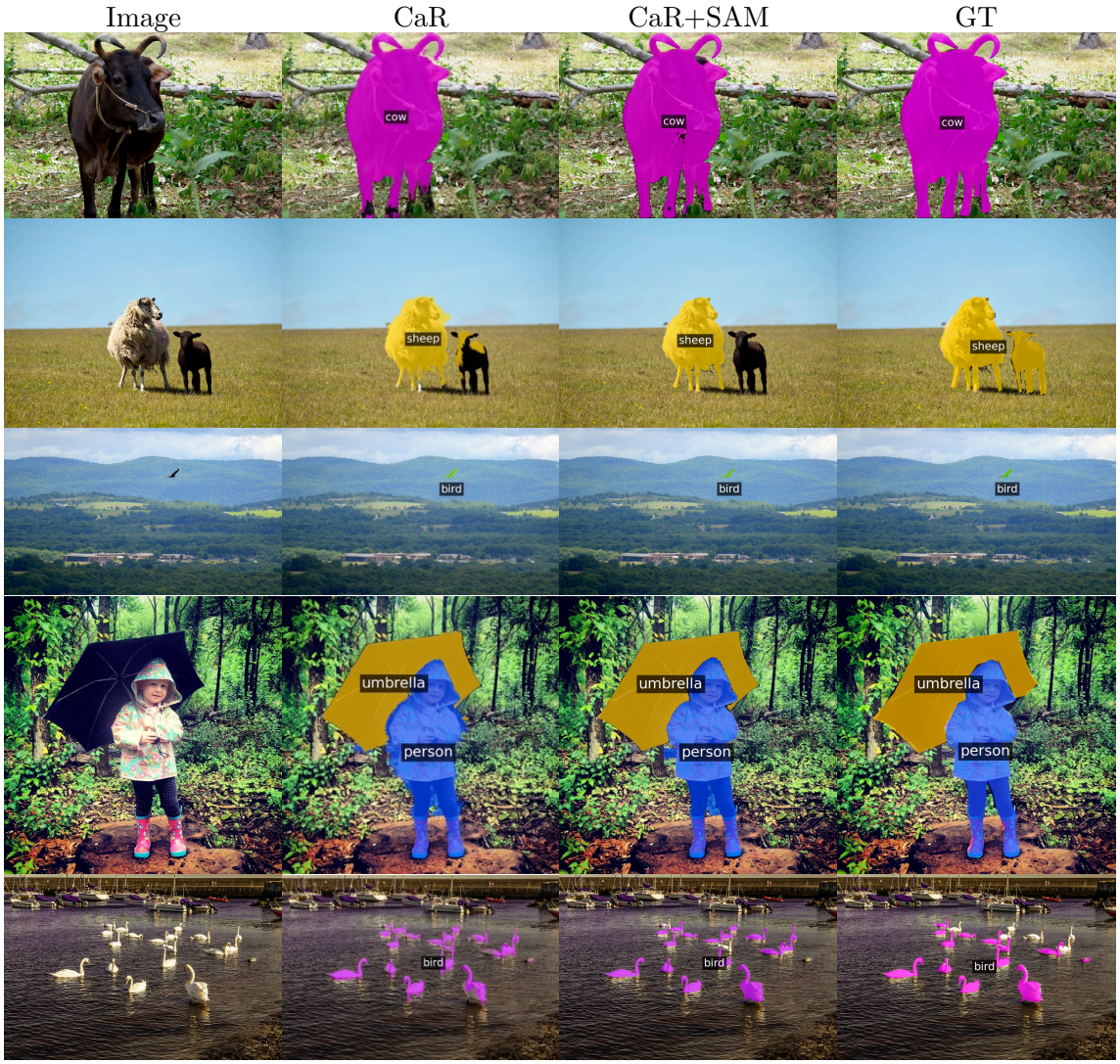


Figure B. Comparison of different post-processors on randomly selected images from COCO Object.

Figure C. Visualization comparison of different open-vocabulary segmentation methods.

Algorithm C Pseudo-code of CLIP as RNN in PyTorch style.

```
import cv2
import numpy as np
import torch
from scipy.ndimage import binary_fill_holes
def apply_visual_prompts(
       image_array,
       mask,
       visual_prompt_type=('circle'),
       visualize=False,
       color=(255, 0, 0),
       thickness=1,
       blur_strength=(15, 15)):
   prompted_image = image_array
   inv_mask = (1 - mask)[:, :, None]
   if 'blur' in visual_prompt_type:
       # blur the part out side the mask
       # Blur the entire image
       blurred = cv2.GaussianBlur(prompted_image, blur_strength, 0)
       # Get the sharp region using the mask
       sharp_region = cv2.bitwise_and(
          prompted_image,
          prompted_image,
          mask=np.clip(mask, 0, 255)))
       # Get the blurred region using the inverted mask

       blurred_region = (blurred * inv_mask)
       # Combine the sharp and blurred regions
       prompted_image = cv2.add(sharp_region, blurred_region)
   if 'gray' in visual_prompt_type:
       gray = cv2.cvtColor(prompted_image, cv2.COLOR_BGR2GRAY)
       # make gray part 3 channel
       gray = np.stack([gray, gray, gray], axis=-1)
       # Get the sharp region using the mask
       color_region = cv2.bitwise_and(
          prompted_image,
          prompted_image,
          mask=np.clip(mask, 0, 255))
       # Get the blurred region using the inverted mask
       inv_mask = 1 - mask
       gray_region = (gray * inv_mask)
       # Combine the sharp and blurred regions
       prompted_image = cv2.add(color_region, gray_region)
   if 'black' in visual_prompt_type:
       prompted_image = cv2.bitwise_and(
          prompted_image,
          prompted_image,
          mask=np.clip(mask, 0, 255))
   if 'circle' in visual_prompt_type:
       mask_center, mask_height, mask_width = mask2chw(mask)
       center_coordinates = (mask_center[1], mask_center[0])
       axes_length = (mask_width // 2, mask_height // 2)
       prompted_image = cv2.ellipse(prompted_image,
                                    center_coordinates,
                                    axes_length, 0, 0, 360, color, thickness)
   if 'rectangle' in visual_prompt_type:
       mask_center, mask_height, mask_width = mask2chw(mask)
       center_coordinates = (mask_center[1], mask_center[0])
       start_point = (mask_center[1] - mask_width //
                      2, mask_center[0] - mask_height // 2)
       end_point = (mask_center[1] + mask_width //
                    2, mask_center[0] + mask_height // 2)
       prompted_image = cv2.rectangle(prompted_image,
                                      start_point,
                                      end_point,
                                      color, thickness)
   if 'contour' in visual_prompt_type:
       # Find the contours of the mask
       # fill holes for the mask
       mask = binary_fill_holes(mask)
       contours, hierarchy = cv2.findContours(mask, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
       # Draw the contours on the image
       prompted_image = cv2.drawContours(
          prompted_image, contours, -1, color, thickness)
   return prompted_image
```

Chapter 8

Discussion

In this chapter, we discuss the contribution and achievements of each research paper listed in the thesis.

Separable Transformer. Chapter 2 illustrates the Separable Transformer (SeT). This chapter delves into the SeT's innovative architecture, the implementation of positional embeddings, and various training strategies specifically tailored to enhance the training of Transformer-based models for visual tasks. Our exploration paves the way for a deeper understanding of how these components interact to improve visual perception capabilities in computational models. Additionally, subsequent studies, such as those by Dai et al. [68] and Chu et al. [55], demonstrate the efficacy of relative positional encoding, underscoring its potential in further refining Transformer architectures. The methodologies we investigate regarding data augmentation, optimization, and regularization have also been incorporated into subsequent developments of this thesis, *e.g.* Visual Parser (ViP) [258].

Vision Transformer with Progressive Sampling. In Chapter 3, we present the Vision Transformer with Progressive Sampling (PS-ViT), featuring an innovative progressive sampling module that redefines the method of segmenting input images into patches. This adaptive technique has not only been embraced by subsequent

research but has also inspired related studies to incorporate similar methodologies within attention mechanisms [302], and extend its application to video analysis [89]. This demonstrates the broad impact and adaptability of our proposed approach in enhancing the capabilities of Transformer-based models for visual data processing.

TransMix Chapter 4 unveils TransMix, a method designed to tackle the discrepancy between the mixed input images and their corresponding label space. By integrating a dynamic attention mechanism, TransMix is able to selectively concentrate on the most relevant portions of an image, thereby enhancing the model's proficiency in handling objects of varying sizes. The concept of adaptive label reassignment introduced by TransMix has spurred a wave of subsequent research endeavors, including MixPro [348], TokenMix [183], PatchMix [357], and the ReClass approach within ReMaX [257], illustrating the significant influence and potential of this innovative approach.

Visual Parser Chapter 5 introduces the Visual Parser (ViP) [258], a model devised to offer a comprehensive framework for visual perception across various levels of granularity. As an innovative approach that establishes a part-whole hierarchy within a Transformer-based design, ViP has inspired subsequent research focused on applying this hierarchical structure to their specific fields, such as point cloud analysis [91, 162], and in studies that bridge different granularities and tasks [284, 342]. The effectiveness of ViP's architecture has led to its adoption in other advanced projects, like SAM [151], which incorporate the part-whole Transformer concept, further attesting to ViP's significant impact and its role in shaping future research directions.

ReMaX In Chapter 6, we present ReMaX [257] that demonstrates the efficacy of utilizing the objective of a simpler task (semantic segmentation) as a relaxation technique to ease the complexity of a more composite task (panoptic segmentation). ReMaX not only accelerates the training process by about $2-3\times$ but also improves

the overall performance for panoptic segmentation through this strategic simplification, all without adding any additional computational burden during inference. Its straightforward design approach and minimal implementation costs make the methodology highly adaptable for broader applications.

CLIP as RNN Chapter 7 introduces CLIP as RNN (CaR) [255], showcasing the use of a pre-trained CLIP model for fine-grained object and pixel-level visual tasks, eliminating the need for additional fine-tuning. We propose a fresh perspective: fine-tuning on datasets with detailed annotations, such as masks and bounding boxes, tends to restrict the vocabulary scope. Through the development of a recurrent architecture, we delve into the potential of visual prompting. This methodology enables the CLIP model, originally trained on image-text pairs, to excel in image segmentation tasks, surpassing previous state-of-the-art approaches without requiring fine-tuning. Remarkably, CaR sets new performance benchmarks across eight diverse datasets, including Pascal VOC (VOC-20 and VOC-21), COCO Object, Pascal Context (PC-60 and PC-59), ADE-150, ADE-847, and Pascal Context 459, in zero-shot semantic segmentation scenarios. Moreover, leveraging its extensive vocabulary, CaR achieves unparalleled results in referring segmentation tasks on RefCOCO, RefCOCO+, and RefCOCOg for zero-shot referring image segmentation. The success of CaR exemplifies the feasibility of moving towards open-world visual segmentation, where models can interpret and segment an expansive array of visual concepts without predefined labels.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. In *OSDI*, 2016.

[2] Nikita Araslanov and Stefan Roth. Single-stage semantic segmentation from image labels. In *CVPR*, pages 4253–4262, 2020.

[3] Anurag Arnab, Sadeep Jayasumana, Shuai Zheng, and Philip HS Torr. Higher order conditional random fields in deep neural networks. In *ECCV*, 2016.

[4] Anurag Arnab and Philip HS Torr. Bottom-up instance segmentation using deep higher-order crfs. In *BMVC*, 2016.

[5] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention. *ICLR*, 2015.

[6] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[7] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *TPAMI*, 2017.

[8] Donghyeon Baek, Youngmin Oh, and Bumsub Ham. Exploiting a joint embedding space for generalized zero-shot semantic segmentation. In *ICCV*, 2021.

[9] Yutong Bai, Jieru Mei, Alan Yuille, and Cihang Xie. Are transformers more robust than cnns? In *NeurIPS*, 2021.

[10] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. In *ICLR*, 2021.

[11] Irwan Bello. Lambda networks: Modeling long-range interactions without attention. *ICLR*, 2021.

[12] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *ICCV*, 2019.

[13] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *ICCV*, 2019.

[14] Andrew Brock, Soham De, Samuel L Smith, and Karen Simonyan. High-performance large-scale image recognition without normalization. *arXiv preprint arXiv:2102.06171*, 2021.

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

[16] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv:2005.14165*, 2020.

[17] Maxime Bucher, Tuan-Hung Vu, Matthieu Cord, and Patrick Pérez. Zero-shot semantic segmentation. *Advances in Neural Information Processing Systems*, 32, 2019.

[18] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *CVPR*, 2018.

[19] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *CVPR*, pages 1209–1218, 2018.

[20] Kaixin Cai, Pengzhen Ren, Yi Zhu, Hang Xu, Jianzhuang Liu, Changlin Li, Guangrun Wang, and Xiaodan Liang. Mixreorg: Cross-modal mixed patch reorganization is a good mask learner for open-world semantic segmentation. In *ICCV*, pages 1196–1205, 2023.

[21] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, pages 6154–6162, 2018.

[22] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018.

[23] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, pages 7291–7299, 2017.

[24] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229. Springer, 2020.

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

[26] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, pages 9650–9660, 2021.

[27] Junbum Cha, Jonghwan Mun, and Byungseok Roh. Learning to generate text-grounded mask for open-world semantic segmentation from only image-text pairs. In *CVPR*, pages 11165–11174, 2023.

[28] Chun-Fu Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *ICCV*, 2021.

[29] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *CVPR*, 2021.

[30] Jie-Neng Chen, Shuyang Sun, Ju He, Philip HS Torr, Alan Yuille, and Song Bai. Transmix: Attend to mix for vision transformers. In *CVPR*, pages 12135–12144, 2022.

[31] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L Yuille, and Yuyin Zhou. Transunet: Transformers make strong encoders for medical image segmentation. *arXiv:2102.04306*, 2021.

[32] Jun Chen, Deyao Zhu, Guocheng Qian, Bernard Ghanem, Zhicheng Yan, Chenchen Zhu, Fanyi Xiao, Mohamed Elhoseiny, and Sean Chang Culatana. Exploring open-vocabulary semantic segmentation without human labels. *arXiv preprint arXiv:2306.00450*, 2023.

[33] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[34] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[35] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. *ICLR*, 2015.

[36] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*, 2015.

[37] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *T-PAMI*, 40(4):834–848, 2017.

[38] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE TPAMI*, 2017.

[39] Liang-Chieh Chen, Alexander Schwing, Alan Yuille, and Raquel Urtasun. Learning deep structured models. In *ICML*, 2015.

[40] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, pages 801–818, 2018.

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

[42] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML*, 2020.

[43] Peijie Chen, Qi Li, Saad Biaz, Trung Bui, and Anh Nguyen. gscorecam: What objects is clip looking at? In *ACCV*, pages 1959–1975, 2022.

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

[45] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. a^2-nets: Double attention networks. *arXiv preprint arXiv:1810.11579*, 2018.

[46] Yunpeng Chen, Marcus Rohrbach, Zhicheng Yan, Shuicheng Yan, Jiashi Feng, and Yannis Kalantidis. Graph-based global reasoning networks. In *CVPR*, 2019.

[47] Bowen Cheng, Liang-Chieh Chen, Yunchao Wei, Yukun Zhu, Zilong Huang, Jinjun Xiong, Thomas S Huang, Wen-Mei Hwu, and Honghui Shi. Spgnet: Semantic prediction guidance for scene parsing. In *ICCV*, 2019.

[48] Bowen Cheng, Anwesa Choudhuri, Ishan Misra, Alexander Kirillov, Rohit Girdhar, and Alexander G Schwing. Mask2former for video instance segmentation. In *CVPR*, 2022.

[49] Bowen Cheng, Maxwell D Collins, Yukun Zhu, Ting Liu, Thomas S Huang, Hartwig Adam, and Liang-Chieh Chen. Panoptic-DeepLab: A Simple, Strong, and Fast Baseline for Bottom-Up Panoptic Segmentation. In *CVPR*, 2020.

[50] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. *CVPR*, 2022.

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

[52] Bowen Cheng, Alexander G Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. In *NeurIPS*, 2021.

[53] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *CVPR*, 2017.

[54] Grace Chu, Okan Arikan, Gabriel Bender, Weijun Wang, Achille Brighton, Pieter-Jan Kindermans, Hanxiao Liu, Berkin Akin, Suyog Gupta, and Andrew Howard. Discovering multi-hardware mobile models via architecture search. In *CVPR*, pages 3022–3031, 2021.

[55] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021.

[56] Xiangxiang Chu, Bo Zhang, Zhi Tian, Xiaolin Wei, and Huaxia Xia. Do we really need explicit position encodings for vision transformers? *arXiv preprint arXiv:2102.10882*, 2021.

[57] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. https://github.com/open-mmlab/mmsegmentation, 2020.

[58] Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. On the relationship between self-attention and convolutional layers. In *ICLR*, 2020.

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

[60] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *CVPR*, 2019.

[61] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *CVPR*, 2019.

[62] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPRW*, 2020.

[63] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPRW*, 2020.

[64] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPR Workshops*, pages 702–703, 2020.

[65] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, pages 764–773, 2017.

[66] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. Up-detr: Unsupervised pre-training for object detection with transformers. In *CVPR*, 2021.

[67] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. Up-detr: Unsupervised pre-training for object detection with transformers. In *CVPR*, 2021.

[68] Zihang Dai, Hanxiao Liu, Quoc V Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes. *Advances in neural information processing systems*, 34:3965–3977, 2021.

[69] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *ACL*, 2019.

[70] Stéphane d'Ascoli, Hugo Touvron, Matthew Leavitt, Ari Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. *arXiv preprint arXiv:2103.10697*, 2021.

[71] Rina Dechter and Robert Mateescu. And/or search spaces for graphical models. *Artificial intelligence*, 171(2-3):73–106, 2007.

[72] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009.

[73] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009.

[74] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[75] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding. *NAACL*, 2019.

[76] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[77] Jian Ding, Nan Xue, Gui-Song Xia, and Dengxin Dai. Decoupling zero-shot semantic segmentation. In *CVPR*, pages 11583–11592, 2022.

[78] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[79] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[80] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[81] Alaaeldin El-Nouby, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. In *NeurIPS*, 2021.

[82] Gamaleldin F Elsayed, Simon Kornblith, and Quoc V Le. Saccader: improving accuracy of hard attention models for vision. *arXiv:1908.07644*, 2019.

[83] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html, 2012.

[84] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88:303–338, 2010.

[85] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88:303–338, 2010.

[86] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *ICCV*, pages 6824–6835, 2021.

[87] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. *arXiv preprint arXiv:2104.11227*, 2021.

[88] Yuxin Fang, Bencheng Liao, Xinggang Wang, Jiemin Fang, Jiyang Qi, Rui Wu, Jianwei Niu, and Wenyu Liu. You only look at one sequence: Rethinking transformer in vision through object detection. *arXiv preprint arXiv:2106.00666*, 2021.

[89] Mohsen Fayyaz, Soroush Abbasi Koohpayegani, Farnoush Rezaei Jafari, Sunando Sengupta, Hamid Reza Vaezi Joze, Eric Sommerlade, Hamed Pirsiavash, and Jürgen Gall. Adaptive token sampling for efficient vision transformers. In *ECCV*, pages 396–414. Springer, 2022.

[90] Wei Gao, Fang Wan, Xingjia Pan, Zhiliang Peng, Qi Tian, Zhenjun Han, Bolei Zhou, and Qixiang Ye. Ts-cam: Token semantic coupled attention map for weakly supervised object localization. In *ICCV*, 2021.

[91] Xiang Gao, Wei Hu, and Renjie Liao. Learning latent part-whole hierarchies for point clouds. *arXiv preprint arXiv:2211.07082*, 2022.

[92] Golnaz Ghiasi, Xiuye Gu, Yin Cui, and Tsung-Yi Lin. Scaling open-vocabulary image segmentation with image-level labels. In *ECCV*, pages 540–557. Springer, 2022.

[93] Ross Girshick. Fast R-CNN. In *ICCV*, 2015.

[94] Ross Girshick. Fast r-cnn. In *ICCV*, pages 1440–1448, 2015.

[95] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[96] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014.

[97] Chengyue Gong, Dilin Wang, Meng Li, Vikas Chandra, and Qiang Liu. Keepaugment: A simple information-preserving data augmentation approach. In *CVPR*, pages 1055–1064, 2021.

[98] Xiuye Gu, Yin Cui, Jonathan Huang, Abdullah Rashwan, Xuan Yang, Xingyi Zhou, Golnaz Ghiasi, Weicheng Kuo, Huizhong Chen, Liang-Chieh Chen, and David A Ross. Dataseg: Taming a universal multi-dataset multi-task segmentation model. *arXiv preprint arXiv:2306.01736*, 2023.

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

[100] Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. *arXiv preprint arXiv:2103.00112*, 2021.

[101] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *ECCV*, 2014.

[102] Ji He, Mari Ostendorf, Xiaodong He, Jianshu Chen, Jianfeng Gao, Lihong Li, and Li Deng. Deep reinforcement learning with a combinatorial action space for predicting popular reddit threads. In *EMNLP*, pages 770–778, 2016.

[103] Ju He, Jie-Neng Chen, Shuai Liu, Adam Kortylewski, Cheng Yang, Yutong Bai, Changhu Wang, and Alan Yuille. Transfg: A transformer architecture for fine-grained recognition. In *AAAI*, 2022.

[104] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pretraining. In *ICCV*, pages 4918–4927, 2019.

[105] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, pages 2961–2969, 2017.

[106] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[107] Wenbin He, Suphanut Jamonnak, Liang Gou, and Liu Ren. Clip-s4: Language-guided self-supervised semantic segmentation. In *CVPR*, pages 11207–11216, 2023.

[108] Xuming He, Richard S Zemel, and Miguel Á Carreira-Perpiñán. Multiscale conditional random fields for image labeling. In *CVPR*, 2004.

[109] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv:1606.08415*, 2016.

[110] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[111] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *ICLR*, 2020.

[112] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *CVPR*, pages 15262–15271, 2021.

[113] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In *ICCV*, 2021.

[114] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. *arXiv preprint arXiv:2103.16302*, 2021.

[115] Geoffrey Hinton. Some demonstrations of the effects of structural descriptions in mental imagery. *Cognitive Science*, 3(3):231–250, 1979.

[116] Geoffrey Hinton. How to represent part-whole hierarchies in a neural network. *arXiv preprint arXiv:2102.12627*, 2021.

[117] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv:1503.02531*, 2015.

[118] Geoffrey E Hinton, Sara Sabour, and Nicholas Frosst. Matrix capsules with em routing. In *ICLR*, 2018.

[119] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. *arXiv:1912.12180*, 2019.

[120] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. *arXiv:1912.12180*, 2019.

[121] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: better training with larger batches. In *CVPR*, 2020.

[122] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. In *CVPR*, pages 8129–8138, 2020.

[123] Weixiang Hong, Qingpei Guo, Wei Zhang, Jingdong Chen, and Wei Chu. Lpsnet: A lightweight solution for fast panoptic segmentation. In *CVPR*, pages 16746–16754, 2021.

[124] Rui Hou, Jie Li, Arjun Bhargava, Allan Raventos, Vitor Guizilini, Chao Fang, Jerome Lynch, and Adrien Gaidon. Real-time panoptic segmentation from dense detections. In *CVPR*, 2020.

[125] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *ICCV*, pages 1314–1324, 2019.

[126] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv:1704.04861*, 2017.

[127] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *CVPR*, 2018.

[128] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *ICCV*, 2019.

[129] Jie Hu, Linyan Huang, Tianhe Ren, Shengchuan Zhang, Rongrong Ji, and Liujuan Cao. You only segment once: Towards real-time panoptic segmentation. *CVPR*, 2023.

[130] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018.

[131] Ping Hu, Stan Sclaroff, and Kate Saenko. Uncertainty-aware learning for zero-shot semantic segmentation. *Advances in Neural Information Processing Systems*, 33:21713–21724, 2020.

[132] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In *CVPR*, 2017.

[133] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, pages 646–661. Springer, 2016.

[134] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, pages 646–661. Springer, 2016.

[135] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[136] Jitesh Jain, Jiachen Li, Mang Tik Chiu, Ali Hassani, Nikita Orlov, and Humphrey Shi. Oneformer: One transformer to rule universal image segmentation. In *CVPR*, 2023.

[137] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, pages 4904–4916. PMLR, 2021.

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

[139] Daniel Kahneman, Anne Treisman, and Brian J Gibbs. The reviewing of object files: Object-specific integration of information. *Cognitive psychology*, 24(2):175–219, 1992.

[140] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. Mdetr-modulated detection for end-to-end multi-modal understanding. In *ICCV*, pages 1780–1790, 2021.

[141] Laurynas Karazija, Iro Laina, Andrea Vedaldi, and Christian Rupprecht. Diffusion models for zero-shot open-vocabulary segmentation. *arXiv preprint arXiv:2306.09316*, 2023.

[142] Lei Ke, Mingqiao Ye, Martin Danelljan, Yifan Liu, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. Segment anything in high quality. *arXiv preprint arXiv:2306.01567*, 2023.

[143] Anna Khoreva, Anna Rohrbach, and Bernt Schiele. Video object segmentation with language referring expressions. In *ACCV*, pages 123–141. Springer, 2019.

[144] Dahun Kim, Jun Xie, Huiyu Wang, Siyuan Qiao, Qihang Yu, Hong-Seok Kim, Hartwig Adam, In So Kweon, and Liang-Chieh Chen. TubeFormer-DeepLab: Video Mask Transformer. In *CVPR*, 2022.

[145] Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In *ICML*, pages 5275–5285, 2020.

[146] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

[147] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[148] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, pages 6399–6408, 2019.

[149] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *CVPR*, pages 9404–9413, 2019.

[150] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *CVPR*, 2019.

[151] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.

[152] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *CVPR*, 2020.

[153] Adam Kortylewski, Qing Liu, Angtian Wang, Yihong Sun, and Alan Yuille. Compositional convolutional neural networks: A robust and interpretable model for object recognition under occlusion. *IJCV*, 129(3):736–760, 2021.

[154] Adam R Kosiorek, Sara Sabour, Yee Whye Teh, and Geoffrey E Hinton. Stacked capsule autoencoders. *arXiv preprint arXiv:1906.06818*, 2019.

[155] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *Advances in neural information processing systems*, 24, 2011.

[156] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *Advances in neural information processing systems*, 24, 2011.

[157] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV workshops*, pages 554–561, 2013.

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

[159] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[160] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. NIPS. In *NIPS*, 2012.

[161] Ananya Kumar, Percy Liang, and Tengyu Ma. Verified uncertainty calibration. In *NeurIPS*, 2019.

[162] Xin Lai, Jianhui Liu, Li Jiang, Liwei Wang, Hengshuang Zhao, Shu Liu, Xiaojuan Qi, and Jiaya Jia. Stratified transformer for 3d point cloud segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8500–8509, 2022.

[163] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. *arXiv preprint arXiv:1605.07648*, 2016.

[164] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *ECCV*, pages 734–750, 2018.

[165] Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and Rene Ranftl. Language-driven semantic segmentation. In *ICLR*, 2022.

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

[167] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv:1908.03557*, 2019.

[168] Peike Li, Yunchao Wei, and Yi Yang. Consistent structural relation learning for zero-shot segmentation. *NeurIPS*, 2020.

[169] Qizhu Li, Xiaojuan Qi, and Philip HS Torr. Unifying training and inference for panoptic segmentation. In *CVPR*, 2020.

[170] Yanghao Li, Haoqi Fan, Ronghang Hu, Christoph Feichtenhofer, and Kaiming He. Scaling language-image pre-training via masking. In *CVPR*, pages 23390–23400, 2023.

[171] Yanwei Li, Xinze Chen, Zheng Zhu, Lingxi Xie, Guan Huang, Dalong Du, and Xingang Wang. Attention-guided unified network for panoptic segmentation. In *CVPR*, 2019.

[172] Yanwei Li, Hengshuang Zhao, Xiaojuan Qi, Liwei Wang, Zeming Li, Jian Sun, and Jiaya Jia. Fully convolutional networks for panoptic segmentation. In *CVPR*, pages 214–223, 2021.

[173] Zhiheng Li, Wenxuan Bao, Jiayang Zheng, and Chenliang Xu. Deep grouping model for unified perceptual parsing. In *CVPR*, pages 4053–4063, 2020.

[174] Zhiqi Li, Wenhai Wang, Enze Xie, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, Tong Lu, and Ping Luo. Panoptic segformer: Delving deeper into panoptic segmentation with transformers. In *CVPR*, 2022.

[175] Feng Liang, Bichen Wu, Xiaoliang Dai, Kunpeng Li, Yinan Zhao, Hang Zhang, Peizhao Zhang, Peter Vajda, and Diana Marculescu. Open-vocabulary semantic segmentation with mask-adapted clip. In *CVPR*, pages 7061–7070, 2023.

[176] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

[177] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017.

[178] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014.

[179] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014.

[180] Yuqi Lin, Minghao Chen, Wenxiao Wang, Boxi Wu, Ke Li, Binbin Lin, Haifeng Liu, and Xiaofei He. Clip is also an efficient segmenter: A text-driven approach for weakly supervised semantic segmentation. In *CVPR*, pages 15305–15314, 2023.

[181] Chang Liu, Henghui Ding, and Xudong Jiang. GRES: Generalized referring expression segmentation. In *CVPR*, 2023.

[182] Huanyu Liu, Chao Peng, Changqian Yu, Jingbo Wang, Xu Liu, Gang Yu, and Wei Jiang. An end-to-end network for panoptic segmentation. In *CVPR*, 2019.

[183] Jihao Liu, Boxiao Liu, Hang Zhou, Hongsheng Li, and Yu Liu. Tokenmix: Rethinking image mixing for data augmentation in vision transformers. In *ECCV*, pages 455–471. Springer, 2022.

[184] Quande Liu, Youpeng Wen, Jianhua Han, Chunjing Xu, Hang Xu, and Xiaodan Liang. Open-world semantic segmentation via contrasting and clustering vision-language embedding. In *ECCV*, pages 275–292. Springer, 2022.

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

[186] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016.

[187] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 10012–10022, 2021.

[188] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021.

[189] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *CVPR*, 2022.

[190] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and

Thomas Kipf. Object-centric learning with slot attention. *arXiv preprint arXiv:2006.15055*, 2020.

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

[192] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015.

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

[194] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv:1711.05101*, 2017.

[195] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017.

[196] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[197] Fan Lu, Guang Chen, Yinlong Liu, Lijun Zhang, Sanqing Qu, Shu Liu, and Rongqi Gu. Hregnet: A hierarchical network for large-scale outdoor lidar point cloud registration. In *ICCV*, pages 16014–16023, 2021.

[198] Timo Lüddecke and Alexander Ecker. Image segmentation using text and image prompts. In *CVPR*, pages 7086–7096, 2022.

[199] Huaishao Luo, Junwei Bao, Youzheng Wu, Xiaodong He, and Tianrui Li. Segclip: Patch aggregation with learnable centers for open-vocabulary semantic segmentation. In *ICML*, pages 23033–23044. PMLR, 2023.

[200] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *CVPR*, pages 11–20, 2016.

[201] Xiaofeng Mao, Gege Qi, Yuefeng Chen, Xiaodan Li, Ranjie Duan, Shaokai Ye, Yuan He, and Hui Xue. Towards robust vision transformer. *CVPR*, 2022.

[202] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *ICCV*, pages 3651–3660, 2021.

[203] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *ICCV*, 2021.

[204] M Minderer, A Gritsenko, A Stone, M Neumann, D Weissenborn, A Dosovitskiy, A Mahendran, A Arnab, M Dehghani, Z Shen, et al. Simple open-vocabulary object detection with vision transformers. arxiv 2022. *arXiv preprint arXiv:2205.06230*, 2022.

[205] Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. Recurrent models of visual attention. *arXiv:1406.6247*, 2014.

[206] Rohit Mohan and Abhinav Valada. Efficientps: Efficient panoptic segmentation. *IJCV*, 129(5):1551–1579, 2021.

[207] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014.

[208] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014.

[209] Jishnu Mukhoti, Tsung-Yu Lin, Omid Poursaeed, Rui Wang, Ashish Shah, Philip HS Torr, and Ser-Nam Lim. Open vocabulary semantic segmentation with patch aligned contrastive learning. In *CVPR*, pages 19413–19423, 2023.

[210] Varun K Nagaraja, Vlad I Morariu, and Larry S Davis. Modeling context between objects for referring expression understanding. In *ECCV*, pages 792–807. Springer, 2016.

[211] Muzammal Naseer, Kanchana Ranasinghe, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. In *NeurIPS*, 2021.

[212] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *ICCV*, 2017.

[213] Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B Hall, Daniel Cer, and Yinfei Yang. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. *arXiv preprint arXiv:2108.08877*, 2021.

[214] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008.

[215] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, 2018.

[216] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[217] Matthew E Peters, Mark Neumann, Robert L Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. Knowledge enhanced contextual word representations. *arXiv preprint arXiv:1909.04164*, 2019.

[218] Lorenzo Porzi, Samuel Rota Bulò, Aleksander Colovic, and Peter Kontschieder. Seamless scene segmentation. In *CVPR*, 2019.

[219] Siyuan Qiao, Liang-Chieh Chen, and Alan Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. In *CVPR*, 2021.

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

[221] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[222] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *CVPR*, 2020.

[223] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *CVPR*, 2020.

[224] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. In *NeurIPS*, 2019.

[225] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. In *NeurIPS*, 2019.

[226] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Swish: a self-gated activation function. *arXiv:1710.05941*, 2017.

[227] Kanchana Ranasinghe, Brandon McKinzie, Sachin Ravi, Yinfei Yang, Alexander Toshev, and Jonathon Shlens. Perceptual grouping in vision-language models. *arXiv preprint arXiv:2210.09996*, 2022.

[228] Abdullah Rashwan, Yeqing Li, Xingyi Zhou, Jiageng Zhang, and Fan Yang. Maskconver: A universal panoptic and semantic segmentation model with pure convolutions. *OpenReview*, 2023.

[229] Pengzhen Ren, Changlin Li, Hang Xu, Yi Zhu, Guangrun Wang, Jianzhuang Liu, Xiaojun Chang, and Xiaodan Liang. Viewco: Discovering text-supervised segmentation masks via multi-view semantic consistency. *arXiv preprint arXiv:2302.10307*, 2023.

[230] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

[231] Ronald A Rensink. The dynamic representation of scenes. *Visual cognition*, 2000.

[232] Lixiang Ru, Yibing Zhan, Baosheng Yu, and Bo Du. Learning affinity from attention: End-to-end weakly-supervised semantic segmentation with transformers. In *CVPR*, pages 16846–16855, 2022.

[233] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[234] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015.

[235] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. *arXiv preprint arXiv:1710.09829*, 2017.

[236] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018.

[237] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018.

[238] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, pages 618–626, 2017.

[239] Zhiqiang Shen, Zechun Liu, Zhuang Liu, Marios Savvides, Trevor Darrell, and Eric Xing. Un-mix: Rethinking image mixtures for unsupervised visual representation learning. *AAAI*, 2022.

[240] Fenfen Sheng, Zhineng Chen, and Bo Xu. Nrtr: A no-recurrence sequence-to-sequence model for scene text recognition. In *ICDAR*, 2019.

[241] Gyungin Shin, Weidi Xie, and Samuel Albanie. Reco: Retrieve and co-segment for zero-shot transfer. *Advances in Neural Information Processing Systems*, 35:33754–33767, 2022.

[242] Inkyu Shin, Dahun Kim, Qihang Yu, Jun Xie, Hong-Seok Kim, Bradley Green, In So Kweon, Kuk-Jin Yoon, and Liang-Chieh Chen. Video-kmax: A simple unified approach for online and near-online video panoptic segmentation. *arXiv preprint arXiv:2304.04694*, 2023.

[243] Aleksandar Shtedritski, Christian Rupprecht, and Andrea Vedaldi. What does clip know about a red circle? visual prompt engineering for vlms. *arXiv preprint arXiv:2304.06712*, 2023.

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

[245] Richard Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The annals of mathematical statistics*, 35(2):876–879, 1964.

[246] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition. *arXiv:2101.11605*, 2021.

[247] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *ICCV*, 2021.

[248] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *ICCV*, 2021.

[249] Robin Strudel, Ivan Laptev, and Cordelia Schmid. Weakly-supervised segmentation of referring expressions. *arXiv preprint arXiv:2205.04725*, 2022.

[250] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *NeurIPS*, 2015.

[251] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *ICCV*, 2019.

[252] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, et al. Sparse r-cnn: End-to-end object detection with learnable proposals. In *CVPR*, 2021.

[253] Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. Eva-clip: Improved training techniques for clip at scale. *arXiv preprint arXiv:2303.15389*, 2023.

[254] Shuyang Sun, Liang Chen, Gregory Slabaugh, and Philip Torr. Learning to sample the most useful training patches from images. *arXiv preprint arXiv:2011.12097*, 2020.

[255] Shuyang Sun, Runjia Li, Philip Torr, Xiuye Gu, and Siyang Li. Clip as rnn: Segment countless visual concepts without training endeavor. *arXiv preprint arXiv:2312.07661*, 2023.

[256] Shuyang Sun, Jiangmiao Pang, Jianping Shi, Shuai Yi, and Wanli Ouyang. Fishnet: A versatile backbone for image, region, and pixel level prediction. *arXiv preprint arXiv:1901.03495*, 2019.

[257] Shuyang Sun, Weijun Wang, Qihang Yu, Andrew Howard, Philip Torr, and Liang-Chieh Chen. Remax: Relaxing for better training on efficient panoptic segmentation. *arXiv preprint arXiv:2306.17319*, 2023.

[258] Shuyang Sun, Xiaoyu Yue, Song Bai, and Philip Torr. Visual parser: Representing part-whole hierarchies with transformers. *arXiv preprint arXiv:2107.05790*, 2021.

[259] Zhiqing Sun, Shengcao Cao, Yiming Yang, and Kris Kitani. Rethinking transformer-based set prediction for object detection. *arXiv:2011.10881*, 2020.

[260] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. In *CVPR*, 2015.

[261] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pages 2818–2826, 2016.

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

[263] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. In *ECCV*, 2020.

[264] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, pages 9627–9636, 2019.

[265] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv:2012.12877*, 2020.

[266] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv:2012.12877*, 2020.

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

[268] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *ICCV*, 2021.

[269] Yao-Hung Hubert Tsai, Nitish Srivastava, Hanlin Goh, and Ruslan Salakhutdinov. Capsules with inverted dot-product attention routing. *arXiv preprint arXiv:2002.04764*, 2020.

[270] Zhuowen Tu, Xiangrong Chen, Alan L Yuille, and Song-Chun Zhu. Image parsing: Unifying segmentation, detection, and recognition. *IJCV*, 63(2):113–140, 2005.

[271] AFM Uddin, Mst Monira, Wheemyung Shin, TaeChoong Chung, Sung-Ho Bae, et al. Saliencymix: A saliency guided data augmentation strategy for better regularization. In *ICLR*, 2021.

[272] Ashish Vaswani, Prajit Ramachandran, Aravind Srinivas, Niki Parmar, Blake Hechtman, and Jonathon Shlens. Scaling local self-attention for parameter efficient visual backbones. *arXiv preprint arXiv:2103.12731*, 2021.

[273] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, volume 30, 2017.

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

[275] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017.

[276] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, pages 6438–6447, 2019.

[277] Devesh Walawalkar, Zhiqiang Shen, Zechun Liu, and Marios Savvides. Attentive cutmix: An enhanced data augmentation approach for deep learning based image classification. *arXiv preprint arXiv:2003.13048*, 2020.

[278] Haoxiang Wang, Pavan Kumar Anasosalu Vasu, Fartash Faghri, Raviteja Vemulapalli, Mehrdad Farajtabar, Sachin Mehta, Mohammad Rastegari, Oncel Tuzel, and Hadi Pouransari. Sam-clip: Merging vision foundation models towards semantic and spatial understanding. *arXiv preprint arXiv:2310.15308*, 2023.

[279] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen.

Max-deeplab: End-to-end panoptic segmentation with mask transformers. In *CVPR*, pages 5463–5474, 2021.

[280] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. In *CVPR*, 2021.

[281] Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In *ECCV*, 2020.

[282] Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Axial-DeepLab: Stand-Alone Axial-Attention for Panoptic Segmentation. In *ECCV*, 2020.

[283] Jiaqi Wang, Wenwei Zhang, Yuhang Zang, Yuhang Cao, Jiangmiao Pang, Tao Gong, Kai Chen, Ziwei Liu, Chen Change Loy, and Dahua Lin. Seesaw loss for long-tailed instance segmentation. In *CVPR*, pages 9695–9704, 2021.

[284] Junke Wang, Zuxuan Wu, Jingjing Chen, Xintong Han, Abhinav Shrivastava, Ser-Nam Lim, and Yu-Gang Jiang. Objectformer for image manipulation detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2364–2373, 2022.

[285] Peng Wang, Lu Yang, Hui Li, Yuyan Deng, Chunhua Shen, and Yanning Zhang. A simple and robust convolutional-attention network for irregular text recognition. *arXiv preprint arXiv:1904.01375*, 2019.

[286] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*, pages 568–578, 2021.

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

[288] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018.

[289] Xinlong Wang, Zhiding Yu, Shalini De Mello, Jan Kautz, Anima Anandkumar, Chunhua Shen, and Jose M Alvarez. Freesolo: Learning to segment objects without annotations. In *CVPR*, pages 14176–14186, 2022.

[290] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic and fast instance segmentation. *arXiv preprint arXiv:2003.10152*, 2020.

[291] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. *arXiv:2011.14503*, 2020.

[292] Mark Weber, Huiyu Wang, Siyuan Qiao, Jun Xie, Maxwell D. Collins, Yukun Zhu, Liangzhe Yuan, Dahun Kim, Qihang Yu, Daniel Cremers, Laura Leal-Taixe, Alan L. Yuille, Florian Schroff, Hartwig Adam, and Liang-Chieh Chen. DeepLab2: A TensorFlow Library for Deep Labeling. *arXiv: 2106.09748*, 2021.

[293] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *CVPR*, pages 4724–4732, 2016.

[294] Dirk Weissenborn, Oscar Täckström, and Jakob Uszkoreit. Scaling Autoregressive Video Models. In *ICLR*, 2020.

[295] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

[296] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Masayoshi Tomizuka, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision. *arXiv:2006.03677*, 2020.

[297] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Masayoshi Tomizuka, Kurt Keutzer, and Peter Vajda. Visual transform-

ers: Token-based image representation and processing for computer vision. *arXiv:2006.03677*, 2020.

[298] Felix Wu, Angela Fan, Alexei Baevski, Yann N Dauphin, and Michael Auli. Pay less attention with lightweight and dynamic convolutions. In *ICLR*, 2019.

[299] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021.

[300] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv:1609.08144*, 2016.

[301] Fangting Xia, Jun Zhu, Peng Wang, and Alan Yuille. Pose-guided human parsing by an and/or graph using pose-context features. In *AAAI*, volume 30, 2016.

[302] Zhuofan Xia, Xuran Pan, Shiji Song, Li Erran Li, and Gao Huang. Vision transformer with deformable attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4794–4803, 2022.

[303] Yongqin Xian, Subhabrata Choudhury, Yang He, Bernt Schiele, and Zeynep Akata. Semantic projection network for zero-and few-label semantic segmentation. In *CVPR*, pages 8256–8265, 2019.

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

[305] Jinheng Xie, Xianxu Hou, Kai Ye, and Linlin Shen. Clims: Cross language image matching for weakly supervised semantic segmentation. In *CVPR*, pages 4483–4492, 2022.

[306] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.

[307] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.

[308] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *ECCV*, 2018.

[309] Yuwen Xiong, Renjie Liao, Hengshuang Zhao, Rui Hu, Min Bai, Ersin Yumer, and Raquel Urtasun. Upsnet: A unified panoptic segmentation network. In *CVPR*, 2019.

[310] Jiarui Xu, Shalini De Mello, Sifei Liu, Wonmin Byeon, Thomas Breuel, Jan Kautz, and Xiaolong Wang. Groupvit: Semantic segmentation emerges from text supervision. In *CVPR*, 2022.

[311] Jiarui Xu, Sifei Liu, Arash Vahdat, Wonmin Byeon, Xiaolong Wang, and Shalini De Mello. Open-vocabulary panoptic segmentation with text-to-image diffusion models. In *CVPR*, pages 2955–2966, 2023.

[312] Jilan Xu, Junlin Hou, Yuejie Zhang, Rui Feng, Yi Wang, Yu Qiao, and Weidi Xie. Learning open-vocabulary semantic segmentation models from natural language supervision. In *CVPR*, pages 2935–2944, 2023.

[313] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015.

[314] Lian Xu, Wanli Ouyang, Mohammed Bennamoun, Farid Boussaid, and Dan Xu. Multi-class token transformer for weakly supervised semantic segmentation. In *CVPR*, pages 4310–4319, 2022.

[315] Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. Learning texture transformer network for image super-resolution. In *CVPR*, 2020.

[316] Tien-Ju Yang, Maxwell D Collins, Yukun Zhu, Jyh-Jing Hwang, Ting Liu, Xiao Zhang, Vivienne Sze, George Papandreou, and Liang-Chieh Chen. Deeperlab: Single-shot image parser. *arXiv:1902.05093*, 2019.

[317] Zhao Yang, Jiaqi Wang, Yansong Tang, Kai Chen, Hengshuang Zhao, and Philip HS Torr. Lavt: Language-aware vision transformer for referring image segmentation. In *CVPR*, pages 18155–18165, 2022.

[318] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv:1906.08237*, 2019.

[319] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *ICLR*, 2016.

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

[321] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *ECCV*, pages 69–85. Springer, 2016.

[322] Qihang Yu, Ju He, Xueqing Deng, Xiaohui Shen, and Liang-Chieh Chen. Convolutions die hard: Open-vocabulary segmentation with single frozen convolutional clip. *arXiv preprint arXiv:2308.02487*, 2023.

[323] Qihang Yu, Huiyu Wang, Dahun Kim, Siyuan Qiao, Maxwell Collins, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Cmt-deeplab: Clustering mask transformers for panoptic segmentation. In *CVPR*, 2022.

[324] Qihang Yu, Huiyu Wang, Siyuan Qiao, Maxwell Collins, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. k-means Mask Transformer. In *ECCV*, 2022.

[325] Qihang Yu, Yingda Xia, Yutong Bai, Yongyi Lu, Alan L Yuille, and Wei Shen. Glance-and-gaze vision transformer. *NeurIPS*, 2021.

[326] Seonghoon Yu, Paul Hongsuck Seo, and Jeany Son. Zero-shot referring image segmentation with global-local context features. In *CVPR*, pages 19456–19465, 2023.

[327] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zi-Hang Jiang, Francis E.H. Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *ICCV*, pages 558–567, October 2021.

[328] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021.

[329] Xiaoyu Yue, Zhanghui Kuang, Chenhao Lin, Hongbin Sun, and Wayne Zhang. Robustscanner: Dynamically enhancing positional clues for robust text recognition. In *ECCV*, 2020.

[330] Xiaoyu Yue, Zhanghui Kuang, Zhaoyang Zhang, Zhenfang Chen, Pan He, Yu Qiao, and Wei Zhang. Boosting up scene text detectors with guided cnn. *arXiv preprint arXiv:1805.04132*, 2018.

[331] Xiaoyu Yue, Shuyang Sun, Zhanghui Kuang, Meng Wei, Philip HS Torr, Wayne Zhang, and Dahua Lin. Vision transformer with progressive sampling. In *ICCV*, pages 387–396, 2021.

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

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

[334] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, pages 6023–6032, 2019.

[335] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. *arXiv preprint arXiv:2303.15343*, 2023.

[336] Xiaohua Zhai, Xiao Wang, Basil Mustafa, Andreas Steiner, Daniel Keysers, Alexander Kolesnikov, and Lucas Beyer. Lit: Zero-shot transfer with locked-image text tuning. In *CVPR*, pages 18123–18133, 2022.

[337] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. *arXiv preprint arXiv:2004.08955*, 2020.

[338] Hao Zhang, Feng Li, Xueyan Zou, Shilong Liu, Chunyuan Li, Jianwei Yang, and Lei Zhang. A simple framework for open-vocabulary segmentation and detection. In *ICCV*, pages 1020–1031, 2023.

[339] Haotian Zhang, Pengchuan Zhang, Xiaowei Hu, Yen-Chun Chen, Liunian Li, Xiyang Dai, Lijuan Wang, Lu Yuan, Jenq-Neng Hwang, and Jianfeng Gao. Glipv2: Unifying localization and vision-language understanding. *Advances in Neural Information Processing Systems*, 35:36067–36080, 2022.

[340] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

[341] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

[342] Li Zhang, Jiachen Lu, Sixiao Zheng, Xinxuan Zhao, Xiatian Zhu, Yanwei Fu, Tao Xiang, Jianfeng Feng, and Philip HS Torr. Vision transformers: From semantic segmentation to dense prediction. *arXiv preprint arXiv:2207.09339*, 2022.

[343] Pengchuan Zhang, Xiyang Dai, Jianwei Yang, Bin Xiao, Lu Yuan, Lei Zhang, and Jianfeng Gao. Multi-scale vision longformer: A new vision transformer for high-resolution image encoding. In *ICCV*, pages 2998–3008, 2021.

[344] Wenwei Zhang, Jiangmiao Pang, Kai Chen, and Chen Change Loy. K-net: Towards unified image segmentation. In *NeurIPS*, 2021.

[345] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *CVPR*, 2018.

[346] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *CVPR*, 2020.

[347] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017.

[348] Qihao Zhao, Yangyu Huang, Wei Hu, Fan Zhang, and Jun Liu. Mixpro: Data augmentation with maskmix and progressive attention labeling for vision transformer. In *ICLR*, 2023.

[349] Minghang Zheng, Peng Gao, Xiaogang Wang, Hongsheng Li, and Hao Dong. End-to-end object detection with adaptive clustering transformer. *arXiv:2011.09315*, 2020.

[350] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *ICCV*, 2015.

[351] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *AAAI*, 2020.

[352] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017.

[353] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *IJCV*, 2019.

[354] Chong Zhou, Chen Change Loy, and Bo Dai. Extract free dense labels from clip. In *ECCV*, pages 696–712. Springer, 2022.

[355] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *ECCV*, pages 350–368. Springer, 2022.

[356] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.

[357] Jinjing Zhu, Haotian Bai, and Lin Wang. Patch-mix transformer for unsupervised domain adaptation: A game perspective. In *CVPR*, pages 3561–3571, 2023.

[358] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *CVPR*, pages 9308–9316, 2019.

[359] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021.

[360] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021.

[361] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021.