LEARNING HIERARCHICAL IMAGE SEGMENTATION FOR RECOGNITION AND BY RECOGNITION

Tsung-Wei Ke[*1] **Sangwoo Mo**[*2] **Stella X. Yu**[1,2]
[1]University of California, Berkeley [2]University of Michigan, Ann Arbor
{twke,stellayu}@berkeley.edu {swmo,stellayu}@umich.edu

ABSTRACT

Image segmentation and recognition occur simultaneously, with recognition relying on the underlying segmentation to form a continuous visual grouping hierarchy. For example, the same object can be parsed into different part-to-whole structures, resulting in varying recognitions. Despite this, most prior works treated segmentation and recognition as separate tasks. In this paper, we aim to devise a learning framework that involves segmentation in the recognition process, utilizing hierarchical segmentation *for* recognition, which is learned *by* recognition. Specifically, we propose CAST, which realizes this concept through designs inspired by vision transformers, enabling concurrent segmentation and recognition with a single model. The core idea of CAST is to employ adaptive segment tokens that group the finest pixels into coarser segments, using the latest embedding to represent the entire image for recognition. Trained solely on image recognition objectives, CAST automatically discovers the hierarchy of segments. Our experiments demonstrate that CAST provides consistent hierarchical segmentation and recognition, which is impossible with state-of-the-art segmentation methods such as SAM. Additionally, CAST offers several advantages over the standard ViT, including improved semantic segmentation, computational efficiency, and object-centric attention.[1]

1 INTRODUCTION

Human vision has two pathways: **1)** bottom-up processing, where the composition of distinctive parts facilitates coarse scene understanding (Biederman, 1987), and **2)** top-down processing, where coarse recognition explains connected parts (Maurer et al., 2002). These two pathways necessitate recognition for both parts and wholes to be done concurrently, influencing each other (Tanaka & Farah, 1993; Tanaka & Simonyi, 2016; Tanaka et al., 2019). Take the famous *girl vs. woman illusion* in Fig. 1, for example. When we look at the *girl*, we perceive her profile face but not the nose of the *old woman*. In contrast, observing a mouth helps us discern the *old woman*. It appears that our recognition of individual parts is intricately connected to our ability to recognize the whole. In computer vision, the recognition of parts and the whole is often formulated as hierarchical segmentation and image recognition problems. We are thus motivated to ask: Can we develop a vision framework integrating this part-to-whole segmentation structure in the loop for image recognition?

Prior works often treat segmentation and recognition as separate tasks, each involving distinct models and annotations. Recognition models are supervised with per-image labels, such as categories (Deng et al., 2009) or instances (Wu et al., 2018). These methods encode input images into global features (He et al., 2016; Dosovitskiy et al., 2020), disregarding local pixel-level information but focusing on the most distinguishing parts (Selvaraju et al., 2017). Conversely, segmentation models are supervised with pixel labels (Long et al., 2015; Kirillov et al., 2023). Segmentation models often inject global information into local pixel features with skip connections (Lin et al., 2017; Cheng et al., 2021), facilitating pixel label prediction. Due to distinct architectural designs, recognition models cannot be used out-of-the-box for segmentation tasks (Ding et al., 2022), but require additional adaptation of architectures and fine-tuning on segmentation labels. Meanwhile, image-level information requires structural modulation so as to be useful for segmentation models (Ahn & Kwak, 2018).

[*]Equal Contribution
[1]Code: https://github.com/twke18/CAST

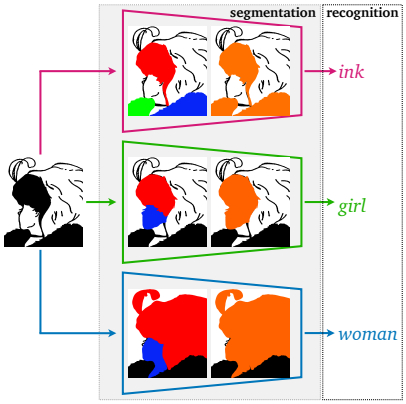


Figure 1: **Motivation.** Our insight is that image segmentation and recognition are a continuum of visual grouping hierarchy at segment- and image- levels, and their *consistency* is more essential than *what* we recognize from the image. Given the *girl vs. woman illusion*, we may recognize *ink*, *girl*, or *woman*. While the identified foreground (colored areas) may vary, there is always a consistent hierarchical segmentation: *individual ink blobs* when *no person* is recognized, or *parts* (e.g., face, hair) of the *person* recognized as *girl* or *woman*. Instead of treating segmentation and recognition as separate tasks, we model them concurrently by *putting segmentation in the loop for recognition*. Consequently, with only recognition objectives *at the image level*, we can learn image segmentation *for free*, and obtain better performance by capturing intrinsic part-whole grouping consistency.

In this paper, we propose a new learning paradigm that incorporates segmentation into the recognition process. Fig. 2 illustrates our concept, which views segmentation and recognition as a *continuum* within a visual grouping hierarchy, rather than as two separate tasks. Specifically, our model processes the input image into fine-to-coarse segmentation, corresponding to part-to-whole recognition, ultimately resulting in a global image embedding that represents the entire image. It is important to note that segmentation serves solely as a means for recognition, not as the end objective. Consequently, the model can be trained solely based on the objectives of the final image embeddings, without the need for fine-grained supervision of intermediate features.

Our concept can be summarized as: *"segmentation of recognition, by recognition, and for recognition."* **1)** "Of" implies our motivation, that segmentation is a citizen of recognition, with each segment participating in the recognition of semantics. **2)** "By" implies our learning process, reliant solely on image-level objectives, without the need for fine-grained annotations. **3)** "For" implies the outcomes of our model, learning part-to-whole structures grounding recognition. Through this process, our model automatically discovers consistent and semantically aligned hierarchical segmentation.

We implement this concept using a method inspired by Vision Transformer (ViT) (Dosovitskiy et al., 2020). To achieve this, we propose two novel components for ViT. **1)** We use superpixels instead of square grids as units for ViT. **2)** We group the tokens as layers progress to form a hierarchy with graph pooling. These two components make the intermediate features correspond to segments, forming a hierarchy with finer ones for lower layers and coarser ones for higher layers. This part-to-whole structure is solely learned from image recognition objectives, either unsupervised (He et al., 2020) or supervised (Touvron et al., 2021). In summary, our model Concurrently learns segmentation and recognition using Adaptive Segment Tokens, so we call it CAST. We remark that superpixels are natural choice for designing a *vision* transformer, unlike text-inspired ViTs using square patches.

We introduce a novel vision model–CAST–that inherently incorporates both bottom-up and top-down pathways and naturally put segmentation in the loop of image recognition. We highlight the insights of this paper:

1. CAST naturally derives hierarchical segmentation by grouping segment tokens in a fine-to-coarse manner. The pixel groupings remain consistent along the hierarchy. In stark contrast, state-of-the-art segmentation models like Segment Anything (SAM) (Kirillov et al., 2023) are trained on huge amounts of segmentation labels (1 billion masks), yet still lack the understanding of part-of-the-whole relationships. SAM fails to generate consistent hierarchical segmentation.
2. CAST learns to generate hierarchical segmentation *for free* from the image recognition objective. Our model adapts such intermediate representations to best explain the final image recognition. Moreover, by optimizing our model with a test-time adaptation (TTA) setup (Sun et al., 2020), our intermediate segmentations improve with the final recognition results (Fig. 3 and Appx. B).
3. CAST concurrently performs segmentation and recognition, achieving competitive performance with prior works—HSG (Ke et al., 2022) for hierarchical segmentation and Swin Transformer (Liu et al., 2021) for classification—which require specially designed architectures for each task.
4. CAST is a natural design of a vision transformer that takes input tokens as superpixels instead of square patches. Such a design enables our model to predict segmentation more accurately

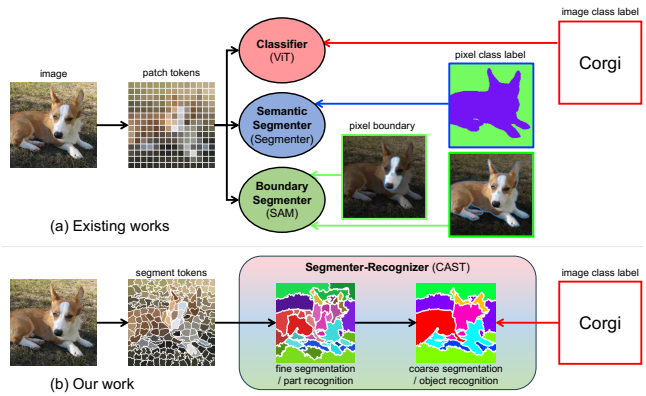


Figure 2: **Concept.** (a) Prior works treat segmentation and recognition as separate tasks using independent models and supervision. (b) CAST considers both tasks as a *continuum* within a visual grouping hierarchy. CAST processes the image from fine to coarse segments, corresponding to part and object recognition. The final features aggregate all the information, leading to image recognition. CAST is trained solely on an image-level objective, without the necessity of fine-grained labels.

compared to prior patch-based ViTs, outperforming on multiple tasks, including unsupervised/supervised semantic segmentation and attention-based figure-ground segmentation.

2 Related Work

Recognition by segmentation. This concept was explored before the deep learning era. Recognition by grouping compatible patches and segmentation by grouping visually similar pixels were simultaneously addressed through detected pixel-patch relations, resulting in object-specific segmentation (Yu et al., 2002; Yu & Shi, 2003b) and figure-ground segmentation (Maire, 2010; Maire et al., 2011). However, these approaches relied on manually crafted features, grouping cues, and pre-trained object part detectors. In contrast, our work does not rely on any such prior information.

Vision transformers. Since the remarkable success of Vision Transformer (ViT) for image recognition (Dosovitskiy et al., 2020), numerous advanced architectures have emerged (Han et al., 2022). One notable direction explores hierarchical transformers, aiming to reduce the number of tokens for improved efficiency. This reduction can be achieved through spatial pooling, drawing on concepts from hierarchical convolution (Liu et al., 2021; Heo et al., 2021; Dong et al., 2022; Ma et al., 2023). Another approach involves measuring token significance scores and pruning tokens accordingly (Goyal et al., 2020; Rao et al., 2021; Marin et al., 2021; Zeng et al., 2022; Bolya et al., 2023). While CAST shares technical similarities with the second approach, **1)** Clustering is a byproduct of implementing our concept, unlike prior works targeting efficiency. **2)** CAST can produce consistent hierarchical segmentation, which was not achievable with previous ViT-based models.

Hierarchical image segmentation. The task is to decompose an image into segments in a fine-to-coarse manner. Classic methods often consider the task as an agglomerative clustering problem (Arbelaez et al., 2010; Sharon et al., 2006), which extracts pixel features, initializes clusterings of pixels, and merges clusters based on feature similarity. Recent works often adopt a supervised approach through top-down decomposition, detecting coarse-semantic instances and breaking each instance into fine-semantic parts (de Geus et al., 2021; Wei et al., 2024; Li et al., 2022b;a). As parts are expensive to annotate, self-supervised methods are used to augment part segmentation of novel categories, through cross-image pixel correspondence (Sun et al., 2023) or feature clustering (Pan et al., 2023). Another line of works predict part- and object-level segmentation separately (Li et al., 2023; Wang et al., 2024; Qi et al., 2024), resulting in misalignment of segmentation across different granularities. CAST modernizes the agglomerative approach, integrating **1)** representation learning, **2)** hierarchical segmentation, and **3)** image recognition within a transformer architecture.

Additional related works, such as superpixels and clustering methods, can be found in Appx. C.

3 Our CAST Framework

Prior arts, such as ViT, are not designed to tackle recognition and segmentation concurrently. **1)** They often use fixed-shape patch tokens, which do not align precisely with object contours (Bolya et al., 2022). **2)** They do not explicitly model segmentation (pixel groupings). The grouping information is encoded implicitly in the attention map and is not enforced to be hierarchical.

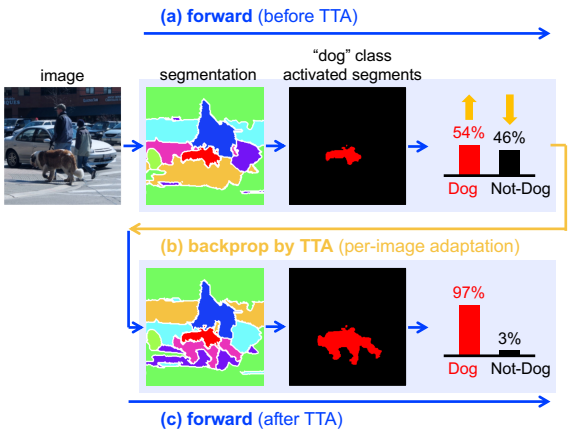


Figure 3: **Adaptive segmentation to image recognition.** Our CAST employs both forward (bottom-up) and backward (top-down) pathways for concurrent segmentation and recognition. (a) Forward process from the self-supervised CAST focuses on all elements in the scene. (b) Backward process adapts the model to recognize the image as a dog through updating parameters test-time adaptation (TTA). (c) Forward process after TTA provides segmentation that better focuses on the dog. These two processes assist visual perception by segmenting and recognizing parts and wholes interactively.

We develop our CAST framework, which explicitly predicts intermediate hierarchical segmentation to support the final image recognition, based on the transformer architecture (Vaswani et al., 2017). Our method has two technical insights: **1)** CAST adopts adaptive segment tokens, not fixed-shape patches, and **2)** CAST adaptively clusters the finer segments into coarser regions with a graph pooling module in the intermediate layers. These design choices not only ensure high-quality and consistent hierarchical segmentation but also unify both segmentation and recognition tasks. In Fig. 3, we show that the intermediate segmentation adapts to the input image with the final image recognition.

Our model alternates between transformer encoder blocks and graph pooling to extract features for each segment token and merge finer segments into coarser segment tokens. We present an overview of the framework in Fig. 4 and show the detailed algorithm in Appx. D.1. We describe segment tokens, the graph pooling module, and implementation details in the following subsections.

3.1 Segment Tokens from Superpixels

For processing an input image, we begin with the finest-level pixel groupings, denoted as S_0, which represent the initial image segmentation. These groupings are based on low-level visual cues and designed to align with image contours. Specifically, we use oversegmentation methods, such as SEEDS (Bergh et al., 2012),[2] to partition the image into locally connected and color-wise coherent regions, known as superpixels. Then, we progressively group these superpixels into coarser regions to create a hierarchical segmentation with precisely localized contours.

To extract features from the superpixels, which can have arbitrary shapes, we initiate the process by generating pixel features from the input image, denoted as X_{cnn}, using convolutional layers. These pixel features are then aggregated within each superpixel in S_0 to create the initial segment tokens, referred to as X_s. These tokens possess dimensions corresponding to the number of superpixels and the number of feature channels. Specifically, we compute X_s by averaging the CNN features X_{cnn} across all pixels i within each superpixel a: $X_s = \frac{1}{|a|}\sum_{i\in a} X_{\text{cnn}}[i] | a \in S_0$. Following this, we append a class token X_{class}, and positional encodings E_{pos} into the initial features X_s. We set E_{pos} to align with the resolution of the CNN features X_{cnn} and then average it within each superpixel. The resulting input segment features are defined as $Z_0 = [X_{\text{class}}; X_s] + [\mathbf{0}; E_{\text{pos}}]$. We apply transformer encoder blocks for extracting visual features from such input tokens. We refer to Dosovitskiy et al. (2020) for more details on the encoder block.

In contrast to existing methods, such as SegSort (Hwang et al., 2019) and HSG (Ke et al., 2022), which handle image segmentation and feature extraction separately, our approach seamlessly integrates image segmentation into the model architecture during feature extraction. We initiate pixel groupings early in the process, allowing us to extract corresponding features for each segment. These segment features are then carried to subsequent layers, enabling the model to derive image segmentations directly from the segment index of each pixel, eliminating the need for post-processing.

[2]Further discussions on the choice of superpixel methods are in Appx. A.2.

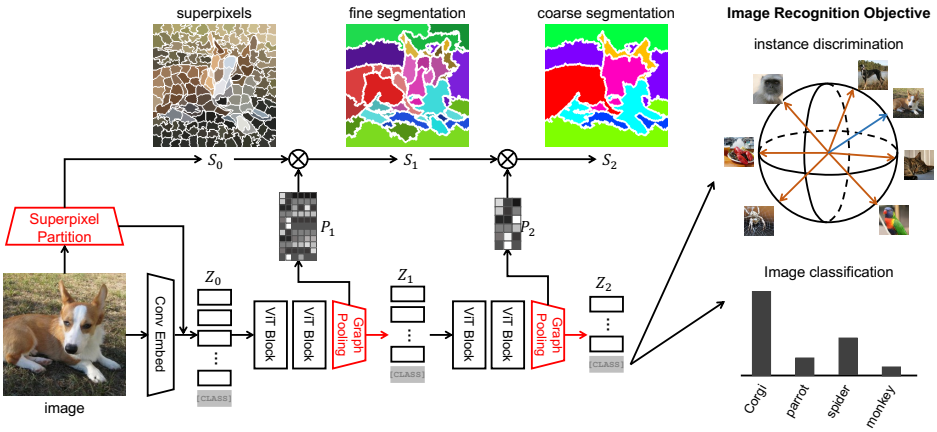


Figure 4: **CAST framework.** We implement our learning principle using a ViT-inspired architecture. In our CAST model, we *concurrently* predict hierarchical image segmentation and recognition by employing *adaptive segment tokens*. CAST employs two technical contributions over ViT: it uses superpixels instead of square patches and applies graph pooling to adaptively cluster them. Our model does not need any fine-grained part-of-the-whole annotation, but learns to generate consistent hierarchical segmentation with an image recognition objective, such as self-supervised instance discrimination or supervised image classification.

3.2 Graph Pooling for Hierarchical Segmentation

Starting with superpixels S_0, we group fine-grained segment tokens Z_0 into L levels of coarser region tokens $(Z_1, \ldots, Z_L)$ to capture a more comprehensive global visual context. This fine-to-coarse segment grouping enables us to create hierarchical image segmentations $(S_1, \ldots, S_L)$ directly for an input image, reframing the task as a multi-scale feature clustering and pooling problem.

To obtain coarser segmentations, we merge finer segments into coarser regions by calculating the soft assignment probability P_l. This probability maps a fine segment a at level $l-1$ to a coarse region c at level l. We initiate the next-level coarse regions by sampling centroids from segment tokens Z_{l-1} at level $l-1$ based on token similarity in the feature space and compute P_l, with C representing the sampled centroid indices. The relationship is defined as follows:

$$P_l = \mathrm{Prob}(a, c) \propto \mathrm{sim}(Z_{l-1}[a], Z_{l-1}[c]), c \in C. \tag{1}$$

The coarser segmentation S_l, representing the segment index of each pixel at level l, is obtained by combining the initial segmentation S_0 with clustering assignments across levels:

$$S_l = S_{l-1} \times P_l = S_0 \times P_1 \times \cdots \times P_l. \tag{2}$$

To derive segment tokens Z_l corresponding to the coarser segmentation S_l, we aggregate previous-level tokens Z_{l-1} using P_l and add them to the sampled token centroids. We apply a non-linear function $f(a, b) = a + \mathrm{MLP}(b)$ to enhance feature extraction:

$$Z_l = f(Z_{l-1}[c] | c \in C, P_l^\top Z_{l-1} / P_l^\top 1). \tag{3}$$

We can obtain hierarchical segmentation by backtracking the clustering assignments (Eqn 2).[3]

In practice, we have two considerations: adjusting the number of coarser segments, and achieving optimal input partitioning. Existing methods (Xu et al., 2022; Ke et al., 2022) use learnable embeddings for clustering, resulting in a fixed segmentation granularity. However, the optimal number of segments varies with image size: larger images require more segments, and smaller ones need fewer. Instead, we conduct feature clustering with an arbitrary number of centroids, where the number of centroids determines the segmentation granularity. Specifically, we use the Farthest Point Sampling (FPS) algorithm (Qi et al., 2017) to select a subset of token features as initial centroids. The FPS algorithm allows for flexible clustering settings and ensures that sampled centroids cover input feature distributions without bias toward dominant features, leading to improved partitioning.

[3] We compare the backtracking strategy for CAST with the naïve segmentation strategy for ViT. For ViT, we apply K-means clustering to the tokens in a hierarchical manner (details in Appx. D.4). Note that ViT fails to segment images accurately (Fig. 5, Fig. 6).

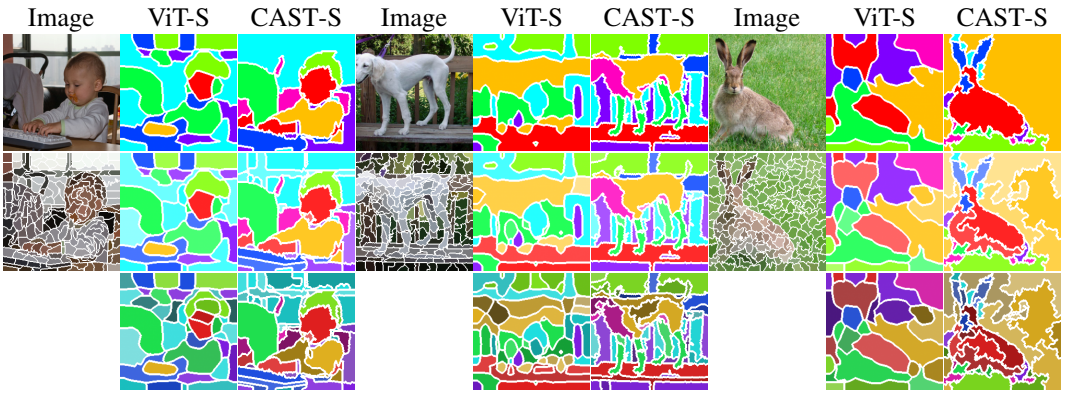


Figure 5: **Hierarchical segmentation.** Segmentation hierarchy of ViT and CAST on ImageNet, where images are divided into 8, 16, and 32 regions from top to bottom, accompanied by corresponding superpixels (which contours). CAST produces consistent and semantic hierarchical segments.

3.3 Architecture and Training

CAST can be integrated into any existing ViT architecture by replacing the patch encoder with our segment encoder and inserting the graph pooling module within the ViT blocks. We use the SEEDS algorithm to extract superpixels and apply the convolutional layer from Xiao et al. (2021) to obtain initial segment features, ensuring a fair comparison with ViT. Following the original ViT configuration, our models are named CAST-(S/B), corresponding to ViT-(S/B). Segmentation granularity is set to $64, 32, 16, 8$ after $3, 3, 3, 2$ encoder blocks, referred to as level $1, 2, 3, 4$ segments, with 196 superpixels as input (level 0 segments). CAST is trained either through supervised (DeiT (Touvron et al., 2021) and Segmenter (Strudel et al., 2021)) or self-supervised learning (MoCo (He et al., 2020)) on ImageNet (Deng et al., 2009) or COCO (Lin et al., 2014) datasets.

4 Experiments

In the experiments section, we demonstrate the following:

- Sec. 4.1. Our CAST produces consistent and semantically aligned *hierarchical segmentation*, which can be utilized for part-to-whole recognition in an unsupervised manner, i.e., without using fine-grained part annotations. This cannot be achieved by state-of-the-art models; for example, Segment Anything (SAM) (Kirillov et al., 2023) fails to capture such semantic hierarchy.
- Sec. 4.2. CAST not only discovers the hierarchy but also enhances flat *semantic segmentation*, indicating that it learns richer dense representations compared to ViT.
- Sec. 4.3. CAST can perform *classification* in addition to segmentation concurrently, matching ViT's accuracy but with greater efficiency. Also, CAST makes *attention* more object-centric.

Outline of the Appendix. We present extra experiments and analyses in the Appendix. **1)** Ablation studies on design choices, including token pooling, centroid initialization, and superpixel methods, are conducted in Appx. A. We compare Graph Pooling with other approaches like ToMe (Bolya et al., 2023). **2)** We demonstrate the adaptiveness of our intermediate hierarchical segmentation to final image recognition using the test-time adaptation (TTA) (Sun et al., 2020) setup. The segmentation quality improves with model adaptation to test images. Detailed setups and results are provided in Appx. B. **3)** Additional methodological details and experiments are described in Appx. D and Appx. E. Lastly, more experimental results are presented in Appx. F. **4)** We summarize the backbone models used in each experiment in Table 10 and Appx. E.

4.1 Hierarchical Segmentation

We demonstrate the superiority of CAST for hierarchical segmentation on the ImageNet (Deng et al., 2009), PartImageNet (He et al., 2022), and DensePose (Alp Güler et al., 2018) datasets.

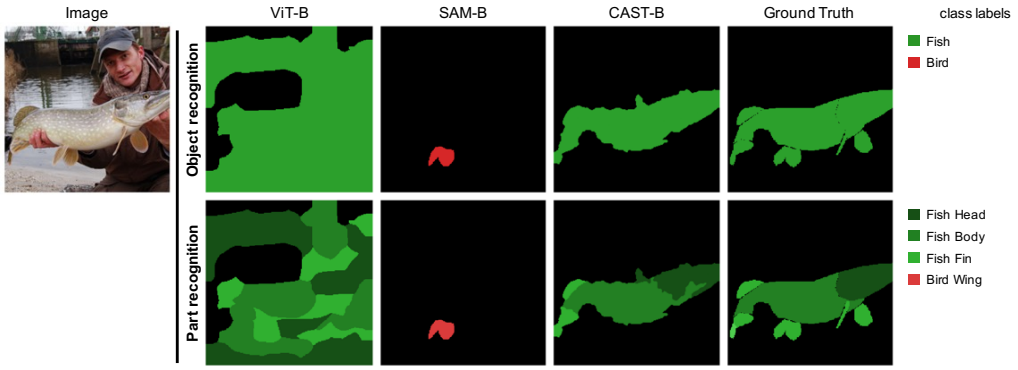


Figure 6: **Part-to-whole recognition (visualization).** Hierarchical segmentation and corresponding part-to-whole recognition results of ViT, SAM, and CAST on PartImageNet. Our CAST effectively captures semantic parts, such as *head*, *body* and *fin* of *fish*, and associates them with their respective regions. In contrast, ViT does not generate fine segmentations that align with object boundaries accurately. SAM does not detect the fish and its part at all. Our CAST can not only generate high-quality fine-grained segments, but also form a part-to-whole hierarchy of the object.

Table 1: **Part-to-whole recognition (evaluation).** mIoU and boundary F-score for ViT, SAM, and CAST on PartImageNet, using category, object, and part-level annotations. CAST outperforms the baselines in most cases, except for SAM, which excels in the boundary F-score for part labels. Note that SAM is effective at drawing clear boundaries but struggles with segmenting semantic regions, as indicated by its F-score and mIoU, respectively. In contrast, ViT roughly captures the semantic regions but falls short in identifying the exact boundaries. CAST performs well in both tasks. Furthermore, it is worth mentioning that SAM requires multiple inferences to parse a scene, resulting in much higher GFLOPS costs compared to the other models.

				← Finer / Coarser →					
				Part		Object		Category	
Model	Training data	Supervised	GFLOPS	mIoU	F-score	mIoU	F-score	mIoU	F-score
ViT-B	IN-1K	-	17.8	11.74	4.64	25.34	10.92	36.68	13.28
SAM-B	SA-1B	✓	488.2	10.15	**7.25**	18.03	20.71	31.36	32.01
CAST-B	IN-1K	-	**12.9**	**13.20**	6.52	**29.66**	**22.32**	**50.75**	**34.38**

ImageNet. We compare the hierarchical segmentations generated by CAST and ViT. Both models are self-supervisedly trained on ImageNet. Fig. 5 displays that CAST captures a better semantic hierarchy of segments than ViT, automatically discovering the part-to-whole structure.

PartImageNet. To validate whether hierarchical segmentation captures part-to-whole structure, we employ PartImageNet, which contains 2 categories (Animals vs. Things), 11 objects (e.g., Biped), and 40 parts (e.g., Biped Head) annotations from a subset of ImageNet. We use an open-vocabulary segment classifier, OVSeg (Liang et al., 2023), to predict the class of each segment. The classifier is applied to two granularities (8 and 16 segments), referred to as coarse and fine. We use the coarse segments for category and object, and the fine segments for part recognition.

We compare CAST with ViT and SAM. Here, we enforce semantic consistency over the hierarchy, where fine segments inherit the part class of their corresponding coarse segments. CAST and ViT utilize the hierarchy obtained from the model, as fine segments cluster to construct coarse segments in the upper layers. However, SAM only provides diverse granularities of segments in a flat manner, lacking hierarchical relationships. Thus, instead of iterating over the tree structure, SAM assigns labels by sequentially moving segments from the largest to smallest ones. Consequently, SAM exhibits worse part-to-whole recognition performance than CAST. The detailed procedure for each model is explained in Appendix D.2. Additionally, note that SAM is supervisedly trained on SA-1B, while CAST and ViT are self-supervisedly trained on IN-1K, utilizing 1000 times more images, even with ground truth segment annotations. Such an explicit hierarchy cannot be achieved through such enormous supervision, whereas CAST can learn in a self-supervised manner.

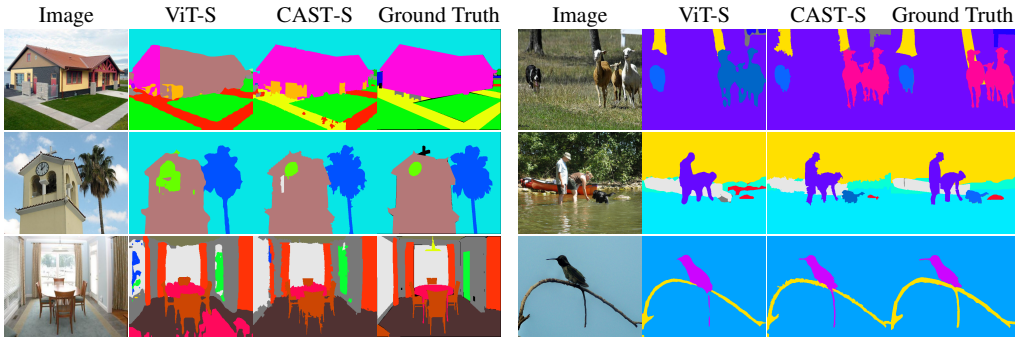


Figure 7: **Semantic segmentation (visualization).** Semantic segmentation results of ViT and CAST on ADE20K (left) and PASCAL Context (right). CAST captures the details more effectively.

Table 3: **Semantic segmentation (evaluation).** mIoU and boundary F-score for ViT and CAST on (a) Pascal VOC, (b) ADE20K and (c) Pascal Context. For (a), we report values before and after fine-tuning, and for (b) and (c), values after fine-tuning. CAST consistently outperforms the baselines in all scenarios. We also compare models using square patch or segment tokens, and using graph pooling or not. Note that only our final CAST can generate segments without post-processing like K-Means. The results indicate that using segment tokens and hierarchical pooling both contribute to performance. This suggests that the improvement from CAST does not solely come from superpixels but rather from the learned hierarchical structures that benefit scene understanding.

(a) Pascal VOC

			before		after	
Model	Token	Pooling	mIoU	F-score	mIoU	F-score
ViT-S	Patch	✗	30.9	16.1	65.8	40.7
	Patch	✓	34.5	19.8	67.2	41.9
	Segment	✗	32.2	21.2	66.5	46.7
CAST-S	Segment	✓	**38.4**	**27.0**	**67.6**	**48.1**

(b) ADE20K

Model	mIoU	F-score
ViT-S	41.7	33.9
CAST-S	**43.1**	**36.5**

(c) PASCAL Context

Model	mIoU	F-score
ViT-S	48.3	42.0
CAST-S	**49.1**	**44.1**

As shown in Fig. 6, CAST produces segments that align better with semantics and enable OVSeg to classify more accurately. Unlike ViT, CAST captures better visual structures such as boundaries. Unlike SAM, CAST exhibits better hierarchical consistency. SAM segments an image at different granularities independently, and pixels belonging to the same fine segment are not guaranteed to be in the same coarse region. Also, Table 1 shows that CAST consistently outperforms ViT and SAM across the semantic hierarchy (+14%, +4.3%, and +1.5% for category, object, and part).

Lastly, CAST is computationally more efficient than SAM. To segment an image, SAM requires **1)** additional mask decoder to predict pixel-segment assignments in the high-dimensional feature space, and **2)** multiple inferences guided by different input points. In contrast, CAST efficiently produces segments from low-dimensional color features through a single inference.

DensePose. We report human part segmentation using DensePose, which contains 4 objects and 14 parts annotations from a subset of COCO. DensePose contains more complex scenes and diverse instances than PartImageNet. We partition each image into 64, 32, and 16 regions and measure the F-score for overlapping regions.

Table 2: **Human part segmentation.** Region F-score on DensePose.

	# region		
Model	64	32	16
HSG (RN50)	17.6	24.8	26.9
CAST-S	17.2	22.8	27.3

We compare CAST with HSG (Ke et al., 2022), the state-of-the-art unsupervised hierarchical segmenter. Both models are self-supervisedly trained on COCO. Table 2 shows that CAST achieves comparable performance to HSG, which is specifically designed for segmentation. In contrast, CAST performs segmentation and recognition concurrently.

Table 4: **Classification.** Linear probing accuracy of MoCo-trained models on ImageNet. CAST outperforms ViT and Swin with a similar model size while reducing GFLOPS by 30%.

Model	GFLOPS	IN-100	IN-1K
ViT-S	4.7	78.1	67.9
Swin-T	4.5	78.3	63.0
CAST-S	**3.4**	**79.9**	**68.1**

Table 5: **Attention.** mIoU of ImageNet-trained models on VOC, measuring similarity between attention and true segmentation on VOC. Visual examples: ViT (left) vs. CAST (right). CAST offers more object-centric attention.

Model	mIoU
ViT-S	45.8
CAST-S	**48.0**

4.2 Semantic Segmentation

We demonstrate the superiority of CAST for semantic segmentation on the Pascal VOC (Everingham et al., 2010), ADE20K (Zhou et al., 2019), and Pascal Context (Mottaghi et al., 2014) datasets. **Pascal VOC.** We compare CAST and ViT, both models self-supervisedly trained on COCO. We test the segmentation of the models both before (unsupervised) and after (supervised) fine-tuning. We follow the setup of SegSort (Hwang et al., 2019) on segment retrieval for unsupervised evaluation, and fine-tune the model alongside a pixel-wise softmax classifier for supervised evaluation.

As shown in Table 3 (a), CAST significantly improves performance in all scenarios, both in terms of region (mIoU) and boundary (F-score) metrics. For instance, CAST outperforms ViT by +7.5% for mIoU and +10.9% for F-score before fine-tuning. We also compare models that use either segment tokens or graph pooling, while CAST uses both, showing that its improvement does not solely rely on superpixels but also on the hierarchical structure. Intuitively, superpixels enhance segmentation by providing better boundaries, while hierarchical pooling enables the model to grasp the structure of entire objects; thus, both are essential. Additionally, note that CAST is the only model capable of predicting segments as is, unlike others that require post-processing like K-Means.

ADE20K and Pascal Context. We use CAST and ViT supervisely trained on ImageNet, and fine-tune them with Segmenter (Strudel et al., 2021) on each dataset. As shown in Table 3 (b) and (c), CAST consistently improves the region and boundary metrics for segmentation.

4.3 Classification and Object-centric Attention

Classification. CAST performs concurrent hierarchical segmentation and image recognition. We compare the classification accuracy on ImageNet, among CAST-S, ViT-S, and Swin-T (Liu et al., 2021). All models are self-supervisedly trained on ImageNet and evaluated using linear probing. Table 4 shows that CAST is on par with those architectures designed for recognition, lacking the ability of segmentation. Additionally, CAST reduces GFLOPS through hierarchical pooling.

Object-centric attention. CAST excels in capturing objectness through self-supervised learning compared to ViT. We compare models self-supervisedly trained on ImageNet and evaluate them on Pascal VOC. We follow Joulin et al. (2010) to generate figure-ground segmentation from multi-head attention and measure mIoU between true segmentations. As shown in Table 5, the attention maps from CAST is more object-centric than ViT, improving mIoU by +2.2%.

5 Conclusion

We introduce a novel learning framework that integrates the segmentation and recognition process, where segmentation serves as the foundation *for* recognition, learned *by* image recognition objectives. The proposed method, CAST, exhibits several advantages over previous methods such as SAM, HSG, and ViT, including hierarchical segmentation, semantic segmentation, and concurrent recognition. We believe our framework opens new avenues for future research, with the aim of creating true *vision* transformer, departing from the text-inspired designs of previous ViT models.

Limitation. CAST relies on off-the-shelf superpixel algorithms and employs a fixed set of segmentation granularities without adapting to different downstream tasks or images. It limits the performance on complex scenes such as Cityscapes. See Appx. A.3 for further discussions.

ACKNOWLEDGEMENTS

This work was supported, in part, by US NSF 2131111, NSF 2215542, NSF 2313151, and Bosch gift funds to S. Yu at UC Berkeley and at the University of Michigan. We thank Jyh-Jing Hwang for many earlier insightful discussions that have led to this work.

ETHICS STATEMENT

Our CAST provides a fine-grained understanding of part-to-whole structures, unlocking new possibilities while also carrying the potential for misuse. For instance, detailed structures like human poses could be used to enhance the quality of DeepFake videos (Masood et al., 2023). However, it's crucial to emphasize that technology itself is not the issue; the responsibility falls on those who wield it. We aspire for our CAST to bring about more benefits than these potential risks.

REPRODUCIBILITY STATEMENT

We provide method and experimental details in Appx. D and Appx. E, and our code on GitHub.

REFERENCES

Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, Sabine Süsstrunk, et al. Slic superpixels compared to state-of-the-art superpixel methods. *PAMI*, 2012.

Jiwoon Ahn and Suha Kwak. Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4981–4990, 2018.

Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. Densepose: Dense human pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7297–7306, 2018.

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

Arindam Banerjee, Inderjit S Dhillon, Joydeep Ghosh, and Suvrit Sra. Clustering on the unit hypersphere using von mises-fisher distributions. *Journal of Machine Learning Research*, 6(Sep): 1345–1382, 2005.

Michael Van den Bergh, Xavier Boix, Gemma Roig, Benjamin de Capitani, and Luc Van Gool. Seeds: Superpixels extracted via energy-driven sampling. In *European conference on computer vision*, pp. 13–26. Springer, 2012.

Irving Biederman. Recognition-by-components: a theory of human image understanding. *Psychological review*, 94(2):115, 1987.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2022.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. In *International Conference on Learning Representations*, 2023.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9650–9660, 2021.

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. *arXiv preprint arXiv:1606.00915*, 2016.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9640–9649, 2021.

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

Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 2002.

Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 702–703, 2020.

Daan de Geus, Panagiotis Meletis, Chenyang Lu, Xiaoxiao Wen, and Gijs Dubbelman. Part-aware panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5485–5494, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Zheng Ding, Jieke Wang, and Zhuowen Tu. Open-vocabulary panoptic segmentation maskclip. 2022.

Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baining Guo. Cswin transformer: A general vision transformer backbone with cross-shaped windows. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12124–12134, 2022.

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. *arXiv preprint arXiv:2010.11929*, 2020.

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 2004.

Brian Fulkerson, Andrea Vedaldi, and Stefano Soatto. Class segmentation and object localization with superpixel neighborhoods. In *2009 IEEE 12th international conference on computer vision*, pp. 670–677. IEEE, 2009.

Raghudeep Gadde, Varun Jampani, Martin Kiefel, Daniel Kappler, and Peter V Gehler. Superpixel convolutional networks using bilateral inceptions. In *ECCV*, 2016.

Stephen Gould, Jim Rodgers, David Cohen, Gal Elidan, and Daphne Koller. Multi-class segmentation with relative location prior. *International journal of computer vision*, 80(3):300–316, 2008.

Saurabh Goyal, Anamitra Roy Choudhury, Saurabh Raje, Venkatesan Chakaravarthy, Yogish Sabharwal, and Ashish Verma. Power-bert: Accelerating bert inference via progressive word-vector elimination. In *International Conference on Machine Learning*, pp. 3690–3699. PMLR, 2020.

Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on vision transformer. *IEEE transactions on pattern analysis and machine intelligence*, 45(1):87–110, 2022.

Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *2011 International Conference on Computer Vision*, pp. 991–998. IEEE, 2011.

Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 447–456, 2015.

Ju He, Shuo Yang, Shaokang Yang, Adam Kortylewski, Xiaoding Yuan, Jie-Neng Chen, Shuai Liu, Cheng Yang, Qihang Yu, and Alan Yuille. Partimagenet: A large, high-quality dataset of parts. In *European Conference on Computer Vision*, pp. 128–145, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11936–11945, 2021.

Jyh-Jing Hwang and Tyng-Luh Liu. Pixel-wise deep learning for contour detection. *arXiv preprint arXiv:1504.01989*, 2015.

Jyh-Jing Hwang, Stella X Yu, Jianbo Shi, Maxwell D Collins, Tien-Ju Yang, Xiao Zhang, and Liang-Chieh Chen. Segsort: Segmentation by discriminative sorting of segments. In *ICCV*, 2019.

Xu Ji, Joao F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9865–9874, 2019.

Armand Joulin, Francis Bach, and Jean Ponce. Discriminative clustering for image co-segmentation. In *CVPR*, pp. 1943–1950. IEEE, 2010.

Hyunwoo Kang, Sangwoo Mo, and Jinwoo Shin. Oamixer: Object-aware mixing layer for vision transformers. *CVPR Transformers for Vision Workshop*, 2022.

Tsung-Wei Ke, Jyh-Jing Hwang, Yunhui Guo, Xudong Wang, and Stella X Yu. Unsupervised hierarchical semantic segmentation with multiview cosegmentation and clustering transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2571–2581, 2022.

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.

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

Liulei Li, Tianfei Zhou, Wenguan Wang, Jianwu Li, and Yi Yang. Deep hierarchical semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1246–1257, 2022a.

Xiangtai Li, Shilin Xu, Yibo Yang, Guangliang Cheng, Yunhai Tong, and Dacheng Tao. Panoptic-partformer: Learning a unified model for panoptic part segmentation. In *European Conference on Computer Vision*, pp. 729–747. Springer, 2022b.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7061–7070, 2023.

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 *European conference on computer vision*, pp. 740–755. Springer, 2014.

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

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *Advances in Neural Information Processing Systems*, 33:11525–11538, 2020.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.

Xu Ma, Yuqian Zhou, Huan Wang, Can Qin, Bin Sun, Chang Liu, and Yun Fu. Image as set of points. In *International Conference on Learning Representations*, 2023.

Michael Maire. Simultaneous segmentation and figure/ground organization using angular embedding. In *European Conference on Computer Vision*, pp. 450–464. Springer, 2010.

Michael Maire, Stella X. Yu, and Pietro Perona. Object detection and segmentation from joint embedding of parts and pixels. In *2011 International Conference on Computer Vision*, pp. 2142–2149. IEEE, 2011.

Jitendra Malik, Serge Belongie, Thomas Leung, and Jianbo Shi. Contour and texture analysis for image segmentation. *IJCV*, 2001.

Dmitrii Marin, Jen-Hao Rick Chang, Anurag Ranjan, Anish Prabhu, Mohammad Rastegari, and Oncel Tuzel. Token pooling in vision transformers. *arXiv preprint arXiv:2110.03860*, 2021.

Momina Masood, Mariam Nawaz, Khalid Mahmood Malik, Ali Javed, Aun Irtaza, and Hafiz Malik. Deepfakes generation and detection: State-of-the-art, open challenges, countermeasures, and way forward. *Applied intelligence*, 53(4):3974–4026, 2023.

Daphne Maurer, Richard Le Grand, and Catherine J Mondloch. The many faces of configural processing. *Trends in cognitive sciences*, 6(6):255–260, 2002.

Sangwoo Mo, Minsu Cho, and Jinwoo Shin. Instagan: Instance-aware image-to-image translation. *International Conference on Learning Representations*, 2019.

Sangwoo Mo, Hyunwoo Kang, Kihyuk Sohn, Chun-Liang Li, and Jinwoo Shin. Object-aware contrastive learning for debiased scene representation. *Advances in Neural Information Processing Systems*, 34:12251–12264, 2021.

Greg Mori, Xiaofeng Ren, Alexei A Efros, and Jitendra Malik. Recovering human body configurations: Combining segmentation and recognition. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pp. II–II. IEEE, 2004.

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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 891–898, 2014.

Yassine Ouali, Celine Hudelot, and Myriam Tami. Autoregressive unsupervised image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020.

Tai-Yu Pan, Qing Liu, Wei-Lun Chao, and Brian Price. Towards open-world segmentation of parts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15392–15401, 2023.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

Lu Qi, Jason Kuen, Weidong Guo, Jiuxiang Gu, Zhe Lin, Bo Du, Yu Xu, and Ming-Hsuan Yang. Aims: All-inclusive multi-level segmentation for anything. *Advances in Neural Information Processing Systems*, 36, 2024.

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 *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34, 2021.

Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *Computer Vision, IEEE International Conference on*, volume 2, pp. 10–10. IEEE Computer Society, 2003.

Zhixiang Ren, Shenghua Gao, Liang-Tien Chia, and Ivor Wai-Hung Tsang. Region-based saliency detection and its application in object recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(5):769–779, 2013.

Saquib Sarfraz, Vivek Sharma, and Rainer Stiefelhagen. Efficient parameter-free clustering using first neighbor relations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8934–8943, 2019.

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 *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

Ramprasaath R Selvaraju, Karan Desai, Justin Johnson, and Nikhil Naik. Casting your model: Learning to localize improves self-supervised representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11058–11067, 2021.

Abhishek Sharma, Oncel Tuzel, and Ming-Yu Liu. Recursive context propagation network for semantic scene labeling. *Advances in Neural Information Processing Systems*, 27, 2014.

Eitan Sharon, Meirav Galun, Dahlia Sharon, Ronen Basri, and Achi Brandt. Hierarchy and adaptivity in segmenting visual scenes. *Nature*, 442(7104):810–813, 2006.

Baifeng Shi, Trevor Darrell, and Xin Wang. Top-down visual attention from analysis by synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2102–2112, 2023.

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

Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 7262–7272, 2021.

Peize Sun, Shoufa Chen, Chenchen Zhu, Fanyi Xiao, Ping Luo, Saining Xie, and Zhicheng Yan. Going denser with open-vocabulary part segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15453–15465, 2023.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pp. 9229–9248. PMLR, 2020.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

James W Tanaka and Martha J Farah. Parts and wholes in face recognition. *The Quarterly Journal of Experimental Psychology Section A*, 46(2):225–245, 1993.

James W Tanaka and Diana Simonyi. The "parts and wholes" of face recognition: A review of the literature. *Quarterly Journal of Experimental Psychology*, 69(10):1876–1889, 2016.

James W Tanaka, Bonnie Heptonstall, and Alison Campbell. Part and whole face representations in immediate and long-term memory. *Vision Research*, 164:53–61, 2019.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *European conference on computer vision*, pp. 776–794. Springer, 2020.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.

Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, and Luc Van Gool. Revisiting contrastive methods for unsupervised learning of visual representations. *arXiv preprint arXiv:2106.05967*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. *International Conference on Learning Representations*, 2021a.

Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3024–3033, 2021b.

Xudong Wang, Ziwei Liu, and Stella X Yu. Unsupervised feature learning by cross-level instance-group discrimination. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12586–12595, 2021c.

Xudong Wang, Shufan Li, Konstantinos Kallidromitis, Yusuke Kato, Kazuki Kozuka, and Trevor Darrell. Hierarchical open-vocabulary universal image segmentation. *Advances in Neural Information Processing Systems*, 36, 2024.

Meng Wei, Xiaoyu Yue, Wenwei Zhang, Shu Kong, Xihui Liu, and Jiangmiao Pang. Ov-parts: Towards open-vocabulary part segmentation. *Advances in Neural Information Processing Systems*, 36, 2024.

Xing Wei, Qingxiong Yang, Yihong Gong, Narendra Ahuja, and Ming-Hsuan Yang. Superpixel hierarchy. *IEEE Transactions on Image Processing*, 27(10):4838–4849, 2018.

Zhirong Wu, Yuanjun Xiong, Stella Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance-level discrimination. *arXiv preprint arXiv:1805.01978*, 2018.

Tete Xiao, Piotr Dollar, Mannat Singh, Eric Mintun, Trevor Darrell, and Ross Girshick. Early convolutions help transformers see better. *Advances in Neural Information Processing Systems*, 34, 2021.

Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *ICCV*, 2015.

Jiarui Xu, Shalini De Mello, Sifei Liu, Wonmin Byeon, Thomas Breuel, Jan Kautz, and Xiaolong Wang. Groupvit: Semantic segmentation emerges from text supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18134–18144, 2022.

Stella X. Yu and Jianbo Shi. Multiclass spectral clustering. In *ICCV*, 2003a.

Stella X. Yu and Jianbo Shi. Object-specific figure-ground segregation. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pp. II–39. IEEE, 2003b.

Stella X Yu and Jianbo Shi. Segmentation given partial grouping constraints. *PAMI*, 2004.

Stella X. Yu, Ralph Gross, and Jianbo Shi. Concurrent object recognition and segmentation by graph partitioning. *Advances in neural information processing systems*, 15, 2002.

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 *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Wang Zeng, Sheng Jin, Wentao Liu, Chen Qian, Ping Luo, Wanli Ouyang, and Xiaogang Wang. Not all tokens are equal: Human-centric visual analysis via token clustering transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11101–11111, 2022.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Xiao Zhang and Michael Maire. Self-supervised visual representation learning from hierarchical grouping. *Advances in Neural Information Processing Systems*, 33, 2020.

Yifan Zhang, Bo Pang, and Cewu Lu. Semantic segmentation by early region proxy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1258–1268, 2022.

Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127:302–321, 2019.

APPENDIX

A Ablation Studies and Analyses **18**
A.1 Ablation Study on Design Choices 18
A.2 Ablation Study on Superpixel Methods 19
A.3 Limitations of Superpixels and Failure Cases 20
A.4 Ablation Study on Inference Latency 20

B Extension to Test-time Adaptation **21**
B.1 Implementation Details 21
B.2 Experimental Results 21

C Additional Related Work **23**

D Additional Method Details **24**
D.1 Algorithm 24
D.2 Open-vocabulary Segmentation 24
D.3 Comparison with Hierarchical Segment Grouping 25
D.4 Generating Segments from ViT 25
D.5 Token Pooling Methods 25

E Additional Experimental Details **26**
E.1 Summary of models used in experiments 26
E.2 Datasets 26
E.3 Image Resolution and Number of Tokens 26
E.4 Hyper-parameters for Training 26
E.5 Inference and Evaluation on ImageNet and VOC 28

F Additional Experimental Results **30**
F.1 Visual Results on Hierarchical Segmentation 30
F.2 Visual Results on Part-to-whole Recognition 31
F.3 Visual Results on Semantic Segmentation 32
F.4 Visual Results on Figure-ground Segmentation 33
F.5 Visual Results on Multi-head Attention Maps 34

A Ablation Studies and Analyses

A.1 Ablation Study on Design Choices

Table 6 presents ablation studies on design choices in our framework. **(a)** We explore various token pooling methods, with our GraphPool module excelling. Notably, FINCH (Sarfraz et al., 2019) lags in performance, showcasing the challenge of adaptive pooling. **(b)** We examine different centroid initialization methods, with Farthest Point Sampling (FPS) significantly outperforming others. FPS not only samples informative tokens but also enhances discriminative token selection, preserving fine-grained visual information. **(c)** We investigate optimal segment granularities, fixed for both training and inference, to achieve a balance between model efficiency and task performance. **(d)** We showcase that our model can adapt to varying segment granularities during inference.

Table 6: **Ablation study.** We compare the design choices for (a) token pooling, (b) cluster centroid initialization, (c) token granularities applied to both training and testing, and (d) token granularities of a trained model varied during testing. We report the linear probing accuracy of MoCo-trained CAST-S on IN-100. † indicates that the methods have been re-implemented.

(a) Pooling	Acc.
Graph Pooling	**79.9**
Random Sampling	55.8
K-Means	73.9
K-Medoids	72.3
FINCH† (Sarfraz et al., 2019)	63.3
Token Pooling† (Marin et al., 2021)	75.8
CTM† (Zeng et al., 2022)	72.2
ToMe† (Bolya et al., 2023)	78.1

(b) Centroids initialization	Acc.
Farthest Point Sampling	**79.9**
Random Sampling	71.2
PoWER-BERT (Goyal et al., 2020)	71.6

(c) Token granularity (Train=Test)	GFLOPS	Acc.
$196, 32, 16, 8$	3.0	78.8
$196, 64, 32, 16$	3.4	**79.9**
$196, 128, 64, 32$	4.3	**79.9**
$324, 64, 32, 16$	4.6	79.8
$324, 128, 64, 32$	5.4	80.4

(d) Token granularity (Train≠Test)	GFLOPS	Acc.
Train: $196, 64, 32, 16$	3.4	**79.9**
Test: $196, 32, 16, 8$	3.0	79.2
Test: $196, 128, 64, 32$	4.3	79.5
Test: $324, 64, 32, 16$	4.6	78.8
Test: $324, 128, 64, 32$	5.4	79.3

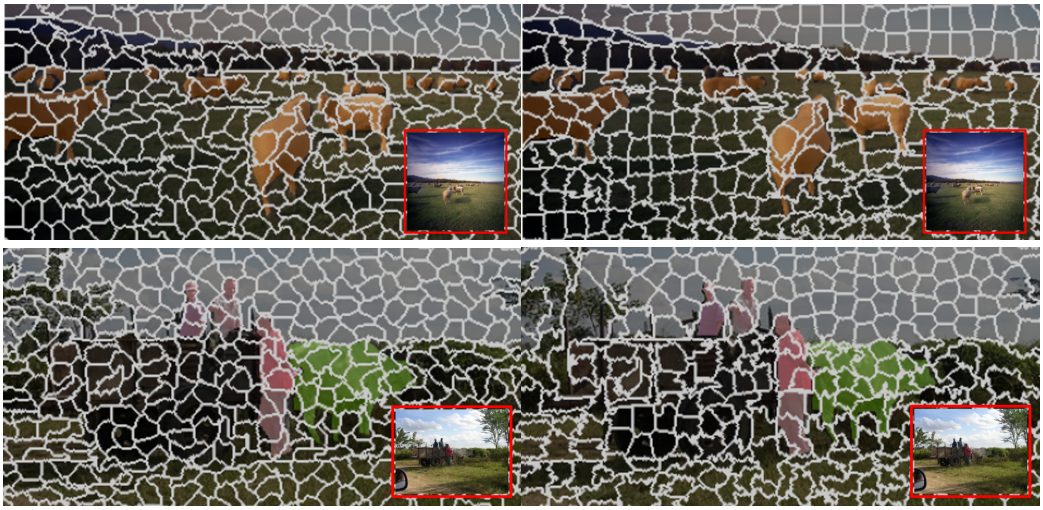
A.2 Ablation Study on Superpixel Methods

We test the robustness of CAST using superpixels generated by different algorithms. In particular, we compare SEEDS (Bergh et al., 2012) against SLIC (Achanta et al., 2012) on classification and semantic segmentation. We train CAST with self-supervised learning on IN-1K and COCO for classification and segmentation. As shown in Table 7, we report the linear probing accuracy on ImageNet-1K and mIoU before (left) and after (right) fine-tuning on VOC. Our method is robust to different choices of superpixel algorithms, while, SEEDS achieves better performance than SLIC.

Table 7: SEEDS outperforms SLIC on classification and semantic segmentation. We report top-1 accuracy and mIoU for classification on IN-1K and segmentation before / after fine-tuning on VOC. Our CAST achieves better performance using superpixels generated by SEEDS.

	Classification	Segmentation
SEEDS	68.1	38.4/67.6
SLIC	65.6	37.7/65.7

Figure 8 illustrates the superpixels obtained using SEEDS and SLIC. SEEDS superpixels exhibit superior alignment with object boundaries, particularly for small objects, compared to SLIC. This improvement stems from SEEDS being a boundary refinement algorithm, iteratively updating superpixels to better capture edges. In contrast, SLIC relies on K-Means clustering based on colors and connected-component constraints, which may be less effective in capturing the boundaries. Consequently, we can conclude that SEEDS superpixels, with their superior shape information, produce more compelling hierarchical segmentation for CAST.



SEEDS SLIC

Figure 8: From left to right, we show the superpixels (white contours) generated by SEEDS and SLIC algorithm, overlaid on the the original images along with the ground-truth semantic segmentation (colored regions). Superpixels generated by SEEDS are more precisely aligned with object boundaries, especially for small objects, than those generated by SLIC. Learning to generate superpixels along with feature learning would enable more precise superpixel partitioning and improve segmentation.

A.3 LIMITATIONS OF SUPERPIXELS AND FAILURE CASES

Our method underforms ViT baselines on Cityscapes dataset. The results show the limitation of using off-the-shelf superpixel algorithms. ViT-S achieves 74.6%, whereas, CAST-S and CAST-SD achieve 72.1% and 74.2% on the benchmark. As shown in Fig. 9, we observe that existing superpixel methods cannot pick out thin structures such as light poles and steel pipes from the scene. One possible reason is that such methods are only developed and tested on object-centric dataset (BSDS). Replacing these superpixel algorithms with learnable approaches could help to address such an issue.

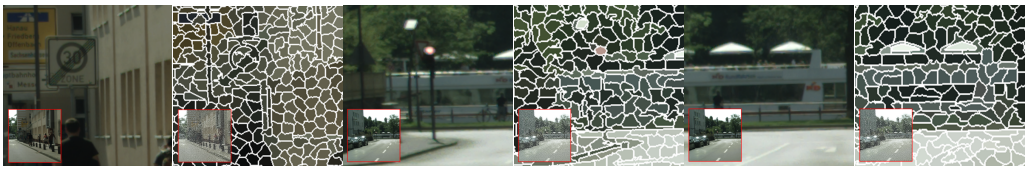


Figure 9: From left to right, we show the image and corresponding superpixels (white contours). Street scenes have many thin structures such as light poles and steel pipes, which superpixel algorithms often fail to pick out. Learning to generate superpixels could help solving such issues.

A.4 ABLATION STUDY ON INFERENCE LATENCY

The computational cost of superpixel generation and graph pooling can be reduced by the decreased number of tokens required for computing self-attention. This advantage becomes more pronounced when using larger models, where self-attention blocks dominate the entire cost.

To validate this, we analyze the latency of model inference and superpixel generation. Our system comprises a 32GB Nvidia Titan V GPU card and two Intel(R) Xeon(R) CPU E5-2630 v4 processors, totaling 20 CPU cores. We utilize the PyTorch machine learning framework with 24 workers, a batch size of 64, and an image resolution set to 224x224.

In our system, CAST-B achieves a lower average inference latency of 217 ms compared to ViT-B with 273 ms. SEEDS takes 73 ms to generate superpixels from the batch of images. However, we remark that the current SEEDS implementation is not fully optimized. Employing GPU implementation or parallelizing the process with more CPU cores can alleviate the bottleneck in superpixel generation. Furthermore, the cost of superpixel generation becomes less significant with larger models, which are commonly used in practice.

In addition, we demonstrate that the cost of self-attention and graph pooling decreases as the number of tokens is reduced. Table 8 presents the computational cost of our graph pooling across layers, showing a significant reduction in cost as the number of tokens per layer decreases.

Table 8: FPS in our graph pool module requires additional computation. We report the inference time (ms) of each module with 384 channel dimensions and 256 batch sizes on IN-100. Optimizing the token sampling technique to increase model efficiency is our future work.

#. of Tokens	Encoder Blocks	GraphPool (FPS)
196	86.43	63.02 (37.64)
64	25.4	18.2 (9.7)
32	12.9	9.6 (3.0)
16	5	6.1 (1.5)

B Extension to Test-time Adaptation

CAST only considers the top-down pathway during training, guided by the recognition objectives. This knowledge is encoded in the model and reflects the bottom-up pathway during inference. While this enables the model to learn the general top-down elements, the segmentations will not change based on what the top layers predict at inference time.

To extend CAST by incorporating the top-down pathway during inference, we employ test-time adaptation (TTA) (Sun et al., 2020), specifically TENT (Wang et al., 2021a), with the classifier trained on top of the CAST backbones. We apply TENT to each sample to adapt the model and maximize its prediction confidence. As a result, CAST refines its object segments to align with its initial belief: If this image depicts the predicted class, which parts contribute to this prediction?

B.1 Implementation Details

We use the CAST model pretrained on ImageNet with the MoCo objective as our base model. We trained a dog vs. non-dog classifier on PartImageNet training data and evaluate it on the validation split. The classifier is defined as the average of normalized embeddings of training data that belong to the class. In other words, we define a class vector as the center of training data for each class. To infer the class, we compute the cosine similarity between the test embedding and class vectors and apply a temperature of 0.07 to the softmax classifier, using these cosine similarities as logits.

For each sample, we calculate the prediction entropy loss and update the model using a single SGD with lr=1.0. We only update the normalization layers while keeping other parameters frozen, following the practice of TTA. Note that no label or batch information is used in this process, and the model is updated solely based on the inference of a single instance.

We evaluate the evolution of segmentation before and after TTA. To this end, we compute the object segmentation mask by assigning a label to each segment. We define this segment label as the majority of the pixel labels, using the ground-truth segmentation masks. We chose this strategy for simplicity, but one can also predict the segment labels using OVSeg, as we discussed previously.

B.2 Experimental Results

Figure 10 showcases the visual examples that TTA improves the segmentation of CAST by better capturing object shapes and less focusing on unnecessary details. The improvement is more substantial for hard samples that the CAST originally fails. To check this, we measure the mIoU of CAST before and after TTA on the hard samples that the mIoU of the original CAST is low.

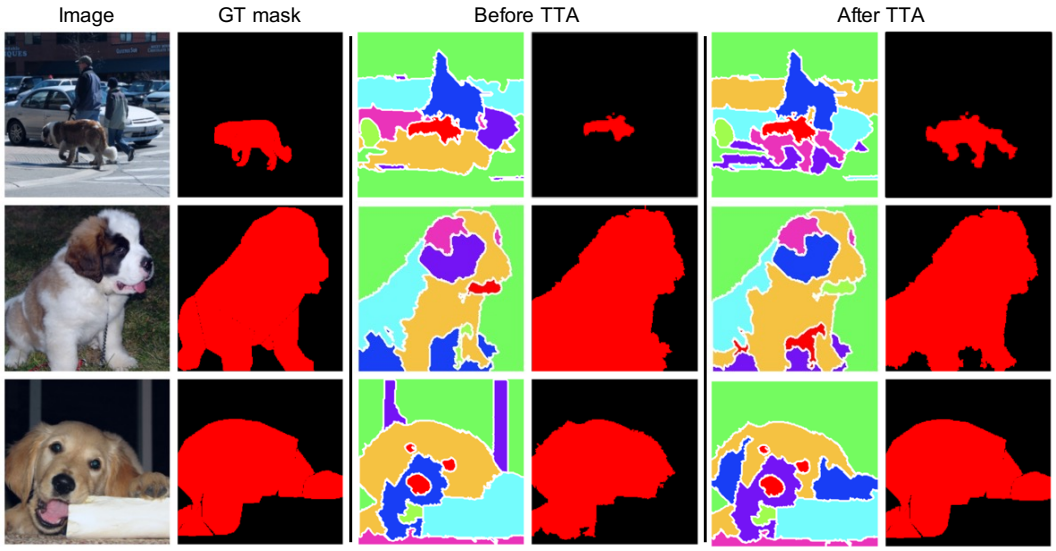


Figure 10: Visual examples of learned clusters (left) and predicted segmentation for dogs (right) using CAST models before and after applying test-time-adaptation (TTA). TTA improves the segmentation of CAST by capturing object shapes better (e.g., not missing legs in rows 1-3) while reducing attention to unnecessary details (e.g., window frames in row 3).

Table 9: We report the mIoU of CAST segmentations both before and after applying TTA. We assess the performance on hard samples where the original CAST fails, based on the mIoU threshold specified in the table. TTA gives a significant improvement in these challenging cases.

	mIoU from the original CAST		
	$< 70\%$	$< 80\%$	$< 90\%$
# of samples	8	22	125
Before TTA	56.9	69.5	83.5
After TTA	71.2	76.6	84.9

C Additional Related Work

Superpixels. Superpixels are sets of locally connected pixels that encapsulate coherent structures, such as colors (Ren & Malik, 2003). They have found applications in various densely labeling tasks, including part parsing (Mori et al., 2004), saliency detection (Ren et al., 2013), and image segmentation (Gould et al., 2008; Fulkerson et al., 2009; Sharma et al., 2014; Gadde et al., 2016; Wei et al., 2018). More recently, Zhang et al. (2022) addressed semantic segmentation by replacing patches with superpixel tokens in ViT architectures. Our model takes a step further by constructing a segment hierarchy and simultaneously performing both segmentation and classification.

Image segmentation and clustering. Image segmentation partitions an image into coherent regions. Classic methods consist of two steps: extracting local features and clustering based on criteria like mode-finding (Comaniciu & Meer, 2002; Banerjee et al., 2005) or graph partitioning (Felzenszwalb & Huttenlocher, 2004; Shi & Malik, 2000; Malik et al., 2001; Yu & Shi, 2003a; 2004). They often output hierarchical segmentation for human perception comparison (Arbelaez et al., 2010). To prevent object boundary ambiguities, common approaches rely on contour detection (Hwang & Liu, 2015; Xie & Tu, 2015), iteratively removing contours for multi-scale segmentations (Arbelaez et al., 2010). In contrast, our work operates directly on segments, without contour proxies.

Self-supervised segmentation and representation learning. Recent works can be categorized into three camps: **1)** Leveraging self-supervised image recognition, models are transferred to segmentation by increasing location sensitivity (Wu et al., 2018; He et al., 2020; Chen et al., 2020; Wang et al., 2021c), implementing contrastive loss across views (Wang et al., 2021b), or applying stronger augmentation and constrained cropping (Selvaraju et al., 2021; Mo et al., 2021). **2)** Learning a pixel-wise cluster predictor maximizes mutual information between cluster predictions on augmented views of the same instance at corresponding pixels (Ji et al., 2019; Ouali et al., 2020). **3)** Learning a pixel-level feature encoder maximizes discrimination between pixels based on contour-induced segments (Hwang et al., 2019), pre-computed region hierarchies (Zhang & Maire, 2020), or feature-induced hierarchical groupings (Ke et al., 2022), deriving segmentation from pixel feature similarities. Our work unifies the first and third approaches, training CAST with a self-supervised image recognition framework while naturally producing unsupervised hierarchical segmentation.

D Additional Method Details

D.1 Algorithm

We present the pseudo code of the Graph Pooling module and our CAST framework.

Algorithm 1: GraphPool

Input: Feature Z and number of clusters N
Output: Coarse feature Y and assignments P
Centroid indices $C \leftarrow \text{FPS}(Z, N)$
Refined feature $U \leftarrow \text{MSA}(Z) + Z$
Normalized feature
$U \leftarrow U - \text{mean}(U) + \text{bias}$
Centroid feature $V \leftarrow \{U[c] | c \in C\}$
$P \leftarrow \text{softmax}(\kappa \frac{UV^\top}{\|U\|\|V\|})$
Project feature $Z^{'} \leftarrow \text{MLP}(Z)$
Average pooled feature $Z^{'} \leftarrow P^\top Z^{'} / P^\top 1$
New centroid feature $Y \leftarrow \{Z[c] | c \in C\}$
Updated centroid feature $Y \leftarrow Y + Z^{'}$

FPS: Farthest Point Sampling.
MSA: Multi-headed Self-Attention.
MLP: MultiLayer Perceptron
FC: Fully Connected Layer.
$\oplus$: Concatenation operator.

Algorithm 2: Overall framework

Input: Input image I, CNN features X_{cnn}, class token X_{class}, position encoding E_{pos}, # of segments N_l at level l
Output: Feature $\mathbf{f}_{\text{class}}$ or $\mathbf{f}_{\text{seg}}$
Input segmentation $S_0 \leftarrow \text{Superpixel}(I, N_0)$
Input tokens $X_s \leftarrow \{\frac{1}{|a|} \sum_{i \in a} X_{\text{cnn}}[i] | a \in S_0\}$
for $l = 0 \ldots L$ **do**
  if $l = 0$ **then**
    Initial segment features
      $Z_0 \leftarrow [X_{\text{class}}; X_s] + [\mathbf{0}; E_{\text{pos}}]$
  else
    Coarsened segment features and clustering assignments
      $Z_l, P_l \leftarrow \text{GraphPool}(Z_{l-1}, N_l)$
    Coarsened segmentation
      $S_l \leftarrow S_{l-1} \times P_l$
  end
  $Z_l \leftarrow \text{ViT_Encoders}(Z_l)$
end
Class token $\mathbf{f}_{\text{class}} \leftarrow Z_L[0]$
Multi-level segment tokens $\mathbf{f}_{\text{seg}} \leftarrow \text{FC}(\oplus(Z_0[1 : N_0], \ldots, \text{Unpool}(Z_L[1 : N_L])))$

To obtain multi-level segment tokens, we unpool coarse tokens to the corresponding fine segments, concatenate the unpooled features with the fine tokens. The segment tokens Z_l are scattered to the affiliated fine segments:

$$Z_{l-1}^{\text{unpool}}[a] = Z_l[c], c = \text{argmax}_{c \in C} P_l[a, c]. \tag{4}$$

D.2 Open-vocabulary Segmentation

We use the open-vocabulary segmentation method OVSeg (Liang et al., 2023) to derive part-to-whole predictions from provided segments. OVSeg employs the CLIP (Radford et al., 2021) classifier on a masked image containing only the segment of interest, with other areas masked in gray. An additional background class is introduced alongside the given foreground labels. For instance, PartImageNet (He et al., 2022) consists of 2 categories (Animals vs. Things), 11 objects (e.g., Biped), and 40 parts (e.g., Biped Head), necessitating 3-way, 12-way, and 41-way classification, respectively. We predict category and object from the coarse segments and parts from fine segments. We use the ViT-B-16 model with pretrained weights released by OpenAI and do not fine-tune it with masked images.

CAST and ViT segments inherently have a hierarchy, so we restrict the part recognition vocabulary based on the object class of the parent segment. In contrast, SAM segments lack this hierarchy, necessitating a workaround to constrain their parent class. The specific procedures for each case are explained below. We applied GaussianBlur with kernel size 7×7 to smooth the segmentations.

CAST (and ViT). CAST progressively clusters segments as the layers advance, resulting in multiple levels of segments. In this context, we utilize the highest level (level 4) with 8 segments for object recognition and its child (level 3) with 16 segments for part recognition. We employ the same strategy for ViT, where the segments are generated using K-means clustering over embeddings at the same level, as discussed in the main text.

SAM. Segment Anything (SAM) (Kirillov et al., 2023) provides segments at different levels but does not consider their hierarchy. Specifically, SAM generates 64 segments with varying levels to parse the original image when no conditions are given. Here, we begin by sorting the segments in descending order of area, implying that the former ones tend to represent objects, while the latter

ones tend to represent parts. We start with an empty semantic mask and iteratively fill the mask by processing the sorted segments to obtain the semantic segmentation.

While following the same procedure, an important distinction arises between object and part recognition. In object recognition, we skip updating the pixel if it is already filled since the earlier object segment has assigned it. Conversely, for part recognition, we override the pixel with the new part segment, enabling OVSeg to consider the fine-grained segments generated by SAM. We tested various approaches, and this strategy yielded the best results.

D.3 Comparison with Hierarchical Segment Grouping

Hierarchical Segment Grouping (HSG) (Ke et al., 2022) is an unsupervised hierarchical semantic segmentation framework that induces grouping hierarchy based on pixel feature similarity. Our CAST is similar to HSG in the sense of inferring hierarchical groupings in terms of merging segments progressively. Both methods predict soft assigned probabilities to map a fine-grained segment to a coarse-grained region at the next level. Multi-scale image segmentation is induced by chaining segment assignments across all levels.

However, our CAST differs from HSG in several aspects. **1)** HSG is designed specifically for unsupervised semantic segmentation. In stark contrast, our CAST is a general backbone architecture for different downstream tasks, e.g. classification, image segmentation, and detection. In addition, our model can be trained both supervisedly and unsupervisedly. **2)** Our hierarchical image segmentation is directly involved in multi-scale feature extraction. From the early stage of our model, every token features correspond to an image segment at different scales. In contrast, HSG builds atop CNN backbones, segmentations are only inferred at the final stage of the model. **3)** Our CAST demonstrates superior model efficiency by involving hierarchical segmentation and image recognition jointly. Segmentation is induced directly from the model architecture. HSG requires additional transformer modules to infer image segmentation.

D.4 Generating Segments from ViT

We generate segments from ViT by applying K-Means clustering to the final output tokens. We bilinearly upscale the feature map and then apply K-Means to the pixel-level features to align the segments with the input image resolution. Similar to CAST, we cross-reference clustering assignments across different levels to achieve hierarchical image segmentation. To maintain a consistent cluster hierarchy, we iteratively run K-Means, reducing the number of clusters by grouping the clusters from the previous iteration into coarser clusters.

D.5 Token Pooling Methods

We re-implemented previous token pooling methods, including FINCH (Sarfraz et al., 2019), Token Pooling (Marin et al., 2021), CTM (Zeng et al., 2022), and ToMe (Bolya et al., 2023), for comparison with our proposed graph pooling module. We employ segment tokens for all methods, only modifying the token pooling layers. For FINCH and Token Pooling, we implemented their clustering algorithms. For TCFormer, we adapted Clustering-based Token Merge (CTM) module from the official implementation[4], removing the convolutional layer to apply it to our superpixel tokens. For ToMe, we used the released codebase[5] and reduced the tokens per layer to 16 to match the latency of other methods.

[4] https://github.com/zengwang430521/TCFormer
[5] https://github.com/facebookresearch/ToMe

E Additional Experimental Details

E.1 Summary of models used in experiments

We pre-train our CAST model self-supervisedly and supervisedly on ImageNet and COCO dataset. Our experiments use different model architectures and pre-training objectives. For clarification, we summarize the backbone models used in each experiment in Table 10.

Table 10: Pre-trained backbone models used in each experiment.

Experiment	Pre-training objective	Model
Hierarchical segmentation: ImageNet (Fig. 5)	self-supervised: IN-1K	CAST-S
Hierarchical segmentation: PartImageNet (Fig. 6 and Table 1)	self-supervised: IN-1K	CAST-B
Hierarchical segmentation: DensePose (Table 2)	self-supervised: COCO	CAST-S
Semantic segmentation: VOC (Table 3 a)	self-supervised: COCO	CAST-S
Semantic segmentation: ADE20K (Table 3 b)	supervised: IN-1K	CAST-S
Semantic segmentation: Pascal Context (Table 3 c)	supervised: IN-1K	CAST-S
Classification: ImageNet (Table 4)	self-supervised: IN	CAST-S
Object-centric attention: VOC (Table 5)	self-supervised: IN	CAST-S
Test-time adaptation: PartImageNet (Figure 10 and Table 9)	self-supervised: IN-1K	CAST-B

E.2 Datasets

Classification Datasets. **ImageNet** (Deng et al., 2009) is a generic image classification dataset, annotated with $1,000$ object categories (IN-1K). The training and validation set includes 1.28M and 50K images, respectively. We follow Tian et al. (2020) to subsample 100 object categories to create IN-100. The subset consists of 127K and 5K images for training and testing.

Segmentation Datasets. **1) Pascal VOC 2012** (Everingham et al., 2010) is an object-centric semantic segmentation dataset that contains 20 object categories and a background class. We use the augmented training set (Hariharan et al., 2011) with $10,582$ images and the validation set with $1,449$ images. **2) MSCOCO** (Lin et al., 2014) is a generic scene dataset with complex contexts and include more objects in each image (7.3 vs. 2.3) than VOC. Following Van Gansbeke et al. (2021), we train on $118,287$ images of *train2017* split and test on the VOC validation set. **3) ADE20K** Zhou et al. (2019) is a complex scene dataset, annotated with 150 object categories. The dataset includes $20,210$ and $2,000$ images for training and validation. **4) Pascal Context** is also a scene dataset with $4,996$ and $5,104$ images for training and validation, labelled with 59 object categories and a background class.

E.3 Image Resolution and Number of Tokens

We closely follow (Chen et al., 2021; Touvron et al., 2021; Van Gansbeke et al., 2021; Caron et al., 2021) to set up image resolution for classification and segmentation. For ViT baselines, on ImageNet, we set $\mathrm{crop_size}$ to 224, resulting in 196 input patch tokens. On VOC, we use 512×512 input images with corresponding 1024 patch tokens for semantic segmentation. We follow the same setup as (Strudel et al., 2021) for experiments on ADE20K and Pascal Context. On ADE20K, we use a 512×512 sliding window and set the stride to 512. On Pascal Context, we use 480×480 sliding window and set the stride to 320. For our CAST, we adopt the same image resolution settings and adjust the granularity of superpixels to match the number of input tokens as ViT baselines.

E.4 Hyper-parameters for Training

We list the hyper-parameters for training using MoCo and DeiT framework in Table 11 and Table 12. We mostly follow the default hyper-parameters used in each framework. We set the same $\mathrm{batch_size}$ and $\mathrm{total_epochs}$ as our CAST and ViT baselines. All the experiments are conducted with either eight TitanX (12 GB) or two A100 (45 GB) Nvidia graphic cards.

Table 11: Hyper-parameters for training our CAST, ViT, and Swin on IN-100, IN-1K, and COCO dataset. Due to computating limitations, we adapt batch_size and $mathrmtotal_epoch$ in our experiments. Otherwise, we follow mostly the same set up as MoCo (Chen et al., 2021).

Parameter	IN-100	IN-1K	COCO
batch_size	256	256	256
crop_size	224	224	224
learning_rate	$1.5e^{-4}$	$1.5e^{-4}$	$1.5e^{-4}$
weight_decay	0.1	0.1	0.1
momentum	0.9	0.9	0.9
total_epochs	200	100	400
warmup_epochs	20	10	40
optimizer	Adam	Adam	Adam
learning_rate_policy	Cosine decay	Cosine decay	Cosine decay
MOCO : temperature	0.2	0.2	0.2
MOCO : output_dimension	256	256	256
MOCO : momentum_coefficients	0.99	0.99	0.99
MOCO : MLP hidden dimension	4096	4096	4096
ViT: # Tokens	$[196]_{\times 11}$		
CAST-S/B: # Tokens	$[196]_{\times 3}, [64]_{\times 3}, [32]_{\times 3}, [16]_{\times 2}$		

Table 12: Hyper-parameters for training our CAST and ViT on IN-1K dataset. We follow mostly the same set up as DeiT (Touvron et al., 2021).

Parameter	IN-1K
batch_size	1024
crop_size	224
learning_rate	$5e^{-4}$
weight_decay	0.05
momentum	0.9
total_epochs	300
warmup_epochs	5
warmup_learning_rate	$1e^{-6}$
optimizer	Adam
learning_rate_policy	Cosine decay
augmentation	RandAug(9, 0.5) (Cubuk et al., 2020)
label_smoothing (Szegedy et al., 2016)	0.1
mixup (Zhang et al., 2017)	0.8
cutmix (Yun et al., 2019)	1.0
ViT: # Tokens	$[196]_{\times 11}$
CAST-S: # Tokens	$[196]_{\times 3}, [64]_{\times 3}, [32]_{\times 3}, [16]_{\times 2}$

E.5 Inference and Evaluation on ImageNet and VOC

For image classification on ImageNet (Fig. 11), we apply the linear probing procedure for evaluation. For semantic segmentation on VOC (Fig. 12), we use the segment retrieval and transfer learning procedure to evaluate model performance.

Image classification: linear probing. For unsupervised classification, we follow MoCo-v3 (Chen et al., 2021) to evaluate image-wise discrimination model performance using a linear probing protocol. We freeze the trained model weights and replace the 3-layer MLP head with a randomly initialized linear projection layer as classifier. We train the linear classifier with ground-truth labels and report the top-1 accuracy. Following Chen et al. (2021), we train the linear classifier with 90 epochs on ImageNet dataset. We set $\mathrm{momentum}$ to 0.9 and $\mathrm{weight_decay}$ to 0 for all experiments. On IN-1K, we set $\mathrm{batch_size}$ to 1024, $\mathrm{learning_rate}$ to 30; on IN-100, we set $\mathrm{batch_size}$ to 256, $\mathrm{learning_rate}$ to 0.8. SGD is used as the optimizer.

Semantic segmentation: segment retrieval. We follow Hwang et al. (2019); Van Gansbeke et al. (2021); Ke et al. (2022) to evaluate semantic segmentation using segment retrieval. We partition an image into several segments and conduct nearest neighbor search to predict the label for each segment. We assign the majority labels from the 20 retrieved segments.

For ViT baselines, we apply the MLP head on each token to generate unit-length output features and upsample the feature maps to the original resolution of the input image. Followed by spherical K-Means clustering algorithm, we partition the image into 40 segments using the output features.

Our CAST does not require additional upsampling and K-Means clustering. For segmentation, our model follows Hypercolumn design (Hariharan et al., 2015) to unpool and fuse multi-level segment tokens. Our model generates the same number of output tokens as the superpixels. We gather pixel features from output tokens based on the superpixel index. Without the need for spherical K-Means clustering, our CAST predicts an image segmentation using the graph pooling modules. We compute normalized segment features according to such image segmentation.

Semantic segmentation: transfer learning. We follow Van Gansbeke et al. (2021) to evaluate model performance using transfer learning protocol. All models are unsupervisedly trained on MSCOCO, and supervisedly fine-tuned on Pascal VOC. We replace the 3-layer MLP head with 2-layer 1×1 convolutional layers. For training ViT baselines, we upscale patch tokens back to the input image resolution using bilinear interpolation. For training our CAST, we scatter segment tokens to per-pixel features using superpixel indices. We use per-pixel ground-truth labels for training both methods. We set the training steps to 30K, $\mathrm{learning_rate}$ to 0.003, $\mathrm{weight_decay}$ to 0.0001, $\mathrm{batch_size}$ to 16, $\mathrm{crop_size}$ to 512. Following Chen et al. (2016), we adopt poly learning rate policy by multiplying base learning rate by $1 - \frac{iter}{max_iter}^{0.9}$. We adopt the SGD optimizer. We use only a single-scale image for inference.

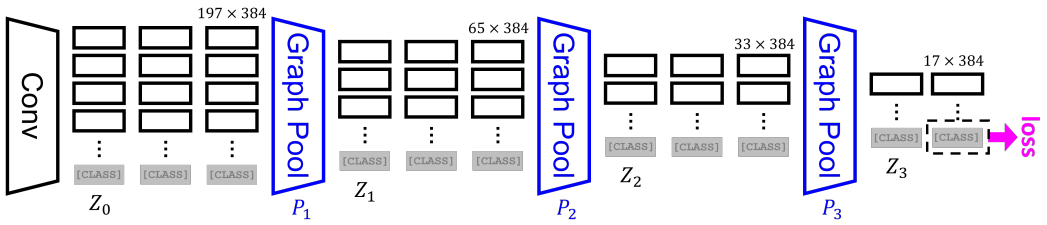
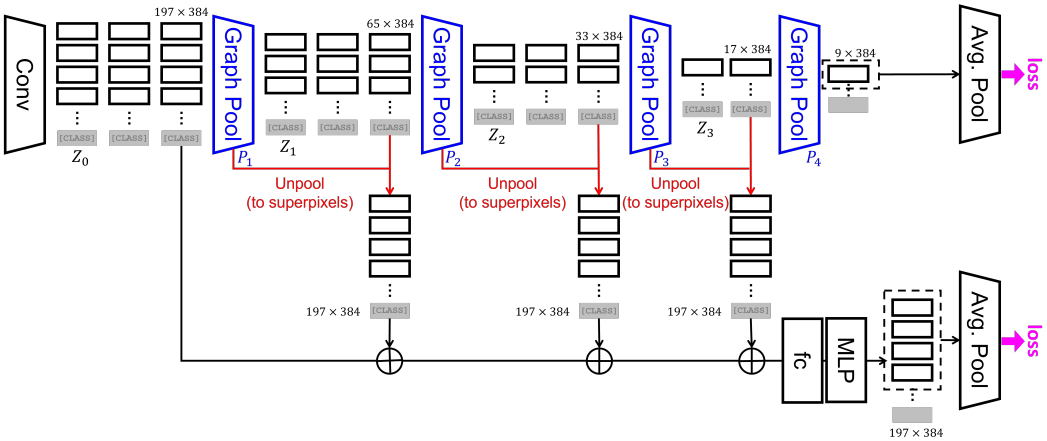
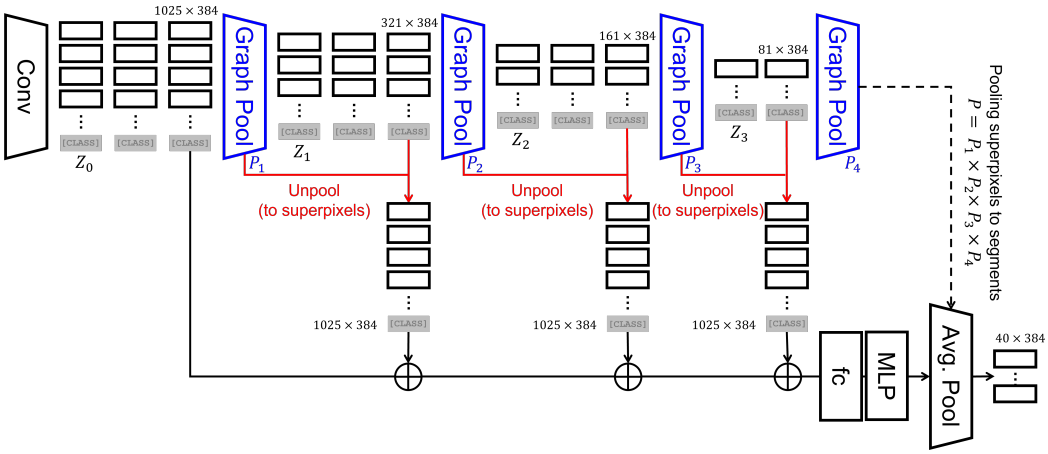


Figure 11: Detailed model architecture for classification. Our CAST aggregates segment tokens by average pooling convolutional features within each superpixel, contextualize segment tokens with transoformer encoder blocks, and coarsen them into fewer region tokens with Graph Pooling module.



(a) our CAST during training



(b) our CAST during inference for nearest segment retrievals

Figure 12: Detailed model architecture for semantic segmentation. Our CAST aggregates segment tokens by average pooling convolutional features within each superpixel, contextualize segment tokens with transoformer encoder blocks, and coarsen them into fewer region tokens with Graph Pooling module. **(a)** Our CAST-S during training. **(b)** Our CAST-S during inference for segment retrievals. We unpool coarsened segment tokens based on the grouping index w.r.t input superpixels. We concatenate ($\oplus$) unpooled segment tokens across multiple scales and fuse them using a fully_connected layer, followed by the 3-layer MLP head. For transfer learning, we replace the 3-layer MLP head with 2-layer 1×1 convolutional layers atop the fused tokens. We also remove the final average pooling layer. For segment retrievals, we learn an additional Graph Pooling module to predict coarse segmentations and average pool tokens as outputs for nearest-neighbor retrievals.

F Additional Experimental Results

F.1 Visual Results on Hierarchical Segmentation

We present additional visualizations of hierarchical segmentations induced by ViT, K-Medoids, and our CAST (Fig. 13). CAST captures image boundaries and semantics more precisely.

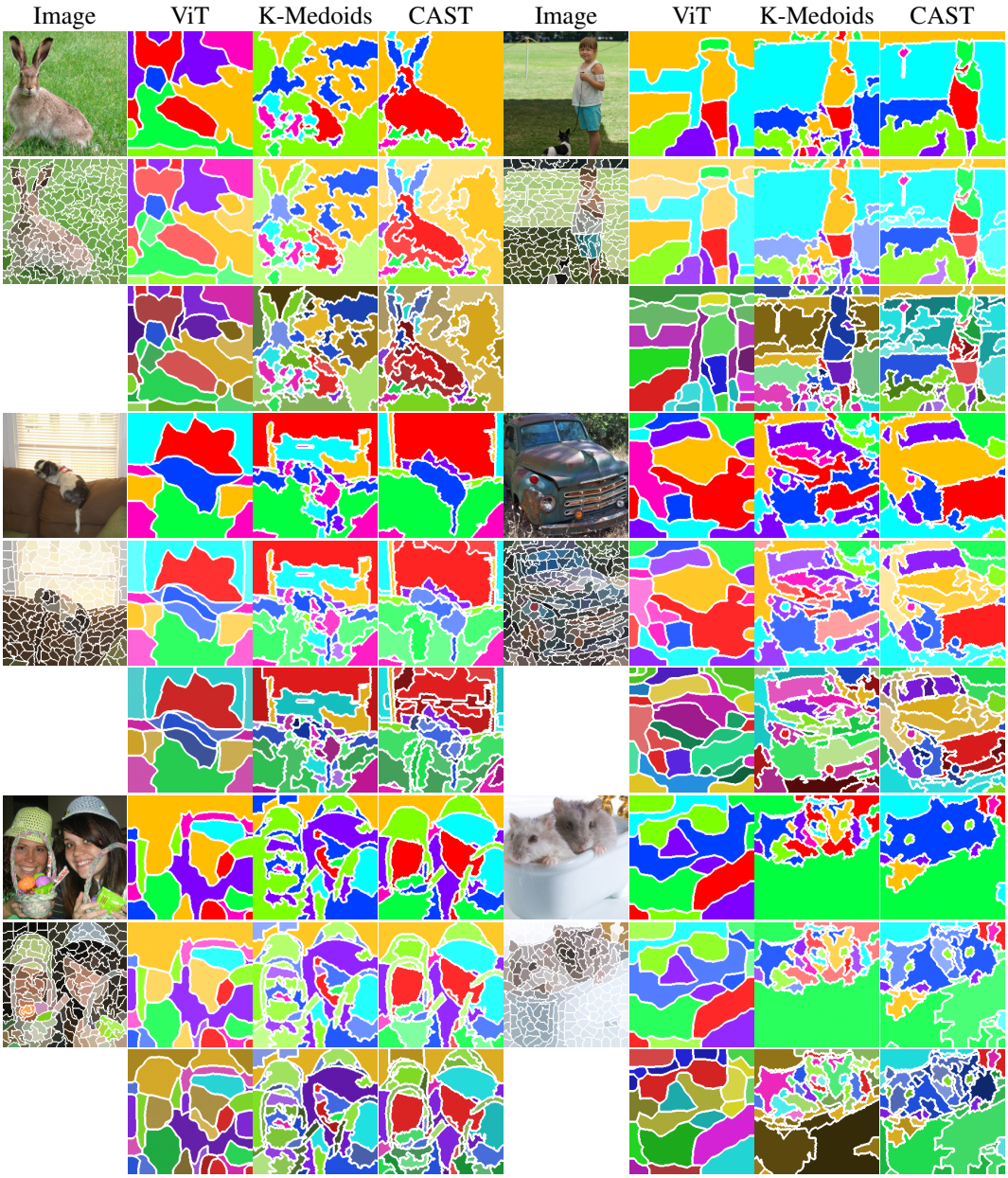


Figure 13: Our CAST generates higher-quality hierarchical image segmentation. From left to right, we show the input image and hierarchical segmentations generated by ViT, K-Medoids clustering, and our CAST. We also show corresponding superpixels (white contours) generated from input images. Our CAST naturally generates hierarchical segmentations without any post-processing. Our segmentations align with image boundaries and capture semantics more precisely.

F.2 Visual Results on Part-to-whole Recognition

We present additional visualizations for part-to-whole recognition, comparing ViT, SAM, and our CAST (Fig. 14). CAST provides consistent and semantically aligned hierarchical segmentation.

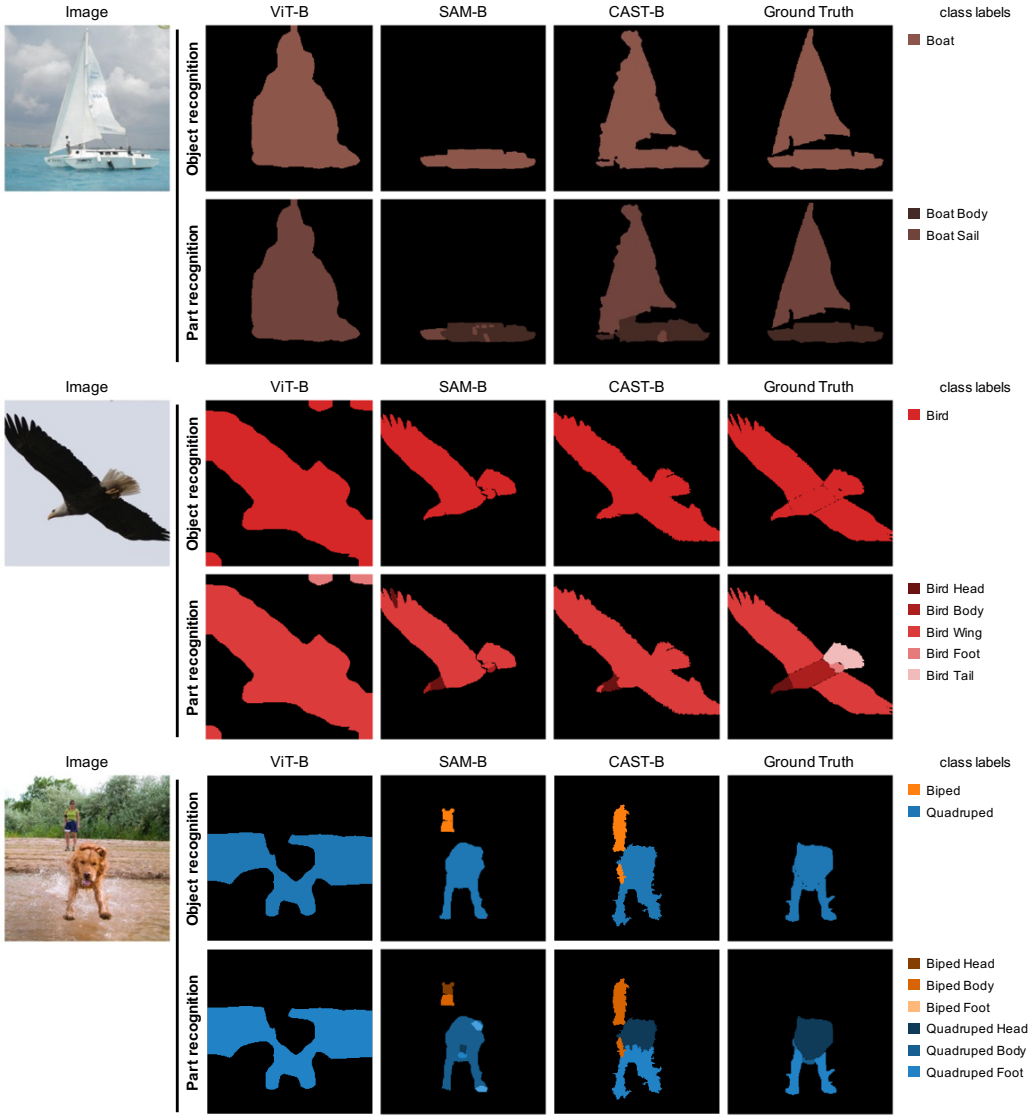


Figure 14: Additional visual results on part-to-whole recognition. Our model captures the more consistent and semantically aligned part and whole segmentations than ViT and SAM.

F.3 VISUAL RESULTS ON SEMANTIC SEGMENTATION

We present more visualization results of before and after fine-tuned semantic segmentation on VOC (Fig. 15). Compared to ViT baselines, our model produces more accurate segmentations. Remarkably, our results align with object contours precisely without the need of additional CRF post-processing.

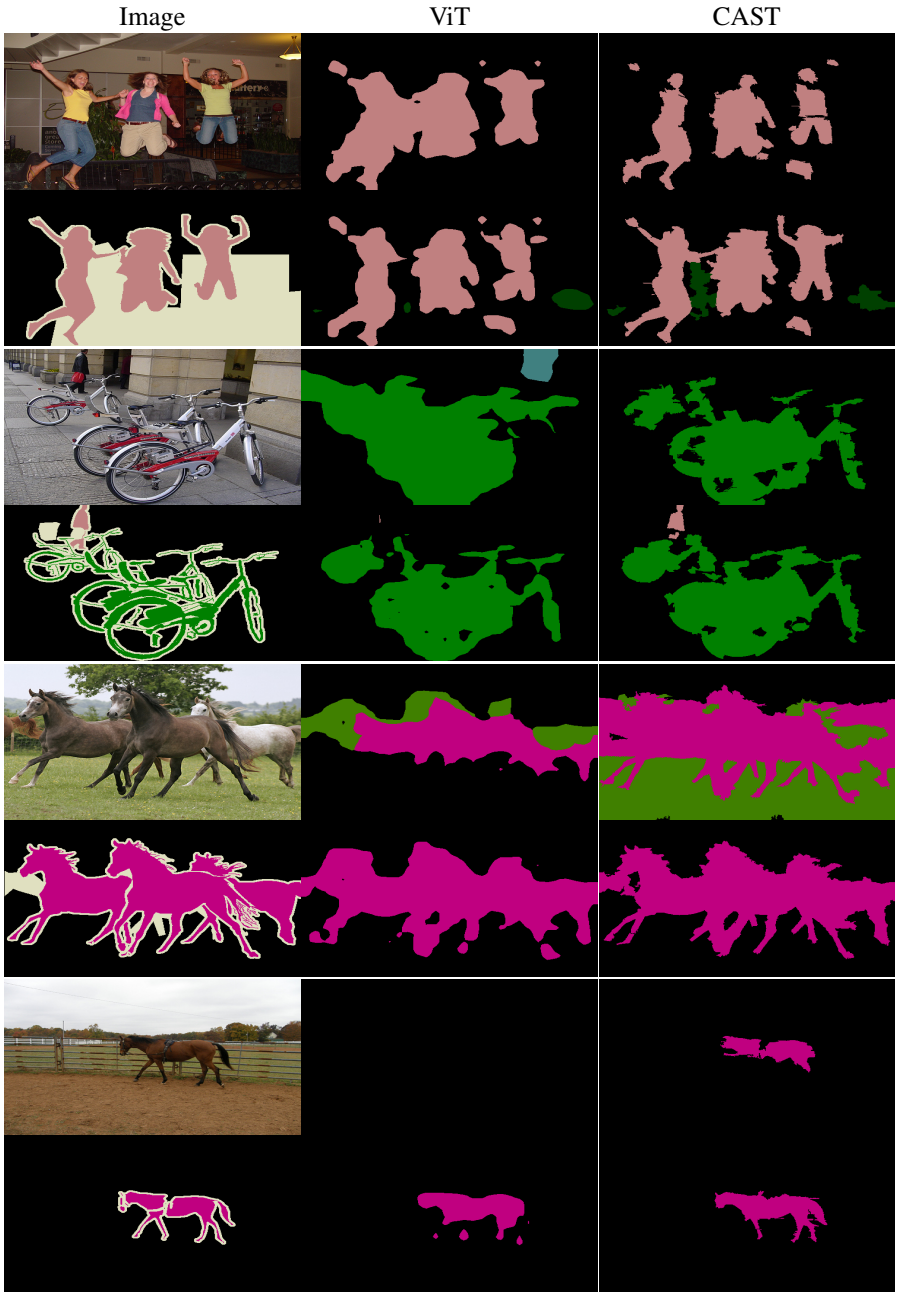


Figure 15: Our model induce much more precise segmentation than patch tokens. Segmentations are predicted based on segment-wise nearest neighbor retrievals (row 1 images) and fine-tuned models (row 2 images). Using segment, not patch, tokens improves our predicted segmentations by a large margin. Notably, our method explicitly produces segmentations without the need of additional K-Means clustering for segment retrievals.

F.4 Visual Results on Figure-ground Segmentation

We present more visual results of figure-ground segmentations generated from multi-head attention maps on VOC. Our CAST attends to foreground semantics more precisely than ViT, and the segmentations preserve object boundaries more accurately. See Fig. 16.

CAST (top) vs. ViT (bottom)

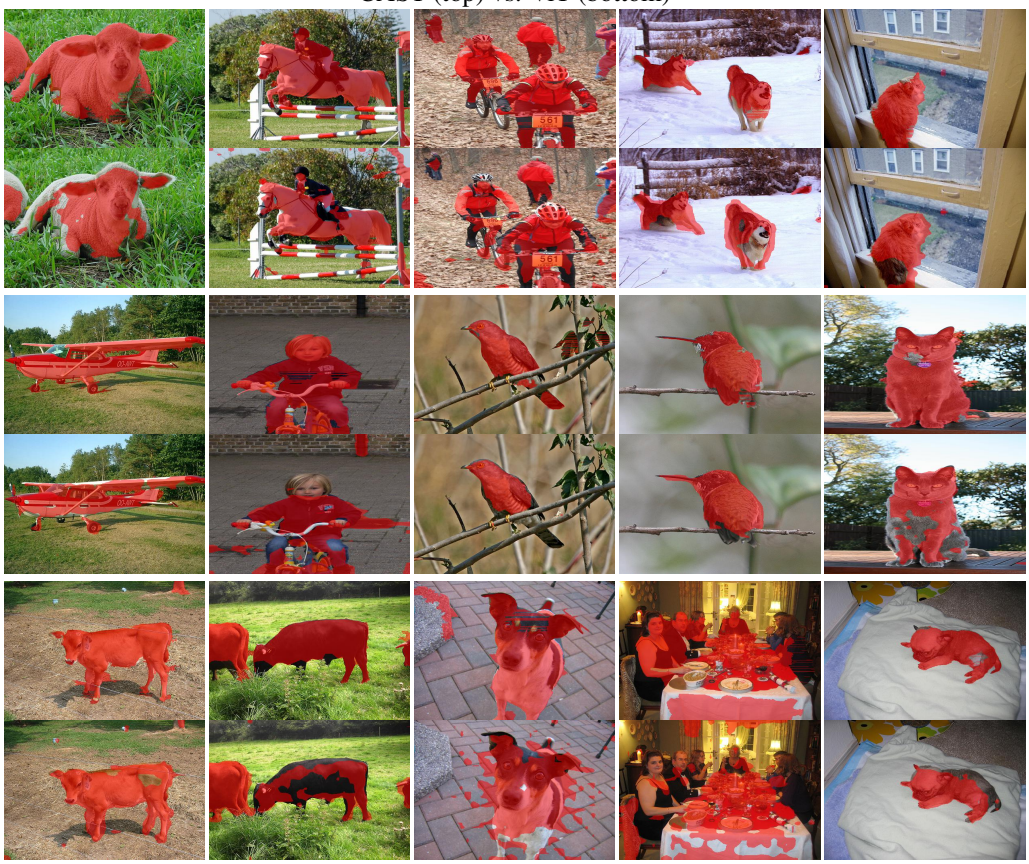


Figure 16: Our CAST (top row) attends to foreground semantics more precisely than ViT (bottom row) and DINO (Caron et al., 2021) on VOC. We adopt the same procedure as DINO to generate foreground segmentation masks from latent multi-head attention maps. All models are trained on IN-1K dataset from scratch. Our CAST and ViT are trained based on MoCo-v3 (Chen et al., 2021).

F.5 VISUAL RESULTS ON MULTI-HEAD ATTENTION MAPS

We visualize the multi-head attention maps of the [CLASS] token to all the other segment tokens in our vision transformer. As the [CLASS] token is optimized for image-wise discrimination, such attention maps indicate the most informative groupings of segments that will induce the most discriminative image-wise representations. We visualize the same attention maps used to generate the figure-ground segmentation, which are the ones in the 9th transformer encoder block. The layer takes 32 coarsened segment tokens as inputs, resulting in 12 heads of 32×32 attention maps. We follow the same procedure as DINO (Caron et al., 2021) to display the binarized attention maps. The threshold is adjusted to keep 60% of the mass. See Caron et al. (2021) for more details.

As shown in Fig. 17, our attention maps reveal parts-of-the-whole information of the image. We observe that the same object parts are together attended in the same attention head, e.g. face vs. ears vs. nose of the dog. It indicates that image-wise recognition requires parts-of-the-whole information. Additionally, our model carries segment, not patch, tokens through the layers, resulting in attention maps better aligned with object boundaries.

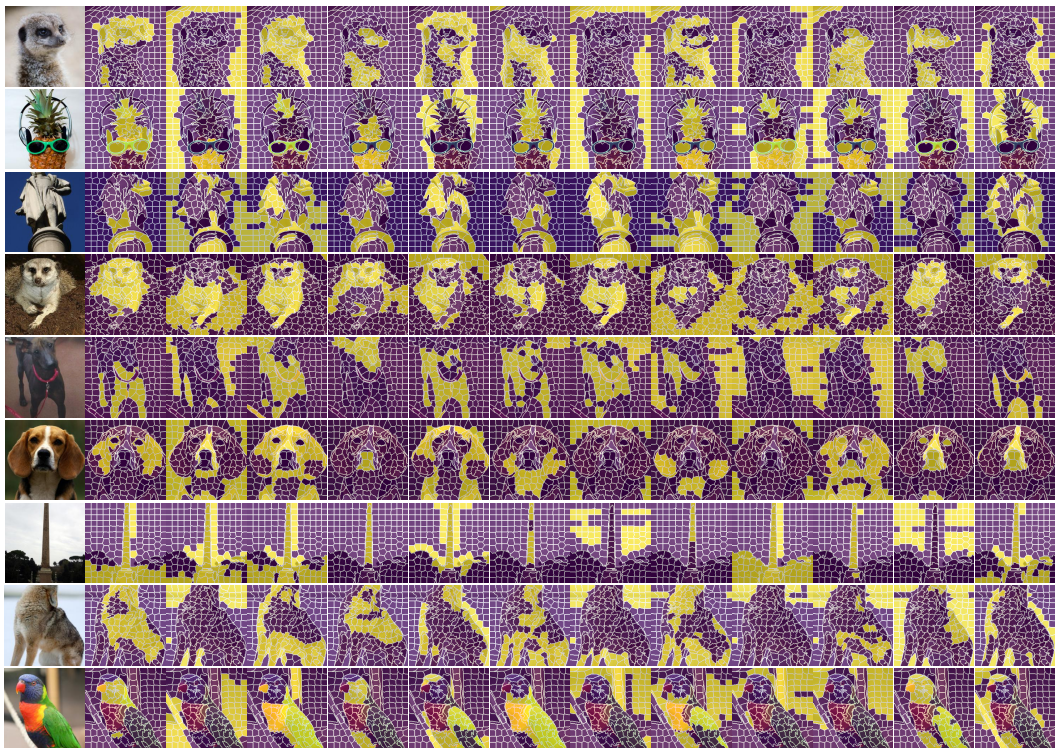


Figure 17: Our multi-head attention maps reveal parts-of-the-whole information of the image on IN-100. **From left to right:** input images and corresponding 12 heads of attention maps of the [CLASS] token to all the other segments. We follow DINO (Caron et al., 2021) to binarize attention maps. We show that the same object parts are together attended in the same head, e.g. face vs. ears vs. nose of the dog. Our model takes segment tokens, resulting in attention maps better aligned with object boundaries.