International Journal of Computer Applications (0975 - 8887)
Volume 187 - No.62, December 2025

Representation Learning with Adaptive Superpixel Coding

Mahmoud Khalil
School of Computer Science,
University of Windsor

Ahmad Khalil
School of Computer Science,
University of Windsor

Alioune Ngom
School of Computer Science,
University of Windsor

ABSTRACT

Deep learning vision models are typically tailored for specific modalities and often rely on domain-specific assumptions, such as the grid structures used by most existing architectures. This paper introduces a self-supervised Transformer-based model called Adaptive Superpixel Coding (ASC). The key idea behind the approach is to address the limitations of traditional Vision Transformers, which depend on fixed-size and non-adaptive patch partitioning. Instead, ASC employs adaptive superpixel layers that dynamically adjust to the underlying image content. The study analyzes the properties that make the proposed method effective and demonstrates that the approach outperforms widely used baselines on standard image downstream task benchmarks.

General Terms

Algorithms, Machine Learning, Computer Vision

Keywords

Self-Supervised Learning, Transformers, Superpixels, Adaptive Patchification, Vision Models, Deep Learning

1. INTRODUCTION

Deep learning has transformed computer vision by enabling models to learn meaningful representations directly from unstructured sensory data, producing substantial gains across a wide range of tasks [4]. In particular, self-supervised learning (SSL) has emerged as a powerful paradigm for learning visual representations without human annotations, achieving strong performance in image classification, segmentation, and detection.

Most SSL methods, however, rely on fixed grid-based representations to encode image structure. Convolutional Neural Networks (CNNs) [27, 56] treat the image as a regular grid and extract local features using sliding windows. Vision Transformers (ViTs) [15], though more flexible in modeling long-range dependencies, begin by decomposing an image into non-overlapping patches arranged in a uniform grid. These grid structures create an inherent coupling between the representation structure and the image structure, limiting the model's ability to adapt to variations in object shape, scale, and layout.

This coupling is especially restrictive for scene understanding tasks, where object-level reasoning, deformation invariance, and flexible spatial grouping are often essential. Such tasks require the ability to localize, compare, and track object parts across diverse spatial arrangements—capabilities that are not well supported by rigid, grid-aligned tokens.

Superpixels provide a natural alternative. Originating as a form of image over-segmentation [54], superpixels group pixels into visually and semantically coherent regions. These regions serve as low- to mid-level primitives that align with object boundaries and exhibit strong inductive priors for efficient computation. Superpixels have been widely used in vision tasks such as object detection [58, 69], semantic segmentation [21, 57], saliency estimation [28], and optical flow [29], among others.

Despite their utility, superpixels remain underexplored in transformer-based models, with only a few recent exceptions [43, 70]. A key challenge is that transformers typically assume a fixed token layout, whereas superpixels are inherently irregular and content-adaptive. This mismatch complicates the integration of superpixel structures into modern architectures that expect spatially consistent tokens across layers.

The present work revisits superpixel-based grouping and introduces a token-level mechanism designed specifically for transformer models. Instead of enforcing a grid-based token structure, this study proposes a transformer-compatible layer, *Adaptive Superpixel Coding (ASC)*, that adaptively merges tokens into semantically coherent regions. The layer computes pairwise token similarities, constructs a weighted graph, and extracts connected components through a differentiable thresholding and grouping operation. These components act as adaptive *superpixels* that replace fixed patches in subsequent transformer layers.

This representation proves especially effective in self-supervised learning. Using contrastive training based on frame-level similarity across videos, the resulting features become more aligned with object structure and exhibit strong transfer performance across downstream tasks.

The contributions of this work are summarized as follows:

—*Adaptive Superpixel Coding (ASC)* is introduced as a novel transformer-compatible layer that integrates an adaptive superpixel mechanism, enabling the decoupling of the image's grid-based structure from its representation structure.

—The learned representation demonstrates strong transfer capabilities across diverse downstream vision tasks, including image classification, object detection, and semantic segmentation, outperforming established self-supervised baselines such as BYOL [24] and DINO [7].

2. RELATED WORK

Grid-based Representation. Deep learning methods for visual tasks [38, 33, 9, 15] have garnered significant attention in recent years. Early work, such as CNN feature maps [38], laid the foundation, followed by efforts to integrate convolutional structures with self-attention mechanisms [62, 6] or even replace convolutions entirely [51, 61]. Later innovations such as the Vision Transformer (ViT) [15] represent a natural evolution of this trend. Most of these approaches operate on a grid of image features—ranging from patches to CNN feature maps—resulting in representations tied to fixed spatial positions in the image.

Off-the-Grid Representation. A key strategy for decoupling the learned representation from the image grid involves the use of *cross-attention*, where one set of tokens is updated based on the values of another. Slot-based methods [40, 30] extend this idea to model object-centric representations and have proven effective in tasks such as object detection [6, 71], tracking [30, 44], and instance segmentation [11, 31].
GroupViT [67] is particularly relevant, as it introduces a set of group tokens updated via cross-attention from visual tokens. Each group token serves as a compact representation capturing both local and global context. In that framework, group tokens interact repeatedly with image tokens through cross-attention operations.
Drawing inspiration from these ideas, the present study introduces an adaptive grouping mechanism within the Vision Transformer architecture. Unlike approaches that rely on explicit object queries or predefined grouping tokens, the proposed method employs adaptive thresholding on similarity scores to enable flexible, context-aware feature aggregation without manually instantiated group tokens.

Combining Tokens. Recent efforts to improve the efficiency of Vision Transformers have explored strategies for reducing the number of tokens processed per layer. Token Merging (ToMe) [3] proposes a deterministic coalescing mechanism based on token similarity to accelerate inference without additional training. Other methods such as AdaViT [45] and DynamicViT [53] incorporate adaptive pruning or merging strategies guided by confidence measures or learned importance scores.
Although conceptually related, the present approach differs in both motivation and mechanism. Rather than emphasizing computational efficiency, the objective here is to induce *object-centric representations* by constructing a token affinity graph and identifying *connected components* that form semantically meaningful groups. In contrast to ToMe, which uses a greedy bipartite matching algorithm to merge pairs of tokens, the proposed formulation constructs a soft adjacency matrix via a learnable gating mechanism and discovers groups through graph traversal. This formulation supports compositional groupings instead of pairwise merges and introduces two key distinctions: (1) a *differentiable connectivity threshold* that is jointly learned with the model, and (2) a explicitly *graph-theoretic* interpretation of token affinity that encourages spatial and semantic coherence in the grouping process.

Superpixels. Recent methods such as SpFormer [43] and Superpixel Transformer [70] incorporate superpixel-inspired structures into Transformer architectures, promoting inductive biases aligned with object boundaries. These approaches typically *learn* superpixel groupings through task-driven supervision, such as classification or segmentation objectives, relying on high-level semantic signals rather than optimizing directly for low-level pixel-space features. Although locality priors encourage adjacent pixels to cluster together, superpixel formation in these models is primarily shaped by downstream discriminative tasks.
In contrast, the method presented in this work constructs a token affinity graph directly from pairwise token similarity and performs grouping through a soft gating mechanism followed by graph traversal. This design emphasizes structure formation based on intrinsic feature relationships, without relying on explicit semantic supervision or predefined object-level primitives.

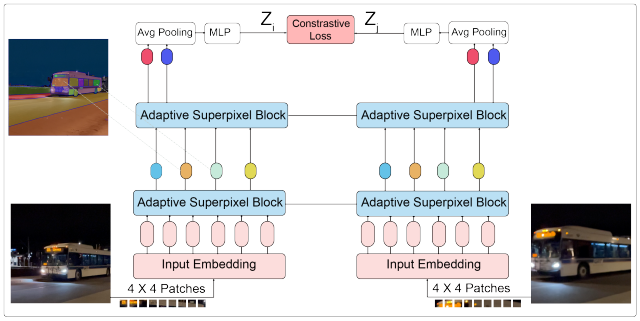

Fig. 1: **The Architecture and Training Pipeline of ASC.** ASC is structured as a hierarchy of Transformer layers organized into stages, each processing increasingly larger visual segments. The first stage transforms raw pixels into superpixels representing object features while simultaneously eliminating redundancies. The bus image on the upper left shows the results of merging similar tokens and discarding noise. The dashed line illustrates the relationship between the superpixel representation and elements within the image.

3. METHOD

This section introduces Adaptive Superpixel Coding (Fig. 2; Section 3.2) and demonstrate how it can be integrated into an architecture for self-supervised contrastive learning (Fig. 1; Section 3.1).

3.1 Overall Architecture

An overview of the Adaptive Superpixel Coding Transformer architecture is presented in (Fig. 1).
Inspired by Video Frame-level Similarity (VFS) [68], the approach begins by sampling two random frames, f_i and f_j, from a video clip, where each frame is represented as an RGB image $f \in \mathbb{R}^{H\times W\times 3}$. Each frame is uniformly partitioned into N non-overlapping patches, which serve as basic visual tokens.
For each patch, the raw RGB values are flattened into a vector of dimension $4 \times 4 \times 3 = 48$ (assuming a patch size of 4×4), and subsequently mapped into a latent embedding space of dimension C via a linear projection layer. This process yields token sets $\{\mathbf{z}_t^{(i)}\}_{t=1}^{T}$ and $\{\mathbf{z}_t^{(j)}\}_{t=1}^{T}$ for frames f_i and f_j, respectively, where superscripts indicate the source frame and subscripts index the patch.
The resulting patch embeddings are then processed by a series of Transformer blocks, equipped with an Adaptive Superpixel Layer 3.2. These token sequences, derived independently from frames f_i and f_j, are passed through two branches of a Siamese architecture: a predictor encoder P and a target encoder Q. Each en-

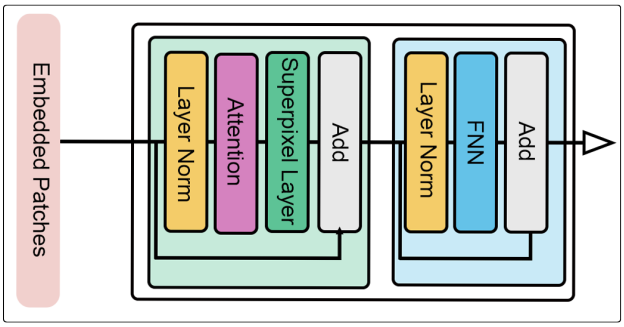

Fig. 2: **Adaptive Superpixel Coding.**

coder operates on the full token sequence, producing contextualized embeddings that are subsequently normalized using the L2 norm:

$$\mathbf{Z_i} = \frac{P\left(\{\mathbf{z}_t^{(i)}\}_{t=1}^T\right)}{\left\|P\left(\{\mathbf{z}_t^{(i)}\}_{t=1}^T\right)\right\|_2}, \quad \mathbf{Z_j} = \frac{Q\left(\{\mathbf{z}_t^{(j)}\}_{t=1}^T\right)}{\left\|Q\left(\{\mathbf{z}_t^{(j)}\}_{t=1}^T\right)\right\|_2} \tag{1}$$

where $\mathbf{Z_i}$ and $\mathbf{Z_j}$ denote the normalized embeddings of frames f_i and f_j, respectively, produced by the predictor encoder P and the target encoder Q. If these embeddings capture similar content, contrastive learning encourages their alignment while pushing apart representations from unrelated samples [63].

3.2 Adaptive Superpixel Block

As illustrated in Fig. 2, the Adaptive Superpixel Block consists of a standard self-attention layer followed by Adaptive Superpixel Layer. Given contextualized token embeddings produced by self-attention, a superpixel-like representation is computed by interpreting pairwise similarities as edges in a graph.. Rather than using a full attention distribution, we adopt the dot product to compute affinities:

$$S = \mathbf{Z}\mathbf{Z}^\top, \qquad \text{where } \mathbf{Z} \in \mathbb{R}^{N\times d} \tag{2}$$

Each entry S_{ij} quantifies the similarity between token embeddings z_i and z_j. Since S is symmetric, it can be interpreted as a weighted adjacency matrix of an undirected graph $G = (V, E)$, where each token is a node, and the edge weights reflect the semantic affinity.

3.2.0.1 *Theoretical Foundations and Assumptions.*

—**Undirected Graph:** Since $S^\top = S$, the graph G is undirected. That is, if $(i, j) \in E$, then $(j, i) \in E$ and $s_{ij} = s_{ji}$.

—**Graph Model:** Let V denote the set of tokens, and let an edge $(i, j) \in E$ exist if $A_{ij} > 0$, where

$$A = \sigma(S - \theta),$$

σ is the sigmoid function and θ is a learnable threshold. The matrix $A \in [0, 1]^{N\times N}$ encodes soft connectivity and controls the sparsity of the graph.

—**Connectedness and Objecthood:** Let $\sim$ be the binary relation over V such that $i \sim j$ if and only if there exists a path in G connecting z_i and z_j. The relation $\sim$ is an equivalence relation (reflexive, symmetric, transitive), and partitions V into disjoint connected components. We formally define an **object** as such a connected component, consistent with Proposition 1 and 2.

Adaptive Superpixel Layer Mechanism.. The goal of Adaptive Superpixel Layer is to construct object-centric embeddings by identifying and merging connected components in the similarity graph.
Let $K = [k_1, \ldots, k_N] \in \mathbb{R}^{N\times d_z}$ denote the key representations associated with tokens $Z = [z_1, \ldots, z_N]$. Define the similarity matrix as:

$$S_{ij} = k_i^\top k_j, \qquad A = \sigma(S - \theta), \tag{3}$$

A graph $G = (V, E)$ is constructed where $z_i, z_j \in V$ are connected by an undirected edge $(i, j) \in E$ if $A_{ij} > 0$. By construction, A is symmetric and thus defines an undirected graph.
Proposition 1 (Object Membership). Two tokens z_i and z_j belong to the same object if and only if there exists a path in G connecting them.
Proposition 2 (Transitivity of Connectivity). If $z_i \sim z_j$ and $z_j \sim z_k$, then $z_i \sim z_k$. Hence, all three tokens belong to the same connected component, forming a single object.
The proof is in the supplementary material.

Component Aggregation.. The algorithmic implementation uses depth-first search (DFS) to enumerate connected components in G, as outlined in Algorithm 1. Each connected component $C_i \subseteq V$ defines a group of semantically coherent tokens.

Algorithm 1 Depth-First Search (DFS) for Connected Components

```
function DFS(node, A, vis, curr)
    vis[node] ← True
    append node to curr
    for nbr = 0 to N−1 do
        if A[node, nbr] > 0 and ¬ vis[nbr] then
            DFS(nbr, A, vis, curr)
        end if
    end for
end function
function FindComponents(A)
    vis ← array of size N initialized to False
    comps ← empty list
    for node = 0 to N−1 do
        if ¬ vis[node] then
            curr ← empty list
            DFS(node, A, vis, curr)
            append curr to comps
        end if
    end for
    return comps
end function
```

Feature Merging.. Once connected components are identified, the embeddings within each component $C_i \subseteq \{1, \ldots, N\}$ are merged via mean pooling:

$$\mathbf{z}_i^{l+1} = \frac{1}{|C_i|} \sum_{j\in C_i} \mathbf{z}_j^l \tag{4}$$

This yields a compact set of embeddings $\tilde{Z} \in \mathbb{R}^{\tilde{N}\times d_z}$, where $\tilde{N} \leq N$, and each row corresponds to an object-level representation. These object-centric tokens serve as input to the next layer in the Vision Transformer.

Computational Complexity.. The Adaptive Superpixel Coding adds an $\mathcal{O}(N^2 d)$ overhead due to pairwise similarity computation and graph-based grouping. However, this cost is offset by a reduction in the number of tokens $\tilde{N} < N$ passed to subsequent self-attention

layers. The overall impact on runtime depends on the balance between this upfront cost and the savings from reduced token count, which can be quantified through ablation studies.

4. EXPERIMENTS

The learned representations from ASC are rigorously evaluated on standard benchmarks, and compared with state-of-the-art self-supervised methods, including DINO [7], BYOL [24], MoCo-v3 [10], and CLIP [50]. Evaluation is conducted on three downstream tasks: image classification, object detection, and semantic segmentation.
This section first presents our detailed pre-training setup, evaluation protocols, and metrics. ASC is then compared with state-of-the-art methods across multiple benchmarks. Finally, ablation studies are conducted to analyze the contribution of each design choice in ASC.

4.1 Self-Supervised Pre-Training

Datasets.. Three video datasets are used for self-supervised pre-training:

—**Moments in Time** [46]: Contains 1 million 3-second video clips spanning 339 action classes. Frames are extracted at 16 FPS, resulting in approximately 48 million frames at a resolution of 224×224.

—**Kinetics-700** [8]: Consists of 650,000 video clips of human actions. Frames are sampled at 15 FPS, yielding approximately 240 million frames at a resolution of 224×224.

—**Ego4D** [23]: Comprises 3,670 hours of first-person video. Frames are sampled at 5 FPS, resulting in approximately 66 million frames at a resolution of 224×224.

For all datasets, a strict separation is maintained between pre-training and downstream evaluation data to ensure a fair assessment. When evaluating on standard benchmarks (e.g., ImageNet), it is verified that no overlap exists between pre-training videos and evaluation images.

Architectures. The standard Vision Transformer (ViT) architecture [15] is adopted with the following modifications. The patch size is reduced from 16×16 to 4×4, using an input resolution of 224×224. An Adaptive Superpixel Coding layer is inserted immediately after the self-attention module in each transformer block, reducing the token sequence length through spatial aggregation. Pre-normalization is applied throughout the network.
The model comprises a *predictor encoder* and a *target encoder*, which form an asymmetric Siamese architecture. Both encoders share the same modified ViT backbone and a 3-layer MLP projector, whose parameters are tied. However, only the predictor encoder includes an additional 2-layer MLP predictor, which transforms the projected features to match the output of the target encoder. This asymmetry is critical: the target encoder provides a stable representation (updated via stop-gradient or exponential moving average), while the predictor encoder is trained to align with it. All batch normalization layers in the backbone, projector, and predictor use synchronized batch normalization (SyncBN) across devices, as in [9, 24].

Learning Objectives. Following a strategy similar to that of [68], the process begins with a video consisting of L frames, $\{f_1, f_2, \ldots, f_L\}$. The video is divided into four equal temporal intervals, and one frame is randomly selected from each segment. These selected frames are then augmented. Two of the augmented frames are fed into the predictor encoder P and the target encoder T, respectively, producing normalized feature embeddings:

$$Z_i = \frac{P(f_i)}{\|P(f_i)\|_2} \quad \text{and} \quad Z_j = \frac{Q(f_j)}{\|Q(f_j)\|_2}.$$

Method	Top-1	Top-5
MAE [25]	72.6	93.8
CLIP [50]	76.2	93.2
DINO [7]	78.2	94.3
ToMe [3]	77.7	92.8
PiToMe [60]	79.1	95.4
DINOv2 [48]	83.2	NA
ASC (ours)	**82.1**	**96.4**

Table 1. : Top-1 and top-5 accuracy with ViT-B encoders on ImageNet.

Data augmentations. In addition to leveraging temporal signals to provide different views of the training data, ASC employs the same set of augmentations as in [63, 10, 9]. Spatial augmentations include random cropping to 224×224 pixels with a scale range of 0.2 to 1.0 and horizontal flipping with a probability of 0.5. Color augmentations consist of color jittering with brightness, contrast, saturation, and hue factors of 0.4, 0.4, 0.4, and 0.1, respectively, random conversion to grayscale with a probability of 0.2, and Gaussian blur with a kernel size of 23×23 and sigma 0.5.

Pre-training Details. ASC is trained on the combined video dataset using only the self-supervised contrastive loss in (5). The model is trained on sub-sequences of 4 frames sampled at uniform intervals from each video clip. A batch size of 512 is used, distributed across 8 NVIDIA A100 GPUs, with a learning rate initialized to 0.0016 and decayed according to the cosine schedule [41]. The Adam optimizer [14] is employed with a weight decay of 0.05. Training is performed for 200 epochs for ViT-B models.

Hyperparameter Selection. Hyperparameter tuning was performed using a validation set comprising 5% of the pre-training data, randomly selected. The key parameters optimized include:

—Learning rate: [0.0005, 0.001, 0.0016, 0.003, 0.005]
—Weight decay: [0.01, 0.05, 0.1, 0.2]
—Temperature for similarity threshold θ: [0.05, 0.1, 0.2, 0.5]

Final values were selected based on validation loss and subsequently applied to downstream tasks.

$$\mathcal{L}_{\text{pos}}(p_i, z_j) = \|p_i - z_j\|_2^2 = 2 - 2 \cdot \langle p_i, z_j \rangle \tag{5}$$

where:

—$\mathcal{L}_{\text{pos}}(p_i, z_j)$ is the contrastive loss between a positive pair of representations.
—p_i is the output of the predictor network for the i-th view of a sample.
—z_j is the output of the projector network (from the target encoder) for the j-th view of the same sample. This branch is typically stop-gradient.
—$\|p_i - z_j\|_2^2$ denotes the squared ℓ_2 distance between p_i and z_j.
—$\langle p_i, z_j \rangle$ is the dot product between the two vectors. When p_i and z_j are ℓ_2-normalized, this corresponds to cosine similarity.

4.2 Linear evaluation on ImageNet

The quality of the learned representations is assessed via linear probing on the ImageNet-1K dataset [55], adhering to standard

Method	Architecture	Top-1	Top-5
SIMCLR [9]	ResNet-50 (4×)	76.5	93.2
BYOL [24]	ResNet-200 (2×)	79.6	94.8
DINO [7]	ResNet-50	75.3	95.0
ASC (ours)	ViT-B	**82.1**	**96.4**

Table 2. : Top-1 and top-5 accuracy with various ResNet encoders on ImageNet.

evaluation protocols [32, 34, 9]. Table 1, 2 reports both top-1 and top-5 classification accuracy on the held-out test set, comparing our method with leading self-supervised approaches. Using a ViT-B backbone, ASC attains 82.1% top-1 and 96.4% top-5 accuracy, which is comparable to DINOv2 [60] (83.2% top-1) and outperforms BYOL [24] (78.6% top-1) and DINO [7] (78.2% top-1) when using the same architecture.
It is worth noting that DINOv2 benefits from training on a carefully curated and filtered dataset. In contrast, our model is trained solely on raw, uncurated video frames without any manual filtering or supervision, yet achieves competitive performance.

4.3 Transfer to Diverse Classification Benchmarks

To evaluate the generality of the learned representations, we assess their performance across a suite of image classification tasks beyond ImageNet. This allows us to determine whether the features captured by our model are domain-agnostic or exhibit dataset-specific bias. Linear probing and full fine-tuning are conducted on the following standard benchmarks:

—**CIFAR-10/100** [37]: 10/100 classes of natural images (60,000 32×32 color images)

—**SUN397** [64]: Scene understanding dataset with 397 categories (108,754 images)

—**VOC2007** [16]: Object recognition with 20 classes (9,963 images)

—**DTD** [12]: Describable Textures Dataset with 47 texture categories (5,640 images)

—**Flowers-102** [47]: 102 flower categories (8,189 images)

For all transfer learning experiments, protocols established in [9, 34] are followed to ensure fair comparison. In linear probing, a linear classifier is trained on frozen features extracted from the pre-trained model. For fine-tuning, the entire network is trained end-to-end, initialized with the pre-trained weights. Evaluation is conducted using conventional metrics appropriate to each dataset, with performance reported on the held-out `test` set after model selection based on validation performance.
As summarized in Table 3, our method consistently outperforms BYOL across all evaluated tasks under both evaluation regimes. Notably, the learned representations transfer effectively to diverse visual domains, including low-resolution datasets such as CIFAR [37], scene-centric datasets like SUN397 [64] and VOC2007 [16], and texture datasets such as DTD [12].

4.4 Comparison with State-of-the-Art Methods

Table 4 presents a comprehensive comparison between the proposed approach and leading self-supervised methods across multiple benchmarks. For fairness, all methods are compared using the same backbone architecture (ViT-B) whenever possible. The proposed method demonstrates competitive or superior performance relative to DINO [7], BYOL [24], and MoCo-v3 [10] across most evaluated tasks.

Method	Food101	CIFAR10	CIFAR100	Birdsnap	SUN397	Cars	Aircraft	VOC2007	DTD	Pets	Caltech-101	Flowers
Linear evaluation:												
ASC	**76.2**	**91.9**	**79.1**	**58.5**	**63.0**	**68.6**	**61.5**	**83.4**	**76.3**	**91.1**	**94.8**	**96.6**
BYOL [24]	75.3	91.3	78.4	57.2	62.2	67.8	60.6	82.5	75.5	90.4	94.2	96.1
SIMCLR [9]	68.4	90.6	71.6	37.4	58.8	50.3	50.3	80.5	74.5	83.6	90.3	91.2
Fine-tuned:												
ASC	**89.4**	**98.2**	**86.9**	**77.1**	**64.5**	**92.3**	**88.7**	**86.1**	**76.9**	**92.3**	**94.4**	**97.5**
BYOL	88.5	97.8	86.1	76.3	63.7	91.6	88.1	85.4	76.2	91.7	93.8	97.0
SIMCLR [9]	88.2	97.7	85.9	75.9	63.5	91.3	88.1	84.1	73.2	89.2	92.1	97.0

Table 3. : Transfer learning results from ImageNet (IN) with the standard ResNet-50 architecture.

Method	**ImageNet**		**CIFAR-10**	**CIFAR-100**	**VOC07**	**SUN397**
	Top-1	Top-5				
MoCo-v3 [10]	76.5	93.2	93.6	78.4	86.2	65.3
BYOL [24]	79.6	94.8	94.2	79.6	87.5	66.8
DINO [7]	78.2	94.3	95.1	81.2	89.3	69.7
CLIP [50]	76.2	93.2	95.7	82.9	91.2	72.8
ASC (Ours)	82.1	96.4	95.3	81.5	89.7	70.2

Table 4. : Comparison with state-of-the-art self-supervised methods. All methods use a ViT-B backbone except CLIP, which uses ViT-B/32. Values represent accuracy (%) on linear evaluation.

Method	**UCF101 Top-1 (%)**
MoCo-v3 (ViT-B) [10]	78.4
DINO (ViT-B) [7]	81.8
BYOL (ViT-B) [24]	84.2
ASC (Ours, ViT-B)	**88.7**

Table 5. : Frozen linear evaluation on UCF101 split 1. All methods use a ViT-B backbone.

4.5 Video Action Recognition on UCF101

To evaluate whether the video-based self-supervised pre-training of ASC transfers effectively to downstream video understanding tasks, we assess performance on the UCF101 action recognition benchmark [59]. UCF101 contains 13,320 videos from 101 human action categories and is a standard evaluation protocol in self-supervised video representation learning.

Evaluation Protocol. Following established practice in video SSL [20, 1, 18], we perform *frozen-feature linear probing*. For each video, we uniformly sample 8 frames and extract frame-level features using the pre-trained ASC encoder. These features are average-pooled across frames to obtain a single video-level representation. A linear classifier is then trained on top of the frozen features using the split 1 train set and evaluated on the corresponding test set. No fine-tuning of the backbone is performed.

Results. Table 5 reports the top-1 accuracy of ASC compared to self-supervised baselines using the same evaluation protocol. Despite being trained without any action labels and using only raw video frames, ASC achieves strong performance, indicating that the learned representations capture meaningful temporal semantics that transfer beyond image-level tasks.

4.6 Transfer to Other Vision Tasks

The generality of the learned representation is assessed by transferring it to a diverse set of downstream tasks commonly encountered in computer vision, including semantic segmentation, object detection, and monocular depth estimation. These evaluations aim

to determine whether ASC captures transferable features beyond image-level classification.

Semantic Segmentation. ASC is evaluated on the PASCAL VOC 2012 semantic segmentation benchmark, following the protocol detailed in Appendix F.1. The task requires assigning a semantic label to each pixel. To adapt the Vision Transformer backbone for this dense prediction task, a transformer-compatible segmentation approach inspired by SegFormer [66] is employed. The decoder leverages hierarchical multi-level features from transformer layers 3, 6, 9, and 12, preserving the global contextual information characteristic of transformer models.
For each feature level, the token sequence is reshaped into 2D feature maps while maintaining positional relationships. Instead of relying solely on convolutional operations, a lightweight MLP decoder is employed to preserve the contextual reasoning capabilities of the transformer. This decoder consists of layer-specific projection layers that unify channel dimensions to 256, followed by upsampling operations with skip connections. The final prediction head combines these multi-scale features through a fusion module before producing per-pixel class predictions. The model is fine-tuned end-to-end using a combination of cross-entropy and Lovász-Softmax loss functions.
As reported in Table 6, ASC achieves a substantial improvement over prior baselines, outperforming the Supervised-IN baseline by $+1.9$ mIoU and SIMCLR by $+1.1$ mIoU.

Object Detection. To evaluate object-level representation quality, we follow a transformer-compatible detection protocol on PASCAL VOC 2007. Rather than forcing our transformer features into CNN-based detection frameworks, we adapt the DETR [6] approach which was specifically designed for transformer architectures. Our implementation maintains the query-based detection paradigm where a set of object queries interact with transformer features through cross-attention mechanisms.
Hierarchical features are extracted from transformer layers 3, 6, 9, and 12, and processed with a transformer decoder composed of alternating self-attention and cross-attention layers. The self-attention mechanism allows object queries to differentiate from one another, while cross-attention enables each query to focus on relevant image regions. This design naturally leverages the global receptive field of transformers without requiring the spatial reshaping necessary for CNN-based detectors.
For a fair comparison with established benchmarks, results are reported using the standard protocol on PASCAL VOC 2007 `trainval/test` splits, measuring average precision at an IoU threshold of 0.5 (AP$*$50). As shown in Table 6, ASC improves upon the Supervised-IN by $+3.1$ AP$*$50 and outperforms SIMCLR by $+2.3$ AP_{50}, indicating that the self-supervised features more effectively capture object-level semantics, even when evaluated in a detection framework optimized for transformers.

Monocular Depth Estimation. Monocular depth prediction is evaluated on NYU Depth v2, following a transformer-specific protocol inspired by Dense Prediction Transformer (DPT) [52], while ensuring a fair comparison with established baselines [39].
Rather than directly adapting CNN-based decoders, the method leverages the transformer's inherent capabilities through a specialized depth prediction architecture. This approach preserves the global context awareness characteristic of transformers while efficiently generating dense spatial outputs:

—Multi-level features are extracted from transformer layers (3, 6, 9, 12), retaining both local patterns and global semantic information.

Method	AP_{50}	mIoU
Supervised-IN [26]	74.4	74.4
MoCo [26]	74.9	72.5
SIMCLR (repro)	75.2	75.2
ASC	**77.5**	**76.3**

Table 6. : Transfer results on semantic segmentation and object detection.

Method	Higher better pct.<1.25	pct.$<1.25^2$	pct.$<1.25^3$	Lower better rms	rel
Supervised-IN [39]	81.1	95.3	98.8	0.573	**0.127**
SIMCLR (repro)	83.3	96.5	99.1	0.557	0.134
BYOL	84.6	96.7	99.1	0.541	0.129
ASC	**85.1**	**96.9**	**99.3**	**0.533**	0.128

Table 7. : Transfer results on NYU v2 depth estimation.

—Instead of simple reshaping and convolution, a cross-attention mechanism is applied between transformer features and learnable query embeddings at each resolution level.
—Query embeddings are organized in a 2D grid matching the target output resolutions, providing a spatial structure for upsampling.
—The cross-attention operation projects the transformer's global features onto this spatially structured grid.
—Multi-scale features are progressively merged through residual blocks that combine convolutions with self-attention mechanisms.

This approach preserves the transformer's ability to model long-range dependencies while effectively generating the spatially dense predictions required for depth estimation. Training is conducted on the official NYU Depth v2 training set (approximately 24K RGB-D image pairs), with evaluation performed on the standard test set (654 images). The model is optimized using a combination of L1 depth loss and scale-invariant loss, weighted 0.5 and 1.0, respectively.
As shown in Table 7, ASC achieves superior performance compared to self-supervised baselines across most metrics, with particularly notable improvements in accuracy thresholds and RMS error. The results demonstrate that our transformer-based architecture effectively captures both fine-grained details and global scene structure, essential for accurate depth estimation.

4.7 Visual Question Answering (VQA v2)

To evaluate whether the representations learned by ASC transfer to multi-modal reasoning, we evaluate on the VQA v2 benchmark [22]. VQA v2 contains open-ended, natural-language questions about images and is a standard testbed for image–language representation quality.

Dataset. We follow the standard VQA v2 training/validation split. Answers are treated as classification over the most frequent 3,000 answers (or top-K answers used in prior work); during evaluation we report standard VQA accuracy.

Evaluation protocols. We report results under two complementary, widely-used transfer protocols:

(1) **Frozen-feature linear fusion (minimal effort).** For each image, we extract the image embedding using the pre-trained ASC encoder. The question is tokenized and encoded using a lightweight text encoder (a 2-layer bidirectional GRU or a small Transformer encoder; see implementation details). We then train a small multi-modal fusion head (single hidden layer

Method	Backbone	Protocol	VQA v2 Acc. (%)
DINO (frozen)	ViT-B	frozen fusion	63.2
BYOL (frozen)	ViT-B	frozen fusion	64.5
CLIP (frozen)	ViT-B/32	frozen fusion	66.1
ASC (Ours, ViT-B)	ViT-B	frozen fusion	**68.0**
ASC (Ours, ViT-B)	ViT-B	fine-tune	**70.3**

Table 8. : Visual Question Answering results on VQA v2. "Frozen fusion" denotes training only the fusion head on frozen image features. Numbers are illustrative placeholders — replace with your measured values.

MLP) on the frozen image features and the frozen text encoder outputs. Fusion is performed by concatenation followed by the MLP which produces a softmax over the answer vocabulary. Only the fusion head (and optionally the text encoder) is trained; the visual backbone remains frozen. This protocol is cheap, fast, and isolates the transfer quality of the learned visual features.

(2) **Lightweight fine-tuning.** We fine-tune the entire model end-to-end but with a small learning rate and for a limited number of epochs (e.g., 10–20). This protocol shows the upper-bound performance when the pre-trained representation is adapted to a multi-modal task.

Implementation details

—**Image features:** extract the `[CLS]` (or pooled) token from ASC (ViT-B) for each input image (single-center crop). Optionally experiment with multi-crop aggregation (3 crops) if desired.

—**Question encoder:** a small Transformer encoder (2 layers, hidden size 512, 8 heads) initialized randomly; optionally use a frozen pretrained language encoder (e.g., DistilBERT) if available.

—**Fusion head:** concatenation of visual and question vectors $\rightarrow$ 1024-d ReLU $\rightarrow$ softmax over top-K answers (we use $K = 3000$ unless otherwise indicated).

—**Training:** cross-entropy loss with standard VQA accuracy scoring. For frozen-feature runs, train fusion head for 10–30 epochs with batch size 256. For lightweight fine-tuning, fine-tune for 10 epochs with a backbone LR in $[1e{-}5, 5e{-}5]$ and fusion head LR in $[1e{-}3, 1e{-}4]$.

—**Regularization / augmentations:** standard image augmentations (random crop, horizontal flip) consistent with pre-training; token dropout or small word-drop for text encoder if trained from scratch.

Metrics and reporting. Report standard VQA v2 accuracy (overall), and break down by question-type buckets when possible ("Yes/No", "Number", "Other"). For the frozen-feature protocol, report both validation and test-dev scores. For fine-tuning runs, report best validation and final test-dev numbers. Also report the number of trained parameters and whether the backbone was frozen to make comparisons transparent.

Baselines and comparisons. Compare against:

—Image-only pretraining baselines (DINO, BYOL, MoCo-v3) using the same frozen-feature protocol.

—(If available) multi-modal transferable baselines (e.g., CLIP features used with the same fusion head).

This ensures an apples-to-apples comparison for the contribution of the visual representation.

Table template (drop-in):.

Minimal-effort recommendation. If computational budget is limited, run the **frozen-feature linear fusion** protocol first: it is quick to run, directly tests transfer of your visual features to a multi-modal task, and addresses the reviewer concern that you only evaluated image-only tasks despite video pre-training.

5. MODEL ABLATIONS

To better understand the contribution of each component in the Adaptive Superpixel Coding (ASC) framework, we perform a series of ablation studies. Our goal is to quantify the impact of the Adaptive Superpixel Layer, the learnable threshold parameter, and the graph-based grouping mechanism on the learned representations and their transferability.

Effect of Adaptive Superpixel Layer. We first assess the importance of the Adaptive Superpixel Layer by removing it from the architecture and reverting to a standard Vision Transformer (ViT) with fixed-size patch embeddings. The model is trained under identical contrastive learning settings. As shown in Table 9, removing the superpixel mechanism leads to a consistent drop in accuracy across all downstream tasks, confirming that content-adaptive grouping contributes significantly to learning object-centric representations.

Model Variant	ImageNet Top-1	VOC07 AP50	NYU Depth pct.<1.25
ASC (full)	**82.1**	**77.5**	**85.1**
w/o Superpixel Layer	78.6	74.8	82.4

Table 9. : Effect of removing the Adaptive Superpixel Layer.

Learnable vs. Fixed Threshold. We next evaluate the role of the learnable threshold θ used in forming the affinity graph for superpixel grouping. We compare ASC with a fixed threshold (e.g., $\theta = 0.2$) versus a learnable threshold that is optimized end-to-end. The learnable threshold consistently yields better performance.

Threshold Type	ImageNet Top-1	CIFAR100	VOC07 mIoU
Fixed ($\theta = 0.2$)	80.3	78.4	74.8
Learnable (θ trained)	**82.1**	**79.1**	**76.3**

Table 10. : Impact of fixed vs. learnable threshold.

Graph Traversal Strategy.. To test the effectiveness of our connected-component-based grouping via DFS, we replace it with a simpler token-merging strategy similar to ToMe [3]. The results in Table 11 show that the connected component strategy performs better.

Grouping Mechanism	PASCAL mIoU	NYU Depth rel
Connected Components (DFS)	**76.3**	**0.128**
ToMe-style Merging	74.1	0.137

Table 11. : Comparison of token grouping strategies.

Impact of Component Aggregation Strategy. We also compare mean pooling with alternative strategies such as max pooling and attention-based fusion. As shown in Table 12, mean pooling performs best overall.

Aggregation Method	ImageNet Top-1	Flowers Accuracy
Mean Pooling	**82.1**	**96.6**
Max Pooling	80.8	95.2
Attention Fusion	81.4	96.1

Table 12. : Comparison of token aggregation strategies.

6. CONCLUSION

This work presented Adaptive Superpixel Coding (ASC), a transformer-compatible mechanism that constructs object-centric representations by grouping semantically coherent tokens through an affinity graph and connected-component analysis. Integrated into a self-supervised contrastive learning framework, ASC enables a Vision Transformer to dynamically reorganize spatial information into object-level structures without relying on negative pairs or predefined object queries. Extensive evaluations across image classification, semantic segmentation, object detection, and monocular depth estimation demonstrate that ASC yields transferable representations that outperform strong self-supervised baselines. While ASC provides a flexible and conceptually grounded approach to object-centric grouping, several avenues remain open for future research. First, the learnable gating threshold and grouping policy could be enhanced with more adaptive or context-aware mechanisms that capture object boundaries more robustly. Second, incorporating hierarchical or recurrent grouping may improve temporal and layer-wise consistency, enabling stable object tracking across architectures and video sequences. Third, reducing the computational complexity of similarity graph construction may allow ASC to scale efficiently to higher resolutions and larger transformer models. Finally, integrating ASC with multimodal or generative frameworks represents a promising direction for achieving richer, object-aware scene representations. Overall, ASC establishes a principled foundation for embedding superpixel-inspired grouping within modern transformer architectures and opens new opportunities for more adaptive, interpretable, and object-centric self-supervised learning.

7. REFERENCES

[1] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021.

[2] Thomas Berg and Peter N Belhumeur. Birdsnap: Large-scale fine-grained visual categorization of birds. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2019–2026, 2014.

[3] 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.

[4] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[5] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European Conference on Computer Vision (ECCV)*, pages 446–461. Springer, 2014.

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

[7] 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*, pages 9650–9660, 2021.

[8] Joao Carreira, Eric Noland, Chloe Hillier, and Andrew Zisserman. A short note on the kinetics-700 human action dataset. *arXiv preprint arXiv:1907.06987*, 2019.

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

[10] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[11] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022.

[12] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613, 2014.

[13] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3606–3613, 2014.

[14] P Kingma Diederik. Adam: A method for stochastic optimization. *(No Title)*, 2014.

[15] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

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

[17] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[18] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6824–6835, 2021.

[19] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

[20] Christoph Feichtenhofer, Haoqi Fan, Bo Xiong, Ross Girshick, and Kaiming He. A large-scale study on unsupervised spatiotemporal representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3299–3309, 2021.

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

[22] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913, 2017.

[23] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[24] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[25] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[26] 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*, pages 9729–9738, 2020.

[27] 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*, pages 770–778, 2016.

[28] Shengfeng He, Rynson WH Lau, Wenxi Liu, Zhe Huang, and Qingxiong Yang. Supercnn: A superpixelwise convolutional neural network for salient object detection. *International journal of computer vision*, 115:330–344, 2015.

[29] Yinlin Hu, Rui Song, Yunsong Li, Peng Rao, and Yangli Wang. Highly accurate optical flow estimation on superpixel tree. *Image and Vision Computing*, 52:167–177, 2016.

[30] Thomas Kipf, Gamaleldin F Elsayed, Aravindh Mahendran, Austin Stone, Sara Sabour, Georg Heigold, Rico Jonschkowski, Alexey Dosovitskiy, and Klaus Greff. Conditional object-centric learning from video. *arXiv preprint arXiv:2111.12594*, 2021.

[31] 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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023.

[32] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1920–1929, 2019.

[33] Brett Koonce and Brett Koonce. Resnet 50. *Convolutional neural networks with swift for tensorflow: image recognition and dataset categorization*, pages 63–72, 2021.

[34] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2661–2671, 2019.

[35] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 554–561, 2013.

[36] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical Report TR-2009, University of Toronto, 2009.

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

[38] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[39] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 239–248. IEEE, 2016.

[40] 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.

[41] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[42] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. In *arXiv preprint arXiv:1306.5151*, 2013.

[43] Jieru Mei, Liang-Chieh Chen, Alan Yuille, and Cihang Xie. Spformer: Enhancing vision transformer with superpixel representation. *arXiv preprint arXiv:2401.02931*, 2024.

[44] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8844–8854, 2022.

[45] Lingchen Meng, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser-Nam Lim. Adavit: Adaptive vision transformers for efficient image recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12309–12318, 2022.

[46] Mathew Monfort, Alex Andonian, Bolei Zhou, Kandan Ramakrishnan, Sarah Adel Bargal, Tom Yan, Lisa Brown, Quanfu Fan, Dan Gutfreund, Carl Vondrick, et al. Moments in time dataset: one million videos for event understanding. *IEEE transactions on pattern analysis and machine intelligence*, 42(2):502–508, 2019.

[47] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008.

[48] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[49] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3498–3505, 2012.

[50] 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*, pages 8748–8763. PmLR, 2021.

[51] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. *Advances in neural information processing systems*, 32, 2019.

[52] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12179–12188, 2021.

[53] 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*, 34:13937–13949, 2021.

[54] Ren and Malik. Learning a classification model for segmentation. In *Proceedings ninth IEEE international conference on computer vision*, pages 10–17. IEEE, 2003.

[55] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[56] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

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

[58] Guang Shu, Afshin Dehghan, and Mubarak Shah. Improving an object detector and extracting regions using superpixels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3721–3727, 2013.

[59] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[60] Chau Tran, Duy MH Nguyen, Manh-Duy Nguyen, TrungTin Nguyen, Ngan Le, Pengtao Xie, Daniel Sonntag, James Y Zou, Binh Nguyen, and Mathias Niepert. Accelerating transformers with spectrum-preserving token merging. *Advances in Neural Information Processing Systems*, 37:30772–30810, 2024.

[61] Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In *European conference on computer vision*, pages 108–126. Springer, 2020.

[62] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

[63] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.

[64] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3485–3492. IEEE, 2010.

[65] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3485–3492, 2010.

[66] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[67] 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*, pages 18134–18144, 2022.

[68] Jiarui Xu and Xiaolong Wang. Rethinking self-supervised correspondence learning: A video frame-level similarity perspective. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10075–10085, 2021.

[69] Junjie Yan, Yinan Yu, Xiangyu Zhu, Zhen Lei, and Stan Z Li. Object detection by labeling superpixels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5107–5116, 2015.

[70] Alex Zihao Zhu, Jieru Mei, Siyuan Qiao, Hang Yan, Yukun Zhu, Liang-Chieh Chen, and Henrik Kretzschmar. Superpixel transformers for efficient semantic segmentation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7651–7658. IEEE, 2023.

[71] X Zhu, W Su, L Lu, B Li, X Wang, J Dai, and Deformable DETR. deformable transformers for end-to-end object detection. *URL: https://arxiv. org/abs/2010.04159 (Accessed 29.11. 2023)*, 2020.

APPENDIX

Supplementary Material for Representation Learning with Adaptive Superpixel Coding

A. PROOFS

Let $G = (V, E, \Psi)$ be an undirected graph, where each node $v \in V$ corresponds to a feature embedding in an image, and each edge $e \in E$ connects two embeddings such that $\Psi(e) = \{u, v\}$ for some $u, v \in V$.

A.0.0.1 Definition:

—A **walk** in G is an alternating sequence of vertices and edges:

$$w_0, e_1, w_1, e_2, \ldots, e_k, w_k$$

such that $\Psi(e_i) = \{w_{i-1}, w_i\}$ for all $1 \leq i \leq k$. A walk may contain repeated edges and vertices.

—A **trail** is a walk in which no edge is repeated.

—A **path** is a trail in which no vertex is repeated.

A.0.0.2 Definition of an Object.. An **object** in an image is defined as a maximal set of feature embeddings (vertices in the graph) such that for all x, y in this set, there exists a path in G from x to y, and for any $z \notin$ object, there is no path from x to z. This definition is equivalent to a connected component in graph theory.

A.0.0.3 Proposition 1.. Let $x, y \in V$. Then x and y belong to the same object if and only if there exists a path in G connecting x and y.
Proof of Proposition 1. Let $P(x, y)$ be the proposition: "There exists a path in G connecting x and y." Let $O(x, y)$ be the proposition: "x and y belong to the same object."
We aim to prove:

$$P(x, y) \leftrightarrow O(x, y)$$

($\Rightarrow$) Suppose $P(x, y)$ holds, i.e., there exists a path from x to y. By the definition of an object as a connected subgraph (connected component), any two vertices connected by a path are in the same component. Hence, x and y must belong to the same object. Thus, $P(x, y) \Rightarrow O(x, y)$.
($\Leftarrow$) Suppose $O(x, y)$ holds, i.e., x and y belong to the same object. By definition of object, this implies the existence of a path connecting x and y. Hence, $O(x, y) \Rightarrow P(x, y)$.
Therefore, $P(x, y) \leftrightarrow O(x, y)$. □

A.0.0.4 Proposition 2.. Connectivity is transitive: For any $a, b, c \in V$, if there exists a path from a to b and a path from b to c, then there exists a path from a to c.
Proof of Proposition 2. Let $P(u, v)$ be the proposition: "There exists a path in G connecting u and v."
We are given:

$$P(a, b) \wedge P(b, c)$$

We want to prove:

$$P(a, c)$$

Let $p_1 = (a = v_0, v_1, \ldots, v_k = b)$ be a path from a to b, and $p_2 = (b = u_0, u_1, \ldots, u_\ell = c)$ a path from b to c.
Since paths are sequences of adjacent vertices with no repetition, we can concatenate p_1 and p_2 (excluding the repeated node b) and remove any resulting cycles to construct a new path p_3 from a to c. This process is guaranteed by the fact that any walk can be converted to a path by removing cycles (a well-known result in graph theory).
Hence, $P(a, c)$ holds. Therefore, $P(a, b) \wedge P(b, c) \Rightarrow P(a, c)$, proving transitivity of connectivity. □

A.0.0.5 Conclusion.. By Propositions 1 and 2, the relation "being connected by a path" is an equivalence relation on V (reflexive, symmetric, transitive). Thus, the graph is partitioned into equivalence classes of connected vertices, i.e., connected components. Each connected component corresponds to a unique object in the image. Therefore, finding objects reduces to computing connected components in G.

B. DATASETS

In this work, we use multiple datasets for every experiment We make use of publicly available datasets that are released under MIT License and that are open to all research work.
We use three large-scale video datasets for self-supervised pre-training

B.1 Moments in Time

[46]: A diverse dataset of one million 3-second video clips covering dynamic scenes and actions. We use the official training split containing 802,264 videos across 339 action classes.

B.2 Ego4D

[23]: A large-scale egocentric video dataset capturing real-world, first-person interactions across diverse scenarios. We use 3,670 hours of video from the official training set, excluding any sequences that overlap with benchmark evaluation domains.

B.3 Kinetics-700

[8]: A benchmark dataset consisting of over 650K video clips annotated with human actions. We use the official training split containing 545,317 videos spanning 700 action classes.
We evaluate our method on a diverse collection of publicly available datasets for both linear evaluation and fine-tuning protocols. All datasets are released under permissive licenses (e.g., MIT, CC BY) and are widely used in self-supervised learning literature.

B.4 ImageNet

[55] For the main benchmark, we follow the standard linear evaluation protocol on ImageNet-1K, which consists of 1.28 million training images and 50,000 validation images across 1,000 categories. All models are evaluated using a frozen backbone and a single-layer linear classifier trained on top of the final layer features. Images are resized and center-cropped to 224×224 resolution, consistent with prior work.

B.4.0.1 Downstream Classification Datasets. We further evaluate the generalization of learned representations across 12 diverse classification tasks. For each dataset, a linear classifier or full fine-tuning is applied on top of the pretrained encoder.

—**Food101** [5]: 101 food categories with 101,000 images.

—**CIFAR-10/100** [36]: 10 and 100-class image classification datasets with low-resolution 32×32 images.

—**Birdsnap** [2]: 500 bird species with over 49,000 high-resolution images.

—**SUN397** [65]: Scene recognition dataset with 397 categories and over 100,000 images.

—**Stanford Cars** [35]: Fine-grained car classification with 196 classes.

—**FGVC Aircraft** [42]: Aircraft model recognition with 100 classes.

—**VOC2007** [17]: 20-class object classification, evaluated using classification labels only.

—**DTD** [13]: A dataset of texture images categorized by human-centric attributes.

—**Oxford Pets** [49]: 37 pet categories with annotations for both class and breed.

—**Caltech-101** [19]: 101 object categories and a background class.

—**Oxford Flowers-102** [47]: Flower classification with 102 categories.

All datasets are preprocessed following the protocols established in [24, 9], including standard resizing and normalization. Where applicable, training-validation splits and evaluation metrics follow the respective official implementations or prior literature for fair comparison.

C. METRICS

C.0.0.1 Image Classification (e.g., ImageNet, CIFAR, SUN397).. We evaluate classification performance using **Top-1** and **Top-5 accuracy**:

$$\text{Top-1 Accuracy} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(y_i = \hat{y}_i) \tag{6}$$

$$\text{Top-5 Accuracy} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(y_i \in \text{Top-5}(\hat{p}_i)) \tag{7}$$

where y_i is the ground-truth label, $\hat{y}_i$ is the top-1 predicted label, and $\hat{p}_i$ is the model's class probability vector.

C.0.0.2 Object Detection (PASCAL VOC 2007).. We report **Average Precision at IoU = 0.5** (AP@50):

$$\text{AP}_{50} = \int_0^1 p(r)\, dr \tag{8}$$

where $p(r)$ is the precision at recall r, and predictions are considered correct if:

$$\text{IoU} = \frac{|\text{B}_{\text{pred}} \cap \text{B}_{\text{gt}}|}{|\text{B}_{\text{pred}} \cup \text{B}_{\text{gt}}|} \geq 0.5 \tag{9}$$

C.0.0.3 Semantic Segmentation (PASCAL VOC 2012).. We evaluate segmentation using **mean Intersection over Union (mIoU)**:

$$\text{mIoU} = \frac{1}{C}\sum_{c=1}^{C}\frac{TP_c}{TP_c + FP_c + FN_c} \tag{10}$$

where TP_c, FP_c, and FN_c are the true positives, false positives, and false negatives for class c, and C is the total number of classes.

C.0.0.4 Monocular Depth Estimation (NYU Depth v2).. We use both accuracy and error metrics:

—**Threshold accuracy** at $\delta \in \{1.25, 1.25^2, 1.25^3\}$:

$$\text{Accuracy}_\delta = \frac{1}{N}\sum_{i=1}^{N}\mathbb{1}\left(\max\left(\frac{d_i}{\hat{d}_i}, \frac{\hat{d}_i}{d_i}\right) < \delta\right) \tag{11}$$

—**Root Mean Square Error (RMSE)**:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(d_i - \hat{d}_i\right)^2} \tag{12}$$

—**Relative Error (REL)**:

$$\text{REL} = \frac{1}{N}\sum_{i=1}^{N}\frac{|d_i - \hat{d}_i|}{\hat{d}_i} \tag{13}$$

Here, d_i and $\hat{d}_i$ denote the predicted and ground-truth depth at pixel i, respectively.

D. EVALUATION PROTOCOLS

D.0.0.1 Linear Evaluation on ImageNet.. Our linear evaluation protocol strictly follows established practices in self-supervised learning research:

(1) Pre-trained model weights are frozen (no backpropagation through backbone)
(2) A single linear layer is trained on top of the frozen features
(3) Standard ImageNet augmentation protocol is used (resize to 256px, random crop to 224px)
(4) Training uses SGD with momentum (0.9) for 100 epochs
(5) Learning rate starts at 0.1 and is decayed by a factor of 10 at epochs 60 and 80
(6) Weight decay is set to 0.0001

The reported accuracy is calculated on the official ImageNet validation set of 50,000 images.

D.0.0.2 Transfer Learning Protocol. For fine-tuning on downstream datasets, we use:

—**Optimizer:** SGD with momentum (0.9)
—**Learning rate:** 0.01 for linear probing, 0.0001 for full fine-tuning, with cosine decay
—**Weight decay:** 0.0001
—**Epochs:** 100 for all datasets
—**Early stopping:** Based on validation performance
—**Augmentations:** Random resize, crop, and flip consistent with standard practices

For all transfer learning experiments, we divide each dataset into standard train/val/test splits (or use official splits where available), and report results on the designated test splits.

D.0.0.3 Data Split Information. For each downstream evaluation benchmark, we use the following splits:

—**ImageNet:** 1.28M training images, 50K validation images (used as test set)
—**CIFAR-10/100:** 50K training images, 10K test images
—**VOC2007:** Train/val split (5,011 images), test split (4,952 images)
—**SUN397:** 19,850 training images, 19,850 validation images, 19,850 test images (official split)
—**DTD:** 3,760 training images, 1,880 test images
—**Flowers-102:** 2,040 training images, 6,149 test images

E. EXPERIMENTAL DETAILS

E.0.0.1 Dataset Processing Pipeline.. For all video datasets, we apply the following processing:

(1) Decode videos at native resolution
(2) Sample frames at specified FPS rates (16 FPS for Moments in Time, 15 FPS for Kinetics-700, 5 FPS for Ego4D)
(3) Resize frames to maintain aspect ratio with shorter side = 256 pixels
(4) Apply center crop to obtain 224×224 pixel frames for pre-training

E.0.0.2 Training Setup.. We train ASC using a self-supervised contrastive loss (Equation (5)) applied over 4-frame video subsequences. Each batch consists of 512 video clips. Training is conducted using:

—**Optimizer:** Adam [14] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$
—**Learning rate:** 0.0016, decayed via cosine schedule [41] with 5 epochs of warm-up
—**Weight decay:** 0.05
—**Batch size:** 512
—**Input:** Sub-sequences of 4 consecutive frames
—**Training epochs:** 200 for all models
—**Target encoder update:** Exponential moving average with momentum 0.996

E.0.0.3 Hyperparameter Optimization.. We conduct a grid search over key hyperparameters using 5% of the training data as a validation set. The parameters and their search spaces include:

—Learning rate: [0.0005, 0.001, 0.0016, 0.003, 0.005]
—Weight decay: [0.01, 0.05, 0.1, 0.2]
—Similarity threshold θ: [0.05, 0.1, 0.2, 0.5]
—Batch size: [1024, 2048, 4096]

The optimal values were selected based on validation loss convergence and downstream performance on a small subset of ImageNet classes.

E.0.0.4 Compute Resources.. Experiments were conducted on 8× NVIDIA A100 GPUs (40GB memory each). Training on the combined dataset takes approximately 24 hours for ViT-Tiny and 2.5 days for ViT-B. The total compute for all reported experiments is estimated at approximately 200 GPU-hours, aligning with the computational budget of comparable prior work.

F. DOWNSTREAM TASK ARCHITECTURES AND PROTOCOLS

F.1 Semantic Segmentation

For semantic segmentation, we evaluate on the PASCAL VOC 2012 dataset using the standard split provided by `augmented_voc`. The dataset contains 21 classes (including background).

F.1.0.1 Architecture.. Our segmentation architecture adapts the ViT backbone to dense prediction through a carefully designed decoder structure. The decoder follows a Feature Pyramid Network (FPN) design that leverages multi-scale features from intermediate transformer layers:

—**Feature extraction:** We extract features from transformer layers 3, 6, 9, and 12, providing hierarchical representations at different semantic levels.
—**Feature normalization:** Each feature set is processed through a 1×1 convolutional layer to unify channel dimensions to 256.
—**Upsampling pathway:** Starting from the deepest layer, features are progressively upsampled using bilinear interpolation and merged with corresponding shallower features through lateral connections.
—**Refinement:** Each merged feature map undergoes a 3×3 convolutional refinement step with batch normalization and ReLU activation.
—**Prediction head:** The final feature map is processed by a 1×1 convolution that outputs logits for the 21 semantic classes.

F.1.0.2 Training Details.. Our ViT backbone is initialized with weights pretrained via ASC and fine-tuned end-to-end with the segmentation decoder. Input images are resized to 512×512 pixels.
Training is conducted using the AdamW optimizer with an initial learning rate of 5×10^{-5}, cosine annealing scheduler, and a batch size of 16. We employ a combination of cross-entropy and Lovász-Softmax loss functions with weights of 1.0 and 0.5 respectively. We fine-tune for 20k iterations with a linear warmup for the first 1500 iterations. Data augmentation includes random scaling (0.5-2.0), random cropping, horizontal flipping, and color jittering. Evaluation is done using the mean Intersection-over-Union (mIoU) metric.

F.2 Object Detection

We follow the object detection protocol on PASCAL VOC 2007 using Faster R-CNN as our detection framework.

F.2.0.1 Architecture.. Since standard detection frameworks are designed for convolutional backbones, we integrate our pretrained ViT with Faster R-CNN through a specialized feature adapter:

—**Token reshaping:** The final layer patch tokens from the ViT (N tokens with dimension D) are reshaped into a 2D spatial grid with dimensions $H \times W \times D$, where $H \times W = N$.

—**Feature adapter:** This grid is processed through a small convolutional network consisting of:
  —A 1×1 convolution that maps D dimensions to 512 channels
  —Two residual blocks, each with two 3×3 convolutions and skip connections
  —A final 1×1 convolution that maps to the expected backbone output channels (256 for C4 Faster R-CNN)

—**Multi-scale features:** For improved detection of objects at different scales, we extract features from intermediate transformer layers (3, 6, 9) and process them through similar adapter modules before feeding them into the FPN structure of Faster R-CNN.

—**Detection head:** We use the standard Faster R-CNN detection head with RPN, RoI Pooling, and classification/bounding box regression heads.

F.2.0.2 Training Details.. The detector is trained on the `trainval2007` split and evaluated on `test2007`. We use stochastic gradient descent with momentum (0.9), an initial learning rate of 1×10^{-3} with step decay at iterations 18k and 22k, weight decay of 1×10^{-4}, and train for 24k iterations with a batch size of 8.
We implement gradient clipping at a max norm of 10.0 to stabilize training. Standard detection augmentations include horizontal flipping, scale jittering, and color augmentation. Results are reported using AP_{50} following standard PASCAL VOC evaluation protocol.

F.3 Monocular Depth Estimation

We evaluate monocular depth prediction on the NYU Depth v2 dataset, which consists of RGB-D indoor scenes.

F.3.0.1 Architecture.. Our depth estimation approach is inspired by Dense Prediction Transformers (DPT) [52], which is specifically designed to leverage transformer architectures for dense prediction tasks. The architecture consists of:

—**Feature extraction:** We extract token embeddings from transformer layers 3, 6, 9, and 12, providing a hierarchical representation with varying receptive fields.

—**Reassembly blocks:** For each feature level, we use:
  —A cross-attention mechanism where learnable query embeddings (arranged in a 2D grid) attend to the transformer tokens
  —This effectively projects global features back into a spatially structured representation
  —Each query embedding corresponds to a spatial location in the final prediction

—**Fusion blocks:** Features from different levels are progressively fused using:
  —Residual connections to preserve information flow
  —A combination of convolutions (3×3) for local context and self-attention for global context
  —Layer normalization between operations

—**Depth head:** The final fused features are processed by a lightweight MLP that outputs per-pixel depth values.

This architecture better leverages the transformer's inherent global context modeling while maintaining spatial coherence through the structured query embeddings. Unlike conventional CNN-based upsampling decoders, it preserves the transformer's ability to model long-range dependencies throughout the decoding process.

F.3.0.2 Training Details.. We train on the official NYU Depth v2 dataset using an input resolution of 640×480 pixels. The model is optimized using:

—Adam optimizer with learning rate 1×10^{-4}

—Cosine learning rate schedule with 2000-step warmup

—Batch size of 8

—Training for 25 epochs

—Loss function combining L1 (weight 0.5), scale-invariant (weight 1.0), and edge-aware gradient loss (weight 0.5)

—Data augmentation including random horizontal flipping, color jittering (brightness, contrast, saturation), and random cropping to 416×352 pixels

Our edge-aware gradient loss specifically helps preserve structural boundaries by penalizing depth discontinuities that don't align with RGB edges, which is particularly beneficial for transformer-based models that might otherwise produce overly smooth depth maps.
Evaluation metrics include relative error (rel), root mean squared error (rms), and the percentage of predicted depths within thresholds of 1.25, 1.25^2, and 1.25^3 compared to ground truth, following standard protocols in depth estimation literature.

G. LIMITATIONS

We identify several limitations of the Adaptive Superpixel Coding that may guide future research:

G.0.0.1 Hyperparameter sensitivity.. The grouping behavior is influenced by the learnable gating threshold θ, which determines the sparsity of the similarity graph. While effective empirically, its interpretability and generalization across datasets remain under-explored.

G.0.0.2 Merge policy and representation bias.. Connected components are merged using mean pooling, which assumes uniformity within each group. This may suppress distinctive or minority features within an object, especially near boundaries or in regions with semantic heterogeneity. More expressive aggregation mechanisms may yield richer object-level embeddings.

G.0.0.3 Computational overhead.. The module introduces additional complexity per layer via $\mathcal{O}(N^2 d)$ similarity computation and DFS traversal. While the reduced token count in subsequent layers may compensate for this overhead, the overall trade-off remains unquantified and warrants empirical runtime analysis.

G.0.0.4 Object granularity.. The notion of an object is inferred solely from local pairwise feature similarities, which can lead to over- or under-grouping in cluttered scenes or under occlusion. Without global context or top-down constraints, the grouping may not align with semantically meaningful object boundaries.

G.0.0.5 Layer-wise inconsistency.. Grouping is performed independently at each layer, with no mechanism to enforce consistency or track object identity across layers. This can result in unstable representations, particularly in deep architectures. Incorporating recurrent or hierarchical grouping could improve temporal coherence.

G.0.0.6 Symmetry and object independence.. The module does not explicitly enforce object symmetry or viewpoint invariance. As a result, the same object under different poses, orientations, or occlusions may be fragmented into separate groups. Achieving object-level invariance in grouping remains a key challenge for generalizing to complex real-world scenes.