# Unsupervised Dense Prediction using Differentiable Normalized Cuts

Yanbin Liu[1] and Stephen Gould[2]

[1] Auckland University of Technology
[2] Australian National University
yanbin.liu@aut.ac.nz, stephen.gould@anu.edu.au

**Abstract.** With the emergent attentive property of self-supervised Vision Transformer (ViT), Normalized Cuts (NCut) has resurfaced as a powerful tool for unsupervised dense prediction. However, the pre-trained ViT backbone (*e.g.*, DINO) is frozen in existing methods, which makes the feature extractor suboptimal for dense prediction tasks. In this paper, we propose using Differentiable Normalized Cuts for self-supervised dense feature learning that can improve the dense prediction capability of existing pre-trained models. First, we review an efficient gradient formulation for the classical NCut algorithm. This formulation only leverages matrices computed and stored in the forward pass, making the backward pass highly efficient. Second, with NCut gradients in hand, we design a self-supervised dense feature learning architecture to finetune pre-trained models. Given two random augmented crops of an image, the architecture performs RoIAlign and NCut to generate two foreground masks of their overlapping region. Last, we propose a mask-consistency loss to back-propagate through NCut and RoIAlign for model training. Experiments show that our framework generalizes to various pre-training methods (DINO, MoCo and MAE), network configurations (ResNet, ViT-S and ViT-B), and tasks (unsupervised saliency detection, object discovery and semantic segmentation). Moreover, we achieved state-of-the-art results on unsupervised dense prediction benchmarks.

## 1 Introduction

Dense prediction aims at understanding detailed aspects of an image, encompassing tasks such as object detection [52], segmentation [26, 37], and correspondence [18, 29]. Modern approaches rely on supervised learning that requires high-quality annotations on bounding boxes, segmentation masks and keypoints, which are laborious to acquire. Moreover, supervised learning is trained on a predefined set of classes, limiting their application in real-world scenarios [50].

In this paper, we focus on unsupervised dense prediction with no human annotation. Recent advances deal with this problem building on two standard techniques: self-supervised Vision Transformer (ViT) [7] and Normalized Cuts [32].

First, distillation with no labels (DINO) [7] found that self-supervised ViT contains sensible object boundaries in the self-attention of the `[CLS]` token in last

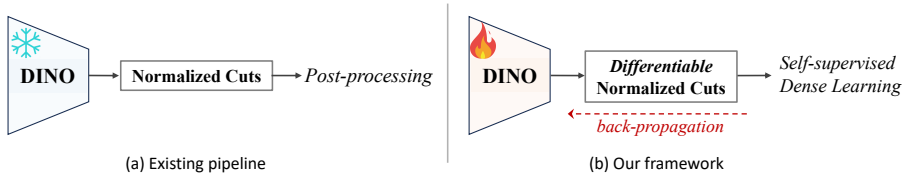


**Fig. 1:** (a) Existing methods [28, 45] feed DINO [7] (*frozen*) features to classical Normalized Cuts algorithm [32], and apply post-processing. (b) We propose using *Differentiable* Normalized Cuts that can back-propagate the gradient from the Self-supervised Dense Learning loss to finetune DINO for better dense prediction.

layer. Motivated by this emergent property, LOST [35] adopted the *key* features of the last attention layer to build a patch similarity graph for unsupervised object discovery. Following LOST, the same features were utilized by DSS [28] and TokenCut [45] to solve more dense prediction tasks such as unsupervised saliency detection and unsupervised video segmentation. Pre-trained DINO has become the default feature extractor for dense prediction [5, 31, 34, 36].

Second, instead of directly using self-attention from DINO, TokenCut [45] and DSS [28] discovered that applying the classical Normalized Cut [32] algorithm on DINO features achieved state-of-the-art performance on unsupervised dense prediction tasks. Specifically, an image graph was constructed from the *key* features of DINO. Then, image segmentation was formulated as a minimum graph cut problem. According to Normalized Cuts, the solution (*i.e.*, segmentation mask) is the *second smallest eigenvector* of the Laplacian matrix of the image graph. Based on this segmentation mask, further post-processing procedures [2, 36, 45] can be applied to obtain a refined mask.

Combining the above two techniques, the existing pipeline for dense prediction is shown in Fig. 1(a). Existing methods directly take the pre-trained DINO model and freeze its weights, thus limiting the capacity for dense prediction. DINO is trained on the global feature level, so it is not aligned with dense prediction tasks requiring pixel-level understanding. In this paper, we try to bridge the gap between the pre-trained self-supervised model (*e.g.*, DINO) and dense feature representation. Our framework is shown in Fig. 1(b), including two components: Differentiable Normalized Cuts and Self-supervised Dense Learning.

Normalized Cuts [32] involves an optimization problem of the generalized eigenvalue system, so it is not straightforward to back-propagate gradients from the solution (*i.e.*, segmentation mask) to the input (*i.e.*, image features). We propose to use a *Differentiable Normalized Cuts* layer, which wraps up Normalized Cuts in the forward pass and derives an efficient formulation for the backward pass. Our approach starts with the traditional differential of an eigenvector according to Magnus *et al.* [27], which contains a time-consuming pseudo-inverse operation. To improve the running time we use eigendecomposition [1] to evaluate the pseudo-inverse using only matrices computed and stored in the forward

pass. In this way, the Differentiable Normalized Cuts layer can be plugged into deep networks for efficient and effective training.

For Self-supervised Dense Learning, we design an effective architecture to utilize the foreground masks from Normalized Cuts, which have not been explored previously in self-supervised learning. Specifically, we take two random augmented crops of an image and feed them to a shared encoder for feature extraction. The extracted features then undergo RoIAlign and Normalized Cuts to generate foreground masks for the overlapping regions of the two crops. A mask-consistency loss is calculated on the overlapping masks, and gradients are back-propagated through RoIAlign and, importantly, Normalized Cuts to update the encoder weights. To combat the issue of collapse in self-supervised learning, we enforce structural regularization tailored for dense feature learning.

Starting from a pre-trained model, our framework only needs to fine-tune for *two epochs* on ImageNet [12], to unlock the dense prediction capability. In the experiments, we have verified the effectiveness of our framework for dense prediction on several diverse tasks (unsupervised saliency detection, object discovery and semantic segmentation), pre-training methods (DINO [7], MAE [19] and MoCoV3 [9]), and network architectures (ResNet [22], ViT-S and ViT-B [13]).

The contribution of our paper is summarized as follows:

- We introduce *Differentiable Normalized Cuts* to the context of unsupervised dense prediction. It enables efficient back-propagation through Normalized Cuts with matrices necessarily computed in the forward pass.
- We design a Self-supervised Dense Learning architecture to improve the dense prediction capability of the pre-trained self-supervised ViT model.
- Our framework generalizes to diverse dense prediction tasks, pre-training methods and network architectures. The state-of-the-art results are achieved on unsupervised prediction benchmarks.

## 2 Related Work

**Self-supervised and Dense Contrastive Learning.** The recent success of self-supervised learning adopts a contrastive learning scheme: two random augmented views of the same image are regarded as positive pairs. A large body of work was proposed, such as SimCLR [8], BYOL [17], MoCo [9,20], SimSiam [11], DINO [7], and SwAV [6] to name a few.

However, these methods learn at the global feature vector level, which is not optimal for dense prediction tasks. Therefore, dense contrastive learning was proposed to learn at pixel-level [44, 47] or region-level [46, 49]. For example, DenseCL [44] extended MoCo-v2 [10] to perform dense pairwise contrastive learning at the pixel level. DenseSiam [49] proposed Dense Siamense Network and leveraged both pixel consistency and region consistency for dense feature learning. Instead of operating on pixel- or region-level, our method utilizes the intrinsic structure of an image (*i.e.*, segmentation mask from Normalized Cuts [32]) and proposes an efficient formulation to back-propagate Normalized Cuts.

**Normalized Cuts and Unsupervised Dense Prediction.** Normalized Cuts (NCut) [3, 23, 32] is a traditional segmentation method, which reframes image segmentation as a graph partitioning problem solved via an eigenvalue system. Based on the unsupervised object attention from self-supervised ViT [7] (*e.g.*, DINO), recent works [28,32] unleashed the strong performance of NCut for dense prediction tasks, including unsupervised segmentation and localization. Their pipeline is shown in Fig. 1(a): they employ a pre-trained DINO as the feature extractor and apply NCut to generate the initial segmentation mask, which is then post-processed using different strategies.

Other dense prediction methods [5,31,36] follow the same pipeline as Fig. 1(a), but they replace NCut by training a post-processing module on an extra DUTS-TR [40] dataset. For example, `FOUND` [36] adopted a *frozen* DINO to discover the background from a selected seed, and train a lightweight $1 \times 1$ convolution layer to refine the DINO dense features. MOVE [5] used a *frozen* DINO as the segmenter and a *frozen* Masked AutoEncoder (MAE) [19] as the inpainter, and then performed adversarial training on inpainted images. Different from existing methods, which might be limited by the frozen DINO, we devise a general framework to improve the dense prediction capability of pre-trained models. Our framework requires only a few epochs of training on ImageNet.

**Backpropagation through Normalized Cuts.** Backpropagation requires differentiating the eigendecomposition (ED) or singular value decomposition (SVD) problem. Several works have been proposed to solve the problem [23,27,30,41,42]. An early work by Magnus *et al.* [27] derived the differential of eigenvalues involving a pseudo-inverse calculation. Then, Papadopoulo *et al.* [30] proposed to estimate the Jacobian of the SVD using an exact analytic technique. Ionescu *et al.* [23] studied matrix backpropagation and derived partial derivatives for both SVD and symmetric ED problems. To improve the numerical stability, Wang *et al.* [41] proposed a hybrid strategy for differentiable ED: utilize SVD during the forward pass and derive the gradients from the Power Iteration (PI). This was further improved in later work [42] by using Taylor expansion to replace PI during the backward gradient derivation. Different from existing works, we focus on the application of unsupervised dense prediction, which has not been explored with differentiable NCut. Instead of using supervised loss in prior works [23, 24, 41], we proposed a self-supervised loss to compute the gradients, entailing practical designs such as stop gradient and asymmetric architecture.

## 3 Methodology

### 3.1 Background

**Self-supervised ViT.** While the original ViT model [13] was trained with image labels, DINO [7] found that segmentation masks emerge after self-distillation training with no labels. Specifically, object masks appear in the self-attention of the last layer `[CLS]` token. This emergent property has been utilized by recent unsupervised object segmentation and localization methods [28,35,45]. However, instead of using self-attention, they adopt the *key* features from the last layer.

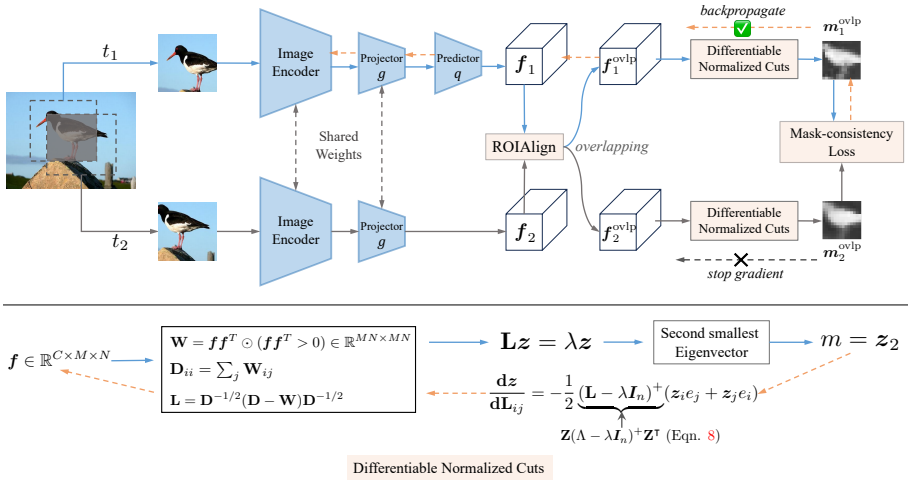


**Fig. 2: Overview of our Self-supervised Dense Learning framework**. (Above) At first, two random crops of an image are augmented (with $t_1, t_2$) and fed into a shared Image Encoder (*e.g.*, DINO [7]), followed by asymmetric Projector $g$ and Predictor $q$ to extract dense features $\boldsymbol{f}_1, \boldsymbol{f}_2$. Then, we use RoIAlign [21] to get the features of overlapping areas, which are then input to the proposed *Differentiable Normalized Cuts* Layer. Normalized Cuts [32] algorithm generates the foreground masks $\boldsymbol{m}_1^{\text{ovlp}}, \boldsymbol{m}_2^{\text{ovlp}}$. Finally, a mask-consistency loss is used to train the model. (Bellow) Detailed forward and backward computation of the *Differentaible Normalized Cuts* Layer.

Formally, given an image $\mathbf{I} \in \mathbb{R}^{3\times M\times N}$, the *key* feature $\boldsymbol{f} \in \mathbb{R}^{C\times M/P\times N/P}$ is extracted from a ViT encoder $\Phi_\theta$, where $C$ is the feature dimension and $P$ is the downsampling factor. An affinity matrix is constructed as

$$\mathbf{W} = \max\{\boldsymbol{f}\boldsymbol{f}^\intercal, 0\} \in \mathbb{R}^{\frac{MN}{P^2}\times\frac{MN}{P^2}}, \tag{1}$$

where max is applied elementwise. Then, $\mathbf{W}$ can be used as the adjacency matrix in Normalized Cuts [32] to generate the foreground mask.

**Normalized Cuts (NCut).** Shi and Malik [32] framed image segmentation as a graph cut problem. For an image, the graph $G = (V, E)$ is constructed with the adjacency matrix $\mathbf{W} = \{w(u, v) : (u, v) \in E\}$. Then, image segmentation translates to a graph partition problem: removing edges to divide $V$ into two disjoint sets, $A, B, A \cup B = V, A \cap B = \emptyset$. The total weight of removed edges is called the *cut*:

$$cut(A, B) = \sum_{u\in A, v\in B} w(u, v)\,. \tag{2}$$

And normalized cut ($Ncut$) is adopted for graph partition:

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)}\,, \tag{3}$$

where $assoc(A, V) = \sum_{u \in A, t \in V} w(u, t)$ is the total weight of edges connecting nodes in $A$ to all graph nodes.

Finding the optimal bipartition of the graph, *i.e.* normalized cut, is then equivalent to minimizing the $Ncut(A, B)$ measure.

### 3.2 Differentiable Normalized Cuts

**Minimizing NCut [32].** Let $\boldsymbol{x} = \{1, -1\}^n$ ($n = |V|$) be a solution for minimizing $Ncut(A, B)$, where $\boldsymbol{x}_i = 1$ if node $i$ is in $A$ and $-1$ otherwise. And $\mathbf{D}$ is a diagonal matrix with $\mathbf{D}_{ii} = \sum_j \mathbf{W}_{ij}$. Introduce two variables $b = \frac{\sum_{\boldsymbol{x}_i > 0} \mathbf{D}_{ii}}{\sum_{\boldsymbol{x}_i < 0} \mathbf{D}_{ii}}$ and $\boldsymbol{y} = (1 + \boldsymbol{x}) - b(1 - \boldsymbol{x})$, then according to [32], the problem can be expressed as minimizing the Rayleigh quotient [16]:

$$\min_{\boldsymbol{y}} \frac{\boldsymbol{y}^\intercal (\mathbf{D} - \mathbf{W}) \boldsymbol{y}}{\boldsymbol{y}^\intercal \mathbf{D} \boldsymbol{y}}\,, \tag{4}$$

with the condition $\boldsymbol{y}(i) \in \{1, -b\}$ and $\boldsymbol{y}^\intercal \mathbf{D}\mathbf{1} = 0$. Since Eqn. 4 is NP-complete, Shi and Malik [32] relax $\boldsymbol{y}$ to take on real values and to obtain the generalized eigenvalue system:

$$(\mathbf{D} - \mathbf{W})\boldsymbol{y} = \lambda \mathbf{D} \boldsymbol{y}\,. \tag{5}$$

Let $\boldsymbol{z} = \mathbf{D}^{1/2}\boldsymbol{y}$, Eqn. 5 can be rewritten as

$$\mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{W})\mathbf{D}^{-1/2}\boldsymbol{z} = \lambda \boldsymbol{z}\,. \tag{6}$$

Obviously, $\lambda_1 = 0$ is the smallest eigenvalue of Eqn. 5 with corresponding eigenvector $\boldsymbol{y}_1 = \mathbf{1}$. As $\boldsymbol{y}_1$ does not have useful information, existing works [28, 45] adopted the *second smallest eigenvector* $\boldsymbol{y}_2$ or $\boldsymbol{z}_2$ for segmentation.

**Differentiating NCut.** In general, the eigenvalue system (Eqn. 5 or 6) does not have a closed-form solution. Thus, given the second smallest eigenvector $\boldsymbol{z}_2$, we cannot directly calculate the gradient of $\boldsymbol{z}_2$ w.r.t. feature $\boldsymbol{f}$ (Eqn. 1) to update the parameters of the ViT encoder $\Phi$. However, we can rely on the following theorem to calculate the gradient.

**Theorem 1.** *(From Magnus et al. [27]) Let $\mathbf{X} \in \mathbb{R}^{n \times n}$ be a real symmetric matrix and $\boldsymbol{z}$ be a normalized eigenvector associated with a simple eigenvalue*[3] *$\lambda$ of $\mathbf{X}$, i.e., $\mathbf{X}\boldsymbol{z} = \lambda \boldsymbol{z}$ and $\boldsymbol{z}^\intercal \boldsymbol{z} = 1$. Then, the differential $\mathbf{d}\boldsymbol{z} = (\lambda \boldsymbol{I}_n - \mathbf{X})^+ (\mathbf{d}\mathbf{X})\boldsymbol{z}$, where $\boldsymbol{I}_n$ is an identity matrix and $^+$ denotes pseudo-inverse.*

Using Theorem 1, we can calculate the gradient of an eigenvector $\boldsymbol{z}$ (with simple eigenvalue) w.r.t. the normalized Laplacian matrix $\mathbf{L} = \mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{W})\mathbf{D}^{-1/2}$ in Eqn. 6:

$$\frac{\mathbf{d}\boldsymbol{z}}{\mathbf{d}\mathbf{L}_{ij}} = -\frac{1}{2}(\mathbf{L} - \lambda \boldsymbol{I}_n)^+ (\boldsymbol{z}_i e_j + \boldsymbol{z}_j e_i)\,, \tag{7}$$

[3] A simple eigenvalue is an eigenvalue with an algebraic multiplicity of one, implying a unique associated eigenvector.

where $e_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ with 1 at the $i$-th position.

Directly computing the pseudo-inverse requires $O(n^3)$ complexity, which is time-consuming. So, we proposed a faster alternative using eigendecomposition [1]. Specifically, $\mathbf{L}$ can be decomposed as $\mathbf{L} = \mathbf{Z}\Lambda\mathbf{Z}^\intercal$, where $\mathbf{Z}$ is an orthogonal matrix with eigenvectors at the columns and $\Lambda$ is a diagonal matrix with eigenvalues at the diagonal. Thus,

$$(\mathbf{L} - \lambda \boldsymbol{I}_n)^+ = \mathbf{Z}(\Lambda - \lambda \boldsymbol{I}_n)^+ \mathbf{Z}^\intercal \,. \tag{8}$$

$(\Lambda - \lambda \boldsymbol{I}_n)^+$ is diagonal and can be efficiently calculated by inverting non-zero elements of $(\Lambda - \lambda \boldsymbol{I}_n)$. Moreover, the elements of $\mathbf{Z}$ and $\Lambda$ are already obtained when solving Eqn. 6, without incurring extra overload.

With Eqn. 8, we can efficiently calculate the gradient $\frac{\mathbf{d}\boldsymbol{z}}{\mathbf{d}\mathbf{L}}$ through the NCut algorithm, and thus back-propagate through the affinity matrix $\mathbf{W}$ and features $\boldsymbol{f}$, making the encoder $\Phi$ end-to-end trainable. Finally, we encapsulate the forward (Eqn. 6) and backward (Eqn. 7 and 8) calculations into a *Differentiable Normalized Cuts* layer—a fast plug-and-play layer for including NCut in deep networks.[4]

### 3.3 Self-supervised Dense Learning Framework

Most existing dense representation learning methods [44,47,49] extend instance-level contrastive learning [8,17] to the pixel- or region-level. However, inherent image structures (*e.g.*, foreground mask from NCut) that provide richer information than pixels have not been explored. One possible reason is the difficulty of backpropagating through such structures. However, with the Differentiable Normalized Cuts layer, we can develop a *Self-supervised Dense Learning* framework that is end-to-end trainable. The overall framework (shown in Fig. 2) shares a similar two-branch structure with recent contrastive methods [11,17], but with a focus on dense representation learning.

**Dense Feature Representation.** Given an image $\mathbf{I} \in \mathbb{R}^{3\times M\times N}$, we take two random crops and apply different augmentations $t_1, t_2$ to generate two views $\mathbf{I}_1 = t_1(\mathbf{I}), \mathbf{I}_2 = t_2(\mathbf{I})$. Then, the two cropped views are input to a shared image encoder $\Phi_\theta$ to extract dense features. We follow BYOL [17] to introduce a Projector $g$ and a Predictor $q$ ($g$ and $q$ are 2D ConvNets) to perform asymmetric feature transformation. Then, transformed features are $\boldsymbol{f}_1 = g(q(\Phi_\theta(\mathbf{I}_1)))$, $\boldsymbol{f}_2 = g(\Phi_\theta(\mathbf{I}_2)) \in \mathbb{R}^{C\times M/P\times N/P}$. Instance-level contrastive learning methods (*e.g.*, DINO [7] and BYOL [17]) only care about the global semantics and make $(\boldsymbol{f}_1, \boldsymbol{f}_2)$ as positive pairs. In contrast, for dense feature learning, we care about the spatial information and structures lying in two views. Therefore, we apply the RoIAlign [21] operation to the two views and transform them to the same overlapping area in the original image as: $\boldsymbol{f}_1^{\text{ovlp}}, \boldsymbol{f}_2^{\text{ovlp}} = \text{RoIAlign}(\boldsymbol{f}_1, \boldsymbol{f}_2, t_1, t_2)$.

[4] While deep learning software libraries, such as PyTorch, implement a differentiable eigendecomposition API, we found experimentally that our approach results in more stable learning.

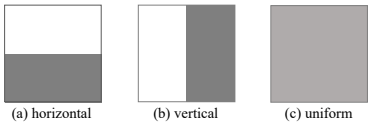


**Fig. 3:** Three mask templates for structural regularization.

**Mask Consistency Learning.** Given $\boldsymbol{f}_1^{\text{ovlp}}, \boldsymbol{f}_2^{\text{ovlp}}$, a straightforward training strategy is to enforce pixel-to-pixel similarities from the same position, as done in [44, 47]. While simple, this strategy treats each pixel individually and ignores any structure information, such as edges and object segmentation. Furthermore, given that leading unsupervised dense prediction techniques, exemplified by TokenCut [45], incorporate Normalized Cuts in their post-processing stages, the pixel-based strategy falls short in maximizing potential benefits.

To align with downstream dense prediction tasks, we use foreground masks generated by the NCut algorithm and assume *the overlapping areas should generate consistent masks*. So we input $\boldsymbol{f}_1^{\text{ovlp}}, \boldsymbol{f}_2^{\text{ovlp}}$ into the Differentiable Normalized Cuts layer to obtain two separate segmentation masks $\boldsymbol{m}_1^{\text{ovlp}}, \boldsymbol{m}_2^{\text{ovlp}} \in \mathbb{R}^{M/P \times N/P}$ (*i.e.*, second smallest eigenvector of Eqn. 6). Then we adopt binary cross-entropy loss $\mathcal{L}_{bce}$ to evaluate the mask consistency. Given $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}_+^n$, $\mathcal{L}_{bce}$ is defined as

$$\mathcal{L}_{bce}(\boldsymbol{a}, \boldsymbol{b}) = -\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{a}_i \log \boldsymbol{b}_i + (1 - \boldsymbol{a}_i) \log(1 - \boldsymbol{b}_i)\,. \tag{9}$$

We can now define the mask-consistency loss as

$$\mathcal{L}_{mask} = \mathcal{L}_{bce}(\boldsymbol{m}_1^{\text{ovlp}}, \texttt{stop\_grad}(\boldsymbol{m}_2^{\text{ovlp}})). \tag{10}$$

Here, `stop_grad` denotes the stop gradient operation. We also swap the dense features before Projector to obtain a symmetric version of $\mathcal{L}_{mask}$. Different from DINO [7], we do not need a momentum encoder [20] as the teacher network [7], which saves GPU memory during training.

With $\mathcal{L}_{mask}$ and Differentiable Normalized Cuts, we can now back-propagate gradients through NCut algorithm and RoIAlign to update the parameters of the shared encoder $\Phi$.

### 3.4 Structural Regularization to Avoid Collapse

For instance-level contrastive learning, there are two common collapse patterns [7]: one-hot feature distribution and uniform feature distribution. However, our dense learning framework operates on 2D masks $\boldsymbol{m}_1^{\text{ovlp}}, \boldsymbol{m}_2^{\text{ovlp}}$ instead of 1D feature vectors. Hence, the collapse patterns of masks might also differ from feature vectors.

In early experiments, we observed three mask collapse patterns: horizontal, vertical and uniform. The first two are specific to our architecture and mask-consistency loss, while the uniform pattern is similar to instance-level contrastive

learning. Motivated by these observations, we design three mask templates $\mathcal{M} = \{\boldsymbol{m}^{\mathrm{h}}, \boldsymbol{m}^{\mathrm{v}}, \boldsymbol{m}^{\mathrm{u}}\}$ shown in Fig. 3 as a simple mechanism to mitigate against collapse. Given the three masks, we design a regularization loss as follows:

$$\mathcal{L}_{reg} = -\sum_{\boldsymbol{m}\in\mathcal{M}} \mathcal{L}_{bce}(\boldsymbol{m}_1^{\mathrm{ovlp}}, \boldsymbol{m})\,. \tag{11}$$

We minimize $\mathcal{L}_{reg}$ to avoid collapse and add it to the mask-consistency loss. The final loss is

$$\mathcal{L} = \mathcal{L}_{mask} + \alpha\mathcal{L}_{reg}\,, \tag{12}$$

where $\alpha$ is the regularization factor.

## 4 Experiments

We evaluate Differentiable Normalized Cuts on three dense prediction tasks: unsupervised saliency detection (Sec. 4.1), unsupervised object discovery (Sec. 4.2) and unsupervised semantic segmentation (Sec. 4.3). Moreover, our method is verified to generalize beyond DION and ViT in Sec. 4.4. We perform ablation studies in Sec. 4.5.

**Training details.** Existing methods freeze an ImageNet pre-trained DINO [7] (ViT-s/16 or ViT-s/8) as the feature extractor and train post-processing modules on an extra DUTS-TR [40] (10,553 images) dataset. In contrast, we **finetune DINO only on ImageNet to obtain a general dense feature extractor**. Specifically, we take two random crops from an image with a range $[0.2, 1.0]$ and resize them to $(224, 224)$. Then, we apply different augmentations $t_1, t_2$ following DINO [7]. We train ViT-s/16 (ViT-s/8) for two epochs with a learning rate of 0.0005 and a batch size of 256 (64). We set $\alpha$ to 0.05 in Eqn. 12 to avoid collapse.

After training, we use our ImageNet-trained model to replace DINO as the feature extractor for dense prediction tasks. When combined with different task-specific methods, we demonstrate that our model is a better plug-and-play feature extractor than DINO.

### 4.1 Unsupervised Saliency Detection

**Datasets.** We evaluate our method on three benchmarks: DUT-OMRON [48] (5,168 images), DUTS-TE [40] (5,019 images) and ECSSD [33] (1,000 images).
**Evaluation metric.** We report results on three metrics: per-pixel mask accuracy (Acc), intersection over union (IoU), and max$F_\beta$. Acc is the percentage of correctly predicted foreground/background pixels. IoU measures the overlap between the binary predicted mask and the ground truth mask. Following previous works [28,36,45], $F_\beta = \frac{(1+\beta^2)\mathrm{Precision}\times\mathrm{Recall}}{\beta^2\mathrm{Precision}+\mathrm{Recall}}$ with $\beta = 0.3$, and max$F_\beta$ is reported as the maximum value of 255 uniformly distributed thresholds. We also reported the results of applying bilateral solver [4] post-processing for mask refinement.
**Comparison Results.** Since our focus is to verify the feature capability for dense prediction, we directly apply our trained model to three state-of-the-art

| Method | Train data | DUT-OMRON | | | DUTS-TE | | | ECSSD | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Acc | IoU | max$F_\beta$ | Acc | IoU | max$F_\beta$ | Acc | IoU | max$F_\beta$ |
| — ***Without post-processing bilateral solver*** — | | | | | | | | | | |
| LOST [35] | ImNet | 79.7 | 41.0 | 47.3 | 87.1 | 51.8 | 61.1 | 89.5 | 65.4 | 75.8 |
| DSS [28] | ImNet | – | 56.7 | – | – | 51.4 | – | – | 73.3 | – |
| TokenCut [45] | ImNet | 88.0 | 53.3 | 60.0 | 90.3 | 57.6 | 67.2 | 91.8 | 71.2 | 80.3 |
| **Ours** + TokenCut | ImNet | **89.5** | **55.7** | **62.8** | **91.5** | **59.8** | **69.8** | **92.9** | **74.2** | **82.8** |
| FreeSOLO [43] | ImNet, COCO | 90.9 | 56.0 | 68.4 | 92.4 | 61.3 | 75.0 | 91.7 | 70.3 | 85.8 |
| FOUND [36] | ImNet, DUTS-TR | 91.2 | 57.8 | 66.3 | 93.8 | 64.5 | 71.5 | 94.9 | 80.7 | 95.5 |
| MOVE [5] | ImNet, DUTS-TR | 92.3 | 61.5 | 71.2 | 95.0 | 71.3 | 81.5 | 95.4 | 83.0 | 91.6 |
| **Ours** + FOUND | ImNet, DUTS-TR | 91.7 | 61.2 | 71.6 | 94.1 | 66.7 | 77.8 | **95.7** | **83.3** | **95.6** |
| **Ours** + MOVE | ImNet, DUTS-TR | **93.3** | **63.5** | **73.6** | **95.3** | **72.4** | **83.8** | 95.6 | 83.1 | 90.5 |
| — ***With post-processing bilateral solver*** — | | | | | | | | | | |
| LOST [35] | ImNet | 81.8 | 48.9 | 57.8 | 88.7 | 57.2 | 69.7 | 91.6 | 72.3 | 83.7 |
| TokenCut [45] | ImNet | 89.7 | 61.8 | 69.7 | 91.4 | 62.4 | 75.5 | 93.4 | 77.2 | 87.4 |
| **Ours** + TokenCut | ImNet | **91.2** | **63.6** | **71.9** | **92.5** | **63.8** | **77.4** | **94.3** | **79.0** | **88.7** |
| FOUND [36] | ImNet, DUTS-TR | 92.2 | 61.3 | 70.8 | 94.2 | 66.3 | 76.3 | 95.1 | 81.3 | 93.5 |
| MOVE [5] | ImNet, DUTS-TR | 93.1 | **63.6** | 73.4 | 95.1 | **68.7** | 82.1 | 95.3 | 80.1 | 91.6 |
| **Ours** + FOUND | ImNet, DUTS-TR | 92.2 | 63.1 | 71.0 | 94.2 | 66.8 | 76.6 | 93.3 | **82.7** | **95.6** |
| **Ours** + MOVE | ImNet, DUTS-TR | **93.6** | 63.0 | **74.7** | **95.2** | 66.9 | **83.7** | **95.6** | 81.4 | **95.6** |

**Table 1: Unsupervised saliency detection.** We apply our trained model (**Ours**) to three task-specific methods: TokenCut [45], FOUND [36] and MOVE [5]. State-of-the-art results are obtained in almost all settings. The best results are highlighted in **bold**.

methods (TokenCut [45], FOUND [36] and MOVE [5]) to obtain the saliency masks. For TokenCut, we directly replace the DINO pre-trained model with ours for evaluation. For FOUND and MOVE, we replace their DINO feature extractors with ours and keep them *frozen*. Then, we strictly follow their original setups to only train the specific post-processing modules on the DUTS-TR [40] dataset for a fair comparison.

From Tab. 1, we find that our trained model improves the performance of TokenCut for all metrics on three datasets. Since TokenCut does not conduct any training, the improvements solely come from the feature backbone, demonstrating the effectiveness of our method as a better dense feature extractor. For methods with extra data for training (FOUND and MOVE), our trained model improves most of the metrics (except for MOVE[5] with bilateral solver, which trades off IoU against max$F_\beta$). This shows the compatibility between our model and the task-specific post-processing methods. Moreover, FOUND and MOVE do not employ Normalized Cuts to generate masks, which means our method can generalize beyond the Normalized Cuts algorithm as a generic feature extractor. Finally, using our trained model, state-of-the-art results are achieved in almost all metrics and settings.

**Qualitative Results.** To understand the difference between our trained model and DINO, we show qualitative results in Fig. 4. Comparing Fig. 4(c) and 4(d), we observe that our model obtains higher-quality and cleaner attention (*second smallest eigenvectors* $\boldsymbol{z}_2$ from NCut [32]) than DINO on the salient object. This is attributed to our self-supervised mask-consistency loss, which improves the dense feature quality by back-propagating through NCut. When combined with

[5] Besides DINO, MOVE [5] used another MAE [19] (ViT-L/16) pre-trained in an adversarial fashion. Since only DINO can be replaced with ours, we conjecture that this leads to the tradeoff between IoU and max$F_\beta$.

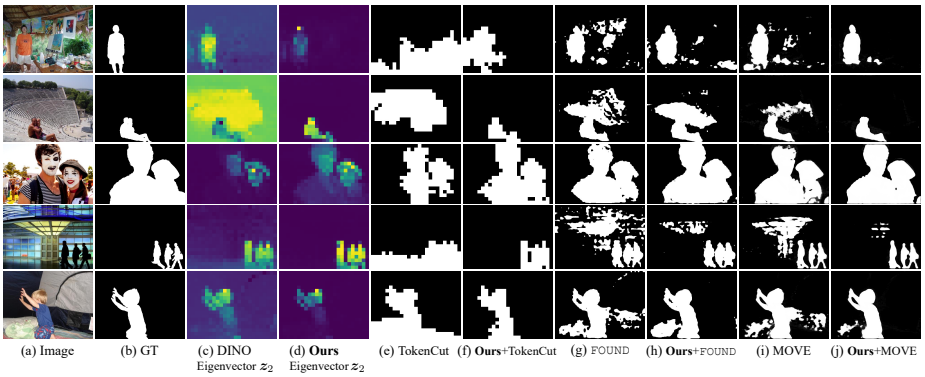


**Fig. 4: Qualitative results of unsupervised saliency detection.** Compared with DINO, our model obtains higher quality and cleaner foreground attentions (*second smallest eigenvector* $z_2$ from NCut [32]). Therefore, applying our model to three tasks-specific methods (TokenCut [45], FOUND [36] and MOVE [5]) can reduce the noisy background and improve the quality of saliency masks.

different task-specific methods, our model significantly improves their results (Fig. 4(e)–(j)) by reducing the background noises.

### 4.2 Unsupervised Object Discovery

**Datasets.** We evaluate our method on three common benchmarks: the `trainval` split of PASCAL VOC07 [14] & VOC12 [15] datasets, and the `train` split of COCO20K [25, 39]. The goal of this task is to detect a single object for each image using unsupervised models.

**Evaluation metric.** As in [31, 36, 45], we adopt Correct Localization (*CorLoc*) as the metric, which calculates the percentage of correctly predicted boxes. A predicted box is regarded as correct if it has an intersection over union (IoU) larger than 0.5 with any ground-truth boxes.

| Method | VOC07 | VOC12 | COCO20k |
|---|---|---|---|
| DINO-seg [7, 35] | 45.8 | 46.2 | 42.1 |
| LOST [35] | 61.9 | 64.0 | 55.7 |
| FreeSOLO [43] | 56.1 | 56.7 | 52.8 |
| DSS [28] | 62.7 | 66.4 | 56.2 |
| TokenCut [45] | 68.8 | 72.1 | 58.8 |
| **Ours** + TokenCut | 71.3 (2.5↑) | 74.2 (2.1↑) | 61.6 (2.8↑) |
| FOUND [36] | 72.5 | 76.1 | 62.9 |
| **Ours** + FOUND | 73.5 (1.0↑) | 76.9 (0.8↑) | 63.6 (0.7↑) |
| MOST [31] | 74.8 | 77.4 | 67.1 |
| **Ours** + MOST | **75.6** (0.8↑) | **78.7** (1.3↑) | **69.4** (2.3↑) |

**Table 2: Unsupervised object discovery**. *CorLoc* score on VOC07, VOC12 and MS-COCO20k datasets. "**Ours** + " indicates applying our trained backbone to different task-specific post-processing methods. The best results are highlighted in **bold**.

**Comparison Results.** To verify the dense prediction capability of our model, we apply our trained model to three post-processing methods: TokenCut [45], FOUND [36] and MOST [31]. As shown in Tab. 2, the performance of all three methods improves after using our model, and state-of-the-art results are obtained by combining our model with MOST post-processing (**Ours** + MOST). Since the original feature extractor for all three methods is pre-trained DINO, this

corroborates that our model provides better dense features suitable for localizing objects.

### 4.3 Unsupervised Semantic Segmentation

**Dataset.** Following [51], we finetune the model on ImageNet-100 [38] or COCO [25] dataset, and evaluate the overclustering (cluster $K = 500$) or unsupervised semantic segmentation ($K = 21$) results on Pascal VOC 2012 [15].
**Evaluation metric.** We report the mean Intersection over Union (mIoU) for both settings. We adopt the ViT-S/16 backbone pretrained either from DINO or our model, and then finetune using the Leopart [51] objective.

| | DINO [7] | Ours | SwAV [6] | MoCo-v2 [10] | DINO+Leopart | Ours+Leopart |
|---|---|---|---|---|---|---|
| $K$=500 | 17.4 | **19.0** | 35.7 | 39.1 | 53.3 | **55.2** |
| $K$=21 | 4.6 | **5.8** | 13.7 | 18.5 | 18.9 | **20.2** |

**Table 3: Unsupervised semantic segmentation**.

**Comparison Results.** As shown in Tab. 3, without any post-processing, our finetuned model outperforms DINO by 1.6% and 1.2%, respectively. After applying the Leopart for clustering, our model also outperforms the pretrained DINO backbone by 1.9% and 1.3%, respectively. Those results verify the superior semantic recognition potential of our model over DINO beyond the saliency and detection tasks.

### 4.4 Generalize Beyond DINO

In the above experiments, we have shown the benefits of our method on top of pre-trained DINO. In this section, we verify that our method can generalize to other pre-training methods beyond DINO and architecture beyond ViT.

To eliminate the dataset-specific factors and directly assess the feature capability, we implement a simple, dataset-agnostic heuristics for post-processing: *taking the second smallest eigenvector* $\boldsymbol{z}_2 > \mathbf{0}$ *as criteria to generate the foreground mask*, denoted as **Simple**. We also include TokenCut and other task-specific methods for comparison.
**Pre-training Models.** We take other pre-training methods: Masked Autoencoders (MAE) [19] and MoCoV3 [9], and adopt the same experiment setup as DINO, *i.e.*, finetuning each pre-trained model for *two epochs on ImageNet*. According to Tab. 4, our method significantly improves the performance of both MAE and MoCoV3 by at least 5% Average IoU. This means our method can consistently improve the dense prediction capability of various pre-training methods beyond DINO. And this capability is obtained in a cheap way, *i.e.*, only two epochs of finetuning on ImageNet.
**Network Architecture.** We report the results of diverse network architectures and configurations in Tab. 4, including ResNet50, ViT-S/8, ViT-S/16, and

| Post-process | Pre-training | Backbone | DUT-OMRON | | | DUTS-TE | | | ECSSD | | | Avg ↑ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Acc | IoU | max$F_\beta$ | Acc | IoU | max$F_\beta$ | Acc | IoU | max$F_\beta$ | |
| Simple $z_2 > 0$ | DINO [7] | ResNet50 | 72.9 | 22.7 | 30.2 | 76.0 | 29.2 | 40.3 | 72.9 | 39.4 | 50.5 | |
| | **Ours** + DINO | | 75.3 | 28.5 (5.8) | 38.2 | 78.4 | 34.7 (5.5) | 47.8 | 76.5 | 47.3 (7.9) | 60.2 | +**6.4** |
| | DINO [7] | ViT-S/16 | 88.2 | 49.7 | 62.3 | 91.1 | 58.2 | 72.6 | 90.1 | 68.6 | 81.2 | |
| | **Ours**+DINO | | 89.5 | 56.4 (6.7) | 65.4 | 91.2 | 61.8 (3.6) | 72.5 | 91.3 | 72.9 (4.3) | 83.9 | +**4.9** |
| | MoCoV3 [9] | ViT-S/16 | 85.8 | 39.0 | 49.7 | 89.9 | 53.5 | 66.1 | 88.5 | 62.1 | 74.5 | |
| | **Ours**+MoCoV3 | | 88.1 | 52.7 (13.7) | 64.2 | 90.2 | 58.9 (5.4) | 71.3 | 91.1 | 72.8 (10.7) | 83.5 | +**9.9** |
| | MAE [19] | ViT-B/16 | 85.5 | 48.6 | 61.4 | 89.6 | 55.9 | 71.1 | 87.6 | 65.4 | 78.5 | |
| | **Ours**+MAE | | 89.9 | 57.8 (9.2) | 69.1 | 92.7 | 63.6 (7.7) | 76.6 | 91.9 | 74.7 (9.3) | 85.5 | +**8.7** |
| TokenCut | DINO [7] | ResNet50 | 68.9 | 25.7 | 29.5 | 73.0 | 31.3 | 36.3 | 73.9 | 44.0 | 49.2 | |
| | **Ours**+DINO | | 73.2 | 30.9 (5.2) | 35.3 | 77.1 | 36.4 (5.1) | 42.0 | 78.3 | 50.5 (6.5) | 56.1 | +**5.6** |
| | DINO [7] | ViT-S/16 | 88.0 | 53.3 | 60.0 | 90.3 | 57.6 | 67.2 | 91.8 | 71.2 | 80.3 | |
| | **Ours**+DINO | | 89.5 | 55.7 (2.4) | 62.8 | 91.5 | 59.8 (2.2) | 69.8 | 92.9 | 74.2 (3.0) | 82.8 | +**2.5** |
| | MoCoV3 [9] | ViT-S/16 | 83.5 | 44.5 | 51.0 | 88.1 | 53.3 | 63.3 | 90.2 | 68.6 | 78.4 | |
| | **Ours**+MoCoV3 | | 88.0 | 53.3 (8.8) | 60.0 | 90.3 | 57.6 (4.3) | 67.2 | 91.8 | 71.2 (2.6) | 80.3 | +**5.2** |
| | MAE [19] | ViT-B/16 | 79.2 | 42.5 | 47.7 | 84.5 | 49.2 | 56.4 | 87.9 | 65.2 | 72.1 | |
| | **Ours**+MAE | | 85.9 | 52.4 (9.9) | 58.7 | 89.3 | 57.1 (7.9) | 66.2 | 92.0 | 73.0 (7.8) | 81.5 | +**8.5** |
| Simple $z_2 > 0$ | DINO [7] | ViT-S/8 | 87.8 | 43.7 | 55.9 | 91.5 | 59.3 | 72.5 | 88.5 | 61.1 | 76.6 | |
| | **Ours**+DINO | | 90.9 | 57.3 (13.6) | 68.8 | 92.7 | 64.3 (5.0) | 76.5 | 91.8 | 71.5 (10.4) | 84.7 | +**9.7** |
| TokenCut | DINO [7] | ViT-S/8 | 89.5 | 57.2 | 65.1 | 91.6 | 61.9 | 73.3 | 92.7 | 74.1 | 85.2 | |
| | **Ours**+DINO | | 91.8 | 60.9 (3.7) | 69.5 | 92.7 | 63.4 (1.5) | 76.7 | 93.3 | 75.4 (1.3) | 87.0 | +**2.2** |
| FOUND | DINO [7] | ViT-S/8 | 91.2 | 57.8 | 66.3 | 93.8 | 64.5 | 71.5 | 94.9 | 80.7 | 95.5 | |
| | **Ours**+DINO | | 91.7 | 61.2 (3.4) | 71.6 | 94.1 | 66.7 (2.2) | 77.8 | 95.7 | 83.3 (2.6) | 95.6 | +**2.7** |
| MOVE† | DINO [7] | ViT-S/8 | 92.3 | 61.5 | 71.2 | 95.0 | 71.3 | 81.5 | 95.4 | 83.0 | 91.6 | |
| | **Ours**+DINO | | 93.3 | 63.5 (2.0) | 73.6 | 95.3 | 72.4 (1.1) | 83.8 | 95.6 | 83.1 (0.1) | 90.5 | +**1.1** |

†Besides DINO, MOVE used an MAE pre-trained in an adversarial fashion. We only replace DINO with ours.

**Table 4: Performance improvements across various pre-training methods, architectures and post-processing methods.** "Simple" refers to our simple dataset-agnostic post-processing heuristics, *i.e.*, *employing the second smallest eigenvector* $\boldsymbol{z}_2 > \mathbf{0}$ *for foreground delineation.* The IoU improvements are shown in the parentheses. The average IoU improvements are shown in the rightmost column.

ViT-B/16. Our method generalizes to all listed architectures and configurations, usually with a significant increase over existing pre-trained models. Although ResNet does not have the emerging object attention property as ViT [7], our method consistently improves pre-trained ResNet50 by over 5% IoU.

For ViT-S/8, we gather different post-processing methods for comparison. Tab. 4 shows that our model consistently improves the performance of all post-processing methods. We also find that using our model, even simple dataset-agnostic heuristics can match the results of well-designed post-processing methods such as TokenCut [45].

### 4.5 Ablation Study

**Mask Collapse.** As a self-supervised dense learning method, we have observed specific mask collapse patterns different from existing self-supervised methods [7, 17]. First, without any regularization, the masks $\boldsymbol{m}_1^{\text{ovlp}}, \boldsymbol{m}_2^{\text{ovlp}}$ converge to horizontal stripes shown in Fig. 5(a). Then, we apply negative BCE regularization on the horizontal pattern (Fig. 3(a)), and observe collapsing to vertical stripes shown in Fig. 5(b). *After applying both horizontal and vertical regularization, the collapse issue is fixed.* But if we remove the projector $g$, a third collapse pattern (Fig. 5(c)) ensembling uniform mask appears. We conjecture


**Fig. 5: Mask Collapse.** (a) Our model collapses to a horizontal pattern without regularization. (b) After penalizing the horizontal pattern with BCE loss, the vertical pattern appears. (c) If we remove the projector $g$, the third collapse pattern appears.

| | DUT-OMRON | DUTS-TE | ECSSD |
|---|---|---|---|
| Ours | **56.4** | **61.8** | **72.9** |
| *w/o* structural regularization | 48.8 | 55.8 | 69.8 |
| *w/o* projector | 52.2 | 59.1 | 66.3 |
| pixel consistency | 46.0 | 54.2 | 66.7 |

**Table 5: Ablation study** of design choices. We use the "Simple" post-processing: *taking* $\boldsymbol{z}_2 > \mathbf{0}$ *as foreground*. IoU is reported.

that constrained by our architecture and the mask-consistency loss, the mask might gradually lose its structure information and collapse to these patterns as a trivial solution. This motivates us to design the three structural regularization patterns in Sec. 3.4.

**Design Choices.** We then study the impact of different design choices on our model, listed in Tab. 5. For pixel consistency, we use cross-entropy loss on $\boldsymbol{f}_1^{\text{ovlp}}$ and $\boldsymbol{f}_2^{\text{ovlp}}$, which is similar to DenseCL [44] and can be seen as a dense extension of DINO. If a model variant collapses, we evaluate the checkpoint before collapsing. Otherwise, we train each model for two epochs. From the comparisons in Tab. 5, we find that it is critical to apply structural regularization and projector to ensure the model does not collapse. The pixel consistency variant only uses pixel-level information, thus performing much worse than our mask consistency method. This demonstrates the effectiveness of the proposed Differentiable Normalized Cuts layer.

## 5 Conclusion

In this paper, we propose an effective self-supervised dense feature learning framework, to improve the dense prediction capability of existing pre-trained models. The core component of our framework is the *Differentiable Normalized Cuts* layer. In the forward pass of this layer, we employ the classical Normalized Cuts algorithm to generate unsupervised foreground masks. In the backward pass, we present an efficient gradient formulation for back-propagation, using matrices only from the forward pass for fast computation. Based on this layer, we devise an effective two-branch architecture and propose a mask-consistency loss to train the model. Our method generalizes to diverse pre-trained models, network architectures, and dense prediction tasks.

## Acknowledgements

This work was supported by an Australian Research Council (ARC) Future Fellowship (project number FT200100421). Yanbin Liu was partly supported by the Google Cloud Research Credits program with the award GCP19980904.

## References

1. Abdi, H.: The eigen-decomposition: Eigenvalues and eigenvectors. Encyclopedia of measurement and statistics pp. 304–308 (2007)
2. Aflalo, A., Bagon, S., Kashti, T., Eldar, Y.: Deepcut: Unsupervised segmentation using graph neural networks clustering. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 32–41 (2023)
3. Bach, F., Jordan, M.: Learning spectral clustering. Advances in neural information processing systems **16** (2003)
4. Barron, J.T., Poole, B.: The fast bilateral solver. In: European conference on computer vision. pp. 617–632. Springer (2016)
5. Bielski, A., Favaro, P.: Move: Unsupervised movable object segmentation and detection. Advances in Neural Information Processing Systems **35**, 33371–33386 (2022)
6. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. Advances in neural information processing systems **33**, 9912–9924 (2020)
7. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9650–9660 (2021)
8. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: Simclr: A simple framework for contrastive learning of visual representations. In: International Conference on Learning Representations. vol. 2 (2020)
9. Chen, X., Xie, S., He, K.: An empirical study of training self-supervised vision transformers. in 2021 ieee. In: CVF International Conference on Computer Vision (ICCV). pp. 9620–9629 (2021)
10. Chen, X., Fan, H., Girshick, R., He, K.: Improved baselines with momentum contrastive learning. arXiv preprint arXiv:2003.04297 (2020)
11. Chen, X., He, K.: Exploring simple siamese representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 15750–15758 (2021)
12. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
13. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2020)
14. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html (2007)

15. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html (2012)
16. Golub, G.H., Van Loan, C.F.: Matrix computations. JHU press (2013)
17. Grill, J.B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al.: Bootstrap your own latent-a new approach to self-supervised learning. Advances in neural information processing systems **33**, 21271–21284 (2020)
18. Han, K., Rezende, R.S., Ham, B., Wong, K.Y.K., Cho, M., Schmid, C., Ponce, J.: Scnet: Learning semantic correspondence. In: Proceedings of the IEEE international conference on computer vision. pp. 1831–1840 (2017)
19. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 16000–16009 (2022)
20. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9729–9738 (2020)
21. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)
22. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
23. Ionescu, C., Vantzos, O., Sminchisescu, C.: Matrix backpropagation for deep networks with structured layers. In: Proceedings of the IEEE international conference on computer vision. pp. 2965–2973 (2015)
24. Law, M.T., Urtasun, R., Zemel, R.S.: Deep spectral clustering learning. In: International conference on machine learning. pp. 1985–1994. PMLR (2017)
25. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. pp. 740–755. Springer (2014)
26. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3431–3440 (2015)
27. Magnus, J., Neudecker, H.: Matrix differential calculus with applications in statistics and econometrics. Wiley series in probability and mathematical statistics Show all parts in this series (1988)
28. Melas-Kyriazi, L., Rupprecht, C., Laina, I., Vedaldi, A.: Deep spectral methods: A surprisingly strong baseline for unsupervised semantic segmentation and localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8364–8375 (2022)
29. Min, J., Lee, J., Ponce, J., Cho, M.: Hyperpixel flow: Semantic correspondence with multi-layer neural features. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3395–3404 (2019)
30. Papadopoulo, T., Lourakis, M.I.: Estimating the jacobian of the singular value decomposition: Theory and applications. In: Computer Vision-ECCV 2000: 6th European Conference on Computer Vision Dublin, Ireland, June 26–July 1, 2000 Proceedings, Part I 6. pp. 554–570. Springer (2000)

31. Rambhatla, S.S., Misra, I., Chellappa, R., Shrivastava, A.: Most: Multiple object localization with self-supervised transformers for object discovery. arXiv preprint arXiv:2304.05387 (2023)
32. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on pattern analysis and machine intelligence **22**(8), 888–905 (2000)
33. Shi, J., Yan, Q., Xu, L., Jia, J.: Hierarchical image saliency detection on extended cssd. IEEE transactions on pattern analysis and machine intelligence **38**(4), 717–729 (2015)
34. Shin, G., Albanie, S., Xie, W.: Unsupervised salient object detection with spectral cluster voting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3971–3980 (2022)
35. Siméoni, O., Puy, G., Vo, H.V., Roburin, S., Gidaris, S., Bursuc, A., Pérez, P., Marlet, R., Ponce, J.: Localizing objects with self-supervised transformers and no labels. In: BMVC 2021-32nd British Machine Vision Conference (2021)
36. Siméoni, O., Sekkat, C., Puy, G., Vobeckỳ, A., Zablocki, É., Pérez, P.: Unsupervised object localization: Observing the background to discover objects. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3176–3186 (2023)
37. Strudel, R., Garcia, R., Laptev, I., Schmid, C.: Segmenter: Transformer for semantic segmentation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 7262–7272 (2021)
38. Tian, Y., Krishnan, D., Isola, P.: Contrastive multiview coding. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16. pp. 776–794. Springer (2020)
39. Vo, H.V., Pérez, P., Ponce, J.: Toward unsupervised, multi-object discovery in large-scale image collections. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16. pp. 779–795. Springer (2020)
40. Wang, L., Lu, H., Wang, Y., Feng, M., Wang, D., Yin, B., Ruan, X.: Learning to detect salient objects with image-level supervision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 136–145 (2017)
41. Wang, W., Dang, Z., Hu, Y., Fua, P., Salzmann, M.: Backpropagation-friendly eigendecomposition. Advances in Neural Information Processing Systems **32** (2019)
42. Wang, W., Dang, Z., Hu, Y., Fua, P., Salzmann, M.: Robust differentiable svd. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(9), 5472–5487 (2021)
43. Wang, X., Yu, Z., De Mello, S., Kautz, J., Anandkumar, A., Shen, C., Alvarez, J.M.: Freesolo: Learning to segment objects without annotations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14176–14186 (2022)
44. Wang, X., Zhang, R., Shen, C., Kong, T., Li, L.: 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 (2021)
45. Wang, Y., Shen, X., Hu, S.X., Yuan, Y., Crowley, J.L., Vaufreydaz, D.: Self-supervised transformers for unsupervised object discovery using normalized cut. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14543–14553 (2022)
46. Xiao, T., Reed, C.J., Wang, X., Keutzer, K., Darrell, T.: Region similarity representation learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10539–10548 (2021)

47. Xie, Z., Lin, Y., Zhang, Z., Cao, Y., Lin, S., Hu, H.: Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16684–16693 (2021)
48. Yang, C., Zhang, L., Lu, H., Ruan, X., Yang, M.H.: Saliency detection via graph-based manifold ranking. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3166–3173 (2013)
49. Zhang, W., Pang, J., Chen, K., Loy, C.C.: Dense siamese network for dense unsupervised learning. In: European Conference on Computer Vision. pp. 464–480. Springer (2022)
50. Zheng, J., Li, W., Hong, J., Petersson, L., Barnes, N.: Towards open-set object detection and discovery. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3961–3970 (2022)
51. Ziegler, A., Asano, Y.M.: Self-supervised learning of object parts for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14502–14511 (2022)
52. Zou, Z., Chen, K., Shi, Z., Guo, Y., Ye, J.: Object detection in 20 years: A survey. Proceedings of the IEEE (2023)