


Patch-Based Deep Unsupervised Image Segmentation Using Graph Cuts

Isaac Wasserman¹ and Jeová Farias Sales Rocha Neto² ^a

¹University of Pennsylvania, Philadelphia PA 19104, U.S.A.

²Bowdoin College, Brunswick ME 04011, U.S.A.

Keywords: Image Segmentation, Unsupervised Learning, Graph Cuts.

Abstract: Unsupervised image segmentation seeks to group semantic patterns in an image without the use of human annotation. Similarly, image clustering searches for groupings of images based on their semantic content. Traditionally, both problems have drawn from sound mathematical concepts to produce concrete applications. With the emergence of deep learning, the scientific community turned its attention to complex neural network-based solvers that achieved impressive results in those domains but rarely leveraged the advances made by classical methods. In this work, we propose a patch-based unsupervised image segmentation strategy that uses the algorithmic strength of classical graph-based methods to enhance unsupervised feature extraction from deep clustering. We show that a simple convolutional neural network, trained to classify image patches and iteratively regularized using graph cuts, can be transformed into a state-of-the-art, fully-convolutional, unsupervised, pixel-level segmenter. Furthermore, we demonstrate that this is the ideal setting for leveraging the patch-level pairwise features generated by vision transformer models. Our results on real image data demonstrate the effectiveness of our proposed methodology.

1 INTRODUCTION


Image segmentation has long been one of the main tasks in computer vision, and it has been widely applied in general image understanding or as a preprocessing step for other tasks such as object detection. It aims to correspond each pixel in an image to a class, in such a way that semantically similar pixels are assigned to the same class. This problem finds various industrial applications such as autonomous driving, medical image analysis, video surveillance, and virtual reality to name a few (Minaee et al., 2021).

On the supervised front, deep learning approaches using convolutional (CNN) and fully convolutional neural networks (FCN) have achieved unprecedented results in image segmentation, as illustrated by the UNet (Ronneberger et al., 2015) and DeepLab (Chen et al., 2017a) models. Recently, however, methods using transformer-based models, such as Segformer (Xie et al., 2021), DETR (Carion et al., 2020) and DINO (Caron et al., 2021), are slowly outperforming established CNN solutions. This shift prompted a recent interest in deep models that utilize image patches rather than individual pixels (Tolstikhin et al., 2021; Han et al., 2022), leading some to believe that patch

representations are the main source of vision transformers' success (Trockman and Kolter, 2022).

These accomplishments come, however, at the cost of long training schemes and the need for vast amounts of annotated data, which hinder their application in many domains where data can be expensive or scarce, such as in biology, or astrophysics (Yu et al., 2018). Such issues are resolved via the application of unsupervised techniques. In this setting, one aims to create a model that automatically discovers semantically important visual features that characterize the various objects in a scene. Classically, we approach this problem using variational, statistical, and graphical methods, exemplified in active contours (Chan and Vese, 2001), conditional random fields (Krähenbühl and Koltun, 2011), and graph cuts (Shi and Malik, 2000; Boykov and Jolly, 2001). Within the deep learning literature, prominent advances in unsupervised deep image clustering (Ren et al., 2022) eventually led to developments in deep image segmentation (Kim et al., 2020; Hamilton et al., 2021; Lin et al., 2021; Hamilton et al., 2022; Wang et al., 2022).

In this work, we introduce GraPL, an unsupervised image segmentation technique that draws inspiration from the success of CNNs for imaging tasks,

^a  <https://orcid.org/0000-0001-8103-7937>

the learning strategies of deep clustering methods, and the regularization power of graph cut algorithms. Here, we alternate the training of a CNN classifier on image patches and the minimization of clustering energy via graph cuts. To the best of our knowledge, this is the first attempt in the deep clustering and image segmentation literature to make use of graph cuts to solve a deep learning-based unsupervised task. We show that our zero-shot approach detects visual segments in an image without onerous unsupervised training on an entire image dataset, automatically finds a satisfactory number of image segments, and easily translates patch-level training to efficient pixel-level inference. Furthermore, because of its patch-level structure, it's able to naturally incorporate pretrained patch embeddings (Caron et al., 2021; Oquab et al., 2023), without relying on them for inference. Finally, we show that this simple approach achieves state-of-the-art results in deep unsupervised image segmentation, demonstrating the potential of graph cuts to improve other patch-based deep segmentation algorithms. Specifically, we make the following contributions with our work:

- We introduce GraPL, an unsupervised segmentation method that learns a fully convolutional segmenter directly from the image's patches, using an iterative algorithm regularized by graph cuts.
- We show that this framework naturally employs patch embeddings for pixel-level segmentation without the need for postprocessing schemes, such as CRF refining.
- We demonstrate that GraPL can compete with the state-of-the-art in deep unsupervised segmentation, despite having just 23k parameters and operating in a zero-shot paradigm.

2 RELATED WORK

2.1 Deep Clustering

With the advancements in deep supervised image classification techniques, interest in deep architectures to solve unsupervised problems followed naturally.

This pursuit led to the task of partitioning image datasets into clusters using deep representations, without human supervision, inaugurating the body of work which is now referred to as "deep clustering." The interested reader is referred to (Ren et al., 2022) for a comprehensive review of the available approaches to deep clustering.

In GraPL, we treat image patches as individual images to be clustered as a pretext task to train our segmenter and efficiently use graph cuts to impose constraints on the patch clusters. To the best of our knowledge, our method is the first to use MRF-based algorithms for clustering CNN-generated visual features.

2.2 Deep Unsupervised Image Segmentation

As deep clustering aims to learn visual features and groupings without human annotation via deep neural models, deep unsupervised image segmentation hopes to use the same models to learn coherent and meaningful image regions without the use of ground-truth labels. To do so, many methods explore strategies that resemble those from deep clustering. Cho *et al.* (Cho et al., 2021) iteratively employ k -means to cluster pixel-level features extracted from a network trained on photometrically and geometrically varying image views. The work in (Ji et al., 2019) efficiently extends a mutual information-based deep clustering algorithm to the pixel-level by recognizing that such a process can be achieved via convolution operations. (Hwang et al., 2019) computes pixel embeddings from a metric learning network and segments each image using a spherical k -means clustering algorithm.

In (Kim et al., 2020), the authors train an FCN with pseudo-labels generated by the same network in a prior step. They attain reliable segmentations by proposing a complex loss function that ensures the similarity among pixels in shared segments while encouraging their spatial continuity and limiting their total count. Our method, while similarly training an FCN, works on patches and reinforces spatial continuity and low segment count via our graph cut approach. Furthermore, GraPL's use of graph cuts allows it to incorporate pairwise patch relationships. Finally, while other patch-based unsupervised solutions require a segmentation refinement stage after a patch feature clustering step (Hamilton et al., 2022; Wang et al., 2022; Wang et al., 2023), we both discover and instill patch knowledge interactively, without the need for postprocessing our result.

2.3 Graph Cuts for Image Segmentation

Modeling image generation as a Markov random field (MRF) has a long history in computer vision, dating its initial theoretical and algorithmic achievements to the works of Abel *et al.* (Abend et al., 1965) and Besag (Besag, 1986). Soon enough, MRFs found ap-

plications in various image processing tasks, such as edge detection, image denoising, segmentation, and stereo (Li, 2009). In particular, the works conducted by Boykov and Jolly (Boykov and Jolly, 2001) and Boykov *et al.* (Boykov *et al.*, 2001) demonstrated that one can apply efficient min *st*-cut-based algorithms to solve image segmentation by modeling it as a Maximum *a Posteriori* (MAP) estimator of an MRF. Their groundbreaking results made possible the emergence of classical graph-based segmentation methodologies such as GrabCut (Rother *et al.*, 2004) and were, more recently, used to improve the training of CNN-based segmenters (Marin *et al.*, 2019). CRF modeling, closely related to MRF, has also played an important role in refining coarse network predictions in recent segmentation methods (Zheng *et al.*, 2015; Chen *et al.*, 2017a; Chen *et al.*, 2017b).

In some ways, our proposed method draws inspiration from the methodologies proposed by Rother *et al.* (Rother *et al.*, 2004), and Marin *et al.* (Marin *et al.*, 2019). In (Rother *et al.*, 2004), the authors propose GrabCut, an algorithm that iteratively bound-optimizes a segmentation energy, requiring the solution of a min *st*-cut problem at each iteration to perform unsupervised regularized fitting of the image’s appearance, which is modeled as a Gaussian mixture model. Our algorithm also uses min *st*-cut solvers iteratively, but here we (1) work on patch data, instead of individual pixels, and (2) fit the image appearance using a CNN classifier. Due to the nature of CNNs, our network can seamlessly translate the patch-level classifier into a pixel-level image segmenter. In (Marin *et al.*, 2019), the authors show how to perform weakly-supervised CNN segmentation via an optimizer that alternates between solving an MAP-MRF problem and gradient computation. In contrast, our method solves a fully unsupervised segmentation problem and does not use our MAP solution to adjust gradient directions.

3 METHODOLOGY

GraPL (Graph Cuts at the Patch Level) is a fully unsupervised segmentation method that operates in a single-image paradigm. Using patch clustering as a pretext task for segmentation, during training it learns distinctive segment features that enable it to effectively segment the image at the pixel level. Although other techniques have previously shown patch clustering to be an effective surrogate task (Ji *et al.*, 2019; Ouali *et al.*, 2020; Wang *et al.*, 2022; Wang *et al.*, 2023), our method demonstrates that clustering the patches of a single image provides sufficient feature

learning to accurately segment it. GraPL’s training is guided by patch-level graph cuts; this intervention imposes spatial coherence priors, which help learn clusters that are conducive to segmentation. At inference, the complexities of the pipeline disappear, leaving only the network. Leveraging a generalization of CNNs, the trained model is “convolved” over the entire image to produce a pixel-level segmentation.

3.1 Algorithm

Let $I : \Omega \mapsto \mathbb{R}^c$ be an image of c channels with pixel set $\Omega = \{1, \dots, n\} \times \{1, \dots, m\}$, and $S : \Omega \mapsto \{1, \dots, K\}$ be a segmentation map of I in K regions. Let \mathcal{P} be a set of patches from I , such that all patches are of the same size, i.e., for each p in \mathcal{P} , $p : \Omega_p \mapsto \mathbb{R}^c, \Omega_p \subset \Omega, |\Omega_p| = \text{const}$. In practice, we populate \mathcal{P} by selecting all patches on a non-overlapping $d \times d$ grid of I , resulting in patches of shape $(w/d, h/d)$. We make this choice of \mathcal{P} based on two factors: (1) efficiency, as this operation can be efficiently performed by most deep learning libraries via their unfolding methods, and (2) simplicity, as it’s one of the simplest ways to generate equal sized patches that span Ω .

Let $F_\theta : \Omega \mapsto [0, 1]^{K \times |\Omega|}$ be an FCN, and $F'_\theta : \Omega_p \mapsto [0, 1]^K$ be a CNN patch classifier. In GraPL, both networks are parametrized by the same parameters θ . F'_θ is used in our training stage and is applied to the patches in \mathcal{P} , while F_θ is employed in our inference phase and is our final segmenter. The full algorithm is shown in Figure 1.

Training Stage. Our goal is to learn θ exclusively from the data in \mathcal{P} and transfer it to F_θ . To do so, our method trains F'_θ by minimizing an energy formulated at the patch level of I . Let $S' : \mathcal{P} \mapsto \{1, \dots, K\}$ be a labeling for the patches in \mathcal{P} . Following the literature on MRF modeling (Boykov and Jolly, 2001), we define the energy of S' for an unknown θ as:

$$E(S', \theta | \mathcal{P}) = U(S', \theta | \mathcal{P}) + \lambda V(S' | \mathcal{P}), \quad (1)$$

with $\lambda \geq 0$. The unary term $U(\cdot)$ is traditionally defined as:

$$U(S', \theta | X) = - \sum_{p \in \mathcal{P}} \sum_{k=1}^K \mathbb{1}(S'(p) = k) [\ln F'_\theta(p)]_k, \quad (2)$$

where $\mathbb{1}(\cdot)$ is the indicator function and $[\cdot]_k$ is the k -th position of a vector. Let α and β be probability distributions in \mathbb{R}^K , and let $H(\alpha, \beta) = - \sum_{k=1}^K [\alpha]_k \ln [\beta]_k$ be their cross entropy. This means that Eq. 2 can be seen as the sum of cross entropies $H(S'(p), F'_\theta(p))$ between $S'(p)$, taken as a one-hot probability distribution, and $F'_\theta(p)$ over all $p \in \mathcal{P}$. The pairwise energy term $V(\cdot)$

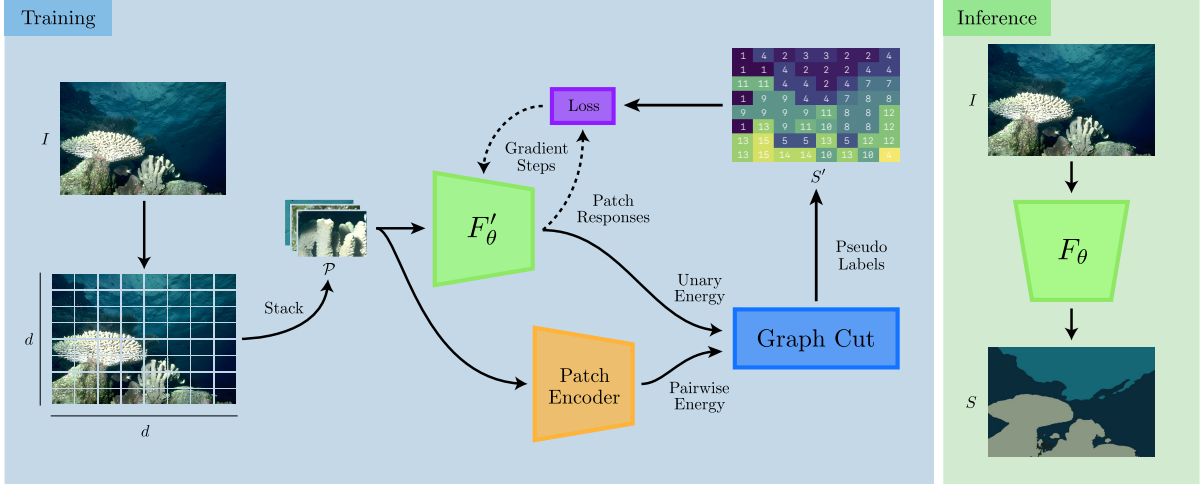


Figure 1: The proposed algorithm. GraPL trains a convolutional neural network to cluster patches of a single image without supervision under the guidance of graph cuts, spatial continuity loss, and a patch affinity encoder. At inference, this patch clustering knowledge is applied to pixel-level segmentation of the image. F'_θ and F_θ share the same parameters.

is given by:

$$V(S'|\mathcal{P}) = \sum_{(p,q) \in \mathcal{P} \times \mathcal{P}} \mathbb{1}(S'(p) \neq S'(q)) \phi(p,q), \quad (3)$$

with the patch similarity function $\phi(\cdot)$ defined as:

$$\phi(p,q) = \frac{1}{\text{dist}(p,q)} \exp\left(-\frac{\text{aff}(p,q)^2}{2\sigma}\right), \quad (4)$$

where the data affinity function $\text{aff}(p,q)$ evaluates the data similarity between p and q , and $\text{dist}(p,q)$ considers the Euclidean distance between the centers of p and q . We select σ as the standard deviation of affinities for all $p, q \in \mathcal{P}$. To compute patch affinities we make use of a patch encoder, which extracts an embedding from each patch in \mathcal{P} .

Inspired by GrabCut (Rother et al., 2004), GraPL minimizes E using a block-coordinate descent iterative strategy, where we alternate between optimizing for θ and S' , keeping the other constant. The current labeling S'_t is updated using the current network parameters θ_{t-1} , now taken as fixed in Eq. 1:

$$S'_t = \arg \min_{S'} E(S'|\theta_{t-1}, \mathcal{P}). \quad (5)$$

The above problem can be approximately solved by a series of minimum st -cut in the form of α -expansion or $\alpha\beta$ -swap moves (Boykov et al., 2001). This step is quickly accomplished due to the efficiency of such graph cut algorithms and the comparatively small size of \mathcal{P} , which presents a further advantage to our patch-based framework. We then compute the updated parameters θ_t via:

$$\theta_t = \arg \min_{\theta} \mathcal{L}(F'_\theta(\mathcal{P}), S'_t), \quad (6)$$

where $F'_\theta(\mathcal{P}) = \{F'_\theta(p)\}_{p \in \mathcal{P}}$. We employ traditional gradient descent-based backpropagation to solve the above problem. The loss $\mathcal{L}(\cdot)$, is designed to predict the outputs of F'_θ on each patch using the labels from S'_t . Keeping S'_t fixed, Eq. 2 conveniently formulates that process as a sum of cross entropy losses, just as one would naturally devise in a supervised segmentation learning scheme. We then follow Kim et al. (Kim et al., 2020) and include a patch-level continuity loss $C(\theta)$:

$$C(\theta) = \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{N}_p} |F'_\theta(p) - F'_\theta(q)|, \quad (7)$$

where $|\cdot|$ is the $L1$ norm and \mathcal{N}_p is the set of patches immediately neighboring p in Ω space. In the general case, one can employ a k -nearest neighbors graph of the elements in \mathcal{P} , considering the Euclidean distance between patch centers. For the $d \times d$ grid from Eq. 1, we choose \mathcal{N}_p to be given by the patches immediately above and to the left of p , resembling what is done in (Kim et al., 2020). This continuity loss brings spatial coherence outside the graph step and encourages smooth boundaries on the network outputs. In practice, we found it to be beneficial to have both the graph step and $C(\theta)$ in our method. Our final loss is then defined as:

$$\mathcal{L}(F'_\theta(\mathcal{P}), S'_{t-1}) = \sum_{p \in \mathcal{P}} H(S'_{t-1}(p), F'_\theta(p)) + \mu C(\theta), \quad (8)$$

where $\mu \geq 0$. As a consequence of the use of both graph cuts and the continuity loss described above, GraPL naturally suppresses extraneous labels arising from irrelevant patterns or textures, automatically promoting model selection. As the alternation continues, F'_θ improves to the point where it no longer

requires the guidance of the graph cuts to produce spatially and semantically coherent patch clusters. At that point, we end our training phase.

Inference Stage. Once F'_θ is trained, our next goal is to classify all possible patches in I of shape equal to the patches in \mathcal{P} . To that end, we first assume that, as a CNN, F'_θ is composed of an initial series of convolutional layers and a final stage of say Q dense layers, along with a softmax function at the end. Assume that the inputs of all layers are unpaddinged, and that each dense layer has s_q units leading to a final output of size K . Now, one can replace each dense layer in F'_θ with a convolutional one of kernel size $\sqrt{s_q}$ and retain its exact functionality. Our resulting FCN, F_θ , is now capable of efficiently being applied to I , by effectively “convolving” it with patch classifier F'_θ .

3.2 Advantages of Using Graph Cut

In the absence of labels, GraPL learns to cluster patches via an iterative procedure. This general formulation allows us to inject knowledge about the domain by designing an apt method for selecting pseudo-labels. While similar methods use k -means (Caron et al., 2018), mixture models (Hwang et al., 2019), or simply argmax (Kim et al., 2020) to transform network outputs into new pseudo-labels, GraPL uses these response vectors to define the unary energy of a patch-level MRF graph of the image.

This approach for patch clustering introduces some advantages to our method. First, while the MRF modeling step is done primarily to impose a spatial coherence prior, due to the known shrinking bias of graph cuts (Kolmogorov and Boykov, 2005), the resultant partition also smooths segment boundaries and reduces the number of distinct segments, leading to natural model selection. The spatial regularization introduced by the proposed graph can also be generalized to accommodate other classical graph formulations that consider segmentation seeds (Boykov et al., 2001), appearance disparity (Tang et al., 2013), curvature (El-Zehiry and Grady, 2010), or target distributions (Ayed et al., 2010). Finally, in contrast to methods that discover objects by clustering patch embeddings arising from pretrained transformers and applying a segmentation head (Hamilton et al., 2022) or CRF refinement (Wang et al., 2022; Wang et al., 2023), GraPL considers patch embeddings only as way to guide its training stage, yielding a final pixel-level segmentation map without postprocessing. We consider our graph cut-based approach to handle rich patch features beneficial, as we do not overly depend on their clustering power, and simply reference them as guidance when regularizing our training.

4 EXPERIMENTS

4.1 Experimental Setup

Segmentation Task. To evaluate the behaviors of GraPL (Section 4.2), the algorithm was tasked with segmenting the 200-image test set of BSDS500 (Arbelaez et al., 2011) using a variety of hyperparameters. Segmentation performance is measured in terms of weighted mean intersection over union (mIoU) (Garcia-Garcia et al., 2017), with predicted segments matched one-to-one with target segments using a version of the Hungarian algorithm modified to accommodate $\hat{K} \neq K^*$, where \hat{K} is the number of distinct segments in the final segmentation, and K^* is the number of segments in the ground truth. Results are averaged across 10 different random seeds for initialization. In addition to BSDS500, we compare GraPL to competing methods using COCO-stuff (Caesar et al., 2018), as configured in (Ji et al., 2019), presenting the standard macro averaged mIoU of each model.

Hyperparameters. Unless otherwise specified, the following configuration was used during testing. Pseudo-labels were initialized according to the SLIC-based (Achanta et al., 2012) algorithm described in Paragraph 4.2. GraPL was run for four training iterations, and the number of gradient steps in the loss minimization at each iteration was 40, 32, 22, and 12 respectively. K_0 , the number of initial segments was set at 14, and d was set to 32. Graph cuts were implemented using pyGCO (Li and Borovec, 2023), and the pairwise energy coefficient, λ , was set to 64. The continuity loss was assigned a weight of $\mu = 3$. The L_2 norm between DINOv2 (Oquab et al., 2023) (ViT-L/14 distilled) patch embeddings was used as an affinity metric to determine pairwise weights.

Network Architecture. An intentionally minimal CNN architecture was used, consisting of $2 \times 3 \times 3$ unpaddinged convolutional layers with 32 and 8 filters, respectively. In F'_θ , this is followed by a dense classification head with K_0 units, and in F_θ it is followed by a $(\frac{h}{d} - 4) \times (\frac{w}{d} - 4)$ convolutional segmentation head with K_0 filters. The network layers are each separated by batch normalization, tanh activations, and dropout with a rate of 0.2. Without padding, our network is subject to certain regularization implications. In CNNs, the use of zero padding has the effect of dropping out some of the weights of the subsequent convolutional layer. As our method requires the training phase to be executed on unpaddinged

images, it is effectively deprived of this regularization feature. We found that applying dropout before the first convolutional layer all but resolved issues arising from the network’s lack of padding. Despite its simplicity, this network is complex enough to achieve reliable segmentations, and more complex networks did not lead to better performance. In our work, we also abstain from using padding on our inference phase, which results in $\hat{S} = F_{\theta}(I)$ being of a size smaller than Ω , due to the convolution operations in F_{θ} . GraPL handles this discrepancy by interpolating \hat{S} to the original dimensions via nearest neighbor interpolation. Networks were implemented using PyTorch 2.0.1 (Paszke et al., 2019) and ran on a Nvidia A100 GPU.

Early Stopping. If a cross-entropy loss of less than 1.0 was reached during the first iteration, it was stopped early, and new pseudo-labels were assigned. During the first iteration, we are fitting the initial pseudo-labels, which are either arbitrary or assigned by SLIC. By imposing this early stopping condition, we are avoiding the local minima where GraPL may be overfitting to a less performant (or worse, arbitrary) segmentation.

4.2 Ablation Studies

Initialization. As an iterative algorithm, proper initialization is an important factor in training GraPL. Although similar deep clustering algorithms have used randomly initialized pseudo-labels (Caron et al., 2018; Kim et al., 2020), we were unsure whether ignoring more principled approaches was leaving performance on the table. To answer this question, we compared four initialization strategies: “patchwise random,” “seedwise random,” “spatial clustering,” and an approach based on SLIC (Achanta et al., 2012). The “patchwise random” approach individually assigns each patch p in \mathcal{P} a random label. In the “seedwise random” strategy, we select K_0 random patches and assign them each one of the K_0 labels; the remaining patches are assigned the label of the patch closest to them. For “spatial clustering,” patches are clustered using k -means according to their (x, y) spatial coordinates to form K_0 clusters of roughly equal size. In the SLIC-based approach, we unfold a K_0 cluster SLIC segmentation with low compactness into the same patches as the input image. The onehot labels of these patches are averaged and normalized to produce soft labels. These soft initializations are an attempt to regularize and retain all salient features of the patch during training.

Tests demonstrated that patchwise random initial-

ization is not an ideal choice for GraPL (Table 1). This is likely because it encourages a disregard for spatial coherence during the first and most important iteration. While SLIC was shown to be the best choice out of the methods tested, seedwise random and spatial clustering initialization performed only 1.0% and 0.6% worse, respectively, and the algorithm could likely be tuned such that they meet or exceed the performance of SLIC. However, in the current configuration, we notice a tendency for both of these methods to result in undersegmentation, in which $\Delta K = K_0 - \hat{K}$ is considerably higher than the SLIC version (Figure 2).

Pairwise Edge Weights. The pairwise energy function (Eq. 4) used by GraPL includes an affinity function $\text{aff}(p, q)$. Designed with vision transformers in mind, this function is defined by the Euclidean distance between some patch metric or embedding $m(p)$ for $p \in \mathcal{P}$.

Though DINOv2 (Oquab et al., 2023) has been shown to produce excellent, fully unsupervised features on the patch level, requiring minimal supervised fine-tuning to produce an effective segmentation model (Oquab et al., 2023). However, it’s unclear whether the features are easily separable using unsupervised methods.

We examined the applicability of three definitions for $m(p)$: DINOv2 embedding, mean RGB color, and patch position. To produce the final DINO embeddings, images were resized to $14d \times 14d$, such that each GraPL patch corresponds to a 14×14 DINO patch. These embeddings were reduced to K_0 dimensions using 2nd-degree polynomial PCA. As a baseline, we also tested a version where the fully connected graph was replaced with a 4-neighborhood lattice of uniformly weighted edges.

In our tests, DINOv2 embeddings were a significantly better metric than distance alone (Table 2). However, they were outperformed by simple RGB color vectors. Acknowledging DINOv2’s ability to act as a feature extractor for supervised segmentation, further research is needed to determine what types of transformations are necessary for converting the em-

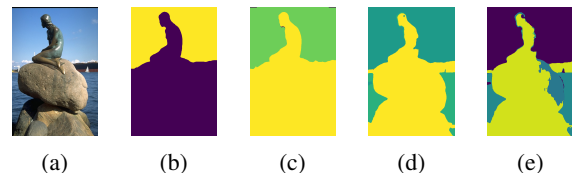


Figure 2: Example of undersegmentation from non-SLIC initialization. (a) Input image. (b) Patchwise Random ($\hat{K} = 2$). (c) Seedwise Random ($\hat{K} = 4$). (d) Spatial Clustering ($\hat{K} = 4$). (e) SLIC ($\hat{K} = 6$).

Table 1: Comparison of pseudo-label initialization methods.

Initializer	mIoU
Patchwise Rand.	0.496
Seedwise Rand.	0.507
Spatial	0.509
SLIC	0.512

Table 2: Comparison of pairwise weighting metrics.

Metric	mIoU
Uniform	0.459
Position	0.476
Color	0.527
DINOv2	0.512

beddings into a better affinity metric.

Warm Start. GraPL is designed to train the same network continuously throughout all iterations. This is in contrast to similar iterative methods which prefer a “cold start,” re-initializing the parameters of the surrogate function before subsequent iterations. Preliminary tests showed that in our case, a “warm start” approach is preferred to re-initializing the network each time. These two approaches produce very different loss curves (Figure 3). Cold starts produce large spikes in loss at the beginning of each training iteration, whereas warm starts require only minor adjustments at these points. We expect that the first iterations of training provide important feature learning to the first layers of the network. By starting cold at each iteration, subsequent iterations are unable to benefit from the learned low-level feature detectors and therefore present a more unstable training phase.

Pairwise Energy Coefficient (λ). GraPL uses graph cuts to generate each new set of pseudo-labels, working on the theory that this graphical representation of the image provides an important spatial co-

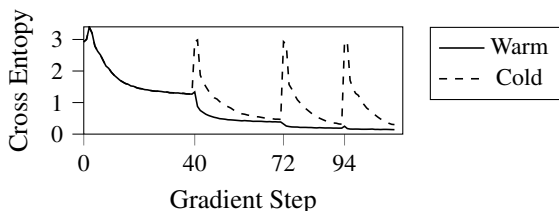


Figure 3: Comparison of loss curves using warm and cold starting methods, averaged over all test images in BSDS500 (Arbelaez et al., 2011). Here we consider the loss value at the end of each gradient step. On the x -axis, we depict the instants where a new training iteration starts.

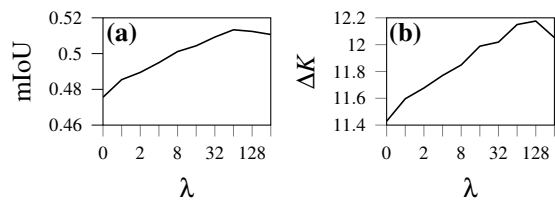


Figure 4: Effects of pairwise energy coefficient. (a) Effect of λ on mIoU. (b) Effect of λ on ΔK .

herence prior, which is perhaps missing from similar unsupervised methods and accounts for its success. Furthermore, GraPL relies on pairwise costs as well as the continuity loss to gradually decrease \hat{K} . To test these ideas, we evaluated the segmentation performance of the algorithm as well as ΔK across different values of λ , the hyperparameter that defines the scale of the pairwise energy as defined in Eq. 4.

When $\lambda = 0$, cutting any non-terminal edge incurs no cost. In this case, the function of the cut is effectively the same as the argmax clustering step found in (Kim et al., 2020), as pseudo-labels are entirely dependent on the current network response vectors. As λ increases, network response vectors are made less influential in the pseudo-label assignment process, as expected.

The results in Figure 4 demonstrate a logarithmic increase in segmentation performance as λ is raised from 0 through 64. However, increasing λ to values higher than 64 tends to result in comparatively poor performance. Because increasing λ strengthens pairwise connections, we would expect it to be closely correlated with ΔK . When $\lambda \leq 64$, we observe this behavior; however, higher values result in a plateau or slight decrease in ΔK .

In a configuration where the pairwise edges were uniformly weighted (or weighted according to spatial distance), we would expect higher than optimal values of λ to push ΔK too high and produce oversimplified segmentations, where multiple target segments are collapsed into a single predicted segment. However, when pairwise edges are weighted by patch encoder embedding affinity, pushing λ too high can instead result in an overly detailed segmentation, in which the graph cut considers the pairwise energy (dictated by the patch embeddings) more than the unary weights learned by GraPL.

Continuity Loss. Spatial continuity loss, first introduced in (Kim et al., 2020), provides GraPL a spatial coherence prior which penalizes the network directly at each gradient step, rather than through the graph cut produced pseudo-labels at the end of each iteration. Though shown effective in (Kim et al., 2020), we instinctively believed that it would be redundant

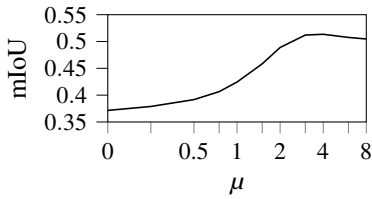


Figure 5: Effect of spatial continuity loss weight on mIoU.

in a graphically guided pipeline like GraPL. However, we observed that the combination of the two different spatial coherence priors produced more accurate segmentations than either one alone (Figure 5).

In practice, we noted that this loss has a different mechanism of action than the graphical coherence prior. In the absence of this spatial loss, GraPL employs a level of trust in the patch encoder that may be unfounded, as the pairwise energy only penalizes the separation of patches with a great affinity; but when using a patch encoder like DINO, which is defined by a large neural network, the edge weights may be high variance. In this case, increasing λ only serves to emphasize the patch encoder’s bias for certain edges. However, increasing the weight of the spatial continuity loss applies a higher penalty for *all* edges. In effect, it could be compared to an additive bias term in the pairwise energy function that raises the cost, no matter the patch affinity.

Patch Size. Patch-based approaches are faced with a choice between granularity (with smaller patches) and the information richness of input (with larger patches). In GraPL’s case, smaller patches also entail more complex graphs that take longer to solve, and larger patches entail higher memory usage. We found that setting d equal to 32 offered both optimal performance and near-optimal efficiency (Table 3).

4.3 Comparison to Other Methods

We compare the segmentation performance of GraPL on BSDS500 to four other unsupervised deep-learning methods: Differentiable Feature Clustering (DFC) (Kim et al., 2020), DoubleDIP (Gandelsman

Table 3: mIoU and segmentation time as a function of patch size. Time measurements are based on segmentation of BSDS500 test set.

d	mIoU	Seconds per Image
8	0.248	3.49
16	0.372	1.72
32	0.512	1.75
64	0.496	6.98

Table 4: Quantitative comparisons to other unsupervised deep-learning methods and baselines. † denotes the use of pretrained DINO ViT.

(a) Performance on BSDS500

Method	mIoU	pAcc
SLIC (RGB features)	0.137	0.416
SLIC (DINOv2 features)	0.258	0.280
W-Net (Xia and Kulis, 2017)	<u>0.428</u>	<u>0.531</u>
DoubleDIP (Gandelsman et al., 2019)	0.356	0.423
IIC (Ji et al., 2019)	0.172	-
DFC (Kim et al., 2020)	0.398	0.505
GraPL (proposed)	0.527	0.569

(b) Performance on COCO-stuff.

Method	mIoU	pAcc
IIC (Ji et al., 2019)	0.067	0.218
PiCIE (Cho et al., 2021)	0.144	0.500
TransFGU†(Zhaoyun et al., 2022)	0.175	0.527
SegDiscover (Huang et al., 2022)	0.143	0.401
STEGO†(Hamilton et al., 2022)	0.282	<u>0.569</u>
ACSeg†(Li et al., 2023)	0.164	-
GraPL (proposed)	<u>0.179</u>	0.613

et al., 2019), Invariant Information Clustering (IIC) (Ji et al., 2019), and W-Net (Xia and Kulis, 2017). We also tested two baselines that use SLIC to segment images based on RGB and DINOv2 (Oquab et al., 2023) patch embeddings (interpolated to the pixel level). These baselines were selected to demonstrate that the success of our method does not simply originate from its initialization or its pretrained guidance. We also compare GraPL to five popular methods on COCO-stuff (Caesar et al., 2018): IIC, PiCIE (Cho et al., 2021), TransFGU (Zhaoyun et al., 2022), SegDiscover (Huang et al., 2022), and ACSIg (Li et al., 2023).

Table 4 summarizes the quantitative comparative results of the above methods, where segmentation performance was measured in terms of both mIoU and pixel accuracy (pAcc) (Garcia-Garcia et al., 2017). As with mIoU, pixel accuracy was computed using a one-to-one label matching strategy. Figure 6 displays some segmentation results from the above methods for qualitative comparison.

Compared to the other unsupervised methods tested on BSDS500, GraPL decomposes complex foregrounds into detailed yet semantically salient components. Notice how GraPL can pick up on small details like sunglasses in the distance while ignoring less relevant features of the image, such as creases in clothing. In many cases, it can handle color gra-

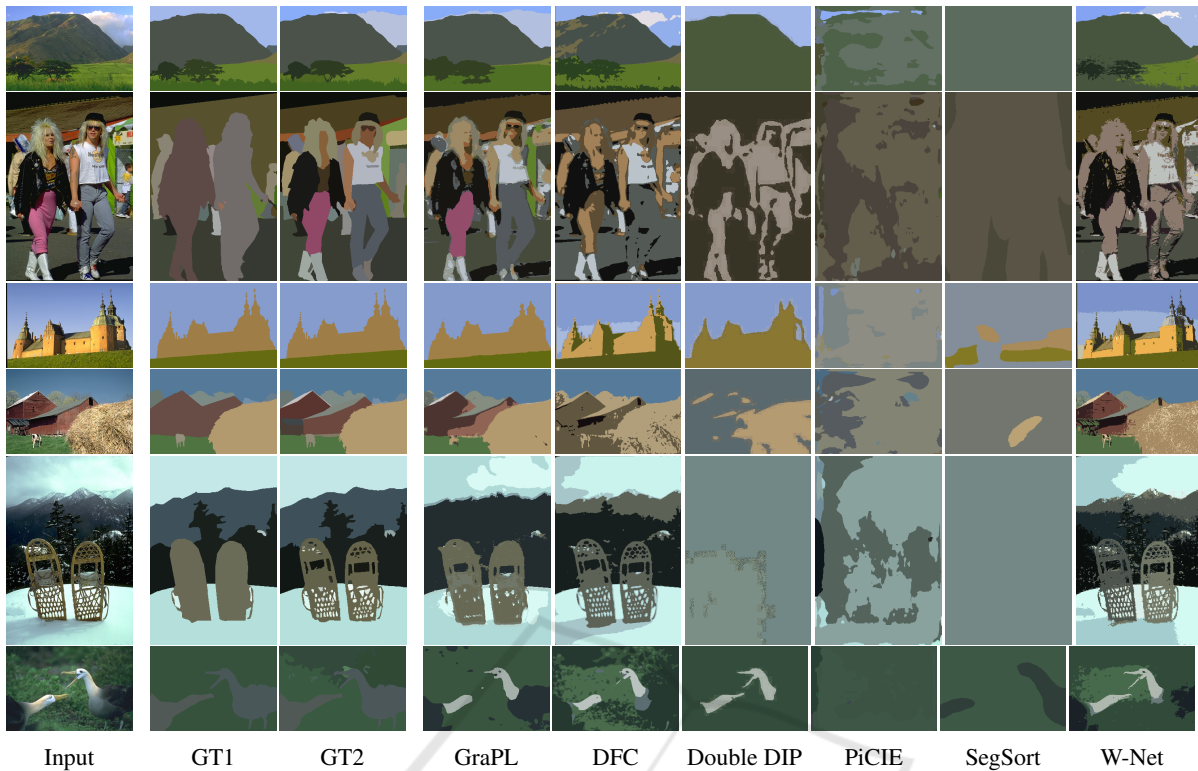


Figure 6: Qualitative comparison of GraPL to other deep learning-based unsupervised segmentation methods. We selected two of the available ground truth segmentations (GT1 and GT2) for comparison, one more detailed and one less detailed.

dent variation, usually present in sky backgrounds or shadow regions. On occasion, GraPL detects segments that are not present in the ground truth, such as the bird heads on the last qualitative example, which, despite being reasonable, decreases its quantitative performance. Finally, it also struggles to detect fine structures, such as castle tops, small holes, and bird beaks. Despite that, our proposed method outperforms all of the compared methods by at least 7.2% in accuracy and 23.1% mIoU on BSDS500 (Table 4a). It is also worth noting the low performance of our baselines when compared to GraPL. This demonstrates that our method does not merely rest on the success of our initializer, SLIC. Instead, GraPL’s success is a product of its training and inference methodology.

While GraPL falls short of first place mIoU on COCO-stuff (Table 4b), it bears repeating that: (1) its default configuration has just 23k parameters, making it 200 times smaller than the next smallest competitor (IIC); (2) it does not require any pretraining, while STEGO, ACSeg, and TransFGU use DINO initialization; and (3) it operates in a zero-shot paradigm, while all competing methods require a full training dataset. In spite of its remarkably low complexity, it scores second in mIoU (or first excluding DINO-based models) and is 7.7% more accurate than its closest com-

petitor. On the other hand, GraPL leads the pixel accuracy (pAcc) performance on both datasets.

5 CONCLUSION

In this paper, we introduce GraPL, a deep learning-based unsupervised segmentation framework that learns a pixel-level classifier by solving a patch clustering surrogate task. GraPL is the first deep learning method to employ a graph cut regularizer during training, encouraging spatial coherence and leveraging the discriminative power of patch embeddings. Furthermore, it seamlessly translates patch-level learning to the pixel-level without the need for postprocessing. Our experiments reveal the promising capabilities of our algorithm, as it can outperform many state-of-the-art unsupervised segmentation methods, despite operating in a zero-shot paradigm and being several orders of magnitude less complex than its closest competitor. As such, GraPL constitutes a remarkably strong baseline for unsupervised segmentation. Our work can be seen as further evidence of the benefit of using graph cuts in deep learning, especially in the context of unsupervised segmentation.

REFERENCES

- Abend, K., Harley, T., and Kanal, L. (1965). Classification of binary random patterns. *IEEE Transactions on Information Theory*, 11(4):538–544.
- Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282.
- Arbelaez, P., Maire, M., Fowlkes, C., and Malik, J. (2011). Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916.
- Ayed, I. B., Chen, H.-m., Punithakumar, K., Ross, I., and Li, S. (2010). Graph cut segmentation with a global constraint: Recovering region distribution via a bound of the bhattacharyya measure. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3288–3295. IEEE.
- Besag, J. (1986). On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 48(3):259–279.
- Boykov, Y., Veksler, O., and Zabih, R. (2001). Fast approximate energy minimization via graph cuts. *IEEE Transactions on pattern analysis and machine intelligence*, 23(11):1222–1239.
- Boykov, Y. Y. and Jolly, M.-P. (2001). Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Proceedings eighth IEEE international conference on computer vision. ICCV 2001*, volume 1, pages 105–112. IEEE.
- Caesar, H., Uijlings, J., and Ferrari, V. (2018). Coco-stuff: Thing and stuff classes in context. In *Computer vision and pattern recognition (CVPR), 2018 IEEE conference on*. IEEE.
- Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. (2020). End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer.
- Caron, M., Bojanowski, P., Joulin, A., and Douze, M. (2018). Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pages 132–149.
- Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. (2021). Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660.
- Chan, T. F. and Vese, L. A. (2001). Active contours without edges. *IEEE Transactions on image processing*, 10(2):266–277.
- Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2017a). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848.
- Chen, L.-C., Papandreou, G., Schroff, F., and Adam, H. (2017b). Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*.
- Cho, J. H., Mall, U., Bala, K., and Hariharan, B. (2021). Picie: Unsupervised semantic segmentation using invariance and equivariance in clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16794–16804.
- El-Zehiry, N. Y. and Grady, L. (2010). Fast global optimization of curvature. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3257–3264. IEEE.
- Gandelsman, Y., Shocher, A., and Irani, M. (2019). "double-dip": unsupervised image decomposition via coupled deep-image-priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11026–11035.
- Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., and Garcia-Rodriguez, J. (2017). A review on deep learning techniques applied to semantic segmentation.
- Hamilton, M., Zhang, Z., Hariharan, B., Snavely, N., and Freeman, W. T. (2021). Unsupervised semantic segmentation by distilling feature correspondences. In *International Conference on Learning Representations*.
- Hamilton, M., Zhang, Z., Hariharan, B., Snavely, N., and Freeman, W. T. (2022). Unsupervised semantic segmentation by distilling feature correspondences. *arXiv preprint arXiv:2203.08414*.
- Han, K., Wang, Y., Guo, J., Tang, Y., and Wu, E. (2022). Vision gnn: An image is worth graph of nodes. *Advances in Neural Information Processing Systems*, 35:8291–8303.
- Huang, H., Chen, Z., and Rudin, C. (2022). Segdiscover: Visual concept discovery via unsupervised semantic segmentation.
- Hwang, J.-J., Yu, S. X., Shi, J., Collins, M. D., Yang, T.-J., Zhang, X., and Chen, L.-C. (2019). Segsort: Segmentation by discriminative sorting of segments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7334–7344.
- Ji, X., Henriques, J. F., and Vedaldi, A. (2019). Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9865–9874.
- Kim, W., Kanezaki, A., and Tanaka, M. (2020). Unsupervised learning of image segmentation based on differentiable feature clustering. *IEEE Transactions on Image Processing*, 29:8055–8068.
- Kolmogorov, V. and Boykov, Y. (2005). What metrics can be approximated by geo-cuts, or global optimization of length/area and flux. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 1, pages 564–571. IEEE.
- Krähenbühl, P. and Koltun, V. (2011). Efficient inference in fully connected crfs with gaussian edge potentials. *Advances in neural information processing systems*, 24.
- Li, K., Wang, Z., Cheng, Z., Yu, R., Zhao, Y., Song, G., Liu, C., Yuan, L., and Chen, J. (2023). Acseg: Adaptive

- conceptualization for unsupervised semantic segmentation.
- Li, S. Z. (2009). *Markov random field modeling in image analysis*. Springer Science & Business Media.
- Li, Y. and Borovec, J. (2023). pygco. <https://github.com/Borda/pyGCO>.
- Lin, Q., Zhong, W., and Lu, J. (2021). Deep superpixel cut for unsupervised image segmentation. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 8870–8876. IEEE.
- Marin, D., Tang, M., Ayed, I. B., and Boykov, Y. (2019). Beyond gradient descent for regularized segmentation losses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10187–10196.
- Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., and Terzopoulos, D. (2021). Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3523–3542.
- Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al. (2023). Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*.
- Ouali, Y., Hudelot, C., and Tami, M. (2020). Autoregressive unsupervised image segmentation. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VII 16*, pages 142–158. Springer.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H., Larochelle, H., Beygelzimer, A., d’Alché Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.
- Ren, Y., Pu, J., Yang, Z., Xu, J., Li, G., Pu, X., Yu, P. S., and He, L. (2022). Deep clustering: A comprehensive survey. *arXiv preprint arXiv:2210.04142*.
- Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015, Proceedings, Part III 18*, pages 234–241. Springer.
- Rother, C., Kolmogorov, V., and Blake, A. (2004). ”grabcut” interactive foreground extraction using iterated graph cuts. *ACM transactions on graphics (TOG)*, 23(3):309–314.
- Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905.
- Tang, M., Gorelick, L., Veksler, O., and Boykov, Y. (2013). Grabcut in one cut. In *Proceedings of the IEEE international conference on computer vision*, pages 1769–1776.
- Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al. (2021). Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems*, 34:24261–24272.
- Trockman, A. and Kolter, J. Z. (2022). Patches are all you need? *arXiv preprint arXiv:2201.09792*.
- Wang, X., Girdhar, R., Yu, S. X., and Misra, I. (2023). Cut and learn for unsupervised object detection and instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3124–3134.
- Wang, Y., Shen, X., Yuan, Y., Du, Y., Li, M., Hu, S. X., Crowley, J. L., and Vafreydaz, D. (2022). Token-cut: Segmenting objects in images and videos with self-supervised transformer and normalized cut. *arXiv preprint arXiv:2209.00383*.
- Xia, X. and Kulis, B. (2017). W-net: A deep model for fully unsupervised image segmentation. *arXiv preprint arXiv:1711.08506*.
- Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., and Luo, P. (2021). Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34:12077–12090.
- Yu, H., Yang, Z., Tan, L., Wang, Y., Sun, W., Sun, M., and Tang, Y. (2018). Methods and datasets on semantic segmentation: A review. *Neurocomputing*, 304:82–103.
- Zhaoyun, Y., Pichao, W., Fan, W., Xianzhe, X., Hanling, Z., Hao, L., and Rong, J. (2022). Transfgu: A top-down approach to fine-grained unsupervised semantic segmentation. In *European Conference on Computer Vision*, pages 73–89. Springer.
- Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., and Torr, P. H. (2015). Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1529–1537.