DeepCut: Unsupervised Segmentation using Graph Neural Networks Clustering

Amit Aflalo   Shai Bagon   Tamar Kashti   Yonina eldar

Faculty of Mathematics and Computer Science, Weizmann Institute of Science
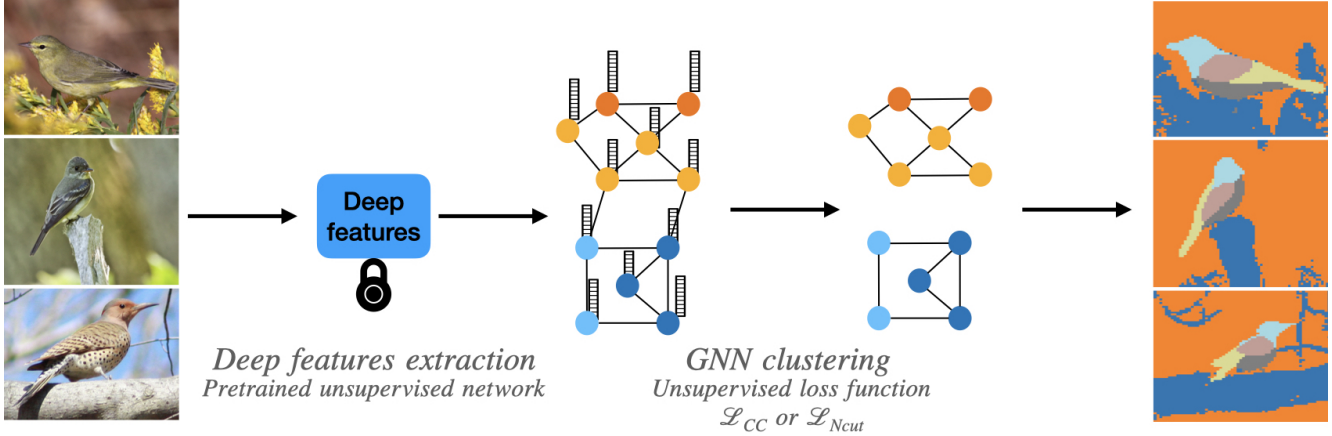
Figure 1. **DeepCut:** We use graph neural networks with unsupervised losses from classical graph theory to solve various image segmentation tasks. We use deep features from a pre-trained vision transformer (ViT) as input to our network, thus avoiding expensive training associated with end-to-end methods.

Abstract

Image segmentation is a fundamental task in computer vision. Data annotation for training supervised methods can be labor-intensive, motivating unsupervised methods. Some existing approaches extract deep features from pretrained networks and build a graph to apply classical clustering methods (e.g., k-means and normalized-cuts) as a post-processing stage. These techniques reduce the high-dimensional information encoded in the features to pair-wise scalar affinities. In this work, we replace classical clustering algorithms with a lightweight Graph Neural Network (GNN) trained to achieve the same clustering objective function. However, in contrast to existing approaches, we feed the GNN not only the pair-wise affinities between local image features but also the raw features themselves. Maintaining this connection between the raw feature and the clustering goal allows to perform part semantic segmentation implicitly, without requiring additional post-processing steps. We demonstrate how classical clustering objectives can be formulated as self-supervised loss functions for training our image segmentation GNN. Additionally, we use the Correlation-Clustering (CC) objective to perform clustering without defining the number of clusters (k-less clustering). We apply the proposed method for object localization, segmentation, and semantic part segmentation tasks, surpassing state-of-the-art performance on multiple benchmarks. [1].

1. Introduction

Object localization and segmentation are used in various real-world applications such as autonomous cars, robotics, and medical diagnosis. This is a long-standing problem in computer vision, with a vast amount of work invested in improving accuracy of these tasks.

Current state-of-the-art performance is achieved using supervised Deep Nural Networks (DNNs). However the lack of annotated data limits applicability of these methods. Data annotation can be labor intensive, thus making it expensive; moreover, in some specific fields like medical imaging, data must be annotated by a domain expert, adding to the complexity of acquiring accurate annotations. Various solutions have been proposed to address this challenge; leveraging color data, adding priors such as boundaries or scribbles [1], semi-supervised learn-

---

[1]Code will be made publicly available.

ing [2], weakly-supervised learning [3], and more. These approaches achieve limited success, as they require some form of annotations or knowledge of the image structure (such as scribbles or boundaries).

One approach to tackle this problem is to consider *unsupervised* methods. Recent research on unsupervised DNNs shows promising results; for instance Caron et al. [4] showed that training Vision Transformers (ViTs) using self-distillation with No labels (DINO) makes their attention maps correspond to semantic segments in the input image. Later, it was shown that the deep features of the trained transformer have strong semantic meaning and can be leveraged for various visual tasks, such as object localization, segmentation, and semantic segmentation [5–7]. Some recent work with deep features demonstrate promising results; for example, current state-of-the-art unsupervised techniques use deep features combined with classical graph theory [6, 7] for object localization and segmentation tasks. Those algorithms are lightweight and rely on pre-trained unsupervised networks. In contrast, end-to-end approaches to unsupervised segmentation must train from scratch, requiring a lot of time and resources.

The lightweight methods above utilize deep features from vision Transformers to construct an adjacency matrix representing a weighted graph without node features. They then apply classical graph clustering methods to the problem. This has a significant disadvantage as it discards most of the deep features high dimensionality information. Here we propose keeping the deep features data for the clustering process and doing so while still relying on classical graph theory. To achieve this goal, we apply *Graph Neural Networks* (GNN) while using classical graph clustering as a loss function for our GNN architecture.

GNN is a class of neural networks which directly operates on graph-structured data. GNNs have been shown to achieve outstanding results in numerous fields, allowing to solve complicated problems, from protein folding with AlphaFold [8] to drug discovery [9] through traffic prediction [10]. Our work employs GNNs for computer vision tasks such as object localization, segmentation, and semantic part segmentation. But unlike previous methods, we do not train the network end to end. We use deep features from a pre-trained network and train the GNN using unsupervised losses from classical graph theory.

More specifically, we propose *DeepCut*, a lightweight unsupervised GNN segmentation algorithm. We employ GNNs to segmentation tasks with unsupervised losses driven by classical graph theory. We use deep features inside the clustering process, while previous methods [6, 7] discard the deep features data and use only the correlation between those features. We achieve two main advantages by using the features directly: first, we observe better object segmentation performance. Second, because we use the deep features in the clustering process, we learn to cluster similar features on different graphs to the same clusters, thus performing semantic segmentation over multiple graphs, where each graph corresponds to a separate image. To further improve segmentation performance, we use two-stage segmentation, first segmenting foreground and background and then applying segmentation individually to each part. This method is derived from the mathematical function used to optimize the GNN approach, which is biased to produce equally sized clusters. We also introduce a method to cluster data without defining in advance the number of clusters k into which to cluster the data, which we refer to as k-less clustering.

Our contributions are as follows:

- Performing object localization and segmentation while surpassing unsupervised state of the art performance.

- Performing two-stage segmentation. Applying background and foreground segmentation separately using our GNN approach.

- Performing semantic part segmentation on multiple images without co-segmentation or any post-processing steps; applying DeepCut only image-wise and not dataset-wise, as appose to most unsupervised methods.

- Utilizing the correlation clustering function for deep features clustering and achieving k-less clustering.

2. Background

Before the deep-learning surge, many classical approaches established quantitative criteria for segmentation founded on principles in graph theory. More specifically, they expressed affinities between image regions as a graph and associated different partitions of the image with cuts in that graph (e.g. [11, 12]). Good image segments corresponded to optimal cuts. These methods required no supervision; all they took were the affinities between image regions. We briefly overview the relevant approaches.

2.1. Graph Clustering

We leverage two graph clustering functionals from classical graph theory in our work, *normalized cut* [11] and *correlation clustering* [12].

Notations Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph induced by an image. Each node represents an image region, and the weights w_{ij} represent the affinity between image regions i and j, $i, j = 1 \ldots n$. Let W be an $n \times n$ matrix whose entries are w_{ij}.

Our goal is to partition this graph into k disjoint sets $A_1, A_2 \ldots A_k$ such that $\cup_i A_i = \mathcal{V}$ and $\forall_{j \neq i} A_i \cup A_j = \emptyset$. This

partition can be expressed as a binary matrix $S \in \{0,1\}^{n \times k}$ where $S_{ic} = 1$ iff $i \in A_c$.

Normalized Cut (N-cut) [11] A good partition is defined as one that maximizes the number of within-group connections, and minimizes the number of between-group connections. The number of between-group connections can be computed as the total weight of edges removed and described in graph theory as a *cut*:

$$cut(A, B) = \sum_{u \in A, v \in B} w(u, v). \tag{1}$$

This objective is formulated by the *normalized cut* (N-cut) functional:

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, \mathcal{V})} + \frac{cut(A, B)}{assoc(B, \mathcal{V})}, \tag{2}$$

where $assoc(A, \mathcal{V}) = \sum_{i \in A, j \in \mathcal{V}} w_{ij}$ is the total affinities connecting nodes of A to all nodes in the graph. The N-cut formulation can be easily extended to $K > 2$ segments. Shi and Malik [11] also suggested an approximated solution to Eq. (2) using spectral methods, known as *spectral clustering*, where the spectrum (eigenvalues) of a graph Laplacian matrix is used to get approximated solution to the N-cut problem.

Correlation Clustering (CC) [12] When the graph, derived from an image contains *both negative and positive* affinities, N-cut is no longer applicable. In this case a good image segment may be one that maximizes the *positive* affinities inside the segment and the *negative* ones across segments [12]. This objective can be formulated by the *correlation clustering* (CC) functional. Given a matrix W, an optimal partition U minimizes:

$$CC(S) = -\sum_{ij} W_{ij} \sum_{c} S_{ic} S_{jc}. \tag{3}$$

Correlation clustering utilizes intra-cluster disagreement (repulsion) to *automatically* deduce the number of clusters K [12] so that it does not need to know K in advance. A thorough probabilistic analysis of this functional can be found in [13, 14].

2.2. Graph Neural Networks

GNN is a class of neural networks which directly operates on graph-structured data. We can divide a GNN layer into two key operations, **message passing** and **aggregation**. **Message passing** is an operation to get information from each node's neighbors, and is applied to every node in the graph. That information can include any combination of node features, edge features, or any other data that is embedded in the graph.

Aggregation is how we fuse all the messages from the message passing phase into one message that updates the current node state.

In our work, we use a Graph Convolutional Network (GCN) [15]. For a GCN in layer k, each node h_v in the GCN is processed by:

$$h_v^{(k+1)} = \sum_{u \in \mathcal{N}(v)} \Theta \frac{h_u^{(k)}}{|\mathcal{N}(v)|}, \tag{4}$$

where Θ are the model parameters. $\mathcal{N}(v)$ denotes the neighbours of a node h_v, and $|\mathcal{N}(v)|$ is the number of neighbours. The GCN aggregation is the summation operator, and the term inside the summation operator is the message passing term. The network is then trained with a loss function objective to optimize the network parameters Θ and thus creating the most meaningful messages to pass in order to optimize loss function.

3. Method

Our method is based on clustering deep features extracted from pretrained unsupervised network. We utilize clustering of the deep features to perform unsupervised object localization, segmentation and semantic part segmentation. We employ GNNs with classical graph theory clustering functions as a loss function for our network, thus applying graph clustering using GNNs. We compare the use of different classical algorithms in conjunction with GNNs and show how to find good pairs of clustering algorithms and unsupervised training methods for deep features extraction such as DINO [4], MoCo (Momentum Contrast for unsupervised Visual Representation Learning) [16] and MAE (Masked Auto-Encoder) [17].

3.1. GNN Image Clustering

3.1.1 From Deep Features to Graphs

As presented in Figure 2, given an image M of size $m \times n$ and p channels, we pass it through a transformer T. The transformer splits the image to $\frac{mn}{p^2}$ patches, where p is the patch size of transformer T. We extract the transformer's internal representation for each of these patches, using the *key* token from the last layer, as it was shown this performs the best in this in various tasks [5, 6]. The result is a feature vector that contains all the extracted features of the different patches $f_{(\frac{mn}{p^2} \times d)}$, where d is the token embedding dimension.

Let $G = (V, E)$ be a weighted graph with W as a weight matrix. We build a patch-wise correlation matrix from ViT obtained features:

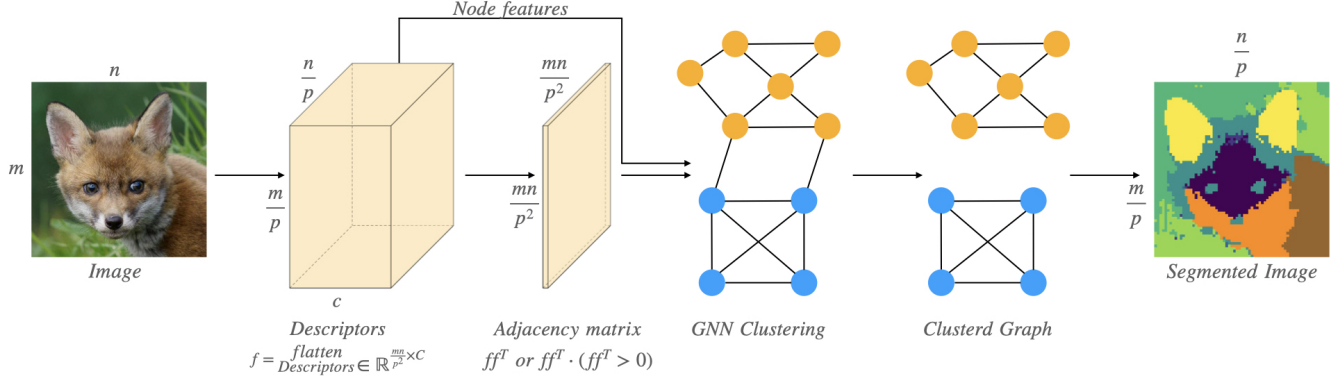$$W = ff^T \in \mathbb{R}^{\frac{mn}{p^2} \times \frac{mn}{p^2}}. \tag{5}$$

Figure 2. **Method overview:** Given an image, we extract deep features from unsupervised trained ViT; using those features, we construct a similarity matrix that describes patch-wise features similarity and serves as our new adjacency matrix. We build a matching graph using this adjacency matrix and our deep features as node features. We use the GNN clustering method described in the paper to split the graph into k distinct clusters for various downstream tasks.

For correlation clustering the negative weights (reputations) are well defined and have valuable data in correlation clustering functional. In contrast, the normalized cut functional only accepts positive weights, therefore we threshold at zero:

$$W = ff^T \cdot (ff^T > 0) \in \mathbb{R}^{\frac{mn}{p^2} \times \frac{mn}{p^2}}. \quad (6)$$

For correlation clustering, we introduce a *k sensitivity* hyper-parameter α. As we cannot choose the number of clusters with correlation clustering, we need a method to adjust the sensitivity of the cluster choosing process, where higher sensitivity equals more clusters. We enforce it in the following manner:

$$W = ff^T - \frac{max(ff^T)}{\alpha}. \quad (7)$$

The *k sensitivity* parameter α ranges between 1 and infinity. Lower α value corresponds to higher repulsion forces between nodes (negative weights in W) and thus higher cluster count, as correlation clustering maximizes negative affinities between segments in the graph.

3.1.2 Graph Neural Network Clustering

Let N be a node feature matrix obtained by one or more layers of of GNN convolution on a graph G with adjacency matrix W, in our case a GCN and patch-wise correlation matrix from ViT obtained features (Eq. (5),Eq. (6),Eq. (7)). Let **S** be the output of a Multi-Layer Perception (MLP) with a softmax function applied on N:

$$\begin{aligned} \hat{N} &= GNN(N, W; \Theta_{GNN}), \\ S &= MLP(\hat{N}; \Theta_{MLP}), \end{aligned} \quad (8)$$

where Θ_{MLP} and Θ_{GNN} are trainable parameters and S, our GNN output, is the cluster assignment matrix consisting

of vectors that define the node's probability to belong to a specific cluster.

This GNN is trained unsupervised, where the loss function is the normalized-cut relaxation proposed in [18], or our new proposed method with correlation clustering objective.

The loss function in the case of normalized cut is:

$$Ncut_{loss} = \frac{Tr(S^T WS)}{Tr(S^T DS)} + \left\| \frac{S^T S}{\|S^T S\|_F} - \frac{\mathbb{I}_K}{\sqrt{K}} \right\|_F, \quad (9)$$

where D is the row-wise sum diagonal matrix of W, K is the number of disjoined sets we want to partition this graph into and, $\mathbb{I}_K$ is the identity matrix. The first term encourages strongly connected components to be clustered together; while the second term encourages the clusters assignments to be orthogonal and to be of similar size.

The loss function for correlation clustering [12] is:

$$CC_{loss} = -Tr(WSS^T). \quad (10)$$

This term encourages intra-cluster agreement where clusters are separated by repulsion (negative affinities). W is chosen to be Eq. (6) for N-cut and Eq. (7) For CC. For CC we will chose a k-sensitivity value to set the sensitivity of the clustering process, for an example see Sec. 7.

3.1.3 Graph Neural Network Segmentation

In Sec. 3.1.1 we suggest to build a graph from deep features extracted from unsupervised trained ViTs, where each node represents a patch of the original image. We cluster those nodes in the graph to disjoint sets that represent different segments in the image. For the clustering process we use graph neural network clustering as in Sec. 3.1.2 with one of the CC or N-cut losses. While previous approaches used
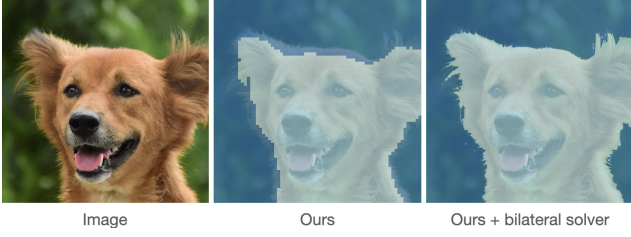
Figure 3. **Segmentation refinement.** Single object segmentation with our method and bilateral solver [19] as a complementary step to refine the obtained segmentation.

Figure 4. **Proposed two-stage clustering:** First, we cluster the image into two disjoined sets, then apply clustering to the foreground and background separately.

the N-cut loss for similar tasks [18], we suggest using the correlation clustering functional as a loss thus utilizing negative features for graph building as in Eq. (5) and Eq. (7). Moreover we use deep features as node features for graph building, while previous methods used only the correlation matrix of the deep feature [6, 7, 18], allowing us to perform feature classification while clustering and perform semantic part segmentation implicitly without using post processing steps used by previous methods [6].

3.1.4 Resolution Manipulation

In this work, we utilize deep features extracted from ViTs. Those features represent the image patches corresponding to the ViT patch size $p \times p$. We perform our segmentation patch-wise thus, for image size $(m \times n)$ our segmentation resolution is $(\frac{m}{p} \times \frac{n}{p})$. For higher resolution segmentation, we change the ViT stride value to $\frac{p}{2}$ instead of p as it doubles the number of patches the ViT uses, doubling the resolution of our segmentation to $(\frac{2m}{p} \times \frac{2n}{p})$.

In order to improve the segmentation resolution further, a bilateral solver [19] can be added as a complementary step to refine the boundaries of the obtained segmentation: see Figure 3.

3.2. Object Localization

Object localization refers to identifying the location of the primary objects in an image and drawing a bounding box around its extent. To perform localization we follow the steps below: (1) Use our GNN clustering method with $k = 2$. (2) Look at the edges of the clustered image and choose the cluster that appears on more then two edges to be the background, and the other to be our main object. (3) Apply a bounding box around it.

3.3. Object Segmentation

Object segmentation refers to the process of separating the foreground object in a given image, also called foreground-background segmentation. The method used here is identical to object localization, without stage 3.
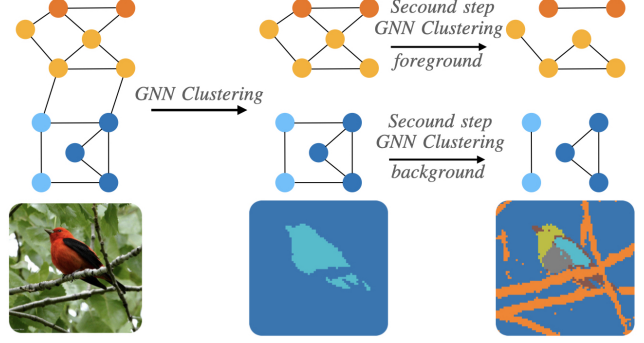
3.4. Semantic Part Segmentation

Our method can perform semantic part segmentation without accessing all of the dataset's images; the deep features encode enough semantic information, so that we can pass images sequentially through our network and still get reliable semantic part segmentation. No costly post-processing for comparing the entire dataset is needed.

In both of our optimization methods, we suggest performing semantic part segmentation in two stages if refined semantic part segmentation is needed, see Figure 4. First, find the object of interest using object segmentation with $k = 2$. Then use the GNN separately on the two clusters created. Using this two-step clustering allows to control the detail level of the segmentation from the image level to the object level and provide more accurate results. This method is derived from our objective functions, which are biased to produce equally sized clusters.

4. Implementation details

For all experiments, we use DINO [4] trained ViT-S/8 transformer for feature extraction; specifically, we use the *keys* features from the last layer of the DINO trained "student" transformer. We use pre-trained weights provided by the DINO paper writers and, trained on ImageNet [20]. **We do not conduct any training of the transformer on any of the tested datasets.**

We solve our optimization problem image-wise; For object localization and object segmentation each image is passed for 10 epochs in the GNN. For semantic segmentation or part segmentation each image is passed for 100 epochs in the GNN. There is no need to train the GNN model on the entire dataset, the model is trained per image and does not need large amount of data to converge.

In this approach we can process approximately five images/sec on a GPU (Tesla V100). We also trained the GNN once on an entire arbitrary image dataset and generalized for

Method	VOC-07	VOC-12	COCO-20k
Selective Search [21]	18.8	20.9	6.0
EdgeBoxes [22]	31.1	31.6	28.8
DINO-[CLS] [4]	45.8	46.2	42.1
LOST [23]	61.9	64.0	50.7
Spectral Methods [6]	62.7	66.4	52.2
TokenCut [7]	68.8	72.1	58.8
DeepCut: CC loss	68.8	67.9	57.6
DeepCut: N-cut loss	**69.8**	**72.2**	**61.6**

Table 1. **Object localization results.** CorLoc metric (percentage of images with $IOU > 0.5$). CC and N-cut denotes correlation clustering and Normalized Cut respectively.

Method	CUB	DUTS	ECSSD
OneGAN [24]	55.5	-	-
Voynov et al. [25]	68.3	49.8	-
Spectral Methods [6]	76.9	51.4	73.3
TokenCut [7]	-	57.6	71.2
DeepCut: CC loss	77.7	56.0	73.4
DeepCut: N-cut loss	**78.2**	**59.5**	**74.6**

Table 2. **Single object segmentation results.** mIOU (mean intersection-over-union) CC denotes correlation clustering and N-cut Normalized Cut.

other datasets, in order to perform inference without training on every image. This accelerated our throughput, but it yields slightly less accurate results.

5. Results

We evaluate our method on three unsupervised tasks: single object localization, single object segmentation, and semantic part segmentation. We use popular benchmarks to compare our method with other unsupervised approaches published on those tasks. We show results for both normalized-cut and correlation clustering GNN objectives.

5.1. Object Localization

In Table 1, we evaluate our performance in unsupervised object localization task on three datasets, PASCAL VOC 2007 [26], PASCAL VOC 2012 [27], and COCO20K (20k images chosen from MS-COCO dataset [28] introduced in previous work [29]). We compare our unsupervised approach to state-of-the-art unsupervised methods. We report our results in the Correct Localization metric (CorLoc), defined as the percentage of images whose intersection-over-union with the ground truth label is greater than 50%. DeepCut with N-cut losses gives the best results on all data sets, and DeepCut with correlation clustering loss surpasses all
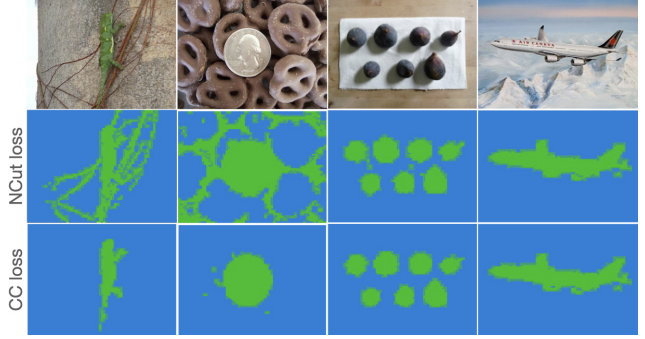

Figure 5. **DeepCut segmentation: N-cut vs. CC losses.** results. Top: original image, middle: DeepCut with N-cut loss, bottom: DeepCut with correlation clustering loss. Note that for images with complex background DeepCut with CC loss outperforms DeepCut with N-cut loss.

previous methods except TokenCut [7].

5.2. Single Object Segmentation

In Table 2, we evaluate our performance in unsupervised single object segmentation on three datasets, CUB (widely-used dataset of birds for fine-grained visual categorization task) [30], DUTS (the largest saliency detection benchmark) [31] and ECSSD (Extended Complex Scene Saliency Dataset) [32]. We report our results in mean intersection-over-union (mIoU). DeepCut with N-cut losses archives the best results on all data sets.Visual comparison of segmentation with our two losses is presented in Figure 5; note how the CC loss segments the foreground better than the Ncut loss when background for specific images that includes different objects and shadows. As this is not the case in the majority of images, the N-cut usually outperforms the CC loss.

5.3. Semantic Part Segmentation

Our DeepCut method performs semantic part segmentation while being exposed to every image sequentially for images from the same modality, such as animals, cars, and people. There is no need for the model to see all the images before performing semantic part segmentation, therefore no co-segmentation or post-processing step to cluster pseudo-labels is needed. That is one of the advantages of learning from deep features with GNN compared to classical methods that learn only from correlations and discard the deep features high dimensionality data. Our model takes advantage of the complex semantic data stored at the deep features and performs semantic part segmentation implicitly.

In Table 3, we evaluate our approach on the CUB [30] dataset. We report our results in Normalized Mutual Info score (NMI) and Adjusted Rand Index score (ARI) measures on the entire data test set. We perform two-step seg-
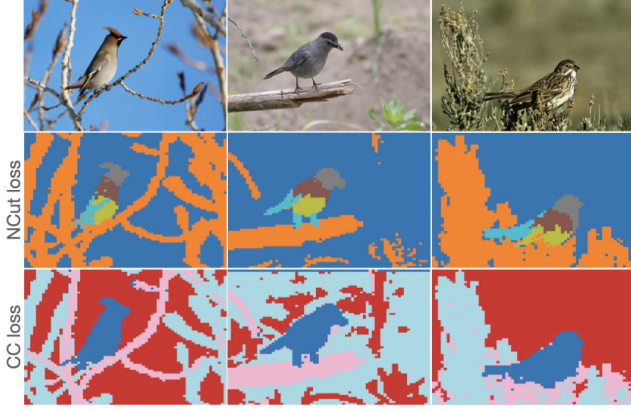
Figure 6. **Semantic part segmentation: N-cut vs. CC losses.** Top: original image, middle: normalized-cut loss with our proposed two-step clustering; bottom: correlation clustering with one-step k-less clustering.

Method	NMI	ARI
SCOPS [33] (model)	24.4	7.1
Huang and Li [34]	26.1	13.2
Choudhury et al. [35]	43.5	19.6
DFF [36]	25.9	12.4
Amir et al. [36]	38.9	16.1
DeepCut: N-cut loss	**43.9**	**20.2**

Table 3. **Semantic part segmentation results.** ARI and NMI over the entire CUB-200 dataset are used to evaluate cluster quality. Predictions were performed with $k = 4$ with our method using the N-cut objective. First three methods use ground truth foreground masks as supervision.

mentation as shown in Figure 4; the first step is foreground-background segmentation ($k = 2$); the second step is to perform semantic part segmentation on the foreground object using $k = 4$. The reason for the two-step segmentation is that the normalized-cut objective enforces cluster sizes to be close to each other, and that prevents obtaining a detailed segmentation for the foreground object. The same process can be applied to the background to get better background segmentation, as in Figure 1 and Figure 6. A comparison between our technique and deep spectral method that uses classical graph theory for deep features based segmentation [18] is presented in Figure 7. As seen at Table 3, Deep-Cut with normalized-cut loss surpasses all other methods including the top three methods [33–35] that uses ground-truth foreground masks as supervision.

6. Deep Features Selection

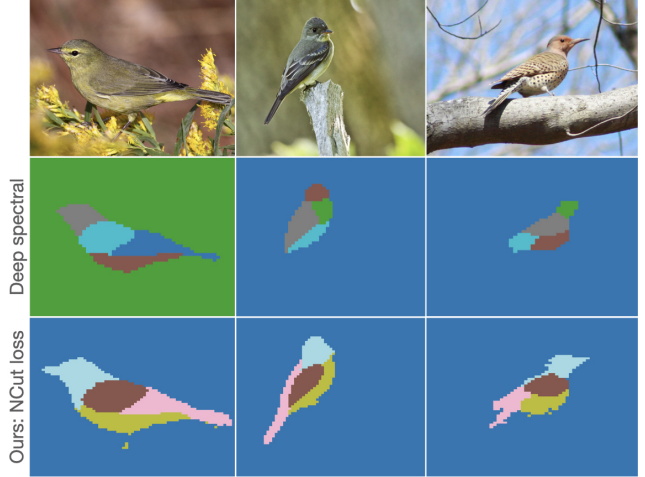Our work and previous unsupervised segmentation methods rely heavily on the correlation between deep fea-

Figure 7. **Semantic part segmentation. deep spectral method vs. DeepCut with N-cut loss.** Top: original image; middle: deep spectral method [6]; bottom: our segmentation with N-cut loss. The deep spectral method failed to preform semantic segmentation across all three images.

Pretraining	DeepCut N-cut	DeepCut CC	Negative percentage
DINO [4]	59.4	59.8	33.6
MoCo-v3 [16]	62.3	43.1	20
MAE [17]	47.2	31.2	5.4

Table 4. **Pretraining.** Object-localization performance on PAS-CAL VOC07 [26] with different unsupervised training methods for acquiring deep features. All methods use ViT-B-16 architecture, and were trained on ImageNet [20]. We provide mIOU (mean Intersection-Over-Union) for correlation clustering (CC) and normalized-cut (N-cut) objective. We also provide the mean percentage (across all the dataset) of negative weights in the corresponding affinity matrix. Note the correlation: the higher the percentage of negative weights of the transformer, the better the mIOU of the DeepCut with CC loss; for DINO DeepCut CC loss outperforms N-cut loss for this dataset.

tures of different patches. In this work, we proposed to use the negative correlations (correlation clustering loss) discarded in previous works, thus utilizing all available information. Unsupervised segmentation methods that rely on deep features can be only as good as the quality of deep features. Moreover, there is no "one size fits all" solution, as different methods need different information to work optimally. For example, this paper presents two techniques, one based on correlation clustering and the other on normalized cut. Correlation clustering relies on positive and negative correlations between features, and normalized cut only accepts positive correlations.

In Table 4 we compare three unsupervised ViT train-

Number of classes	3 classes	4 classes	5 classes
Connected Components	33.3 $\pm$0	25.0 $\pm$0	20.0 $\pm$0
Spectral Clustering	85.5 $\pm$14.1	77.9 $\pm$6.6	82.8 $\pm$6.9
DeepCut: cc loss	98.3 $\pm$0.9	97.1 $\pm$1.7	99.27 $\pm$0.6

Table 5. k-**less clustering.** Clustering with DeepCut and correlation clustering loss on Fashion Product Images Dataset [37]. We take a subset of images from 5 classes: Top-wear, Shoes, Bags, Eye-wear, and Belts. and cluster them to different classes using our k-less method. We show out result as classification purity. we use *k sensitivity = 3* as defined in Eq. (7). results are the *mean* and *std* of multiple experiments.

ing approaches: DINO [4], MoCo [16] and MAE [17]. We observe that correlation clustering performs best with a method that supplies deep features with the largest amount of negative correlation between features (DINO 33.6%). We also observe that the small amount of positive correlation is counterproductive even with normalized cut (MAE 5.4%). We must consider those factors to deduce the best combinations of unsupervised training methods and segmentation. This insight helps us improve unsupervised training methods to get more valuable information for segmentation.

7. k-**less Clustering**

In this part we consider clustering, using as an example image classification. For most clustering methods, we need to define the number of clusters (or classes) k to which we want to classify the data. However, we need a specific understanding of the data to find k, Which can be a significant disadvantage. Our DeepCut method presents a GNN model with correlation clustering loss to perform k-less clustering and derives the number of clusters from the data. We demonstrate this valuable property using the Fashion Product Images Dataset [37]. We sample a subset of 500 images from 5 classes: Top-wear, Shoes, Bags, Eyewear, and Belts. We perform three types of classification experiments, each one 10 times: one with only three classes, one with four, and one with all of the classes. We perform each experiment with our GNN approach using correlation clustering loss to achieve clustering of images to different groups. We choose α (*k sensitivity* defined in Eq. (7)) to be 3, where our features are the class tokens extracted from unsupervised trained DINO [4] ViT base transformer with patch size 16. We observe the pseudo-labels from our unsupervised method and report our result as the classification purity in Tab. 5. We apply class permutation and random subsets sampling of each class to prove the algorithm's robustness.

This k-less nature of correlation clustering can be applied to segmentation tasks as in Figure 6. We present two baselines, connected component and spectral clustering while choosing the number of clusters using eigen-gap

[38]. The reason for using spectral clustering, instead of our DeepCut method with N-cut loss, is that it is not feasible to choose the number of clusters with our GNN N-cut optimization. The connected components method fails completely, and clusters all the images to the same cluster for all experiments, therefore its STD is zero. Spectral clustering provides better results but with high variance. Our proposed DeepCut with the correlation clustering loss results with the highest accuracy and lowest variance. This method can be applied for image segmentation as in Fig. 6.

Deep Cut with correlation clustering loss helps understand the natural split of the deep features and thus provides us a better understanding of what unsupervised pretraining methods learn, as we see here in the example of classifying different fashion items.

8. Conclusion

This paper introduces DeepCut, a new unsupervised segmentation method that relies on classical graph theory, GNNs, and self-supervised pretrained networks. We demonstrated the performance of DeepCut on tasks such as object localization, object segmentation, and semantic part segmentation, surpassing state-of-the-art performance. This paper suggests two loss functions for GNN training derived from classical graph theory; those functions can be changed to fit a specific use, thus improving segmentation models.

Contrary to previous approaches, our graph includes node features, and thus we use more information for the clustering process, allowing us to perform semantic part segmentation implicitly, without post processing methods. This approach can be applied to various downstream tasks, such as video segmentation and image matting. More importantly, we motivate the importance of choosing the right combination between the unsupervised training method and clustering method. We improve our understanding of deep features with techniques such as k-less clustering, as we can observe how the deep features data splits without choosing the number of clusters.

References

[1] Di Lin, Jifeng Dai, Jiaya Jia, Kaiming He, and Jian Sun. Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3159–3167, 2016. 1

[2] Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Armand Joulin, Nicolas Ballas, and Michael Rabbat. Semi-supervised learning of visual features by non-parametrically predicting view assignments with support samples. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8443–8452, 2021. 2

[3] Zhongzheng Ren, Zhiding Yu, Xiaodong Yang, Ming-Yu Liu, Yong Jae Lee, Alexander G Schwing, and Jan Kautz. Instance-aware, context-focused, and memory-efficient weakly supervised object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10598–10607, 2020. 2

[4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660, 2021. 2, 3, 5, 6, 7, 8

[5] Shir Amir, Yossi Gandelsman, Shai Bagon, and Tali Dekel. Deep vit features as dense visual descriptors. *arXiv preprint arXiv:2112.05814*, 2021. 2, 3

[6] Luke Melas-Kyriazi, Christian Rupprecht, Iro Laina, and Andrea Vedaldi. 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*, pages 8364–8375, 2022. 2, 3, 5, 6, 7

[7] Yangtao Wang, Xi Shen, Shell Xu Hu, Yuan Yuan, James L Crowley, and Dominique Vaufreydaz. Self-supervised transformers for unsupervised object discovery using normalized cut. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14543–14553, 2022. 2, 5, 6

[8] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021. 2

[9] Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. *Advances in neural information processing systems*, 31, 2018. 2

[10] Weiwei Jiang and Jiayun Luo. Graph neural network for traffic forecasting: A survey. *Expert Systems with Applications*, page 117921, 2022. 2

[11] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000. 2, 3

[12] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine learning*, 56(1):89–113, 2004. 2, 3, 4

[13] Shai Bagon and Meirav Galun. Large scale correlation clustering optimization. *arXiv preprint arXiv:1112.2903*, 2011. 3

[14] Shai Bagon and Meirav Galun. A multiscale framework for challenging discrete optimization. In *NIPS Workshop on Optimization for Machine Learning*, 2012. 3

[15] Max Welling and Thomas N Kipf. Semi-supervised classification with graph convolutional networks. In *J. International Conference on Learning Representations (ICLR 2017)*, 2016. 3

[16] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9640–9649, 2021. 3, 7, 8

[17] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022. 3, 7, 8

[18] Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. Spectral clustering with graph neural networks for graph pooling. In *International Conference on Machine Learning*, pages 874–883. PMLR, 2020. 4, 5, 7

[19] Jonathan T Barron and Ben Poole. The fast bilateral solver. In *European conference on computer vision*, pages 617–632. Springer, 2016. 5

[20] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 5, 7

[21] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013. 6

[22] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *European conference on computer vision*, pages 391–405. Springer, 2014. 6

[23] Oriane Siméoni, Gilles Puy, Huy V Vo, Simon Roburin, Spyros Gidaris, Andrei Bursuc, Patrick Pérez, Renaud Marlet, and Jean Ponce. Localizing objects with self-supervised transformers and no labels. In *BMVC-British Machine Vision Conference*, 2021. 6

[24] Yaniv Benny and Lior Wolf. Onegan: Simultaneous unsupervised learning of conditional image generation, foreground segmentation, and fine-grained clustering. In *European Conference on Computer Vision*, pages 514–530. Springer, 2020. 6

[25] Andrey Voynov, Stanislav Morozov, and Artem Babenko. Object segmentation without labels with large-scale generative models. In *International Conference on Machine Learning*, pages 10596–10606. PMLR, 2021. 6

[26] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html. 6, 7

[27] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 6

[28] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 6

[29] Huy V Vo, Patrick Pérez, and Jean Ponce. Toward unsupervised, multi-object discovery in large-scale image collections. In *European Conference on Computer Vision*, pages 779–795. Springer, 2020. 6

[30] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. Cub-dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 6

[31] Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan. Learning to detect salient objects with image-level supervision. In *CVPR*, 2017. 6

[32] Qiong Yan, Li Xu, Jianping Shi, and Jiaya Jia. Hierarchical saliency detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1155–1162, 2013. 6

[33] Wei-Chih Hung, Varun Jampani, Sifei Liu, Pavlo Molchanov, Ming-Hsuan Yang, and Jan Kautz. Scops: Self-supervised co-part segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 869–878, 2019. 7

[34] Zixuan Huang and Yin Li. Interpretable and accurate fine-grained recognition via region grouping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8662–8672, 2020. 7

[35] Subhabrata Choudhury, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Unsupervised part discovery from contrastive reconstruction. *Advances in Neural Information Processing Systems*, 34:28104–28118, 2021. 7

[36] Edo Collins, Radhakrishna Achanta, and Sabine Susstrunk. Deep feature factorization for concept discovery. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 336–352, 2018. 7

[37] Param aggarwal. Fashion Product image dataset. https://www.kaggle.com/datasets/paramaggarwal/fashion-product-images-dataset. Accessed: 2020-27-10. 8

[38] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007. 8