Learning the Superpixel in a Non-iterative and Lifelong Manner

Lei Zhu[1,3,4] Qi She[2] Bin Zhang[4] Yanye Lu[1,3,5] Zhilin Lu[2] Duo Li[2] Jie Hu[2]
[1]Institute of Medical Technology, Peking University Health Science Center, Peking University
[2]Bytedance AI Lab [3]Department of Biomedical Engineering, Peking University
[4]Beijing University of Posts and Telecommunications
[5]Institute of Biomedical Engineering, Peking University Shenzhen Graduate School

Abstract

Superpixel is generated by automatically clustering pixels in an image into hundreds of compact partitions, which is widely used to perceive the object contours for its excellent contour adherence. Although some works use the Convolution Neural Network (CNN) to generate high-quality superpixel, we challenge the design principles of these networks, specifically for their dependence on manual labels and excess computation resources, which limits their flexibility compared with the traditional unsupervised segmentation methods. We target at redefining the CNN-based superpixel segmentation as a lifelong clustering task and propose an unsupervised CNN-based method called LNS-Net. The LNS-Net can learn superpixel in a non-iterative and lifelong manner without any manual labels. Specifically, a lightweight feature embedder is proposed for LNS-Net to efficiently generate the cluster-friendly features. With those features, seed nodes can be automatically assigned to cluster pixels in a non-iterative way. Additionally, our LNS-Net can adapt the sequentially lifelong learning by rescaling the gradient of weight based on both channel and spatial context to avoid overfitting. Experiments show that the proposed LNS-Net achieves significantly better performance on three benchmarks with nearly ten times lower complexity compared with other state-of-the-art methods.

1. Introduction

Superpixel segmentation aims to over-segment an image into hundreds of compact partitions, *i.e.* superpixel, by clustering the pixels based on both low-level color features and spatial features. Benefiting from concerning the spatial cues, the superpixel can be efficiently generated with high contour adherence. Therefore, it is widely used by both traditional machine learning (ML) and convolution neural network (CNN) to reduce computational complexity or perceive the contours of objects[32, 12, 3, 31].

Many superpixel segmentation methods arise in the last decade including the gradient-based[1, 2, 18, 19, 5] and the graph-based methods[17, 15, 13, 16]. The gradient-based methods iteratively cluster the pixels in RGB or LAB space with limited spatial distance to refine the initialized cluster centers. This type of method has high efficiency, but suffers from low adherence due to their insufficient features. On the other hand, the graph-based algorithms usually have high adherence because they enrich the features by constructing an undirected graph. Afterwards, the subgraphs are generated as superpixel by cutting or adding edges to optimize a target energy function, which costs a lot of time.

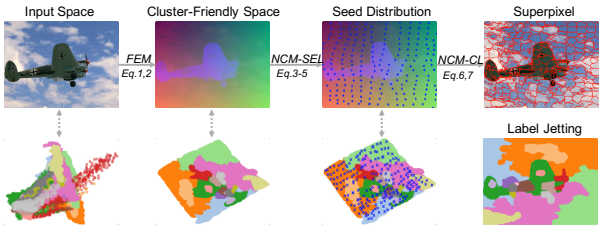


Figure 1. The illustration of the workflow for the proposed LNS-Net, where the top row is the visualization of the features and the bottom is the distribution of the labels jetting in the feature space. The blue "x" is the generated seed node.

Recently, benefiting from the prosperity of the CNN, some approaches employ the CNN to learn a suitable embedding space for superpixel segmentation and then cluster the pixels in this new feature space with clustering methods[11, 27, 30, 28]. Even though they improve the performance by a large margin, some problems come into being simultaneously. Firstly, majority of the CNN-based methods[30, 11, 28] need human-labeled ground truth to supervise the network training which requires additional human resources to label all the pixels in images. Secondly, their offline training step needs to store all the training samples, which demands large amounts of memory and limits their flexibility to transfer the network into other domains. Finally, some CNN-based methods still need to iteratively update the coarse cluster centers (usually the center position

of each grids), which is inconvenient and time-consuming.

To solve these problems, we redefine the CNN-based superpixel segmentation as a lifelong learning task[22, 9, 4] which can sequentially learn a unified model online. In addition, a lightweight unsupervised CNN-based superpixel segmentation method called LNS-Net is proposed to learn superpixel in a non-iterative and lifelong manner. The LNS-Net is composed of three parts: *feature embedder module* (FEM), *non-iterative clustering module* (NCM) and *gradient rescaling module* (GRM). Specifically, in the forward step shown in Fig. 1, FEM firstly embeds the original feature into a cluster-friendly space to protect detail cues with low complexity. Based on the cluster-friendly features, the proposed NCM uses a seed estimation layer (SEL) to learn the spatial shift of the central position, which directly estimates the optimal cluster centers, *i.e.* the seed nodes. Then, the superpixel can be non-iteratively generated by the cluster layer (CL) of NCM that assigns the cluster for each pixel based on their similarity with the feature of seed nodes. Moreover, the GRM is proposed to solve the catastrophic forgetting caused by lifelong learning during backward step. It is consisted of gradient adaptive layer (GAL) and gradient bi-direction layer (GBL), which are used to avoid over-fitting by rescaling the gradient of each weight parameter based on channel importance and spatial context. A range-limited cluster loss is also proposed to effectively train our network without any manual labels.

In a nutshell, our main contributions are threefold: 1) To our knowledge, our work is the first to define the superpixel segmentation as lifelong learning task theoretically and give a corresponding solution. 2) A lightweight LNS-Net is proposed to non-iteratively generate the superpixel, which can be lifelong trained without any manual label. 3) Experiments show that our LNS-Net has higher performance than other unsupervised methods and is also comparable with the supervised CNN-based methods.

2. Related Works

Traditional Superpixel Methods: The traditional superpixel segmentation methods include the gradient-based methods and the graph-based methods. The former iteratively cluster the pixels with limited spatial distance solely based on their color feature. Achanta *et al.* proposed the simple linear iteratively clustering (SLIC)[1] to efficiently generate superpixel by limiting the search range of k-means. To further improve the efficiency, Achanta *et al.* subsequently enabled the method to update cluster center and arrange the label of pixels simultaneously by proposing the simple non-linear iteratively clustering (SNIC)[2]. Liu *et al.* designed the manifold simple linear iteratively clustering (MSLIC)[18, 19], which adopts an adaptive search range for SLIC. Shen *et al.* utilized another robuster cluster method called the density-based spatial clustering with noise (DBSCN)[24] to generate superpixel with stronger spatial consideration. Different with the gradient-based methods, the graph-based methods firstly construct an undirected graph based on the feature of input image and then generate superpixel by creating sub-graphs. Shen *et al.* proposed the lazy random walk (LRW)[23], which adds a self-loop into the random walk graph to make the walking process lazy and can be extended into the superpixel segmentation with the help of a shape-concerned energy term. Liu *et al.* elaborated an entropy rate superpixel (ERS)[17] that maximizes the random walk entropy by continually adding edges into the graph model. Li *et al.* proposed the linear spectral superpixel clustering (LSC)[16] to approximate the normalized cut (NCut)[25] energy by weighted k-means cluster. Recently, Kang *et al.* designed the dynamic random walk (DRW)[15, 13], which efficiently improves the adherence of superpixel by proposing a weighted random walk entropy with limited walk range.

CNN-based Superpixel Methods: The CNN-based superpixel segmentation methods use the CNN to extract features and then cluster the pixels based on these features. Tu *et al.* firstly adopted the CNN in superpixel segmentation by proposing a segmentation-aware loss (SEAL)[28]. It uses the ground truths of semantic segmentation (or boundary detection) to supervise the feature learning. However, SEAL cannot generate superpixel in an end-to-end mode because it adopts the time-consuming ERS[17] as post-processing. Jampani *et al.* proposed an end-to-end superpixel segmentation network called superpixel sample network (SSN)[11] by integrateing SLIC. SSN can be easily used to assist other vision tasks such as semantic segmentation with the task-specific loss. But, it still needs manual labels to supervise the network training and requires iteratively updating the predefined cluster centers to generate superpixel. Yang *et al.* designed a fully-connected convolutional network (S-FCN)[30] that adopts an encoder-decoder structure, which simplifies the iteratively clustering step of SSN by assigning each pixel into the 9-neighbor grid. Though S-FCN improves the segmentation efficiency, it is still supervised by the segmentation labels, and needs upsampling the input images to generate large number of superpixel. Recently, Suzuki utilized the CNN to unsupervisely generate superpixel with regular information maximization (RIM)[27]. It trains a randomly initialized CNN to reconstruct the input image while minimizing the entropy among each superpixel. However, it needs to reinitialize the parameters of the network and takes a long time to reach convergence when generating superpixel for each image.

3. Method

In this section, we begin with defining the superpixel segmentation as the lifelong learning task, where the segmentation process of each image can be viewed as an inde-

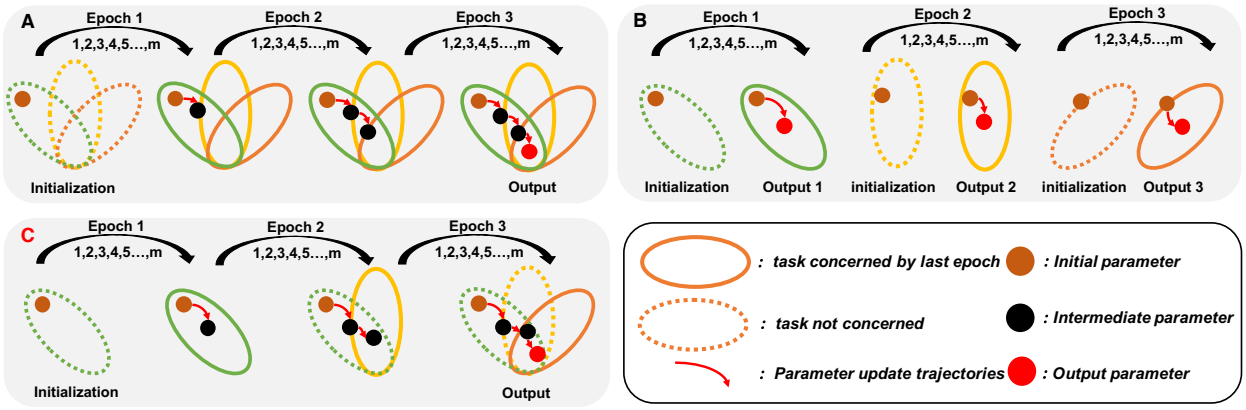


Figure 2. The training strategy of our LSN-Net and other learning-based superpixel segmentation methods. "Ellipses" with different colors mean different clustering tasks (images). Each "dot" means the parameters of the network during training progress. A. The multi-task learning strategy, which learns *a unified embedding* by optimizing *the whole task set*. B. The isolated learning strategy, which respectively learns *a unique embedding* for *each task*. C. The lifelong learning strategy of our proposed LNS-Net, which learns *a unified embedding* by separately optimizing *each task*.

pendent clustering task. Then, we propose a convolutional network structure called LNS-Net which contains: 1) feature embedder module (FEM); 2) non-iterative clustering module (NCM); 3) gradient rescaling module (GRM). Finally, we give our loss function, which does not require any manual labels to supervise training process.

3.1. Problem Definition

In general, the learning strategy of existing CNN-based superpixel segmentation methods can be divided into two categories. One is the multi-task learning strategy[11, 30], which learns a unified embedding based on the whole image set depicted in Figs. 2 A. It updates the weight parameter based on all images during the whole training process which requires large amounts of computation resources. The other is the isolated learning strategy[27], which respectively learns a unique embedding for each image as shown in Figs. 2 B. Though this strategy does not require to maintain all the images, a unique parameter space needs to be repeatedly found for each image, which is time-consuming and cannot generalize to other images. In order to overcome these drawbacks, our proposed LNS-Net sequentially refines the unified embedding based on a certain image, which is a classic case of lifelong learning. As shown in Figs. 2 C, our lifelong learning strategy only focuses on one image per epoch and intends to maintain the performance for the images learned in prior epoches simultaneously.

To theoretically define our sequential learning strategy, we start with the segmentation of a specific image I, which aims to segment the image I into K compact regions by assigning the label for each pixel of the entire image map $\boldsymbol{L}^I$. It can be seen as a clustering task T^I where each pixel i with feature $\boldsymbol{x}_i = \{r, g, b, p_x, p_y\}$ forms the samples $\boldsymbol{X}^I = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N\}$. Supposing the index set of the cluster centers is $\boldsymbol{S}^c$, a following cluster step $\boldsymbol{L}^I = \boldsymbol{c}(\boldsymbol{Z}^I|\boldsymbol{S}^c)$ is used to generate the label for each pixel, where $\boldsymbol{c}(\cdot)$ is a cluster function. $\boldsymbol{Z}^I = \boldsymbol{e}(\boldsymbol{X}^I|\mathbf{W}_e)$ is a learned embedding map to project the samples $\boldsymbol{X}^I$ into a clustering-friendly space with function $\boldsymbol{e}(\cdot)$. The learning weight $\mathbf{W}_e$ can be optimized by $\mathbf{W}_e = \mathbf{W}_e - \alpha * \mathrm{d}\mathbf{W}_e$ with $\mathrm{d}\mathbf{W}_e = \frac{\partial \mathcal{L}}{\partial \mathbf{W}_e}$, where $\mathcal{L}$ is the loss function and α is the learning rate.

Further, assuming that we have a set of images $\mathbb{I} = \{I_1, I_2, ..., I_n\}$, the segmentation of $\mathbb{I}$ can be modeled as a series of clustering tasks $\mathbb{T} = \{T^1, T^2, ..., T^n\}$. Different from the existing models that either obtain the embedding $\boldsymbol{e}(\boldsymbol{X}|\mathbf{W}_e)$ by optimizing $\mathbf{W}_e$ based on the whole set $\mathbb{T}$[11, 30] or separately training an embedding $\boldsymbol{e}^i(\boldsymbol{X}^i|\mathbf{W}_e^i)$ for each task T^i[27] to obtain the cluster-friendly feature $\boldsymbol{Z}$, we aim at optimizing each task T^i separately to generate a unified embedding function $\boldsymbol{e}(\boldsymbol{X}^i|\mathbf{W}_e)$. During the optimization, the retentivity of $\mathbf{W}_e$ for prior tasks is also enhanced by a scaling function $\boldsymbol{\psi}(\mathrm{d}\mathbf{W})$. Finally, with the cluster-friendly features $\boldsymbol{Z}$, pixels can be labeled by the cluster function $\boldsymbol{L} = \boldsymbol{c}(\boldsymbol{Z}|\boldsymbol{S})$ with optimal seed nodes $\boldsymbol{S}$.

Algorithm 1 Flow of the proposed LNS-Net

Input: Cluster tasks T, Feature set X, Max epoches M

```
Initialize the learning rate and parameters
for T^i in 𝕋 do
    Select the pixels feature X^i, Set m = 0
    for m < M do
        Get the Z^i = e(X^i|W_e) by FEM.
        Get the labels L = c(Z^i|S) by NCM.
        Rescale the gradient: dW_e = ψ(dW_e) by GRM.
        Backward update W_e and other parameters.
    end for
end for
```

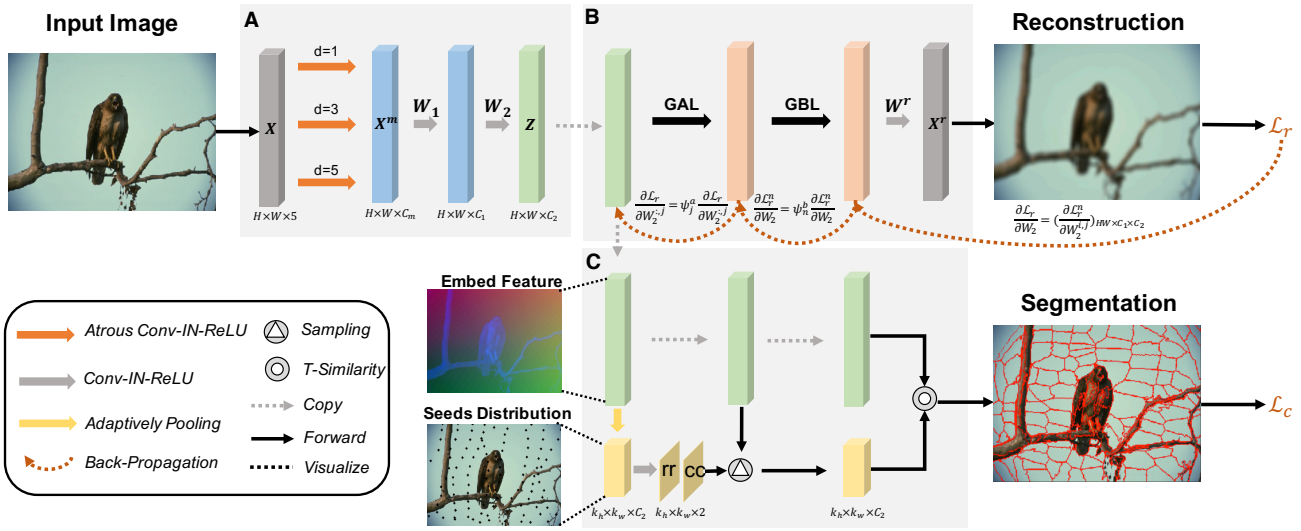


Figure 3. The network structure of our LNS-Net. "Seeds distribution" visualizes the seed node generated by NCM. "Embed feature" visualizes the cluster-friendly feature map $\boldsymbol{Z}$ with the help of PCA dimension reduction. A. The structure of our FEM. B. The structure of our GRM. C. The structure of our NCM.

The flow of our LNS-Net is given in Algorithm 1. We separately optimize each clustering task T^i and train a network that contains three proposed modules to implement the functions $\boldsymbol{e}(\cdot)$, $\boldsymbol{c}(\cdot)$ and $\boldsymbol{\psi}(\cdot)$ respectively. Once T^i has been optimized, we start to focus on the next task T_{i+1} until all tasks are trained.

3.2. Network Design

The structure of proposed LNS-Net shown in Fig. 3 contains three parts: 1) the proposed lightweight FEM embeds the original feature into a cluster-friendly space; 2) the proposed NCM assigns the label for pixels with the help of a seed estimation module, which automatically estimates the indexes of seed nodes; 3) the proposed GRM adaptively rescales the gradient for each weight parameter based on the channel and spatial context to avoid catastrophic forgetting for the sequential learning.

Feature Embedder Module: Actually, superpixel segmentation is based on the low-level color and spatial features rather than the high-level semantic features. We argue that the feature embedders used by other CNN-based methods[11, 30, 27] are too redundant for the superpixel segmentation, due to their large number of channels and receptive field. As a alternative, our FEM only uses two convolution layers with an additional atrous spatial pyramid pooling (ASPP)[7] to enlarge the receptive field rather than go deeper with spatial pooling, which can better preserve details with fewer parameters. As shown in Fig. 3 A, the LAB (or RGB) features and the position indexes of pixels are concatenated and fed into the ASPP structure to capture multi-scale information:

$$\boldsymbol{X}^m = \sigma(concat(\boldsymbol{X} * \mathbf{H}^0, \boldsymbol{X} * \mathbf{H}^1, \boldsymbol{X} * \mathbf{H}^2)) \quad (1)$$

where "$*$" is the convolutional operator, $\boldsymbol{X} \in \mathbb{R}^{N\times 5}$ is the input feature and $\boldsymbol{X}^m \in \mathbb{R}^{N\times C_m}$ is the multi-scale feature. $\mathbf{H}^d \in \mathbb{R}^{5\times \frac{C_m}{3}}$ is the convolution with dilation range d, σ is the non-linear function implemented by ReLU. Then two 3×3 convolution are used to embed the multi-scale feature $\boldsymbol{X}^m$ into the cluster-friendly space:

$$\boldsymbol{Z} = \sigma(\sigma(\boldsymbol{X}^m * \mathbf{W}_1) * \mathbf{W}_2)) \quad (2)$$

where $\boldsymbol{Z} \in \mathbb{R}^{N\times C_2}$ is the cluster-friendly feature, $\mathbf{W}_1 \in \mathbb{R}^{C_m\times C_1}$, $\mathbf{W}_2 \in \mathbb{R}^{C_1\times C_2}$ are the parameter matrixes.

Non-iterative Clustering Module: Once the embedding feature $\boldsymbol{Z}$ has been obtained, the superpixel can be generated by clustering the features in $\boldsymbol{Z}$ with the initialized cluster centers $\boldsymbol{S}^c$. However, those cluster centers usually have a coarse distribution such as the center of grid. The cluster step with time complexity $O(N * K)$ needs to iteratively refine the distribution of the centers. Such refining process is unintegrable in majority cases. Though the recent work[11] makes it integrable, it still needs nearly 10 iterators to reach convergence. To avert this time-consuming process, our NCM uses a seed estimation layer (SEL) to estimate a satisfactory cluster center distribution based on $\boldsymbol{Z}$ by learning the offsets to shift those coarse centers into a more reasonable distribution, *i.e.* the seed nodes $\boldsymbol{S}$.

As shown in Fig. 3 C, $\boldsymbol{Z}$ is adaptively pooled into a low-resolution $\boldsymbol{Z}^k \in \mathbb{R}^{K\times C_2}$, where K is the number of target superpixel. Then, a linear project with sigmoid activation is used to learn the offsets contained by a two-dimension vectors $\boldsymbol{F}_i \in \mathbb{R}^{K\times 2}$:

$$\boldsymbol{F} = sigmoid(\boldsymbol{Z}^k * \mathbf{W}_s) \quad (3)$$

where $\mathbf{W}_s \in \mathbb{R}^{C_2\times 2}$ is the parameter matrix of the linear

project, which can be learned by Adam[14]. The two dimensions of $\boldsymbol{F} = \{\boldsymbol{rr}, \boldsymbol{cc}\}$ can be viewed as the crosswise shift ratio $\boldsymbol{rr}$ and the longitudinal shift ratio $\boldsymbol{cc}$. Then, we restrict their shift scopes inside the corresponding grid by:

$$\triangle \boldsymbol{r} = (\boldsymbol{rr} - 0.5) * R, \qquad \triangle \boldsymbol{c} = (\boldsymbol{cc} - 0.5) * C \tag{4}$$

where R, C are the number of rows and columns of the image, respectively. Next, the offsets $(\triangle \boldsymbol{r}, \triangle \boldsymbol{c})$ are added on the corresponding center to obtain the optimized seed nodes $\boldsymbol{S}$:

$$\boldsymbol{S} = \boldsymbol{S}^c + (\triangle \boldsymbol{r} * R + \triangle \boldsymbol{c}) \tag{5}$$

where $\boldsymbol{S}^c$ is the coarse clustering center indexes and $\boldsymbol{S}$ is the output seed node indexes.

Finally, the clustering layer (CL) of NCM is used to assign the labels $\boldsymbol{L}$ for pixels based on $\boldsymbol{S}$. The CL firstly adopts the t-distribution kernel to measure the similarity between each pixel and seed node:

$$\boldsymbol{P}_{ik} = \frac{(1 + ||\boldsymbol{Z}_i - \boldsymbol{Z}_{\boldsymbol{S}_k}||^2)^{-\frac{1}{2}}}{\sum_k (1 + ||\boldsymbol{Z}_i - \boldsymbol{Z}_{\boldsymbol{S}_k}||^2)^{-\frac{1}{2}}} \tag{6}$$

where $\boldsymbol{P} =\in \mathbb{R}^{N*K}$ is the soft assignment between each pixel and seed node. Finally, the label of each pixel can be obtained by selecting the seed with maximal similarity:

$$\boldsymbol{L}_i = argmax_k(\boldsymbol{P}_{i0}, \boldsymbol{P}_{i1}, ..., \boldsymbol{P}_{ik}) \tag{7}$$

Gradient Rescaling Module: Considering the images are sequential in our learning step, the network will face the catastrophic forgetting that the network over-fits the domain of current task without concerning prior tasks. To overcome this problem, our proposed GRM serves two purposes: 1) using the proposed gradient adaptive layer (GAL) to perceive the importance for the gradient on different *feature channels* to avoid over-fitting; 2) using the proposed gradient bi-direction layer (GBL) to generate confrontation based on the *spatial context* to improve generalizability.

Specifically, both GAL and GBL are backed by a reconstruction head that reconstructs the clustering-friendly feature into the original input features (both spatial and color features) with an additional linear project:

$$\boldsymbol{X}^r = \boldsymbol{Z} * \mathbf{W}^r \tag{8}$$

where $\boldsymbol{X}^r \in \mathbb{R}^{N\times 5}$ is the reconstruction feature whose first three columns are the color features (RGB/LAB) and the rest are spatial features (position indexes), which is respectively supervised by the reconstruction loss $\mathcal{L}_r$. $\mathbf{W}^r \in \mathbb{R}^{C_2\times 5}$ is the parameter for the linear project.

Based on the reconstruction head, the mean reconstruction strength $\boldsymbol{g}(\mathbf{W}^r)$ can be defined to represent the importance for the channel of the cluster-friendly feature:

$$\boldsymbol{g}(\mathbf{W}^r) = (\sum_{i=1,2,3} |\mathbf{W}^r_{:,i}/3|) \odot (\sum_{i=4,5} |\mathbf{W}^r_{:,i}/2|)^T \tag{9}$$

where $\odot$ is the Hadamard product. The higher $\boldsymbol{g}(\mathbf{W}^r)_{:,c}$ is, the more $\boldsymbol{Z}_{:,c}$ contributes for reconstructing $\boldsymbol{X}$ in forward-propagation, *i.e.* this channel has already better fit the domain of current task. Thus, even though $\boldsymbol{g}(\mathbf{W}^r)_{:,c}$ drops in the following tasks, which causes a high gradient $\mathrm{d}\mathbf{W}_{:,c}$, this weight $\mathbf{W}^r_{:,c}$ should be maintained to avoid over-fitting. To achieve this, a vector $\boldsymbol{m} \in \mathbb{R}^{1\times C_2}$ is defined to preserve the historical $\boldsymbol{g}(\mathbf{W}^r)_{:,c}$, which is initialized as an all-one tensor and progressively updated during the sequential training step:

$$\boldsymbol{m} = \lambda * \boldsymbol{g}(\mathbf{W}^r) + (1 - \lambda) * \boldsymbol{m} \tag{10}$$

where λ is to adjust current and history gradient scale.

Based on $\boldsymbol{m}$, our GAL is designed to rescale the gradient of the weight parameter in FEM to avoid overfitting, which works as a "pseudo-function" $\boldsymbol{R}^a(\cdot)$ with the following forward- and back-propagation:

$$\begin{aligned} &\boldsymbol{R}^a(\boldsymbol{X}_{n,:}) = \boldsymbol{X}_{n,:} * \boldsymbol{I} \\ &\frac{\mathrm{d}\boldsymbol{R}^a}{\mathrm{d}\boldsymbol{X}_{n,:}} = \boldsymbol{\psi}^a * \boldsymbol{I} = \frac{\boldsymbol{g}(\mathbf{W}^r)}{\boldsymbol{g}(\mathbf{W}^r) + \boldsymbol{m}} * \boldsymbol{I} \end{aligned} \tag{11}$$

where $\boldsymbol{I} \in \mathbb{R}^{C_2\times C_2}$ is the identity matrix. In the forward-propagation, GAL acts as an identity transform which percepts the importance for each channel by $\boldsymbol{g}(\mathbf{W}^r)$ to preserve the historical memory matrix $\boldsymbol{m}$. During back propagation, GAL scales the gradient of the weight parameters, which lowers the gradient of weights corresponding to the channel with high $\boldsymbol{m}_c$ to avoid over-fitting the current task.

Though the proposed GAL can avoid over-fitting by concerning the historical strength of the channel, it treats each pixel equally without considering their spatial context. Actually, the superpixel segmentation is a dense prediction task, which aims to balance the contour adherence and spatial compactness. This requires the model biasing the color features for pixels near contours, while concerning both the color and the spatial features for pixels in smooth areas. To compensate this, GBL is proposed to rescale the gradient based on the spatial context to avoid overfitting. It generates bi-direction gradient scale based on the contour map $\boldsymbol{B}$ to confound the reconstruction strength for the spatial features of the pixels near contours. The forward- and back-propagation of our GBL are:

$$\begin{aligned} &\boldsymbol{R}^b(\boldsymbol{X}_{n,c}) = \boldsymbol{X}_{n,c} \\ &\frac{\mathrm{d}\boldsymbol{R}^b}{\mathrm{d}\boldsymbol{X}_{n,c}} = \boldsymbol{\psi}^b_n = \begin{cases} 1 & , \quad \boldsymbol{B}_n \le \epsilon \\ -\boldsymbol{B}_n & , \quad \boldsymbol{B}_n > \epsilon \end{cases} \end{aligned} \tag{12}$$

where $\boldsymbol{R}^b(\cdot)$ is the "pseudo-function" of our GBL. In the forward step the GBL also acts as an identity map. While in the backward step, the GBL generates a bi-direction gradients for the different pixels i based on their contour map $\boldsymbol{B}_i$, which makes the pixels near contours bias the color feature reconstruction even though having a confounding spatial information to enhance generalizability.

3.3. Loss Function

A two-terms loss is used to supervise the sequential training step for our network, which can be formulated as:

$$\mathcal{L} = \mathcal{L}_c + \beta * \mathcal{L}_r \tag{13}$$

where $\mathcal{L}_c$ is the clustering loss that encourages the network to group pixels with similar probability. $\mathcal{L}_r$ is the reconstruction loss to help the cluster-friendly feature $\boldsymbol{Z}$ concern both color details and spatial information. β is used to balance the two losses.

Cluster Loss: We propose a range-limited cluster loss to train our network without requiring manual label. It can be formulated as a regularized KL divergence between the limited range soft assignment $\widetilde{\boldsymbol{P}}$ with its reference distribution $\widetilde{\boldsymbol{Q}}$:

$$\mathcal{L}_c = \sum_i \sum_k \widetilde{\boldsymbol{Q}}_{ik} log \widetilde{\boldsymbol{Q}}_{ik} - \widetilde{\boldsymbol{Q}}_{ik} log \widetilde{\boldsymbol{P}}_{ik} + l(\boldsymbol{P}) \tag{14}$$

where $l(\cdot)$ is a regular term. The limited range soft assignment $\widetilde{\boldsymbol{P}}$ enhances the probability for allotting pixels into its "Top-n" nearest seed nodes, which improves the compactness of the segmentation result. Specifically, the spatial distance $\boldsymbol{D}_{ik}$ between the pixel i and the seed node k is firstly calculated based on the l_1 distance on their spatial indexes. Then, we define $\mathbb{V}_i = \text{Top-}n_k(\boldsymbol{D}_{i0}, \boldsymbol{D}_{i1}, ..., \boldsymbol{D}_{ik})$ as the "Top-n" seeds set for pixel i and use it to build a mask matrix, which masks the elements between the pixel-seed pairs with large distance:

$$\boldsymbol{M}_{ik} = \begin{cases} 0, & k \in \mathbb{V}_i \\ 1, & k \notin \mathbb{V}_i \end{cases} \tag{15}$$

Finally, the limited range soft assignment $\widetilde{\boldsymbol{P}} = \boldsymbol{P} \odot \boldsymbol{M}$ can be obtained by adding masks on the original assignment $\boldsymbol{P}$, *i.e.*, where $\odot$ is the Hadamard product.

To improve the cluster purity and penalize the superpixel with too large size, we follow Xie *et al.*[29] and define $\widetilde{\boldsymbol{Q}}$ without requiring the manual labels:

$$\widetilde{\boldsymbol{Q}}_{ik} = \frac{\widetilde{\boldsymbol{P}}_{ik}^2 / \sum_i \widetilde{\boldsymbol{P}}_{ik}}{\sum_j (\widetilde{\boldsymbol{P}}_{ik}^2 / \sum_i \widetilde{\boldsymbol{P}}_{ik})} \tag{16}$$

A regularized term is also added to avoid the local optimum where pixels are assigned into the seed node that not in $\mathbb{V}_i$:

$$l(\boldsymbol{P}) = \frac{\boldsymbol{P} \odot \boldsymbol{M}}{\boldsymbol{P} \odot (\boldsymbol{1} - \boldsymbol{M})} \tag{17}$$

Reconstruction Loss: Reconstruction loss is a crucial part for our proposed GRM to rescale the gradient of weight parameter. As discussed in Sec.3.2, $\mathcal{L}_r$ supervises both reconstruction of the input color and spatial features, which can be define as:

$$\mathcal{L}_r = \mathcal{L}_{r_c} + \phi * \mathcal{L}_{r_s} \tag{18}$$

$\mathcal{L}_{r_c}$ is the reconstruction loss of color feature, $\mathcal{L}_{r_s}$ is the reconstruction loss of spatial feature and ϕ controls the trade-off between $\mathcal{L}_{r_c}^i$ and $\mathcal{L}_{r_s}^i$. Specifically, MSELoss between the reconstruction result and original input is used as the reconstruction loss for $\mathcal{L}_{r_c}$ and $\mathcal{L}_{r_s}$.

From another view, due to the bi-direction gradient generated by our GBL, the reconstruction loss for our network is also equivalent to :

$$\mathcal{L}_r = \sum_{i \notin \mathbb{V}_b} (\mathcal{L}_{r_c}^i + \phi * \mathcal{L}_{r_s}^i) + \sum_{i \in \mathbb{V}_b} (\mathcal{L}_{r_c}^i - \boldsymbol{B}_i * \phi * \mathcal{L}_{r_s}^i) \tag{19}$$

where $\mathbb{V}_b = \{n | \boldsymbol{B}_n > \epsilon\}$ is the counter pixel set. In Eq. (19), the spatial reconstruction part for pixels near contours, *i.e.* $\sum_{i \in \mathbb{V}_b} (\boldsymbol{B}_i * \phi * \mathcal{L}_{r_s}^i)$, serves as a regularization term that avoids the cluster-friendly feature map $\boldsymbol{Z}$ paying much attention on the spatial feature for the pixels in $\mathbb{V}_b$.

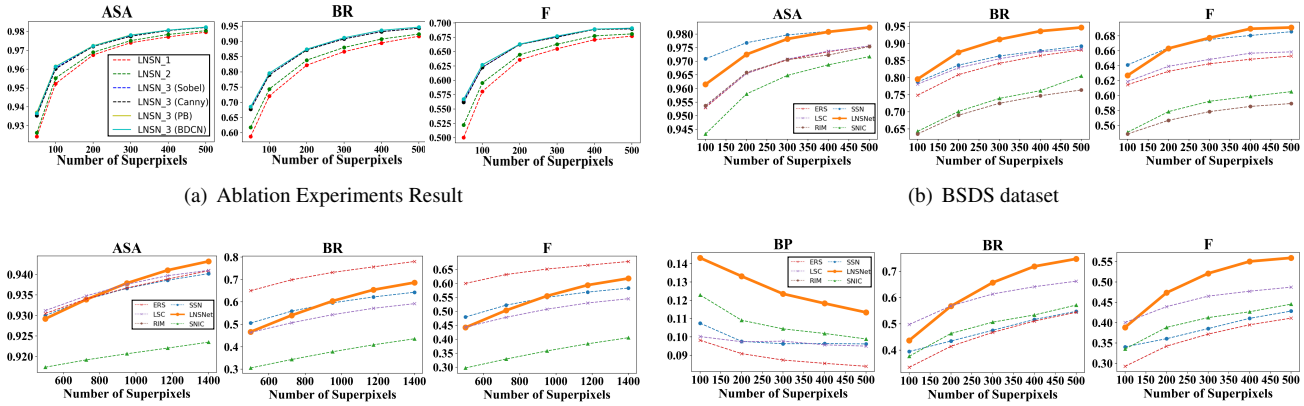
4. Experiment

We conduct experiments on three datasets to demonstrate the effectiveness of the proposed model. We firstly introduce the settings of our experiment including the implementation details of our LNS-Net and the evaluation metrics. Then, ablation studies are performed on BSDS500 dataset to evaluate the different modules of our LNS-Net. Finally, we compare our proposed LNS-Net with other superpixel segmentation methods.

4.1. Settings

Implementation Details: Our LNS-Net is implemented with PyTorch. The numbers of the three channels in FEM are set as $C_m = 10$, $C_1 = 10$, $C_2 = 20$. For the loss function, we set the balance parameter β, ϕ and the neighbor number n as 10, 1, 9 respectively. During the sequential training step, each image is sequentially trained 50 epoches where the first 40 epoches focus on feature learning so $\mathbf{W}_s$ of the seed estimator layer is locked. The last 10 epoches serve as updating the seed distribution, where all the weights of FEM are locked. Adam [14] with learning rate 0.0003 is used to optimize the parameters. Note that our training step do not require any manual label. And in the test step, only our FEM and NCM are used to generate superpixel efficiently.

Evaluation Metrics: In our experiments, the Boundary Recall (BR), the boundary Precision (BP), the Achievable Segmentation Accuracy (ASA) and the F-beta Score (F) are used to evaluate the superpixel segmentation. Considering that the recall is more important than the precision for superpixel segmentation, beta is set 4 for F-beta. For the dataset that has more than one groundtruth such as the BSDS500 dataset, we choose the best one among all the ground truths as the listed score. Moreover, like SSN[11] and RIM[27], we also use same strategy to enforce the spatial connectivity before calculating the evaluation metric.



(a) Ablation Experiments Result

(b) BSDS dataset

(c) DRIVE dataset

(d) DME dataset

Figure 4. The experimental results for ablation strudies of the proposed LNS-Net and the comparison for different superpixel segmentation methods on the BSDS, DRIVE, DME datasets. Better view in color and zoom in four times.

4.2. Ablation Study

Ablation studies are conducted on the BSDS500 dataset[21] to show the effectiveness of the proposed modules. We explain three type LNS-Net structures in details: *LNS-Net*$_1$ only uses FEM to embed the feature and cluster the pixels with grid seed node; *LNS-Net*$_2$ adds SEL of our proposed NCM to automatically generate seed nodes; *LNS-Net*$_3$ further adds GRM to adaptively rescale the gradient of weight parameter to avoid over-fitting by concerning feature channel and spatial context. Note that, the contour map generated by both unsupervised learned methods (Sobel, Canny[6]) and supervised learned methods (PB[20], BDCN[10]) are also tested for the proposed GRM.

The performance of these models is shown in Fig. 4(a). It can be seen that, even using the simple grid seed nodes (LNS-Net$_1$), our model outperforms the unsupervised method RIM[27] by a large margin profited by the cluster-friendly feature space generated by our FEM. While, adding the proposed seed estimation layer to automatically generate seed nodes (LNS-Net$_2$), the BR, ASA, F are further improved facilitated by the more suitable seed distribution. Next, when the proposed GRM is added (LNS-Net$_3$), overfitting is avoided, bringing an obvious improvement in the four evaluation criteria. Moreover, it can be also seen that unsupervised contour (dotted line) are comparable to the supervised-learned contours (full line), which means our GRM is not sensitive to different contour priors.

4.3. Results

In this section, three datasets from different domains are used to compare the performance of our proposed LNS-Net with other methods, including the graph-based ERS[17], LSC[16], the gradient-based SNIC[2] and the CNN-based SSN[11], RIM[27]. Visualization of their segmentations results on the three datasets are shown in Fig. 5.

BSDS500 dataset[21] is the standard benchmark for superpixel segmentation which contains 200 training images, 100 validation images and 200 test images. The size of image in this dataset is 481×321. Each image has more than 5 segmentation ground truths labeled by different person. Thus, we choose one of the ground truth that can achieve the highest segmentation scores in this study. Considering that SSN[11] is a supervised method that needs training the model on the training set and validation set to optimize the parameters, we only compare the performance on the test set for all superpixel segmentation method mentioned above. Quantitative results on BSDS dataset are shown in Fig. 4(b), it can be seen that our LNS-Net has the highest performance among all the unsupervised superpixel segmentation methods (ERS[17], SNIC[2], LSC[16], RIM[27]). This benefits from our sequential training strategy, which can unsupervisely optimize the model parameters. Moreover, our LNS-Net is more sensitive to the contours in a broad sense rather than only the semantic boundaries as shown in Fig. 5. This trait contributes to our higher BR than the supervised segmentation method SSN.

DRIVE dataset[26] is consisted of 40 retinal images with size 565×584 and the corresponding ground truth for their blood vessel. The domain of DRIVE dataset are very different from the images of BSDS500 as shown in Fig. 5, thus the same models trained on BSDS500 from each learning-based method are used to test their generalizability. Experimental results of the different methods on these 40 retinal images are listed in Fig. 4(c). It can be seen that only graph-based methods ERS[17] has higher BR and F than our LNS-Net, because its graph model concerns more global structure of the blood vessel than the other methods. Nevertheless, our LNS-Net is 46 times faster than ERS and has more regular shape of superpixel as shown in Fig. 5. Moreover, our LNS-Net has the highest ASA, indicating that the su-

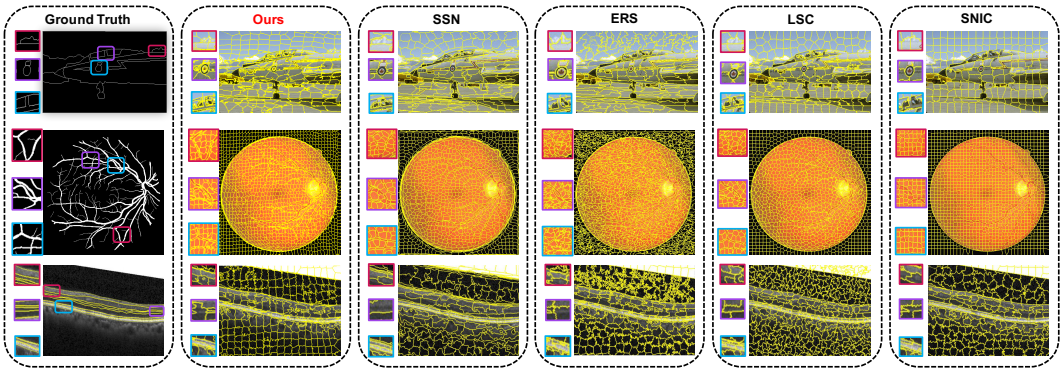


Figure 5. The quantitative results for different superpixel segmentation methods on the BSDS dataset (top row), DRIVE dataset (middle row) and DME dataset (bottom row). Better view in color and zoom in four times.

perpixel generated by our LNS-Net has the highest upper bound for adhering the blood vessel.

Duck DME dataset[8] contains 610 B-scans from 10 subjects who have Diabetic Macular Edema (DME). The size of each B-scan is 565×584 and only 110 of them have manual label for the retina border near their macular. Thus, we select 110 labeled B-scans and crop them into 464×496 to focus their macular area. For each learning-based method, we also use the same model trained on BSDS500 to segment the three-channel B-scans that expanded from gray scale. Experimental results are shown in Fig. 4(d), and it can be seen that all scores of our LNS-Net are much more higher than the others, indicating that its effectiveness on catching weak contours for the medical images. Further, the style of B-scans in the DME dataset is also very different from the images in BSDS500 and contain noise as shown in Fig. 5. We can see that the performance of learning-based method SSN deteriorate seriously in the DME dataset, while our LNS-Net can still have a satisfactory result, showing its robust generalizability.

Discussion: In general, benefiting from the proposed online training step, both the visual impression and the quantitative results demonstrate that our proposed LNS-Net is able to generate better superpixel compared with the unsupervised methods. Even though using an unsupervised sequential training strategy, the superpixel segmentation results generated by LNS-net are still comparable with the supervised learning-based methods. Moreover, LNS-Net has better generalizability with much less complexity *(9 times and 20 times lower in Flops and model size than SSN, respectively)* as shown in Table. 1.

However, there are still some drawbacks in our LNS-Net, which expected to be addressed in future study. Firstly, due to the sequential training strategy, our model cannot reach complete convergence as the other learning-based methods do. This leads to the existence of trivial regions in the superpixel generated by LNS-Net and needs post-processing to remove them. Secondly, LNS-Net can generate superpixel with relatively regular shapes in the smooth area promoted by the spatial consideration of GBL. But, when facing background with complex texture, the boundary map that assists GBL will contain noises and make the shape of superpixel irregular. Finally, although our LNS-Net uses a lightweight convolutional network and achieves real-time segmentation using GPU, the cluster step still needs to generate distance matrix with $N * K$ dimension, which is inefficient when calculated by CPU with a large K.

Table 1. The performance and complexity of methods for generating 100 superpixel on BSDS dataset with image size $481 * 321$

	Time(ms)	Flops(G)	Size(K)	ASA	Labels	Device
SNIC	85	-	-	0.943	×	CPU
LSC	269	-	-	0.953	×	CPU
ERS	2540	-	-	0.953	×	CPU
SSN	260	13.85	214.5	0.970	✓	GPU
RIM	34842	64.15	416.14	0.953	×	GPU
Ours	**55**	**1.15**	**11.22**	**0.962**	×	**GPU**

5. Conclusion

To our best knowledge, this paper is the first work that views superpixel segmentation as a lifelong clustering task. Based on this basis, we propose a CNN-based superpixel segmentation method called LNS-Net. The proposed LNS-Net contains three parts: FEM, NCM, GRM, which is respectively used for feature generation, non-iteratively clustering, and over-fitting avoidance. Experiments show the effectiveness of our LNS-Net in three benchmarks including two medical images datasets. Our method is both efficient and accurate, enabling low latency superpixel generation.

6. Acknowledgements

This work was supported by the Shenzhen Science and Technology Program (1210318663); the National Biomedical Imaging Facility Grant; the Shenzhen Nanshan Innovation and Business Development Grant;

References

[1] R Achanta, A Shaji, K Smith, A Lucchi, P Fua, and S Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(11):2274–2282, 2012. 1, 2

[2] Radhakrishna Achanta and Sabine Susstrunk. Superpixels and polygons using simple non-iterative clustering. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 4895–4904, 2017. 1, 2, 7

[3] Mohamed R. Amer, Siavash Yousefi, Raviv Raich, and Sinisa Todorovic. Monocular extraction of 2.1d sketch using constrained convex optimization. *Int. J. Comput. Vis.*, 112(1):23–42, 2015. 1

[4] H. Bae, E. Brophy, Rhm Chan, B. Chen, and L. Zhou. Iros 2019 lifelong robotic vision: Object recognition challenge [competitions]. *IEEE Robotics and Automation Magazine*, 27(2):11–16, 2020. 2

[5] Michael Van Den Bergh, Xavier Boix, Gemma Roig, Benjamin De Capitani, and Luc Van Gool. Seeds: Superpixels extracted via energy-driven sampling. *Int. J. Comput. Vis.*, 111(3):298–314, 2013. 1

[6] John Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, PAMI-8(6):679–698, 1986. 7

[7] L. C. Chen, G Papandreou, I Kokkinos, K Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4):834–848, 2018. 4

[8] Stephanie J. Chiu, Michael J. Allingham, Priyatham S. Mettu, Scott W. Cousins, Joseph A. Izatt, and Sina Farsiu. Kernel regression based segmentation of optical coherence tomography images with diabetic macular edema. *Biomedical Optics Express*, 6(4):1172–1194, 2015. 8

[9] F. Feng, R. H. M. Chan, X. Shi, Y. Zhang, and Q. She. Challenges in task incremental learning for assistive robotics. *IEEE Access*, 8:3434–3441, 2020. 2

[10] Jianzhong He, Shiliang Zhang, Ming Yang, Yanhu Shan, and Tiejun Huang. Bdcn: Bi-directional cascade network for perceptual edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, PP(99):1–1, 2020. 7

[11] Varun Jampani, Deqing Sun, Ming-Yu Liu, Ming-Hsuan Yang, and Jan Kautz. Superpixel sampling networks. In *Eur. Conf. Comput. Vis. (ECCV)*, pages 352–368, 2018. 1, 2, 3, 4, 6, 7

[12] Zhaoyin Jia. A learning-based framework for depth ordering. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 294–301, 2012. 1

[13] X. Kang, L. Zhu, and A. Ming. Dynamic random walk for superpixel segmentation. *IEEE Trans. Image Process.*, 29:3871–3884, 2020. 1, 2

[14] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Computer Science*, 2014. 5, 6

[15] Zhu Lei, Kang Xuejing, Ming Anlong, and Zhang Xuesong. Dynamic random walk for superpixel segmentation. In *Asi. Conf. Comput. Vis. (ACCV)*, pages 540–554, 2018. 1, 2

[16] Zhengqin Li and Jiansheng Chen. Superpixel segmentation using linear spectral clustering. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 1356–1363, 2015. 1, 2, 7

[17] Ming Yu Liu, O Tuzel, S Ramalingam, and R Chellappa. Entropy rate superpixel segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 2097–2104, 2011. 1, 2, 7

[18] Yong Jin Liu, Cheng Chi Yu, Min Jing Yu, and Ying He. Manifold slic: A fast method to compute content-sensitive superpixels. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 651–659, 2016. 1, 2

[19] Y. J. Liu, M. Yu, B. J. Li, and Y. He. Intrinsic manifold slic: A simple and efficient method for computing content-sensitive superpixels. *IEEE Trans. Pattern Anal. Mach. Intell.*, PP(99):1–1, 2017. 1, 2

[20] D. R. Martin, C. C. Fowlkes, and J Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):530–549, 2004. 7

[21] Arbelez P, Maire M, Fowlkes C, and Malik J. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916, 2011. 7

[22] Q. She, F. Feng, X. Hao, Q. Yang, and Rhm Chan. Openloris-object: A robotic vision dataset and benchmark for lifelong deep learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020. 2

[23] Jianbing Shen, Yunfan Du, Wenguan Wang, and Xuelong Li. Lazy random walks for superpixel segmentation. *IEEE Trans. Image Process.*, 23(4):1451–1462, 2014. 2

[24] Jianbing Shen, Xiaopeng Hao, Zhiyuan Liang, Yu Liu, Wenguan Wang, and Ling Shao. Real-time superpixel segmentation by dbscan clustering algorithm. *IEEE Trans. Image Process.*, 25(12):5933–5942, 2016. 2

[25] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 1997. 2

[26] J. Staal, M.D. Abramoff, M. Niemeijer, M. A. Viergever, and B. Van Ginneken. Ridge-based vessel segmentation in color images of the retina. *IEEE Transactions on Medical Imaging*, 23(4):501–509, 2004. 7

[27] Teppei Suzuki. Superpixel segmentation via convolutional neural networks with regularized information maximization. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020. 1, 2, 3, 4, 6, 7

[28] Wei-Chih Tu, Ming-Yu Liu, Varun Jampani, Deqing Sun, Shao-Yi Chien, Ming-Hsuan Yang, and Jan Kautz. Learning superpixels with segmentation-aware affinity loss. In *Int. Conf. Comput. Vis.*, pages 568–576, 2018. 1, 2

[29] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. *Computer ence*, 2015. 6

[30] Fengting Yang, Qian Sun, Hailin Jin, and Zihan Zhou. Superpixel segmentation with fully convolutional networks. 2020. 1, 2, 3, 4

[31] Lizhu Ye, Lei Zhu, Xuejing Kang, and Anlong Ming. Adaptive occlusion boundary extraction for depth inference. In *IEEE Int. Conf. Image Process. (ICIP)*, pages 4025–4029. IEEE, 2019. 1

[32] Donghun Yeo, Jeany Son, Bohyung Han, and Joon Hee Han. Superpixel-based tracking-by-segmentation using markov chains. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 511–520, 2017. 1