

Hierarchical Superpixel Segmentation via Structural Information Theory

Minhui Xie* Hao Peng*[✉] Pu Li* Guangjie Zeng* Shuhai Wang† Jia Wu‡
Peng Li* Philip S. Yu§

Abstract

Superpixel segmentation is a foundation for many higher-level computer vision tasks, such as image segmentation, object recognition, and scene understanding. Existing graph-based superpixel segmentation methods typically concentrate on the relationships between a given pixel and its directly adjacent pixels while overlooking the influence of non-adjacent pixels. These approaches do not fully leverage the global information in the graph, leading to sub-optimal segmentation quality. To address this limitation, we present SIT-HSS, a hierarchical superpixel segmentation method based on structural information theory. Specifically, we first design a novel graph construction strategy that incrementally explores the pixel neighborhood to add edges based on 1-dimensional structural entropy (1D SE). This strategy maximizes the retention of graph information while avoiding an overly complex graph structure. Then, we design a new 2D SE-guided hierarchical graph partitioning method, which iteratively merges pixel clusters layer by layer to reduce the graph's 2D SE until a predefined segmentation scale is achieved. Experimental results on three benchmark datasets demonstrate that the SIT-HSS performs better than state-of-the-art unsupervised superpixel segmentation algorithms. The source code is available at <https://github.com/SELGroup/SIT-HSS>.

Keywords: Superpixel Segmentation, Structural Entropy, Hierarchical Graph Partitioning

1 Introduction

Superpixel segmentation is a key technique in computer vision, intending to group pixels into compact regions with a consistent semantic context [7]. It usually works as the basis for higher-level computer vision tasks by reducing the number of primitives [21]. Various downstream applications can benefit from superpixel segmentation, including image segmentation [13, 18], image classification [22, 42], saliency detection [20, 41], and 3D reconstruction [9, 40].

The requirements of superpixel segmentation vary for different downstream application scenarios, where some commonly agreed requirements include **Connectivity**, **Boundary Adherence**, **Compactness**, and

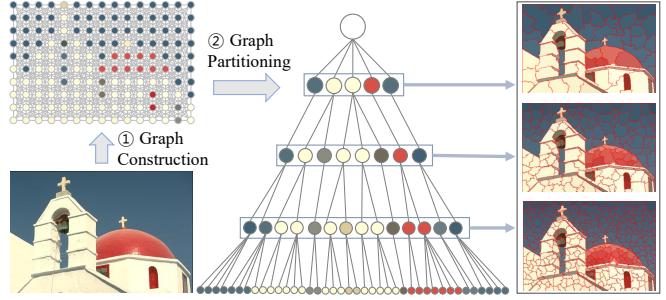


Figure 1: An illustration of superpixel segmentation using SIT-HSS. It performs superpixel segmentation by first constructing a weighted graph, then obtaining an optimal encoding tree, and finally achieving multi-scale superpixel segmentation.

Computational Efficiency. In the last two decades, superpixel segmentation has made considerable progress with the emergence of numerous techniques, such as watershed-based [35], density-based [28], path-based [26], clustering-based [1], graph-based [30] and more recently, deep learning-based methods [32]. Among them, density-based methods cannot determine the number of resulting superpixels, whereas path-based methods generate superpixels with poor compactness [7]. Although some methods [5] focus on a single requirement, most involve trade-offs between different requirements to cover a wide range of downstream applications. Graph-based methods construct graphs from the given image and partition these graphs into subgraphs as superpixels. This graph construction step helps them control the trade-off between boundary adherence and compactness.

Various graph-based methods have been proposed since NCut [24] is utilized as a superpixel segmentation technique for image classification [21]. They typically map the image as a graph and optimize a specific objective, such as the entropy rate [17] or normalized cuts [21] to achieve graph partitioning for superpixel segmentation. Recently, the hierarchy of superpixels has been addressed, as the multi-resolution representation of hierarchical superpixel segmentation is believed to benefit downstream applications [30, 31]. Despite the significant advancement achieved by these methods,

*School of Cyber Science and Technology, Beihang University, China. {xieminhui, penghao, zengguangjie}@buaa.edu.cn, lp2024626@gmail.com, lp_academy@163.com.

†College of Information Science and Technology, Shijiazhuang Tiedao University, China. wsh36302@126.com.

‡Macquarie University, Australia. jia.wu@mq.edu.au.

§University of Illinois at Chicago, Illinois, USA. psyu@uic.edu.

✉ Corresponding author.

existing graph-based methods overlook the relationships of non-adjacent pixels. In the graph construction step, they connect only pixels to their directly adjacent pixels. These adjacent pixel graphs do not fully capture the relationships between pixels and inevitably impede overall segmentation performance.

To address the aforementioned issues, we present SIT-HSS, a hierarchical superpixel segmentation method based on structural information theory. As illustrated in Fig. 1, we achieve superpixel segmentation by graph partitioning on a pixel graph, where the graph is constructed by maximizing 1D SE, and graph partitioning is conducted by minimizing 2D SE. Maximizing 1D SE retains more information between pixels of the image in the graph, delivering high-quality superpixel segmentation with accurate capture of object boundaries. Firstly, we construct an undirected pixel graph from the image. In this graph, graph nodes represent pixels, and weighted edges connect spatially close pixels, measuring their colour and spatial space similarities. To extract adequate information from the image, we devise a novel graph construction strategy by maximizing the 1D SE of the graph. This strategy calculates the graph's 1D SEs with increasing search radii and selects the radius where the 1D SE plateaus. Pixels connect to neighbour pixels within this radius, extracting sufficient information while avoiding complex, dense graphs. Secondly, we propose a hierarchical graph partitioning algorithm based on merging using 2D SE minimization. This algorithm iteratively merges the partition pair with the most significant decrease in 2D SE until the number of partitions matches the pre-defined superpixel segmentation scale. Additionally, the merging is performed only on pairs of partitions directly adjacent to the image, ensuring the connectivity of the superpixels generated.

We comprehensively evaluated the SIT-HSS in three datasets compared to nine baselines. Experimental results demonstrate that SIT-HSS outperforms state-of-the-art superpixel segmentation methods regarding segmentation accuracy while maintaining competitive efficiency. The contributions of this paper are summarized below.

- An interpretable hierarchical superpixel segmentation method based on structural information theory, namely SIT-HSS, is presented.
- A novel graph construction strategy based on 1D SE maximization is devised to retain adequate information from the image while avoiding complex, dense graphs.
- A hierarchical graph partitioning algorithm based on 2D SE minimization is proposed to achieve efficient and accurate superpixel segmentation.

- Extensive comparative experiments and visualizations are conducted on three representative datasets, demonstrating the state-of-the-art (SOTA) performance of SIT-HSS on superpixel segmentation.

2 Preliminaries

In this section, we briefly introduce the structural information theory on which our SIT-HSS is based. First proposed by Li and Pan [14], structural information theory comprehensively defines structural entropy, quantifying graph complexity and connectivity patterns through encoding trees. It has achieved successful applications in many fields, including skin lesion segmentation [36], graph clustering [25, 37], classification [33] and graph structure learning [44], social bot detection [19], social event detection [4], and reinforcement learning [38, 39]. The structural entropy of a graph is defined as the minimum average number of communities required to encode the graph nodes reachable in a random walk. The detailed definitions of encoding trees and structural entropy are given below.

DEFINITION 1. (Encoding Tree). *Given a graph $G = (V, E, \mathbf{W})$, an encoding tree T is a hierarchical partition of the graph G . The encoding tree satisfies the following conditions:*

1) *The root node λ of the encoding tree corresponds to the entire set of graph nodes, $T_\lambda = V$. Each node α in the tree is associated with a subset of graph nodes $T_\alpha \subseteq V$.*

2) *For any leaf node γ , T_γ contains exactly one graph node, i.e., $T_\gamma = \{v\}$, where $v \in V$. For any node α in the tree, its children are denoted as β_1, \dots, β_k , so $(T_{\beta_1}, \dots, T_{\beta_k})$ forms a partitioning of T_α .*

3) *Each tree node α has a height denoted as $h(\alpha)$. For any leaf node γ , $h(\gamma) = 0$; for a node α^- (the parent of α), $h(\alpha^-) = h(\alpha) + 1$. The height of the tree T , denoted as $h(T)$, is defined as the maximum height of all nodes, i.e., $h(T) = \max_{\alpha \in T} \{h(\alpha)\}$.*

DEFINITION 2. (Structural Entropy). *The structural entropy of a graph G with respect to an encoding tree T is defined as:*

$$(2.1) \quad H^T(G) = - \sum_{\alpha \in T, \alpha \neq \lambda} \frac{g_\alpha}{V_G} \log \frac{V_\alpha}{V_{\alpha^-}},$$

where g_α is the cut, i.e., the weighted sum of edges with exactly one endpoint in T_α . V_α , V_{α^-} , and V_G denote the volumes, i.e., the sums of node degrees within T_α , T_{α^-} , and G , respectively.

The structural entropy of different dimensions corresponds to encoding trees of different heights, reflecting various levels of structural information in the graph.

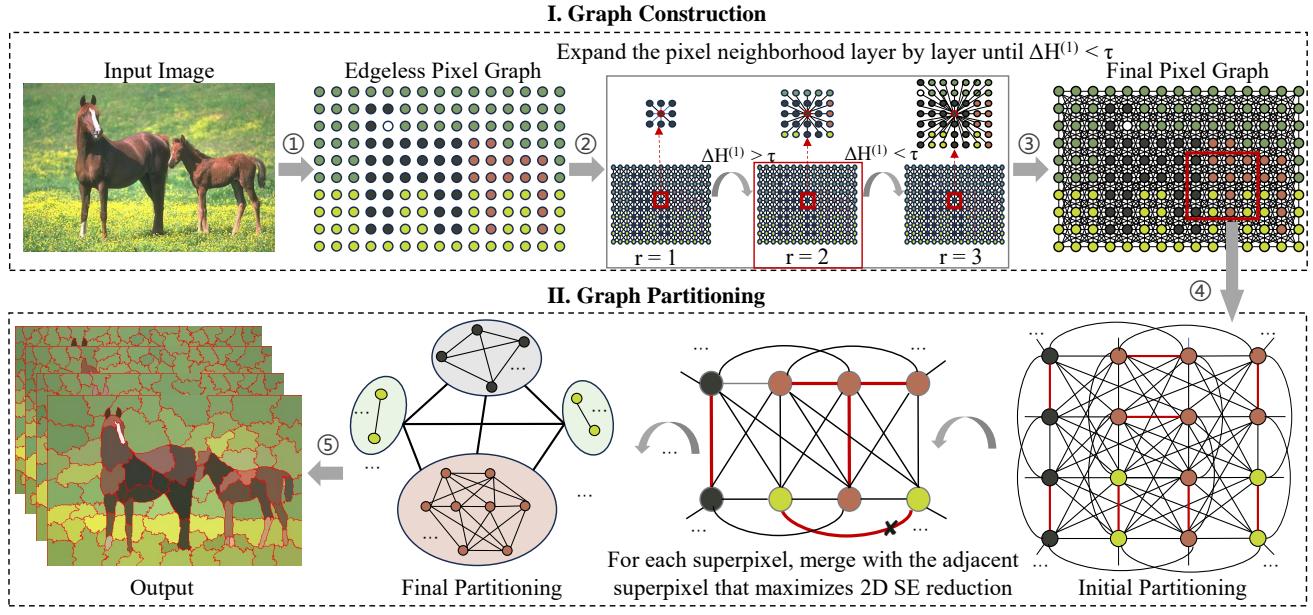


Figure 2: The framework of the SIT-HSS. I and II represent the weighted graph construction and partitioning processes, respectively. We add edges to the edgeless pixel graph by progressively expanding the neighbourhood. Then, for each superpixel, we iteratively merge neighbouring superpixel pairs that maximally reduce the 2D SE until the target size is reached.

The D -dimensional SE $H^{(D)}(G)$ of G is defined as:

$$(2.2) \quad H^{(D)}(G) = \min_{\forall T: h(T)=D} \{H^T(G)\},$$

which is realized by obtaining an optimal encoding tree of height D that minimizes the interference caused by noise or random variations.

3 Methodology

Superpixel segmentation groups pixels with similar colours and proximity. We approach this as an interpretable graph partitioning problem, where pixels are represented as graph nodes, and closely related nodes are grouped into partitions as superpixels. As illustrated in Fig. 2, our proposed SIT-HSS framework consists of two modules: graph construction and partitioning. In the graph construction module, we build a multilayer neighbourhood graph based on 1D SE maximization. This entropy maximization strategy enhances the expressiveness of graphs by retaining more information. In the graph partitioning module, we employ a heuristic node merging strategy to partition graphs efficiently using 2D SE minimization. We introduce each module in detail in the following subsections.

3.1 Problem Formalization. Given an image as the input, the task of superpixel segmentation can be accomplished by constructing and partitioning a weighted graph $G = (V, E, \mathbf{W})$, where $V = \{v_1, \dots, v_N\}$

represents the pixel points of the image, E denotes the edges, and the weight matrix \mathbf{W} measures the similarities between pixels. For two pixel points v_i and v_j , their distance is:

$$(3.3) \quad \rho_{i,j} = \|c_i - c_j\|_2^2 \cdot \|s_i - s_j\|_2,$$

where c_i and c_j represents the color features of v_i and v_j , respectively, and s_i and s_j represents the position features of v_i and v_j , respectively. $\|\cdot\|_2$ denotes Frobenius norm. Then we normalize this distance to represent the weight $\mathbf{W}_{i,j}$ between the two points:

$$(3.4) \quad \mathbf{W}_{i,j} = \exp\left(-\frac{\rho_{i,j}}{t \cdot \frac{1}{|\rho|} \sum_{k=1}^{|\rho|} \rho_k}\right),$$

where $|\rho|$ is the total number of edges, ρ_k is the distance between the endpoints of the k -th edge, and t is a normalization parameter. The edge set E is initially empty and will be expanded by the graph construction process. Partitioning G results in a set $\{p_1, \dots, p_n\}$, where $\bigcap_{i \neq j} (p_i \cap p_j) = \emptyset$ and $\bigcup_{i=1}^n p_i = V$. Each partition p_i represents a segmented superpixel.

3.2 Graph Construction. Existing graph-based superpixel segmentation methods focus only on the relationships between adjacent pixels and overlook the relationships between non-adjacent pixels, leading to inferior performance. We propose a graph construction

strategy based on 1D SE to retain as much information from the image as possible. As shown in Fig. 2 (I), we construct an edgeless graph G with pixels as nodes and add edges to it by progressively expanding the neighbourhood. For each graph node v_i , we define the neighbourhood $\mathcal{N}_{v_i}^r$ as the set of pixels (graph nodes) within a distance of r and connect v_i to all nodes in $\mathcal{N}_{v_i}^r$. When the search radius $r = 1$, the neighbourhood $\mathcal{N}_{v_i}^r$ of v_i consists of the eight directly adjacent pixels. When r is more considerable, $\mathcal{N}_{v_i}^r$ expands to include pixels at a more considerable distance. The 1D SE of G at a given search radius r is given by:

$$(3.5) \quad H_r^{(1)}(G) = - \sum_{i=1}^{|V|} \frac{d_i}{\mathcal{V}_G^r} \log \frac{d_i}{\mathcal{V}_G^r},$$

where \mathcal{V}_G^r is the volume of G under r , and d_i is the sum of weights of edges connecting v_i . We tend to construct a graph with the most information retained, corresponding to the graph with the largest 1D SE. As the search radius r increases, the 1D SE of G typically increases. However, a large value of r leads to a dense complex G , making the subsequent optimization of graph partitioning difficult. Moreover, with further increases in r , the rate of growth of the 1D SE often decreases. Therefore, we select an appropriate value of r to construct a graph that retains adequate information while preventing an overly complex graph structure. Formally, we select r as the following:

$$(3.6) \quad r = \arg \min_i \left\{ i \mid H_{i+1}^{(1)}(G) - H_i^{(1)}(G) < \tau \right\},$$

where τ is a predefined SE threshold for 1D SE change.

3.3 Graph Partitioning. The 2D SE of a given graph G represents the minimum SE achievable through traversing encoding trees at the height of 2, which corresponds to graph partitionings. We perform superpixel segmentation by minimizing the 2D SE of the constructed weighted graph G . As illustrated in Fig. 2 (II), we employ a method based on heuristic node merging for the 2D SE minimization.

For a weighted pixel graph G , the 2D SE of G given by a partitioning $\mathcal{P} = \{p_1, \dots, p_n\}$ is defined as follow:

$$(3.7) \quad H_{\mathcal{P}}^{(2)}(G) = - \sum_{p_i \in \mathcal{P}} \left(\frac{g_{p_i}}{\mathcal{V}_G} \log \frac{\mathcal{V}_{p_i}}{\mathcal{V}_G} + \sum_{j \in p_i} \frac{d_j}{\mathcal{V}_G} \log \frac{d_j}{\mathcal{V}_{p_i}} \right),$$

where d_j is the sum of edge weights connecting p_i and other graph nodes, \mathcal{V}_{p_i} and \mathcal{V}_G are the sum of node degrees in partition p_i and graph G , respectively, and g_{p_i} is the sum of edge weights between nodes inside and outside partition p_i . At first, each pixel belongs to an

Algorithm 1 Proposed SIT-HSS

```

1: Input: An image  $I \in \mathbb{R}^{H \times W}$ , target size  $K$ ,  $t$ ,  $\tau$ .
2: Output: Superpixel blocks of images  $\mathcal{P}$ .
3: // Graph Construction.
4:  $G(V) \leftarrow$  Convert image  $I$  to an edgeless graph;
5: for  $r \in 1, 2, 3, \dots$  do
6:   for pixel  $v_i$  in  $V$  do
7:     for  $v_j$  in  $\mathcal{N}_{v_i}^r$  do
8:       add  $(v_i, v_j)$  to  $E_r$ ,  $\rho_{i,j} \leftarrow$  Eq. 3.3;
9:    $\mathbf{W}_r \leftarrow$  Eq. 3.4;
10:   $H_r^{(1)}(G) \leftarrow$  Eq. 3.5;
11:  if  $H_r^{(1)} - H_{r-1}^{(1)} \leq \tau$  then
12:     $G \leftarrow G(V, E_{r-1}, \mathbf{W}_{r-1})$ ;
13:    Break;
14: // Graph Partitioning.
15:  $\mathcal{P} = \{\{v_1\}, \{v_2\}, \dots, \{v_n\}\}$ ;
16: while True do
17:   for superpixel  $p_i$  in  $\mathcal{P}$  do
18:      $p_{tgt}^i \leftarrow \emptyset$ ,  $\Delta_{max} \leftarrow 0$ ;
19:     for adjacent superpixel  $p_j$  in  $\mathcal{N}_{p_i}$  do
20:        $\Delta H_{p_i, p_j}^{(2)} \leftarrow$  Eq. 3.8;
21:       if  $\Delta H_{p_i, p_j}^{(2)} > \Delta_{max}$  then
22:          $p_{tgt}^i \leftarrow p_j$ ,  $\Delta_{max} \leftarrow \Delta H_{p_i, p_j}^{(2)}$ ;
23:       Merge  $p_i$ ,  $p_{tgt}^i$  to  $p'_i$ ;
24:     Update  $\mathcal{P}$  and all superpixels' volumes and cuts;
25:   if  $|\mathcal{P}| = K$  then
26:     Return  $\mathcal{P}$ ;

```

independent partition, i.e. $p_x = \{v_x\}$. We minimize the 2D SE objective defined by Eq. 3.7 via the merging operator [14]. For a pair of directly adjacent superpixels p_i and p_j , $MERGE(p_i, p_j)$ adds a new superpixel p'_i to G , whose child nodes are the union of the child nodes of p_i and p_j , and removes p_i and p_j from G . After merging, the new partition is $\mathcal{P}' = \{p_1, \dots, p'_i, \dots, p_n\}$, where $p'_i = p_i \cup p_j$. The decrease amount of $H_{\mathcal{P}'}^{(2)}(G)$ after merging is given by:

$$(3.8) \quad \begin{aligned} \Delta H_{p_i, p_j}^{(2)} &= H_{\mathcal{P}}^{(2)} - H_{\mathcal{P}'}^{(2)} \\ &= \frac{1}{\mathcal{V}_G} ((\mathcal{V}_{p_i} - g_{p_i}) \log \mathcal{V}_{p_i} + (\mathcal{V}_{p_j} - g_{p_j}) \log \mathcal{V}_{p_j} \\ &\quad - (\mathcal{V}_{p'_i} - g_{p'_i}) \log \mathcal{V}_{p'_i} + (g_{p_i} + g_{p_j} - g_{p'_i}) \log \mathcal{V}_G), \end{aligned}$$

where $H_{\mathcal{P}}^{(2)}$ and $H_{\mathcal{P}'}^{(2)}$ denote the 2D SE of the graph under the partitioning \mathcal{P} and \mathcal{P}' , respectively. Therefore, we select a superpixel directly adjacent for each superpixel, which results in the most significant decrease in 2D SE after merging in each iteration. This adjacent partition selection is formalized as follows:

$$(3.9) \quad p_{tgt}^i = \arg \max_{p_j \in \mathcal{N}_{p_i}} (\Delta H_{p_i, p_j}^{(2)}),$$

where p_{tgt}^i is the target superpixel for p_i and \mathcal{N}_{p_i} is the set of directly adjacent superpixels of p_i . At the end of each iteration, we dynamically update the volume of each partition \mathcal{V}_{p_i} and the external edge weights g_{p_i} . Finally, the iteration process stops when the target number of superpixels, known as the target size (K), is reached.

3.4 Time Complexity. The overall time complexity of the proposed SIT-HSS algorithm is $O(|E|)$, as shown in Algorithm 1, where $|E|$ denotes the number of edges in the pixel graph. Specifically, the time complexity for searching the optimal radius during the graph construction phase is $O(l_1 \cdot |V| \cdot k) = O(|E|)$ (lines 4-13), where l_1 is the number of search iterations and k is the number of neighbouring pixels. Additionally, the time required for graph partitioning is $O(l_2 \cdot |\mathcal{P}| \cdot m) \leq O(|E|)$ (lines 15-26), where l_2 is the number of iterations, $|\mathcal{P}|$ (with $|\mathcal{P}| \leq |V|$) is the number of superpixels, and m is the average number of neighbors for each superpixel. Therefore, the overall time complexity of SIT-HSS is $O(|E| + |E|) = O(|E|)$. We further discuss the algorithm’s convergence in detail in the appendix. Notably, we employ tensor operations to optimize actual running time, allowing all superpixels to nearly parallelize the selection and merging of the optimal neighbouring pixels in each iteration. Additionally, the process can be further accelerated through GPU processing. By leveraging GPUs’ computing capabilities and tensor operations, SIT-HSS is competitive with the fastest existing algorithms in terms of efficiency.

4 Experiments

In this section, we conduct a series of extensive experiments to evaluate the performance of the SIT-HSS. Through these experiments, we aim to address the following four research questions: **RQ1**: How does the proposed SIT-HSS perform on different datasets compared to the baselines? **RQ2**: How do the visualization results of SIT-HSS image segmentation perform, and does the algorithm exhibit interpretability? **RQ3**: How do the hyperparameters affect the performance of the SIT-HSS algorithm? **RQ4**: How does SIT-HSS perform in terms of efficiency compared to baselines?

Datasets. We perform experiments on three representative datasets, including the Berkeley Segmentation Dataset and Benchmark 500 (BSDS500) [3], the Stanford Background Dataset (SBD) [8] and the PASCAL-S dataset [16]. The BSDS500 dataset includes 500 grayscale and colour images from natural scenes, providing an empirical foundation for image segmentation research. On average, each image in the BSDS500 is segmented into five distinct regions. The SBD dataset

is created by selecting complex outdoor scenes from several datasets, totalling 715 images. The PASCAL-S dataset contains 850 images with different object categories and sizes.

Evaluation Metrics. We employ four widely used metrics [7] to evaluate the quality of superpixel segmentation, including Achievable Segmentation Accuracy (ASA), Boundary Recall (BR), Undersegmentation Error (UE), and Explained Variation (EV). ASA, BR, and UE depend on expert ground truth annotations, while EV is independent of any ground truth. Specifically, ASA measures the achievable fraction of correctly labeled superpixels, BR evaluates how well the superpixel boundaries align with the ground truth boundaries, UE assesses how much a superpixel spans multiple ground-truth segments, and EV measures the extent of variations that superpixels capture.

Baselines. **1) Graph-based methods** treat each pixel as a node in a graph and generate superpixels by minimizing a cost function defined on the graph: ERS [17], and DRW [11]. **2) Clustering-based methods** group pixels into clusters (i.e., superpixels) and iteratively refine them until certain convergence criteria are met: SLIC [1], DBSCAN [23], LSC [6], SNIC [2], SEEDS [27] and ETPS [34]. **3) Unsupervised deep learning-based methods** generate superpixels through end-to-end clustering without the need for labeled data: LNSNET [43].

Implementation Details. For all datasets, we set the normalization parameter $t = 0.1$. For the BSDS500 and PASCAL-S datasets, we set the SE increment threshold $\tau = 2e - 7$; for the SBD dataset, we set $\tau = 1e - 6$. All experiments are conducted on one server, which has an Intel(R) Xeon(R) Platinum 8336C CPU operating at 2.30 GHz and an RTX 4090 GPU with 24GB of memory.

4.1 Quantitative Result Comparison (RQ1). Table 1 shows the quantitative results in both datasets when $K = 600$. SIT-HSS emerges as the most superior method, outperforming all baselines across both three datasets. Specifically, for boundary adherence, SIT-HSS achieves the highest performance on the BR metric. This is owing to SIT-HSS’s consideration of the influence of non-adjacent nodes and global information, which allows it to capture more details and improve boundary adherence. Regarding UE, our method also outperforms other algorithms, clearly demonstrating its capability to reduce under-segmentation errors and excel in weak boundary detection. Regarding the consistency of pixels within each superpixel, SIT-HSS shows the best performance on the metric EV on two datasets and the runner-up performance on the third dataset. The advantage of SIT-HSS on consistency lies

Table 1: Quantitative results of superpixel segmentation methods when $K = 600$. The best results are **bolded**, and the second-best results are underlined.

Method (%)	BSDS500				SBD				PASCAL-S			
	ASA↑	BR↑	UE↓	EV↑	ASA↑	BR↑	UE↓	EV↑	ASA↑	BR↑	UE↓	EV↑
SLIC	95.890	89.903	3.967	84.456	94.909	85.721	5.201	90.863	98.401	87.967	8.092	89.255
SEEDS	94.673	<u>97.018</u>	5.337	87.375	93.825	<u>97.673</u>	6.349	93.276	98.520	<u>96.099</u>	7.668	89.035
ERS	95.699	94.887	4.025	82.335	94.893	90.026	5.280	87.005	98.503	95.087	7.652	89.915
ETPS	<u>96.258</u>	91.197	<u>3.552</u>	<u>89.051</u>	<u>95.203</u>	89.002	<u>4.867</u>	<u>94.115</u>	<u>98.776</u>	95.376	<u>7.032</u>	91.052
DBSCAN	95.694	89.889	4.145	85.225	94.530	86.497	5.547	90.827	98.376	91.668	8.090	90.042
LSC	95.640	89.745	4.400	85.823	94.602	87.141	5.590	90.441	98.524	95.028	7.640	89.986
SNIC	96.122	89.772	3.820	86.005	94.900	85.708	5.109	92.457	98.215	90.449	8.278	90.543
LNSNET	95.705	90.510	4.197	82.590	94.506	88.290	5.510	89.386	98.267	90.360	8.059	88.569
DRW	95.644	94.577	4.209	85.110	94.703	90.118	5.201	90.838	98.262	94.143	7.809	89.120
SIT-HSS	96.824	97.981	3.076	89.078	95.250	99.701	4.729	94.392	98.822	97.248	6.178	90.870
relative gain	0.566	0.963	0.476	0.027	0.047	2.208	0.138	0.277	0.046	1.149	0.854	-0.182

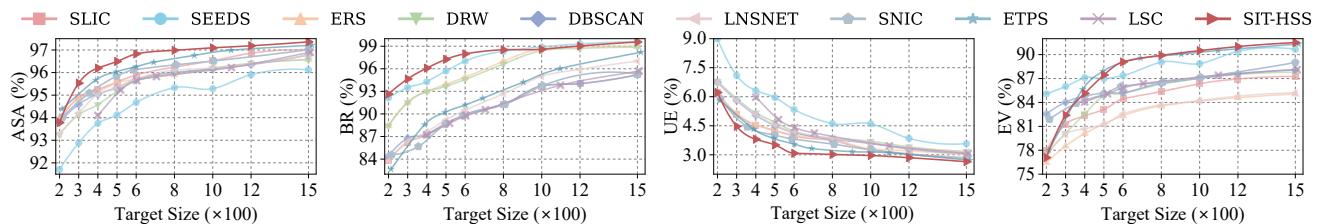


Figure 3: ASA, BR, UE, and EV Curve of our SIT-HSS compared with other state-of-the-art superpixel segmentation algorithms on the BSDS500 dataset.

in the structural entropy-based cost function, which effectively captures similarities between pixels. At the same time, the contour evolution-based method ETPS achieves competitive EV values due to its sensitivity to continuous feature variations. Additionally, SIT-HSS performs well on the ASA metric, indicating that the superpixel generated by our SIT-HSS has a reasonable upper bound for adhering to the object boundaries. In summary, the SIT-HSS demonstrates exceptional performance across multiple metrics, meeting the universal standards for superpixel segmentation. Figure 3 presents the quantitative comparisons with the SOTA algorithms for different K values in the BSDS500 dataset. For the results of the SBD and PASCAL-S datasets, please refer to the appendix.

4.2 Qualitative Result Comparison (RQ2). Fig. 4 shows a visual comparison of seven representative algorithms on images from three datasets with $K = 100$. Among these algorithms, the segmentations ERS, SEEDS, ETPS, LNSNET, and SIT-HSS exhibit good boundary adherence, with most of the boundaries preserved. However, ERS and LNSNET generate distorted boundaries, leading to low compactness. SEEDS and ETPS fail to capture the boundaries of small objects, such as the puppy’s eye in the third row. The

segmentations of SLIC and SNIC can produce relatively regular superpixels in a regular shape. Nevertheless, they cannot adhere to object boundaries very well. In all, SIT-HSS achieves good boundary adherence with high compactness.

To further analyze the performance and demonstrate the interpretability of SIT-HSS in superpixel segmentation, we visualize the weighted pixel graph and the structural entropy encoding trees, as shown in Fig. 5. Specifically, we construct the weighted pixel graph as described in Sect. 3.2 and calculate the positions of the graph nodes using the spring embedder with the Fruchterman-Reingold force-directed algorithm [12]. Figs. 5 (b) and (e) show the graph embedding with ground-truth labels and superpixel labels segmented by SIT-HSS, respectively. The weighted pixel graph clearly distinguishes the clusters in the ground truth, validating the effectiveness of the SIT-HSS graph construction method. We construct encoding trees from ground truth labels and superpixel labels and visualize them in Fig. 5 (c) and (f), respectively. In these trees, the square tree nodes represent superpixels, and the circular leaf nodes represent pixels. After segmentation, the superpixel labels closely match the ground truth, indicating that SIT-HSS correctly segmented most objects into superpixels. This validates the effectiveness of the graph

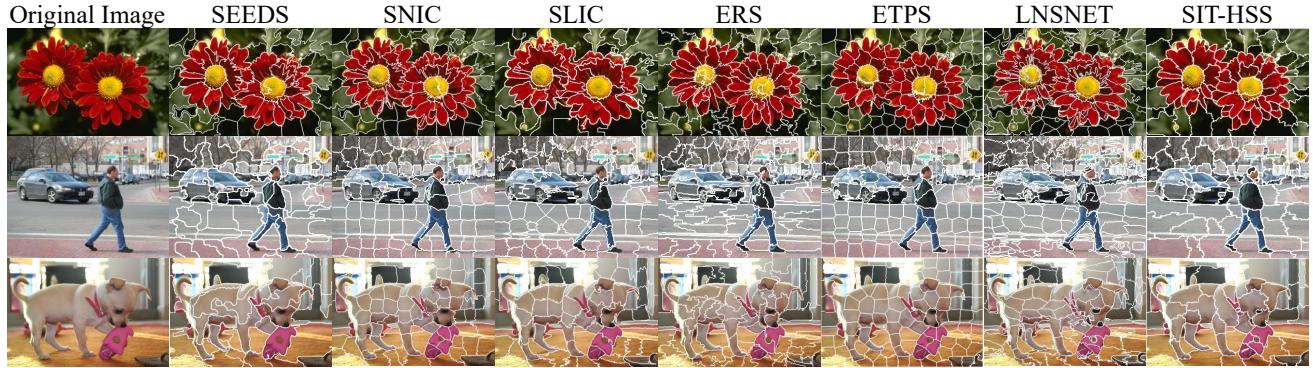


Figure 4: Visualization of different superpixel segmentation algorithms on different datasets when $K = 100$.

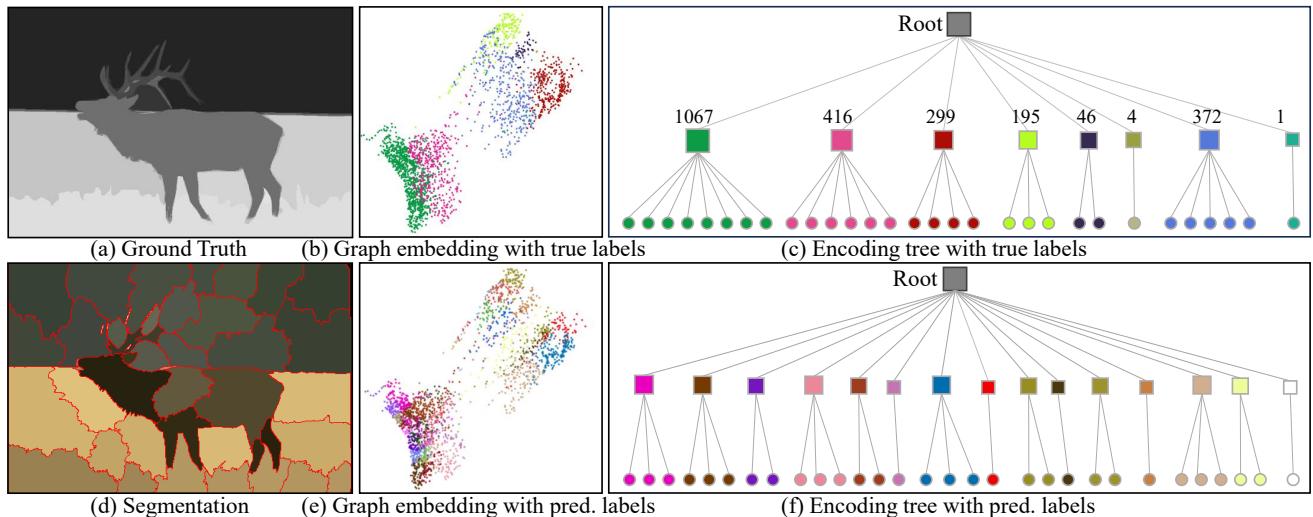


Figure 5: Visualization of the effectiveness validation of graph construction and partitioning.

partitioning process.

4.3 Hyperparameter Study (RQ3). As shown in Fig. 6, we conduct the hyperparameter experiments on the BSDS500 dataset to investigate the impact of the weight normalization parameter t (which controls the sensitivity of pixel similarity to distance in Eq. 3.4) and the SE threshold τ (used to determine the search radius r). Fig. 6a illustrates that when t is small (e.g. 0.05), the weight normalization function changes rapidly, leading to significant weight differences even with minor changes in the distance of the pixels. This sensitivity to pixel variation increases the BR value but also makes the segmentation result more susceptible to noise, reducing ASA and EV. In contrast, when t is large (e.g. 0.20), the weight function becomes less sensitive to distance, resulting in smoother segmentation. This reduces over-segmentation but potentially causes a loss of boundary details and slightly lowers BR value. When $t = 0.10$, ASA and EV reach their peak values, indicating that a moderate t balances segmentation accuracy and boundary preservation well. Furthermore, Fig. 6b shows that when the SE threshold τ is small (e.g. 1e-7), the selected neighborhood radius r is large, making the graph overly dense, potentially containing unimportant information or even noise. When τ is large (e.g., 4e-7), the radius of the neighbourhood becomes small, resulting in an overly sparse graph structure that retains insufficient information. An appropriate value of τ helps to prevent an overly complex graph structure while retaining valuable information in the graph. For hyperparameter studies on other datasets, please refer to the Appendix.

ing that a moderate t balances segmentation accuracy and boundary preservation well. Furthermore, Fig. 6b shows that when the SE threshold τ is small (e.g. 1e-7), the selected neighborhood radius r is large, making the graph overly dense, potentially containing unimportant information or even noise. When τ is large (e.g., 4e-7), the radius of the neighbourhood becomes small, resulting in an overly sparse graph structure that retains insufficient information. An appropriate value of τ helps to prevent an overly complex graph structure while retaining valuable information in the graph. For hyperparameter studies on other datasets, please refer to the Appendix.

4.4 Efficiency (RQ4). Table 2 reports the average processing time of each method per image. SIT-HSS achieves top-tier efficiency among these methods. Compared to the fastest method, DBSCAN, SIT-HSS requires only an additional 0.084 seconds per image, demonstrating a comparable speed. In particular, SIT-

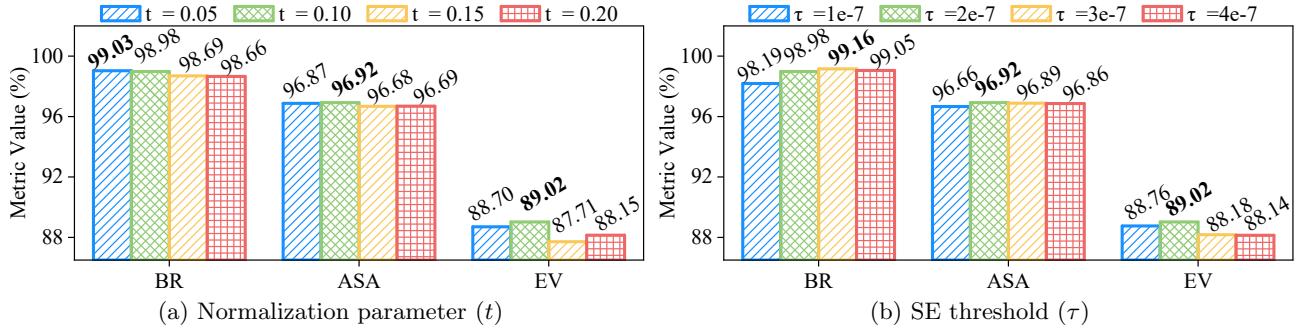


Figure 6: Performance of the SIT-HSS on BSDS500 with different hyperparameter settings.

Table 2: Computational time (s) of different superpixel segmentation algorithms on BSDS500, SBD, and PASCAL-S datasets when $K = 600$.

Methods	SLIC	SEEDS	ERS	ETPS	DBSCAN	LSC	SNIC	LNSNET	DRW	SIT-HSS
BSDS500	0.138	0.102	0.361	0.124	0.034	0.159	0.139	2.860	0.888	0.141
SBD	0.069	0.060	0.176	0.091	0.026	0.075	0.076	1.401	0.572	0.059
PASCAL-S	0.138	0.183	0.574	0.169	0.042	0.167	0.213	4.045	0.980	0.154
Average	0.115	0.115	0.370	0.128	0.034	0.134	0.143	2.769	0.813	0.118

HSS speeds up 3 to 10 times faster than other graph-based methods, such as ERS and DRW. Compared to the deep learning-based LNSNET method, which requires an average of 2.86 seconds, SIT-HSS significantly reduces processing time. Although SLIC, SEEDS, and DBSCAN perform marginally better in speed, SIT-HSS achieves a better balance between segmentation accuracy and efficiency. Furthermore, by adopting more powerful GPUs and advanced hardware, the processing time of SIT-HSS can be further reduced, making it easily scalable for large-scale image data processing.

5 RELATED WORK

In the last two decades, superpixel segmentation has made significant progress in terms of quality and efficiency, and numerous approaches have emerged. Among them, prevalent approaches include graph-based, cluster-based, and deep learning-based methods. Graph-based methods model the segmentation problem using graph theory and generate superpixels by minimizing a cost function defined on the graph [11, 17, 24]. Clustering-based methods group pixels and iteratively refine them until certain convergence criteria are met [1, 15, 27]. More recently, several deep learning-based methods [29, 32, 43] have been introduced, utilizing the high representation power of neural networks. However, their lack of interpretability and high computational complexity limit their range of applications.

Hierarchical superpixel segmentation highlights the importance of multiresolution representations. These approaches generate a set of superpixel segmentations

at different scales, where a coarser superpixel is composed of finer ones. LASH [10] agglomerates pixels into superpixel hierarchy based on a similarity function by reinforcement learning. SH [30] constructs the superpixel hierarchy using minimum spanning trees (MST). CRTrees [31] extend SH’s undirected MST into directed structures based on 1-NN graph theory, allowing for the automatic determination of the number of superpixels at each hierarchy. HHTS [5] iteratively partitions the segments top-down by thresholding. SIT-HSS achieves hierarchical agglomerative superpixel segmentation by hierarchical partitioning on pixel graphs, which retains adequate information through 1D SE maximization.

6 CONCLUSION

This paper introduces SIT-HSS, a novel hierarchical superpixel segmentation method via structural information theory. SIT-HSS achieves superpixel segmentation via hierarchical graph partitioning on pixel graphs based on the structural entropy. Unlike existing graph-based methods that consider only the relations of directly adjacent pixels, SIT-HSS constructs graphs by incrementally exploring the pixel neighborhood. This graph construction strategy maximizes the retention of graph information while avoiding an overly complex graph structure by maximizing 1D SE. Based on the pixel graph, SIT-HSS iteratively merges graph nodes into a hierarchical graph partitioning by 2D SE minimization, forming a hierarchical superpixel segmentation. Extensive experiments demonstrate that SIT-HSS achieves SOTA performance with top-tier efficiency.

Acknowledgments

This work is supported by the NSFC through grants 62322202, 62441612, and 62432006, the Shijiazhuang Science and Technology Plan Project through grant 231130459A, and the Guangdong Basic and Applied Basic Research Foundation through grant 2023B1515120020.

References

- [1] R. ACHANTA, A. SHAJI, K. SMITH, A. LUCCHI, P. FUA, AND S. SUSSTRUNK, *Slic superpixels compared to state-of-the-art superpixel methods*, IEEE TPAMI, 34 (2012), pp. 2274–2282.
- [2] R. ACHANTA AND S. SUSSTRUNK, *Superpixels and polygons using simple non-iterative clustering*, in Proceedings of the CVPR, 2017, pp. 4651–4660.
- [3] P. ARBELEZ, M. MAIRE, C. FOWLKES, AND J. MALIK, *Contour detection and hierarchical image segmentation*, IEEE TPAMI, 33 (2011), pp. 898–916.
- [4] Y. CAO, H. PENG, Z. YU, AND S. Y. PHILIP, *Hierarchical and incremental structural entropy minimization for unsupervised social event detection*, in Proceedings of the AAAI, vol. 38, 2024, pp. 8255–8264.
- [5] T. V. CHANG, S. SEIBT, AND B. VON RYMON LIPINSKI, *Hierarchical histogram threshold segmentation—auto-terminating high-detail oversegmentation*, in Proceedings of the CVPR, 2024, pp. 3195–3204.
- [6] J. CHEN, Z. LI, AND B. HUANG, *Linear spectral clustering superpixel*, IEEE TIP, 26 (2017), pp. 3317–3330.
- [7] S. DAVID, H. ALEXANDER, AND L. BASTIAN, *Superpixels: An evaluation of the state-of-the-art*, Computer Vision and Image Understanding, 166 (2018), pp. 1–27.
- [8] S. GOULD, R. FULTON, AND D. KOLLER, *Decomposing a scene into geometric and semantically consistent regions*, in Proceedings of the ICCV, IEEE, 2009, pp. 1–8.
- [9] T.-K. HUANG, Y.-H. WANG, T.-K. LIN, AND Y.-Y. CHUANG, *A robust automatic object segmentation method for 3d printing*, in Proceedings of the ICME, IEEE, 2016, pp. 1–6.
- [10] V. JAIN, S. C. TURAGA, K. BRIGGMAN, M. HELMSTAEDTER, W. DENK, AND H. SEUNG, *Learning to agglomerate superpixel hierarchies*, Proceedings of the NeurIPS, 24 (2011), pp. 1–9.
- [11] X. KANG, L. ZHU, AND A. MING, *Dynamic random walk for superpixel segmentation*, IEEE TIP, 29 (2020), pp. 3871–3884.
- [12] S. G. KOBOUROV, *Spring embedders and force directed graph drawing algorithms*, arXiv preprint arXiv:1201.3011, (2012), pp. 1–23.
- [13] T. LEI, X. JIA, Y. ZHANG, S. LIU, H. MENG, AND A. K. NANDI, *Superpixel-based fast fuzzy c-means clustering for color image segmentation*, IEEE Transactions on Fuzzy Systems, 27 (2018), pp. 1753–1766.
- [14] A. LI AND Y. PAN, *Structural information and dynamical complexity of networks*, IEEE TIT, 62 (2016), pp. 3290–3339.
- [15] H. LI, Y. JIA, R. CONG, W. WU, S. T. W. KWONG, AND C. CHEN, *Superpixel segmentation based on spatially constrained subspace clustering*, IEEE TII, 17 (2020), pp. 7501–7512.
- [16] Y. LI, X. HOU, C. KOCH, J. M. REHG, AND A. L. YUILLE, *The secrets of salient object segmentation*, in Proceedings of the CVPR, 2014, pp. 280–287.
- [17] M.-Y. LIU, O. TUZEL, S. RAMALINGAM, AND R. CHELLAPPAA, *Entropy rate superpixel segmentation*, in CVPR, 2011, pp. 2097–2104.
- [18] T. C. NG, S. K. CHOY, S. Y. LAM, AND K. W. YU, *Fuzzy superpixel-based image segmentation*, Pattern Recognition, 134 (2023), p. 109045.
- [19] H. PENG, J. ZHANG, X. HUANG, Z. HAO, A. LI, Z. YU, AND P. S. YU, *Unsupervised social bot detection via structural information theory*, ACM Transactions on Information Systems, 42, pp. 1–42.
- [20] Y. QIU, J. MEI, AND J. XU, *Superpixel-wise contrast exploration for salient object detection*, Knowledge-Based Systems, 292 (2024), p. 111617.
- [21] X. REN AND J. MALIK, *Learning a classification model for segmentation*, Proceedings of the ICCV, (2003), pp. 10–17.
- [22] P. SELLARS, A. I. AVILES-RIVERO, AND C.-B. SCHÖNLIEB, *Superpixel contracted graph-based learning for hyperspectral image classification*, IEEE TGRS, 58 (2020), pp. 4180–4193.
- [23] J. SHEN, X. HAO, Z. LIANG, Y. LIU, W. WANG, AND L. SHAO, *Real-time superpixel segmentation by dbSCAN clustering algorithm*, IEEE TIP, 25 (2016), pp. 5933–5942.
- [24] J. SHI AND J. MALIK, *Normalized cuts and image segmentation*, IEEE TPAMI, 22 (2000), pp. 888–905.
- [25] L. SUN, Z. HUANG, H. PENG, Y. WANG, C. LIU, AND S. Y. PHILIP, *Lsenet: Lorentz structural entropy neural network for deep graph clustering*, in Proceedings of the ICML, 2024, pp. 1–12.
- [26] D. TANG, H. FU, AND X. CAO, *Topology preserved regular superpixel*, in Proceedings of the ICME, IEEE, 2012, pp. 765–768.
- [27] M. VAN DEN BERGH, X. BOIX, G. ROIG, B. DE CAPITANI, AND L. V. GOOL, *Seeds: Superpixels extracted via energy-driven sampling*, IJCV, 111 (2012), pp. 298–314.
- [28] A. VEDALDI AND S. SOATTO, *Quick shift and kernel methods for mode seeking*, in Proceedings of the ECCV, Springer, 2008, pp. 705–718.
- [29] Y. WANG, Y. WEI, X. QIAN, L. ZHU, AND Y. YANG, *Ainet: Association implantation for superpixel segmentation*, in Proceedings of the CVPR, 2021, pp. 7078–7087.
- [30] X. WEI, Q. YANG, Y. GONG, N. AHUJA, AND M.-H. YANG, *Superpixel hierarchy*, IEEE TIP, 27 (2018), pp. 4838–4849.
- [31] T. YAN, X. HUANG, AND Q. ZHAO, *Hierarchical su*

- perpixel segmentation by parallel crtrees labeling*, IEEE TIP, 31 (2022), pp. 4719–4732.
- [32] F. YANG, Q. SUN, H. JIN, AND Z. ZHOU, *Superpixel segmentation with fully convolutional networks*, in Proceedings of the CVPR, 2020, pp. 13961–13970.
- [33] R. YANG, H. PENG, A. LI, P. LI, C. LIU, AND S. Y. PHILIP, *Hierarchical abstracting graph kernel*, IEEE Transactions on Knowledge and Data Engineering, (2024), pp. 724–738.
- [34] J. YAO, M. BOBEN, S. FIDLER, AND R. URTASUN, *Real-time coarse-to-fine topologically preserving segmentation*, in Proceedings of the CVPR, 2015, pp. 2947–2955.
- [35] Y. YUAN, Z. ZHU, H. YU, AND W. ZHANG, *Watershed-based superpixels with global and local boundary marching*, IEEE TIP, 29 (2020), pp. 7375–7388.
- [36] G. ZENG, H. PENG, A. LI, Z. LIU, C. LIU, S. Y. PHILIP, AND L. HE, *Unsupervised skin lesion segmentation via structural entropy minimization on multi-scale superpixel graphs*, in Proceedings of the ICDM, IEEE, 2023, pp. 768–777.
- [37] G. ZENG, H. PENG, A. LI, J. WU, C. LIU, AND S. Y. PHILIP, *Scalable semi-supervised clustering via structural entropy with different constraints*, IEEE Transactions on Knowledge and Data Engineering, 37 (2025), pp. 478–492.
- [38] X. ZENG, H. PENG, AND A. LI, *Effective and stable role-based multi-agent collaboration by structural information principles*, in Proceedings of the AAAI, vol. 37, 2023, pp. 11772–11780.
- [39] X. ZENG, H. PENG, AND A. LI, *Effective exploration based on the structural information principles*, in Proceedings of the NeurIPS, 2024, pp. 1–35.
- [40] G. ZHANG, C. XUE, AND R. ZHANG, *Supernerf: High-precision 3d reconstruction for large-scale scenes*, IEEE TGRS, 62 (2024), p. 3435743.
- [41] J. ZHANG, J. CHEN, Q. WANG, AND S. CHEN, *Spatiotemporal saliency detection based on maximum consistency superpixels merging for video analysis*, IEEE TII, 16 (2019), pp. 606–614.
- [42] C. ZHAO, B. QIN, S. FENG, W. ZHU, W. SUN, W. LI, AND X. JIA, *Hyperspectral image classification with multi-attention transformer and adaptive superpixel segmentation-based active learning*, IEEE TIP, 32 (2023), pp. 3606–3621.
- [43] L. ZHU, Q. SHE, B. ZHANG, Y. LU, Z. LU, D. LI, AND J. HU, *Learning the superpixel in a non-iterative and lifelong manner*, in Proceedings of the CVPR, 2021, pp. 1225–1234.
- [44] D. ZOU, H. PENG, X. HUANG, R. YANG, J. LI, J. WU, C. LIU, AND P. S. YU, *Se-gsl: A general and effective graph structure learning framework through structural entropy optimization*, in Proceedings of the ACM Web Conference 2023, 2023, pp. 499–510.

Appendix

6.1 Quantitative Result Comparison. Fig. 7 and 8 illustrate the quantitative result of ASA, BR, UE, and EV of nine advanced superpixel segmentation algorithms and our SIT-HSS on SBD and PASCAL-S datasets. SIT-HSS demonstrates strong performance in multiple metrics. Its improvement in the ASA metric highlights the algorithm's ability to maintain similarity to the target, particularly notable at high superpixel counts. In contrast, while other algorithms perform well in certain scenarios, they generally fall short compared to SIT-HSS. In terms of the BR and UE metrics, SIT-HSS also excels, effectively capturing target boundaries and reducing under-segmentation errors, thereby enhancing the accuracy of the segmentation results. Regarding the EV metric, SIT-HSS maintains a high level of feature retention capability and exhibits excellent structural consistency, further affirming its advantages in superpixel segmentation tasks.

6.2 Hyperparameter Study. As t and τ vary, we examine the changes in the performance of SIT-HSS on SBD and PASCAL-S datasets. The results are presented in Fig. 9 and 10.

Weight normalization parameter t : The selected values of t are 0.05, 0.10, 0.15, and 0.20. Figs. 9a and 10a indicate that SIT-HSS performs best when $t = 0.1$. Smaller t values result in a decrease in the overall precision of the segmentation, while larger t values lead to less refined boundaries.

Structural entropy threshold τ : The selection of the value τ is based on the characteristics of the datasets and the model sensitivity analysis. For the SBD dataset, where the image complexity is relatively lower, larger thresholds help reduce unnecessary noise. Therefore, we set the values of τ at $1e - 6$, $2e - 6$, $3e - 6$, and $4e - 6$. As shown in Fig. 9b, when τ is set to $1e - 6$, SIT-HSS performs best on SBD. In contrast, as illustrated in Fig. 10b, when τ is set to $2e - 7$, SIT-HSS achieves optimal performance on PASCAL-S. Smaller τ values result in lower boundary adhesion, while larger τ values cause the model to focus too much on the boundaries, resulting in a slight decrease in the consistency of the region and the overall quality of segmentation.

6.3 Convergence Analysis. The proposed algorithm is capable of rapidly converging in a few iterations, achieving accurate superpixel segmentation. To validate this, we randomly select an image from each dataset and record the changes in the number of superpixels and the number of edges between them during each iteration. As shown in Fig. 11, as iterations progress, both the number of superpixels and the num-

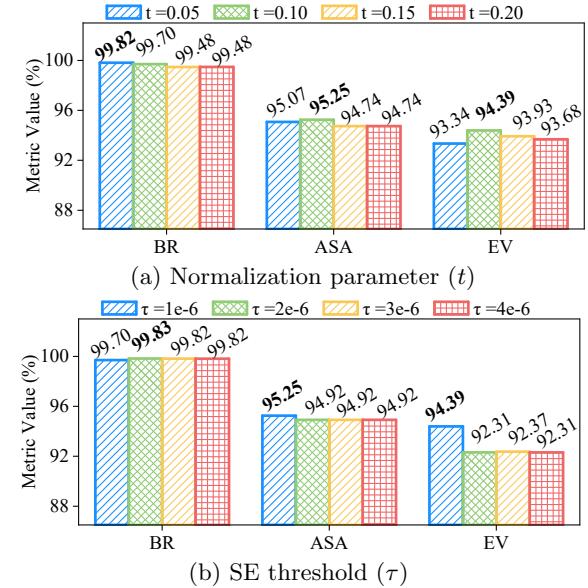


Figure 9: Performance of the SIT-HSS on SBD with different hyperparameter settings.

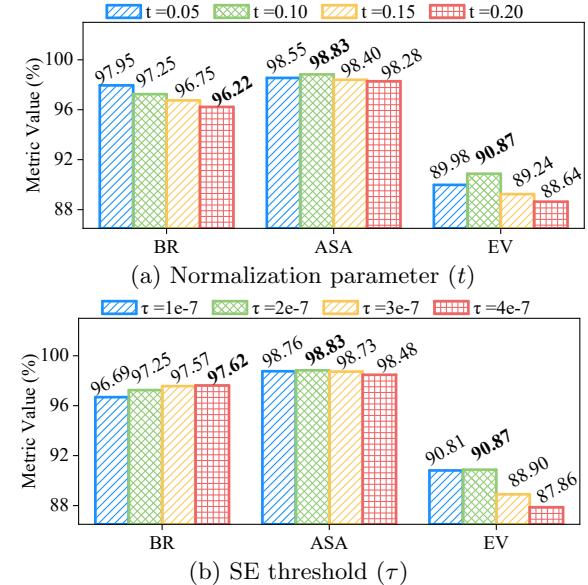


Figure 10: Performance of the SIT-HSS on PASCAL-S with different hyperparameter settings.

ber of edges exhibit an exponential decline. On average, by the seventh iteration, the number of superpixels approaches the target segmentation size, and by the 10th to 12th iteration, it precisely reaches the target size of 600. Note that the algorithm's time complexity is linearly related to the number of edges between superpixels. As the number of edges decreases, the computation time required for subsequent iterations also decreases accordingly.

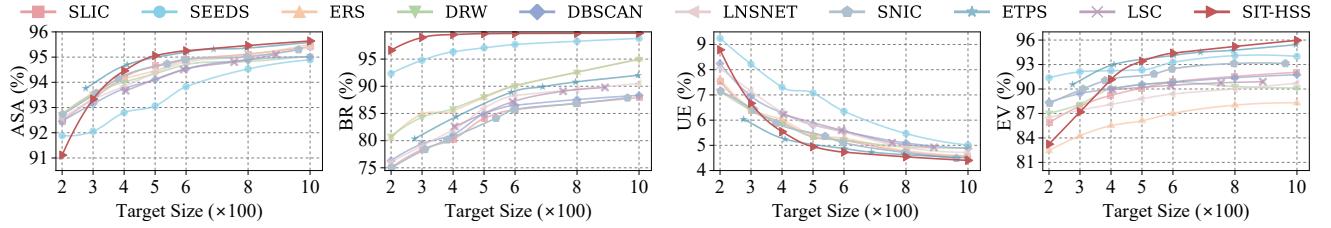


Figure 7: ASA, BR, UE, and EV Curve of our SIT-HSS compared with other state-of-the-art superpixel segmentation algorithms on the SBD dataset.

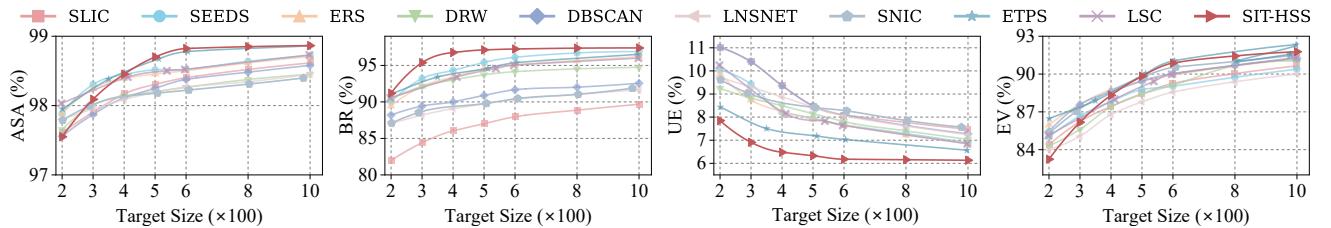


Figure 8: ASA, BR, UE, and EV Curve of our SIT-HSS compared with other state-of-the-art superpixel segmentation algorithms on the PASCAL-S dataset.

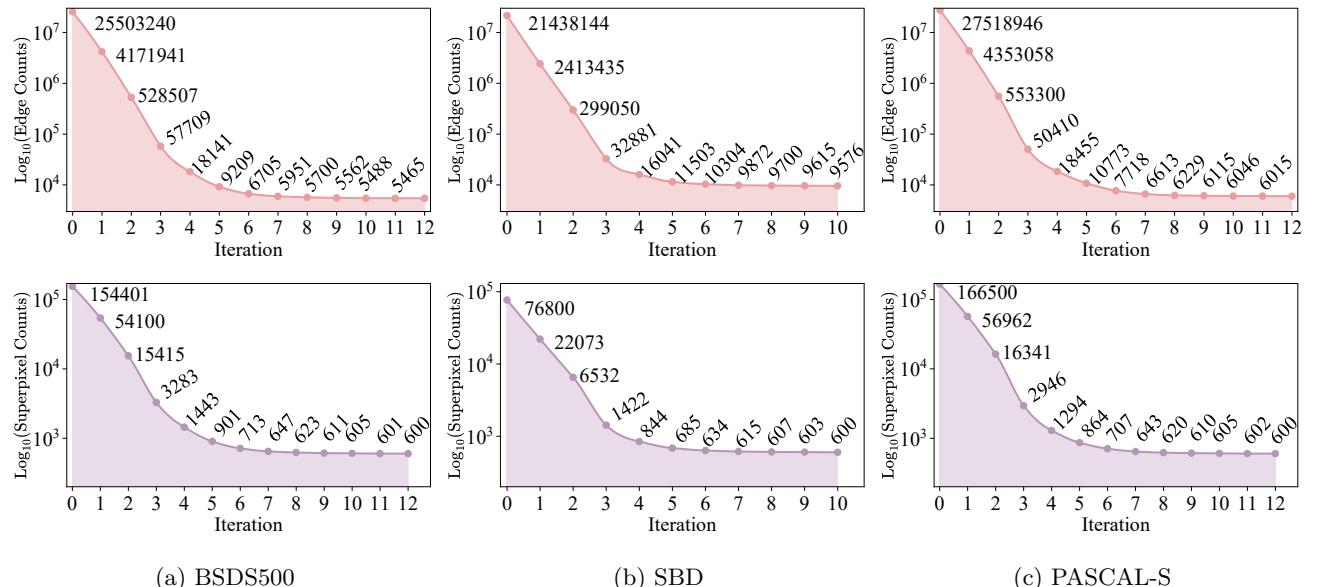


Figure 11: The changes in the edges numbers and superpixel numbers between them at each iteration from images in BSDS500, SBD, and PASCAL-S datasets.