HHS Public Access

Author manuscript

IEEE Trans Image Process. Author manuscript; available in PMC 2021 June 11.

Superpixel Embedding Network

Utkarsh Gaur [Member, IEEE], **B. S. Manjunath [Fellow, IEEE]**
UCSB

Abstract

Superpixel segmentation is a fundamental computer vision technique that finds application in a multitude of high level computer vision tasks. Most state-of-the-art superpixel segmentation methods are unsupervised in nature and thus cannot fully utilize frequently occurring texture patterns or incorporate multi-scale context. In this paper, we show that superpixel segmentation can be improved by leveraging the superior modeling power of deep convolutional autoencoders in a fully unsupervised manner. We pose the superpixel segmentation problem as one of manifold learning where pixels that belong to similar texture patterns are assigned near identical embedding vectors. The proposed deep network is able to learn image-wide and dataset-wide feature patterns and the relationships between them. This knowledge is used to segment and group pixels in a way that is consistent with a more global definition of pattern coherence. Experiments demonstrate that the superpixels obtained from the embeddings learned by the proposed method outperform the state-of-the-art superpixel segmentation methods for boundary precision and recall values. Additionally, we find that semantic edges obtained from the superpixel embeddings to be significantly better than the contemporary unsupervised approaches.

Keywords

Superpixel; Embedding; Convolutional Neural Networks

I. Introduction

SUPERPIXEL segmentation is an important unsupervised pre-processing technique that reduces an input image from millions of pixels to a few thousand clusters of photometrically similar pixels. Superpixels improve the efficiency of higher level computer vision tasks and have found usage in numerous applications including object localization [1], multi-class segmentation [2], [3], classification and visual search [4], optical flow [5], body model estimation [6], object tracking [7], and depth estimation [8].

Superpixel algorithms ordinarily group pixels by optimizing the trade-off between the photometric similarities and the spatial distances between the pixels. Since for natural scenes the local regions of an object demonstrate photometric correlation, this optimization often results in semantically and perceptually meaningful clusters without having to explicitly specify a threshold value.

Traditional superpixel algorithms rely solely on local image context, which works out if the foreground and the background are separated by simple texture patterns. However, complex texture patterns pose challenges to such methods. In particular, these methods often fail to recognize frequently co-occurring similar/dissimilar texture patterns across the image or the dataset. Semantically and perceptually important nonlocal cues are often critical to correctly segment complex texture patterns. Directly accounting for global pixel or texture relationships for all the images is NP-hard due to the exponential growth of connectivity parameters with a linear increase in features. Nevertheless, learning and accounting for global context is critical for a better superpixel segmentation and considerably improves performance, as evidenced by our experiments.

Deep learning methods such as convolutional neural networks (CNNs) have achieved very high performance for a number of supervised computer vision tasks. Their success is attributed in-part to the availability of vast quantities of data, fast GPUs with large memories and discovery of regularizing techniques and activation functions that counter vanishing gradient issues. CNNs possess powerful representational ability due to their high capacity and high VC dimension, $O(\rho^2)$ for ρ weights [9]. Deep CNNs have found use in unsupervised domains as autoencoders, where research has noted that deep features from the *bottleneck* layer automatically separate instances of different classes on the hyperplane [10]. Even so, it is not straightforward how superpixel segmentation algorithms can benefit from the powerful representational ability of CNNs.

As the research towards understanding the effectiveness of CNNs has progressed, research works [11], [12] have noted that CNN classifiers tend to implicitly learn object and object part models at various abstraction levels in their convolutional layers. Recent research works [13], [14], [15] have leveraged this information to compute dense object similarities and cosegmentation with only image-level label availability.

This paper describes a method to leverage this implicit modeling ability of CNNs to aid in computing semantically and perceptually coherent superpixels that can learn complex, object-specific texture patterns from across the dataset. These superpixels adhere to object boundaries better and result in more precise object segmentation. In addition to photometric similarity and spatial distances, we utilize deep feature space cues learned by the network to ultimately improve superpixel segmentation performance in a completely unsupervised manner [1]. Visual examples of superpixels obtained based on the proposed method are shown in figure 2.

We pose the superpixel segmentation problem as one of learning dense embeddings for each image such that the embedding vector $e_i \in \mathbb{R}^D$ for a pixel $p_i \in \mathbb{R}^3$ should be closer to another pixel on the manifold if they belong to a semantically similar region, and farther otherwise. We postulate that across a dataset, semantically similar patterns frequently co-occur in the same neighborhood and thus can be learned. Similarly, we can learn the concept of dissimilarity between image patterns, even if some of these patterns are similar in the photometric space.

[1]Code is planned for release after publication

Since this work aims to learn image patterns in a general setting without the assumption of class labels, we utilize geometric proximity of texture patterns as a proxy to similarity. Texture patterns in natural scenes can be comprised of pixel neighborhoods constituted by very different color values. However across multiple instances of a given texture pattern, the same pixel neighborhoods show up very often. The notion of utilizing geometric proximity for texture similarity is not new and has been used in recent works to compute dense object correspondences, albeit in a weakly supervised setting [16], [14].

In order to learn these patterns, we first compute superpixel segmentations for all the images of the dataset at multiple scale and compactness values with a standard algorithm [17] in a black-box fashion. Superpixel segmentation (clustering) of an image provides us with a label map where pixels belonging to the same superpixel are assigned the same integer label. Note that the numeric values of the labels in the label maps are arbitrary for each image and do not play any role in our learning process. The state-of-the-art superpixel algorithms cluster pixels based on a combination of pixel location and color based features to produce superpixels that nicely preserve object boundaries [17], [18], [19], [20]. For these methods, the clustering is performed in local image neighborhoods and no additional learning step is performed. Pixels belonging to the same texture patterns get clustered into the same superpixel consistently across the images of the dataset. We propose to use these superpixel label maps as a pseudo ground-truth label to learn texture similarity/dissimilarity cues. Specifically, we learn a deep convolutional autoencoder to learn pixel embeddings such that pixels that consistently fall into the same superpixel across the dataset are embedded closer on the manifold, whereas pixels that consistently fall into different superpixels are embedded far from each other on the manifold.

Learning pixel embeddings based on comparing local neighborhood texture patterns often leads to locally smooth embeddings. For robust texture similarity determination, it is important to induce contextual information from a large area in the image. To achieve this, we propose to cluster embeddings from across the same image that are very similar to each other. This is accomplished by introducing a recurrent mean-shift clustering layer which is fully differentiable and thus can used in the end-to-end training of the network. Once the network is trained end-to-end, the distances between the embedded pixels on the manifold are used to formally compute the refined superpixel segmentation via a standard superpixel algorithm. These refined superpixels reflect the effectiveness of the learned CNN manifold in the form of improved boundary F_1-score and achievable segmentation accuracy.

A. Contributions

This paper describes a novel technique to improve superpixel segmentation performance by learning texture pattern similarity/dissimilarity cues in a completely unsupervised fashion. We propose to embed pixels in a manifold where pixels belonging to the same texture pattern fall close and vice-versa. These embeddings are learned via a novel variable-margin loss function which adjusts the penalty of being dissimilar on the manifold based on the average dissimilarity in the photometric space. We propose a recurrent clustering layer to additionally induce image-wide feature context. The only hyperparameter of the clustering

layer, kernel bandwidth, is learned automatically end-to-end. The superpixels obtained via our method are shown to significantly improve boundary precision and recall.

B. Paper Organization

The organization of this paper is as follows: Section II describes the research literature that deals with computing superpixels, previous works that model global context, and works that have used CNNs in conjunction with superpixel segmentation. In section III, the process of modeling semantic pixel similarities via manifold learning is described and the variable-margin penalty term that embeds a pixel pair relative to their superpixel feature attribute similarities is detailed. Section IV explains the need to cluster pixel embeddings to induce image-wide feature context and describes a differentiable, in-network mean shift module to achieve it in an end-to-end fashion. Section V describes the architecture of the encoder-decoder fully convolutional CNN that combines all the aforementioned components. Finally, the quantitative and qualitative experiments are detailed in section VI and the paper is summarized in section VII.

II. Related Work

A. Graph-theoretic and Energy Minimization Approaches

Over the years, many approaches have been proposed to partition an image into perceptually meaningful atomic regions termed superpixels. Many traditional superpixel algorithms take a graph theoretic approach to the segmentation problem. Here, the pixels of a given image constitute the nodes of a graph whereas the edges denote the affinity between neighboring pixels. A superpixel segmentation is then obtained by partitioning this graph to minimize some error objective. A normalized cuts based technique presented in [21] computes superpixels by recursively computing normalized cut on the image at various scales. A bottom-up minimum spanning tree based approach is presented in [22] where pixels are progressively merged based on total variation of affinity between neighboring pixels. A recent graph-based approach, entropy rate superpixels (ERS) [20], computes superpixels via graph partition. Here the optimization objective is to maximize the entropy rate of a random walk on the image graph while imposing superpixels to have similar sizes via a regularization (balancing) term.

The work in [23] formulates the superpixel segmentation problem as one of energy minimization, and optimizes it via graph cuts to encourage superpixel regularity. Another energy maximization based method termed SEEDS [24] uses fast hill-climbing optimization to obtain superpixel segmentation. The optimization directly manipulates boundaries of an initial regular grid of superpixels to encourage smoother regions with homogeneous color histograms. A revised implementation of this algorithm called reSEEDS [25] improves on the performance by introducing a compactness term to the optimization criterion. The compactness term leads to higher connectivity among superpixels and achieves lower run time.

B. Clustering based Superpixels

Clustering local photometric features is perhaps the most popular technique to computer superpixel segmentation. One of the most prominent superpixel algorithms called SLIC [18] performs K-means clustering on CIELAB color and pixel locations jointly on a regular grid to obtain simple yet effective superpixels. Similar to SLIC, the work in LSC [19] method applies a kernel mapping to the the CIELAB color and pixel locations to project them into a new 10-dimensional manifold. They further show that applying a K-means clustering to the projected features in this new manifold is equivalent to a normalized cut in the original space, without the added computational cost. In the same vein, the work in Manifold SLIC [26] projects the features to a 2-dimensional feature space, which is sensitive to the content-richness of an image location. Clustering this feature space leads to smaller superpixels in content-dense regions and larger superpixels in content-sparse regions. The success of these methods suggests that improving the feature representation of the pixels to include richer context leads to improved superpixels.

The work in [27] is an improvement over SLIC where the superpixels are constrained to adhere to prominent contours determined by local gradient information. However this constraint increases the computational cost of the algorithm significantly compared to SLIC. Another recent approach that constraints superpixels boundaries by object contours via contour pattern matching is proposed in [28]. The idea is additionally extended to videos using optical flow to yield temporally consistent supervoxels. The work in [29] proposes a hybrid approach that combines local and global analysis to simplify image over-segmentations. It is shown that image regions can be merged using spectral mean only to produce meaningful and robust segmentations. A recent notable work for superpixel segmentation is SNIC [17], which is a non-iterative improvement on the SLIC algorithm. Unlike SLIC, SNIC explicitly enforces pixel connectivity from the beginning and achieves state-of-the-art results while accessing pixels only once.

C. Superpixels with CNNs

A number of research works have proposed using superpixel cues to improve segmentation [30], [31], [32]. Despite several works using superpixels, no other work has used CNNs to compute superpixels in an unsupervised manner to the best of authors' knowledge. Recently the work in [31] proposed an excellent approach to compute pixel affinity maps such that a superpixel segmentation should be consistent with some ground-truth segmentation. The pixel affinity maps could then be fine-tuned for a variety of tasks including superpixel segmentation, semantic segmentation and edge detection. Our approach is similar to this approach, however, unlike [31] we compute dense embeddings in an end-to-end fashion via a CNN and more importantly, our method does not require dense semantic segmentation masks for training. A recent work in [32] proposed computing feature embeddings which are grouped using hierarchical feature selection (HFS) algorithm [33] to produce effective image segmentations. The system proposed in this work outputs superpixels which can be directly plugged to any vision related task, in addition to image segmentation. We also introduce a differentiable recurrent mean-shift clustering layer into the superpixel network model which improves the performance of the system via its ability to associate embeddings from across the image.

III. Variable-Margin Superpixel Embedding Loss

Superpixels can be formally defined as a local set of p pixels, $l_i = \{p_1, \ldots, p_k\}$ that are disjoint $l_i \cap l_j = \varnothing$, $\forall i, j$ and decompose the image, $\cup_i l_i = \mathcal{I}$. Superpixel algorithms aim to compute clusters of pixels that closely adhere to the boundaries of the objects present in the scene. Operating on superpixels in lieu of raw pixels can provide significant performance improvement albeit at the cost of pixel resolution.

Traditional superpixel segmentation algorithms cluster pixel color and location values in small image neighborhoods without any additional context or learning involved. This may lead to frequently co-occurring patterns getting incorrectly clustered across different images. Research works in the past have brought attention to this lack of global context [19], however, directly accounting for global pixel or texture relationships for all the images is NP-hard due to the exponential growth of connectivity parameters with a linear increase in features. Accommodating global context through complex and computationally expensive methods is unjustified since it defeats the need for superpixels in the first place.

In this paper we introduce image and dataset wide context for the superpixel segmentation by training a CNN to learn which patterns are similar, and more importantly, which patterns aren't. This problem is posed as one of CNN-based manifold learning where embeddings for pixels that should belong to the same pattern (superpixel) are projected closer to each other and farther away from unrelated pixels. Since for natural scenes semantically related patterns frequently appear in the same neighborhood, an initial set of superpixel segmentations, however imperfect, can furnish the CNN with a good sense of semantic similarities. We use these superpixel segmentation maps as a pseudo ground-truth label to learn a new manifold. Feature distances on this manifold act as a proxy for semantic similarity by combining pixels' photometric similarities as well as texture pattern similarity, as learned from the dataset.

Formally, let $e_i, e_j \in \mathbb{R}^D$ be the embedding vectors for any two pixels i, j randomly sampled from superpixels l_i, l_j respectively. Now, if the two pixels were sampled from the same superpixel i.e. $l_i = l_j$, we want their corresponding embeddings e_i, e_j to be close to each other on the manifold. Specifically, the distance between e_i and e_j should be *at most* α on the manifold. To prevent the optimization from learning the same embedding for all pixels, we can similarly define a non-matching pixel penalty term. Specifically, if the two pixels were sampled from different superpixels i.e. $l_i \neq l_j$, then their embeddings e_i, e_j should be *at least* β distance apart on the manifold. If the distances between the two embeddings is given by $\Psi(e_i, e_j)$, then the optimal embedding for that pixel pair should minimize the loss given by:

$$l_{i,j} = \begin{cases} [\Psi(e_i, e_j) - \alpha]_+ & \text{if } l_i = l_j \\ [\beta - \Psi(e_i, e_j)]_+ & \text{otherwise} \end{cases} \quad (1)$$

This double margin embedding loss for matching and non-matching pixel pairs is visualized in figure 4. In practice, we L^2 normalize the embeddings $e \in \mathbb{R}^D$ to project them on a unit

hypersphere. The distance function $\Psi(\cdot, \cdot)$ between two pixels i and j can then defined as the cosine distance between their corresponding L^2 normalized embedding vectors:

$$\Psi(e_i, e_j) = \frac{1}{2} - \frac{\sum_{k=0}^{D} e_{k,i} e_{k,j}}{2\sqrt{\sum_{k=0}^{D} e_{k,i}^2}\sqrt{\sum_{k=0}^{D} e_{k,j}^2}} \tag{2}$$

Even though the Euclidean distance is a popular choice for embedding and metric learning research works [34], [35] have shown that cosine distance has some clear advantages when it comes to comparing deep feature maps. The works of [36], [37] recommend normalizing features to unit L^2 norm in order to stabilize the gradients for training. As opposed to Euclidean distance, which is unbounded, the cosine distance Ψ can be scaled to lie in a fixed interval. This allows for the values of the margin parameters α and β to be interpretable and thus leads to an easier analysis. Additionally, since any distance metric based on cosine similarity is invariant to the magnitude of the embedding vector e, the loss is easily decoupled from network hyperparameters such as weight decay or regularization, that can limit the range of Euclidean distances [38].

Variable Margin:

Since superpixels have an upper bound on their size by design, the superpixel segmentation algorithms induce arbitrary boundaries which often cut across texture patterns belonging to the same category. This design *limitation* is also reflected in the superpixel labeling l where two or more neighboring superpixels can have exact same appearance statistics. This design choice of the superpixel segmentation can drive our network to incorrectly segregate very similar patterns, thus confusing the network and leading to imperfect embeddings. To counter this, we propose the use of a variable penalty margin β in the embedding loss that adjusts the penalty of being dissimilar on the manifold based on the average dissimilarity in the photometric space. More specifically, for any pixel pair obtained from superpixels l_i and l_j, the penalty margin is proportional to the distance between the average $L^*a^*b^*$ color between l_i and l_j. Let C be the average $L^*a^*b^*$ color of a superpixel l, then penalty margin for pixel indices i, j is:

$$\beta_{i,j} \propto \|C_i - C_j\|_2 \tag{3}$$

The variable margin aspect of the penalty term is visually shown in figure 4. The reason that the variable margin is only introduced for the penalty term β is twofold. First, the superpixel size upper-limit constraint primarily leads to similar image regions getting split into multiple superpixels. Secondly, pixels that belong to the same superpixel share the same photometric statistics and thus cannot contribute to the variable margin.

For the final evaluation, the embedding loss is accumulated for pixel pairs across the image. Since the number of potential pairwise comparisons grow quadratically with a linear increase in the number of pixel, we limit the comparisons to pixel pairs in a neighborhood around each pixel, similar to [34]. The total loss function can then be defined as:

$$\mathscr{L}_{total} = \sum_{i \in N} \sum_{j \in \mathscr{N}_i} \frac{\mathbb{1}_{l_i = l_j} [\Psi(e_i, e_j) - \alpha]_+}{N_{l_i = l_j}} + \frac{\mathbb{1}_{l_i \neq l_j} [\beta_{i,j} - \Psi(e_i, e_j)]_+}{N_{l_i \neq l_j}} \quad (4)$$

Here $\mathscr{N}_i$ is the spatial neighborhood of pixel i. For each pixel, we perform 9 pairwise comparisons based on three 3x3 neighborhoods of *atrous* factors 1, 2, and 3 [39].

IV. Learnable Recurrent Clustering Module

The method described thus far aims to learn pixel embeddings that are cognate with superpixels via the cosine distances between them. However, the loss objective that is directly responsible for learning the embeddings only accounts for neighbors in the immediate vicinity of the pixel in question due to the complexity of the problem and the *NP*-hard nature of accommodating global context. This local nature of the embedding loss also means that the image-wide 'global' context may get marginalized, thereby resulting in deficient embeddings.

To account for global context and avoid embeddings getting pigeonholed into optimizing local distances, we propose the use of a differentiable mean-shift clustering recurrent module as a part of the network. The aim of this module is to cluster features from across the feature map of a given image to produce cluster centers that represent better estimates of the global density. Subsequent iterations ensure local embeddings stick to the global clusters more closely. Kernel bandwidth, the hyperparameter of the module which influences the number of modes the algorithm will converge to, is learned automatically via back-propagation.

A. Mean-Shift Kernel Density Estimation

The mean-shift algorithm is a non-parametric density estimation algorithm that operates by placing a kernel on each data sample and shifting this kernel to a higher density region until convergence. Convergence of the mean-shift algorithm implies that a shift in direction cannot accommodate more points inside the kernel. Mean-shift does not require the number of cluster centers (modes) to be known *a priori*. The number of modes that it converges to is directly influenced by the kernel bandwidth parameter. For instance, very small kernel bandwidth would result in each pixel embedding getting assigned as its own cluster. Conversely, large kernel bandwidth would lead to all the pixel embeddings converging towards a single, mean cluster. The kernel bandwidth parameter defines the spatial scope/scale of a given kernel and certainly one fixed parameter is not suitable for the entire dataset.

Mean-shift in its matrix form was first introduced by [40] and the work in [38] demonstrated the benefit of collapsing similar embedding vectors for the instance segmentation task. In this work, we borrow the idea for learning superpixel embeddings from [38] and additionally, learn the bandwidth parameter in-network to automatically learn meaningful clusters.

Formally, let $X \in \mathbb{R}^{D \times N}$ be N pixels embedded into a manifold of dimensionality D. Let $Q \in \mathbb{R}^{N \times N}$ be the degree matrix that captures the connectivity information of a pixel embedding by accumulating similarities of that pixel embedding to the rest of the embeddings. Based on the kernel matrix K, the degree matrix can be computed as $Q = diag(K^T \mathbf{1})$. A mean shift vector M can be computed for the kernel-weighted embeddings X by first smoothing X with the local means $X \leftarrow XKQ^{-1}$ and then subtracting the smoothed version with the original values $\mathbf{M} = \mathbf{XKQ}^{-1} - \mathbf{X}$ [38][40].

For each iteration of the mean shift recurrent module with a step-size η, the embeddings are shifted towards their kernel-weighted mean:

$$\mathbf{X} \leftarrow \mathbf{X} + \eta\mathbf{M} \tag{5}$$

For $\eta = 1$, the above average rule is termed Gaussian Blurring Mean Shift (GBMS) [40] and can be simplified to:

$$\mathbf{X} \leftarrow \mathbf{XKQ}^{-1} \tag{6}$$

Unlike the common practice of keeping the kernel matrix K constant across the iterations, in an GBMS update rule, the kernel matrix is recomputed after each iteration. The work in [40] showed this update rule to have cubic convergence, hence we use it in our network.

Traditionally for mean-shift kernel density estimation, a Gaussian kernel with bandwidth σ centered at each sample is assumed and the density can be approximated by computing the average distance of a point with all the other points via the Gaussian kernel. However, the embeddings learned in this work are L^2 normalized and compared via cosine similarities, hence we use the von Mises-Fisher kernel, which is defined for the unit hypersphere as $K = C_\kappa exp\left(\kappa X^T X\right) \in \mathbb{R}^{N \times N}$ with normalizing factor C_κ and kernel bandwidth κ, which represents the local context on the unit hypersphere to compare a data sample to.

To avoid a very deep computational graph that may result in gradient vanishing issues, we accumulate the superpixel embedding loss for the original (unclustered) superpixel embeddings, as well as the superpixel embeddings over all iterations of the mean-shift recurrent clustering module. In this manner, the gradients are backpropagated through each of the unrolled loops and further propagated to the deep autoencoder network. This combination of the U-net and clustering module is end-to-end trainable.

In figure 6, we demonstrate the effect of mean-shift clustering on the embeddings obtained from the network. The embeddings learned through the loss function in equation 4 encode local-neighborhood similarities. This tends to make the embeddings piecewise smooth, as seen in figure 6 (top). By collapsing very similar embeddings across the image into distinct modes, we can induce context knowledge from a much larger area. Figure 1 shows more examples of embeddings produced by the network reduced to 3 dimensions for visualization. Figure 6 (bottom) visualizes the L^2 normalized, $3D$ embeddings on the sphere, before and after the discretization from the mean-shift module.

V. Network Architecture

The success of CNNs for classification tasks has lead to a number of approaches to image segmentation by the means of dense pixel classification. The first CNN based segmentation approaches classified a pixel based on a square window around it. This patch-based CNN would operate on every pixel of an input image resulting in redundant feature computations and poor efficiency. Long et al proposed using Fully Convolutional Network (FCN) [13] which directly produces a segmentation mask as output with a single forward pass of an input image. At their core, FCN retain the feature extractor part of the standard CNNs (encoder) and replace the fully connected layers by convolutional layers that output feature maps. These feature maps are upsampled via learnable deconvolutional layers (decoder) or other similar mechanism to produce dense pixel-wise classification scores. The end-to-end processing achieved by FCNs not only significantly reduced the training and inference time for segmentation networks, but also set state-of-the-art results for multiple segmentation datasets and benchmarks.

Multiple improvements have since been proposed that extend the FCN architecture, the most significant being the U-Net [41] architecture. This architecture uses shortcut or ‘skip’ connections between the encoder and the decoder part of the CNN to combine low-level feature maps with higher-level ones, which allows it to be pixel-precise. A large number of feature channels in upsampling part allows propagating context information to higher resolution layers. CNNs with some form of ‘skip’ architecture have set the state-of-the-art results in image segmentation across multiple tasks and datasets [42], [39].

We find that convolutional network architectures which employ ‘skip’ connections that connect the encoder and the decoder components work especially well to learn superpixel embeddings. In addition to learning image patterns that are common across the dataset via the long route, the numerous shortcut ‘skip’ connections help the superpixel embeddings by memorizing and propagating very local pixel/texture patterns. These local patterns are of high importance in good superpixel computation since superpixels should be sensitive to local photometric patterns.

This work uses the encoder-decoder architecture similar to U-net [41] with skip connections for all the experiments. The block diagram of the architecture is shown in figure 5. The encoder part of our network follows the typical architecture of a deep CNN where each layer comprises of a convolutional operation, batch normalization, and non-linear activation function. As we progress through the layers, we downsample the feature maps by half via striding while doubling the number of channels via increase in the number of filters.

The decoder section of the network consists of convolution layers followed by upsampling layers. In a fashion opposite of the encoder, the spatial resolution of the feature maps is doubled at each step while the number of channels is reduced by half. We use ‘nearest’ upsampling layer to double the spatial resolution of the feature maps. These upsampled feature maps are concatenated with the low-level feature maps from the corresponding encoder feature maps of the network via the shortcut connections. The output of the network is a feature map with the same spatial resolution as the input image. The U-net architecture

with skip connections has proven to be very useful for problems where the amount of dense annotations is limited [43]. The feature maps are sampled for 5 times each in the encoder and decoder sections of our network so we ensure that input image dimensions are divisible by 32 (2^5).

VI. Experiments

We use images from the Pascal VOC 2007 [44] and the Berkeley Segmentation Dataset (BSDS) 300 [45] to compute the pseudo ground-truth, no other data (ground-truth or otherwise) is utilized. For the convolution layers, reflection padding is used instead of zero padding. We also find that training the network with only 50% of pixels of the input image achieves the same results at double efficiency. We randomly sample the pixels to be processed per image per minibatch. The minibatch is perturbed with an additive normal noise which we found to output better embeddings. Data augmentation is performed by horizontally flipping the image and label at random.

We use the fast SNIC method described in [17] to compute superpixels from the embeddings output by the network. SNIC avoids the iterative clustering step typical in superpixel segmentation algorithms by starting from certain local centroids and growing the superpixel boundary outwards. Connectivity is explicitly enforced by sequentially processing a priority queue composed the neighbors of the centroids, and their neighbors and so on. SNIC outperforms SLIC on multiple quantitative benchmarks. We use the same superpixel distance metric as SNIC, albeit modified to replace pixel similarity term with normalized embedding similarity term. For a pair of embedding vectors e_j and e_k:

$$d_{j,k} = \sqrt{\frac{\|p_j - p_k\|_2^2}{s} + \frac{\Psi(e_j, e_k)}{m'}}, \tag{7}$$

where $p_\phi = [x_\phi, y_\phi]^T$ is the spatial location of some embedding vector e_ϕ in the image space. The normalizing parameters s and m' are retained from SNIC and balance the trade off between the spatial distances and the embedding distances. The scale parameter s is set to be $\sqrt{N/K}$ where N is the total number of pixels in the image and K is the desired number of superpixels. The compactness parameter m' is a user-provided constant and influences the superpixel compactness. We note that the clustering distance metric in [17] equally weights the color and the location vectors. We follow the same scheme to keep the relative weighting of the location and appearance-based embeddings equal and account for the increased dimensionality of the embedding vectors.

The quantitative results are reported on the BSDS500 dataset images [46] for number of superpixels ranging from 50 to 500. The quantitative comparisons can be viewed in figures 7, 8 and 9. A visual comparison of our method with the other state-of-the-art methods is provided in Fig 12. We used the very useful benchmarking toolkit provided by [47] for all the experiments. We use the standard superpixel segmentation metrics undersegmentation error (CUSE), achievable segmentation accuracy (ASA), and boundary F_1-score to evaluate the performance of our method. CUSE and ASA evaluate the superpixel overlap with ground-truth regions, whereas F_1-score is a measure of superpixel contour adherence to

ground-truth contours and is correlated to the other two metrics. We briefly describe these metrics below.

A. *Boundary F_1-score*

In the context of boundaries, precision is the ratio of correctly detected boundaries to the total number of boundaries detected whereas recall is the ratio of correctly detected boundaries to the total number of contours present in the ground-truth segments. Even though boundary recall is one of the most reported and compared metrics for segmentation performance evaluation, it is not well-suited on its own. Precision and recall represent a trade-off and should be reported together since it is possible to increase the value of one at the expense of other.

Similar to the state-of-the-art, we describe a boundary map $B_{\mathcal{S}}$ outputted by the network to be a match to the ground-truth boundary map $B_{\mathcal{G}}$ if the overlap is within a neighborhood of ϵ pixels. if The boundary recall can then be defined as:

$$BR(\mathcal{S},\mathcal{G}) = \frac{\sum_i \mathbb{1}[min_j \left\| B^i_{\mathcal{S}} - B^j_{\mathcal{G}} \right\| < \epsilon]}{\left| B_{\mathcal{G}} \right|} \tag{8}$$

and the boundary precision can be defined as:

$$BP(\mathcal{S},\mathcal{G}) = \frac{\sum_i \mathbb{1}[min_j \left\| B^i_{\mathcal{S}} - B^j_{\mathcal{G}} \right\| < \epsilon]}{\left| B_{\mathcal{S}} \right|} \tag{9}$$

For our experiments, we report the F_1-score (aka F-measure), which combines both boundary recall and precision with equal weights:

$$F_1 - score = 2\frac{BP \cdot BR}{BP + BR} \tag{10}$$

In our evaluations, we set the value of ϵ to 2, similar to the benchmark method [17] and other state-of-the-art methods.

B. Corrected Undersegmentation Error (CUSE)

The undersegmentation error quantifies the overlap error between a superpixel segment and the ground-truth segment that it has the maximum overlap with. The authors of [24] noted that the traditional undersegmentation error fully penalized superpixels on both sides of the boundary of an object for even a single pixel error along the boundary. They proposed the *corrected* version of undersegmentation error which we use here:

$$CUSE(\mathcal{S},\mathcal{G}) = \frac{\sum_i \left| S_i - argmax_{g \in G} \left| S_i \cap g \right| \right|}{\sum_j \left| G_j \right|} \tag{11}$$

Here S is the set of superpixel segments outputted by the proposed algorithm, G is the set of ground-truth segments and $|\cdot|$ denotes the size of the segment. The CUSE penalty for the overlap error between superpixel and ground-truth segment is proportional to the magnitude of the mistake. Additionally, the error is only accumulated a single time for one side of the superpixel.

C. Achievable Segmentation Accuracy (ASA)

Achievable Segmentation Accuracy describes an upper bound on the segmentation performance of a superpixel labeling. It labels superpixels according to their underlying ground truth segments and counts the correctly labeled pixels [20]:

$$ASA(\mathcal{S},\mathcal{G}) = \frac{\sum_i max_k |S_i \cap G_k|}{\sum_j |G_j|} \tag{12}$$

By labeling each superpixel with the label of the ground truth segment that has the largest overlap, ASA can be computed as the fraction of correctly labeled pixels.

Our method outperforms all the other methods for boundary F_1-score and performs as well as SNIC for ASA and CUSE. We note that the boundaries obtained from our method adhere better to the object and is more robust to object scale changes. We also note that our method significantly outperforms the other methods for smaller number of superpixels. The difference gets smaller with larger number of superpixels since the margin of error increases, however our lead still persists.

The time efficiency for the proposed method is compared to other state of the art methods with respect to boundary F_1-score in table I. The results are averaged for BSDS500 images of 481x321 resolution and number of superpixels segmentations desired equal to 50. All the parameters for the algorithms were set to their default values.

SEN network inference was performed on an Nvidia Quadro P5000 GPU and the time reported here is an aggregate of network inference for 16 dimensional embedding computation, clustering, and SNIC superpixel computation. The method demonstrates improved performance over other state of the art methods albeit at a fraction of time cost penalty. We note that the clustering module makes use of matrix inversion operations and backpropagation through it is currently not optimized. These operations are responsible for a non-trivial chunk of time cost.

Clustering module ablation: To test the efficacy of the clustering layer, an ablation experiment was performed, where the clustering layer was removed and the variable margin loss was applied directly to the output of the convolutional layer. SNIC was used again to compute superpixels from the resulting embeddings. The boundary F_1-score comparing the performance of the network with and without the clustering module for the BSDS500 dataset is reported in figure 10.

We observe that the lack of embedding clustering leads to a drop in the boundary F_1-score. Clustering layer allows merging similar embeddings to the same point on the manifold. This

allows the network to learn to associate patterns. We conjecture that the clustering layer helps the network to better differentiate the different 'semantic regions' of the object by well separating the patterns on the manifold and ultimately producing better superpixel segmentations.

Contour detection: In addition to superpixel segmentation, we experimentally evaluated the ability of our method to detect object contours. Contour detection is one of the most fundamental and widely studied problem in computer vision. Deep network-based approaches have managed to significantly improve the state-of-the-art for contour detection. A popular CNN-based approach called HED [48] can compute high quality edge maps for a given image by learning rich hierarchical representations guided by deep supervision. The work in [49] extends this idea by fusing hierarchical convolutional feature maps for crack detection. Most state-of-the-art contour detection approaches are supervised and require dense, pixel-level labeling. We utilized the proposed method to detect high quality object contours, without the need for ground-truth labels.

To obtain contours from superpixels, we extract superpixels for 13 different sizes ranging from 50 to 700 and average the superpixel boundaries. Edges that the network deems prominent consistently become superpixel boundaries across different superpixel sizes. Thus averaging the superpixel boundaries results in prominent edges maintaining strong signal whereas spurious edges get weaker signal due to being averaged out. The resulting average edge image is thresholded to remove low signal components before applying simple non-maximum suppression obtained from Piotr's structured edge detection toolbox [50]. The contours obtained via the proposed method significantly outperform other unsupervised contour detectors as shown in figure 11. We note that the PR curve for this work (SEN) is not monotonic due to the filtering of the average edge image to remove low signal components, without which the edge non-maximum suppression doesn't perform well and the obtained contours are worse.

VII. Summary

Superpixel segmentation is an important pre-processing step for numerous computer vision tasks. Most state-of-the-art superpixel segmentation methods are unsupervised in nature and hence do not learn complex, frequently occurring texture patterns or incorporate multi-scale context. This work proposed a deep convolutional neural network based *unsupervised* approach to not only account for global context but also learn to disambiguate between object/background patterns. This network learns dense pixel embeddings based on pixel similarities as inferred from superpixel segmentations at multiple scales and detail resolutions. The pixel embedding on the manifold are found to correspond to the different objects present in the scene. To learn these embeddings, a novel variable-margin contrastive loss is proposed that adapts the penalty to misclassify matching pixels in proportion to the photometric similarities of their parent superpixels. Finally, usage of an in-network recurrent clustering module is proposed to avoid embeddings getting pigeonholed into optimizing local distances and be more sensitive to image-wide context. Experiments demonstrate that the superpixels obtained from the embeddings learned by our method outperform the state-of-the-art superpixel segmentation methods for boundary precision and recall values.

Acknowledgements

This work was supported by award #HD059217 from the National Institutes of Health.

Biographies

Utkarsh Gaur received the M.S. degree in computer science from the University of California, Riverside, USA, and the Ph.D. degree in computer science from the University of California, Santa Barbara, USA. His research interests involve applying machine learning methods to computer vision tasks including object segmentation, recognition and biometrics. He is currently a computer vision and machine learning architect with Synaptics, San Jose, CA.

B. S. Manjunath (F'05) received the Ph.D. degree in electrical engineering from the University of Southern California, Los Angeles, CA, USA, in 1991. He is currently a Distinguished Professor of Electrical and Computer Engineering at the University of California at Santa Barbara, where he directs the Center for Multimodal Big Data Science and Healthcare. He has authored or coauthored about 300 peer-reviewed articles and served as an Associate Editor for the IEEE Transaction on Image Processing, the IEEE Transactions on Pattern Analysis and Machine Intelligence, the IEEE Transactions on Multimedia, the IEEE Transactions on Information Forensics, and the IEEE Signal Processing Letter. His current research interests include image processing, machine learning, computer vision, and media forensics.

References

[1]. Fulkerson B, Vedaldi A, and Soatto S, "Class segmentation and object localization with superpixel neighborhoods," in Computer Vision, 2009 IEEE 12th International Conference on IEEE, 2009, pp. 670–677.

[2]. Gould S, Rodgers J, Cohen D, Elidan G, and Koller D, "Multiclass segmentation with relative location prior," International Journal of Computer Vision, vol. 80, no. 3, pp. 300–316, 2008.

[3]. Pourian N, Karthikeyan S, and Manjunath BS, "Weakly supervised graph based semantic segmentation by learning communities of image-parts," in Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1359–1367.

[4]. Pourian N and Manjunath B, "Pixnet: A localized feature representation for classification and visual search," IEEE Transactions on Multimedia, vol. 17, no. 5, pp. 616–625, 2015.

[5]. Menze M and Geiger A, “Object scene flow for autonomous vehicles,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3061–3070.

[6]. Mori G, “Guiding model search using segmentation,” in Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, vol. 2 IEEE, 2005, pp. 1417–1423.

[7]. Wang S, Lu H, Yang F, and Yang M-H, “Superpixel tracking,” in Computer Vision (ICCV), 2011 IEEE International Conference on IEEE, 2011, pp. 1323–1330.

[8]. Zitnick CL and Kang SB, “Stereo for image-based rendering using image over-segmentation,” International Journal of Computer Vision, vol. 75, no. 1, pp. 49–65, 2007.

[9]. Sontag ED, “Vc dimension of neural networks,” NATO ASI Series F Computer and Systems Sciences, vol. 168, pp. 69–96, 1998.

[10]. Xie J, Girshick R, and Farhadi A, “Unsupervised deep embedding for clustering analysis,” in International conference on machine learning, 2016, pp. 478–487.

[11]. Zeiler MD and Fergus R, “Visualizing and understanding convolutional networks,” in Proc. European Conference on Computer Vision, 2014.

[12]. Mahendran A and Vedaldi A, “Understanding deep image representations by inverting them,” in Proc. Computer Vision and Pattern Recognition, 2015.

[13]. Long JL, Zhang N, and Darrell T, “Do convnets learn correspondence?” in Advances in Neural Information Processing Systems, 2014, pp. 1601–1609.

[14]. Taniai T, Sinha SN, and Sato Y, “Joint recovery of dense correspondence and cosegmentation in two images,” in Proc. Computer Vision and Pattern Recognition, 2016.

[15]. Choy CB, Gwak J, Savarese S, and Chandraker M, “Universal correspondence network,” in Advances in Neural Information Processing Systems, 2016.

[16]. Gaur U and Manjunath BS, “Weakly supervised manifold learning for dense semantic object correspondence,” in The IEEE International Conference on Computer Vision (ICCV), Oct 2017.

[17]. Achanta R and Süsstrunk S, “Superpixels and polygons using simple non-iterative clustering,” in Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on Ieee, 2017, pp. 4895–4904.

[18]. Achanta R, Shaji A, Smith K, Lucchi A, Fua P, and Süsstrunk S, “Slic superpixels compared to state-of-the-art superpixel methods,” IEEE transactions on pattern analysis and machine intelligence, vol. 34, no. 11, pp. 2274–2282, 2012. [PubMed: 22641706]

[19]. Li Z and Chen J, “Superpixel segmentation using linear spectral clustering,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1356–1363.

[20]. Liu M-Y, Tuzel O, Ramalingam S, and Chellappa R, “Entropy rate superpixel segmentation,” in Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on IEEE, 2011, pp. 2097–2104.

[21]. Ren X and Malik J, “Learning a classification model for segmentation,” in The IEEE International Conference on Computer Vision, 2003.

[22]. Felzenszwalb PF and Huttenlocher DP, “Efficient graph-based image segmentation,” International Journal of Computer Vision, 2004.

[23]. Veksler O, Boykov Y, and Mehrani P, “Superpixels and supervoxels in an energy optimization framework,” in European Conference on Computer Vision, 2010.

[24]. Van den Bergh M, Boix X, Roig G, de Capitani B, and Van Gool L, “Seeds: Superpixels extracted via energy-driven sampling,” in European conference on computer vision Springer, 2012, pp. 13–26.

[25]. Stutz D, Hermans A, and Leibe B, “Superpixels: An evaluation of the state-of-the-art,” Computer Vision and Image Understanding, vol. 166, pp. 1–27, 2018.

[26]. Liu Y-J, Yu C-C, Yu M-J, and He Y, “Manifold slic: A fast method to compute content-sensitive superpixels,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.

[27]. Zhang Y, Li X, Gao X, and Zhang C, “A simple algorithm of superpixel segmentation with boundary constraint,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 27, no. 7, pp. 1502–1514, 2016.

[28]. Lee S-H, Jang W-D, and Kim C-S, “Contour-constrained superpixels for image and video processing,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[29]. Hu Z, Li Q, Zhang Q, Zou Q, and Wu Z, “Unsupervised simplification of image hierarchies via evolution analysis in scale-sets framework,” IEEE Transactions on Image Processing, vol. 26, no. 5, pp. 2394–2407, 2017.

[30]. Gadde R, Jampani V, Kiefel M, Kappler D, and Gehler PV, “Superpixel convolutional networks using bilateral inceptions,” in European Conference on Computer Vision Springer, 2016, pp. 597–613.

[31]. Tu W-C, Liu M-Y, Jampani V, Sun D, Chien S-Y, Yang M-H, and Kautz J, “Learning superpixels with segmentation-aware affinity loss,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018.

[32]. Liu Y, Jiang P-T, Petrosyan V, Li S-J, Bian J, Zhang L, and Cheng M-M, “Del: Deep embedding learning for efficient image segmentation,” in International Joint Conference on Artificial Intelligence, 2018, pp. 864–870.

[33]. Cheng M-M, Liu Y, Hou Q, Bian J, Torr P, Hu S-M, and Tu Z, “Hfs: Hierarchical feature selection for efficient image segmentation,” in European Conference on Computer Vision Springer, 2016, pp. 867–882.

[34]. Adam IK. Harley W. Derpanis Konstantinos G.Segmentation-aware convolutional networks using local attention masks; IEEE International Conference on Computer Vision (ICCV); 2017.

[35]. Gould S, Zhao J, He X, and Zhang Y, “Superpixel graph label transfer with learned distance metric,” in European Conference on Computer Vision Springer, 2014, pp. 632–647.

[36]. Schroff F, Kalenichenko D, and Philbin J, “Facenet: A unified embedding for face recognition and clustering,” in Proc. Computer Vision and Pattern Recognition, 2015.

[37]. Oh Song H, Xiang Y, Jegelka S, and Savarese S, “Deep metric learning via lifted structured feature embedding,” in Proc. Computer Vision and Pattern Recognition, 2016.

[38]. Kong S and Fowlkes C, “Recurrent pixel embedding for instance grouping,” arXiv preprint arXiv:1712.08273, 2017.

[39]. Chen L-C, Papandreou G, Kokkinos I, Murphy K, and Yuille AL, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” IEEE transactions on pattern analysis and machine intelligence, vol. 40, no. 4, pp. 834–848, 2018. [PubMed: 28463186]

[40]. Carreira-Perpinán MA, “Generalised blurring mean-shift algorithms for nonparametric clustering,” in Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on IEEE, 2008, pp. 1–8.

[41]. Ronneberger O, Fischer P, and Brox T, “U-net: Convolutional networks for biomedical image segmentation,” in International Conference on Medical image computing and computer-assisted intervention Springer, 2015, pp. 234–241.

[42]. Iglovikov V, Rakhlin A, Kalinin A, and Shvets A, “Pediatric bone age assessment using deep convolutional neural networks,” arXiv preprint arXiv:1712.05053, 2017.

[43]. Iglovikov V, Mushinskiy S, and Osin V, “Satellite imagery feature detection using deep convolutional neural network: A kaggle competition,” arXiv preprint arXiv:1706.06169, 2017.

[44]. Everingham M, Eslami SMA, Van Gool L, Williams CKI, Winn J, and Zisserman A, “The pascal visual object classes challenge: A retrospective,” International Journal of Computer Vision, vol. 111, no. 1, pp. 98–136, 1 2015.

[45]. Martin D, Fowlkes C, Tal D, and Malik J, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics,” in The IEEE International Conference on Computer Vision, 2001.

[46]. Arbelaez P, Maire M, Fowlkes C, and Malik J, “Contour detection and hierarchical image segmentation,” IEEE Transactions on Pattern Analysis and Machine Intelligence, 2011.

[47]. Stutz D, Hermans A, and Leibe B, “Superpixels: an evaluation of the state-of-the-art,” Computer Vision and Image Understanding, vol. 166, pp. 1–27, 2018.

[48]. Xie S and Tu Z, “Holistically-nested edge detection,” International Journal of Computer Vision, vol. 125, pp. 3–18, 2017.

[49]. Zou Q, Zhang Z, Li Q, Qi X, Wang Q, and Wang S, “Deepcrack: Learning hierarchical convolutional features for crack detection,” IEEE Transactions on Image Processing, 2018.

[50]. Dollár P and Zitnick CL, “Structured forests for fast edge detection,” in The IEEE International Conference on Computer Vision, 2013.

Fig. 1:
Visualizing the embeddings produced by the network reduced to three dimensions via PCA.

Fig. 2:
Visual examples from the BSDS500 dataset. Each image is overlaid by three segments corresponding to 1000/500/200 superpixels using the proposed superpixel embeddings.

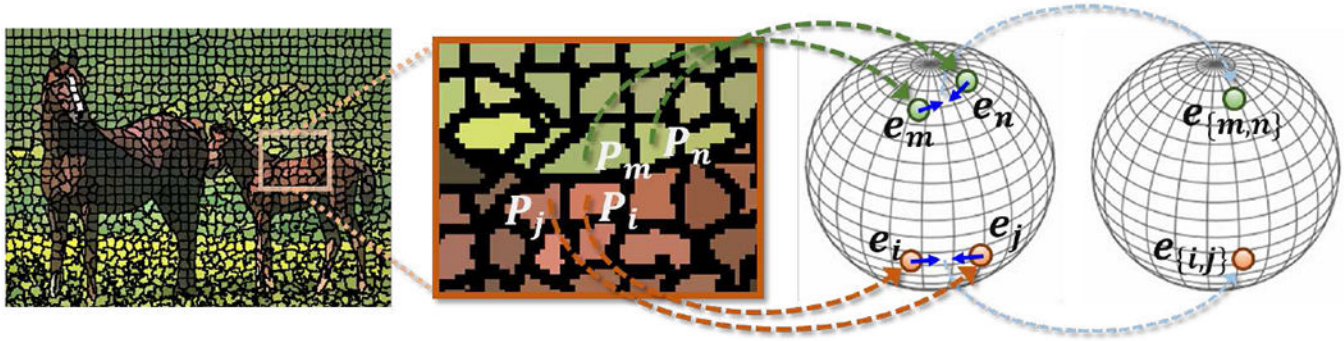

Fig. 3:

Due to the superpixel size constraint, pixels belonging to the same semantic region may get split up into different superpixels. Variable-margin penalty term ensures that embeddings for pixels that have near similar superpixel feature attributes can still be mapped closer on the manifold. Neighboring embeddings from across the image may get coalesced later by the clustering module (blue arrows).

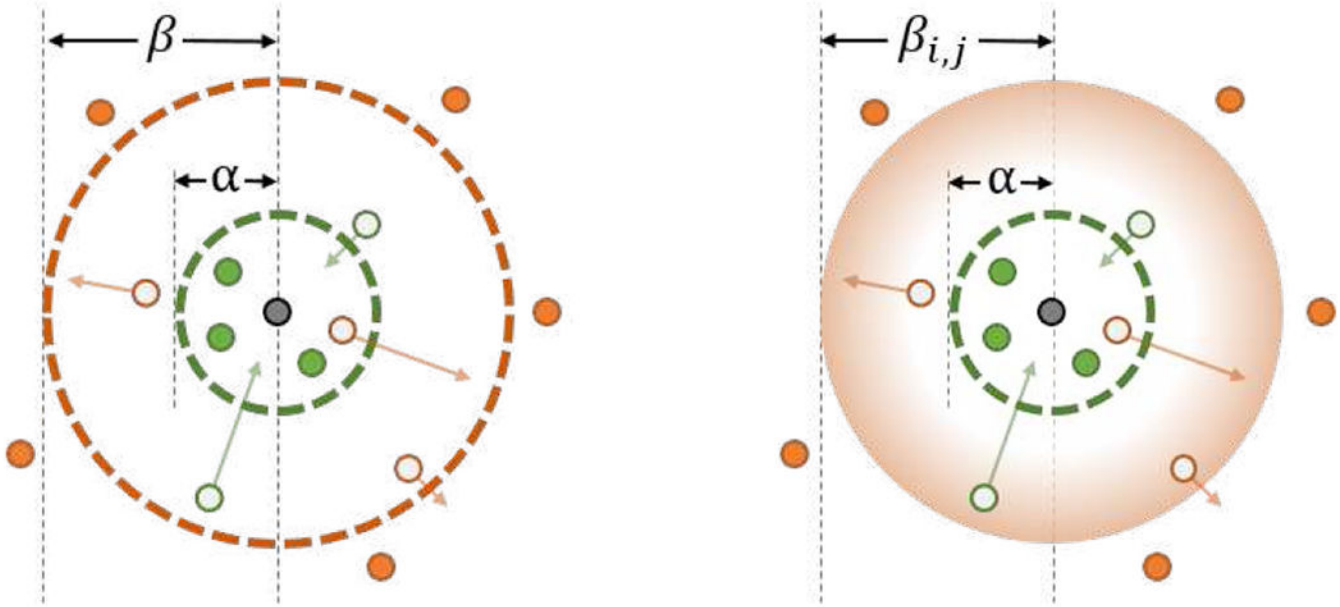

Fig. 4:

Left: Visualization of the double margin contrastive loss for a sample pixel embedding (black) with embeddings of matching pixels (green) and non-matching pixels (orange). A penalty is incurred if a matching pixel's embedding is farther than α or a non-matching pixel's embedding is closer than β. **Right:** Instead of a fixed penalty margin β, this work utilizes a variable margin $\beta_{i,j}$ (represented by the orange gradient) that is based on the photometric distance between the superpixels that cover pixel i and pixel j respectively.

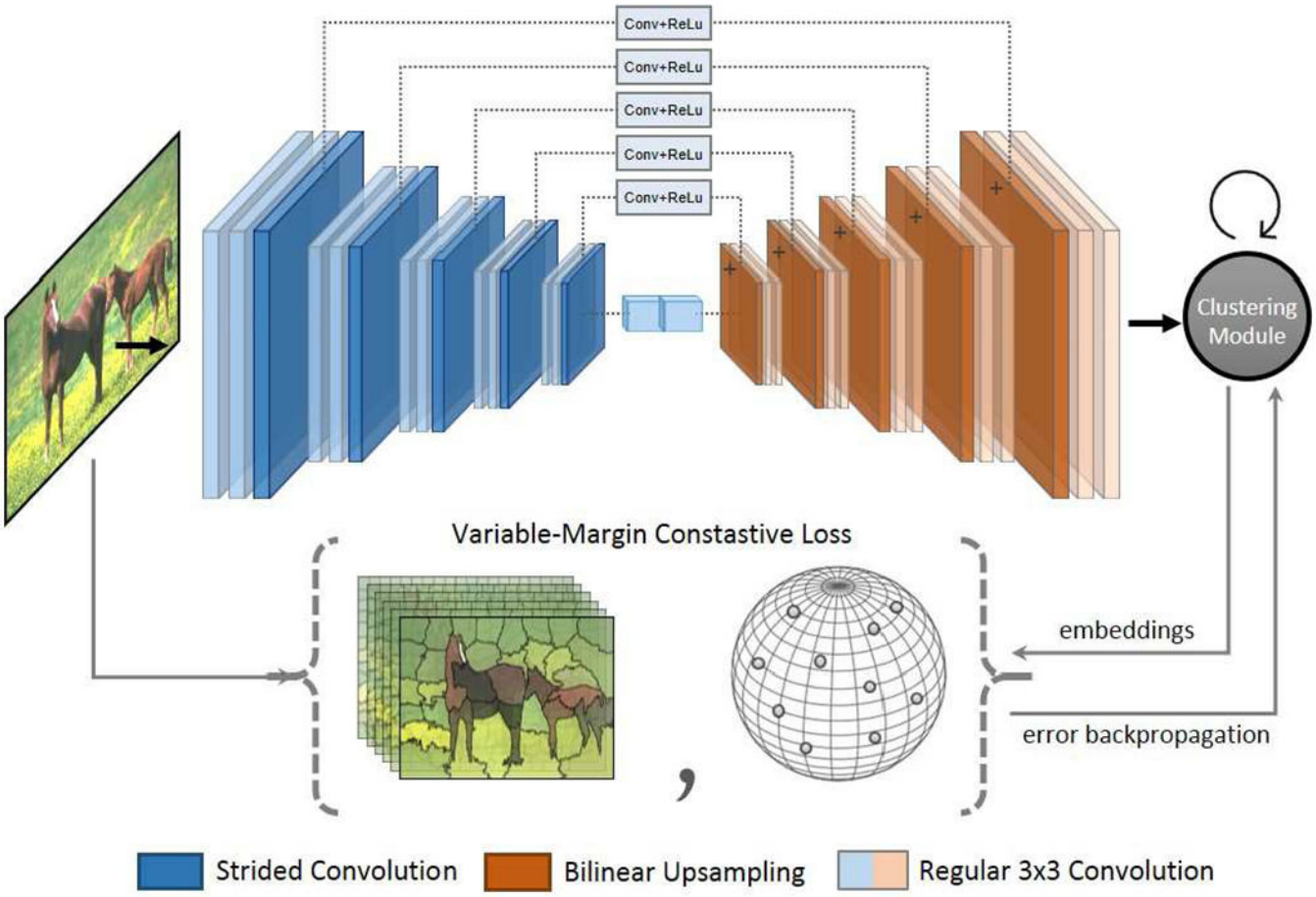

Fig. 5:

Schematic diagram of the proposed network. The input image is processed by a fully convolutional autoencoder with shortcut connections to output initial pixel embeddings. These embeddings are refined to accommodate image-wide context via the recurrent clustering module to output the final dense embeddings. The variable-margin contrastive loss compares the embedding distances with a superpixel segmentation generated with a randomly selected scale and detail attribute.

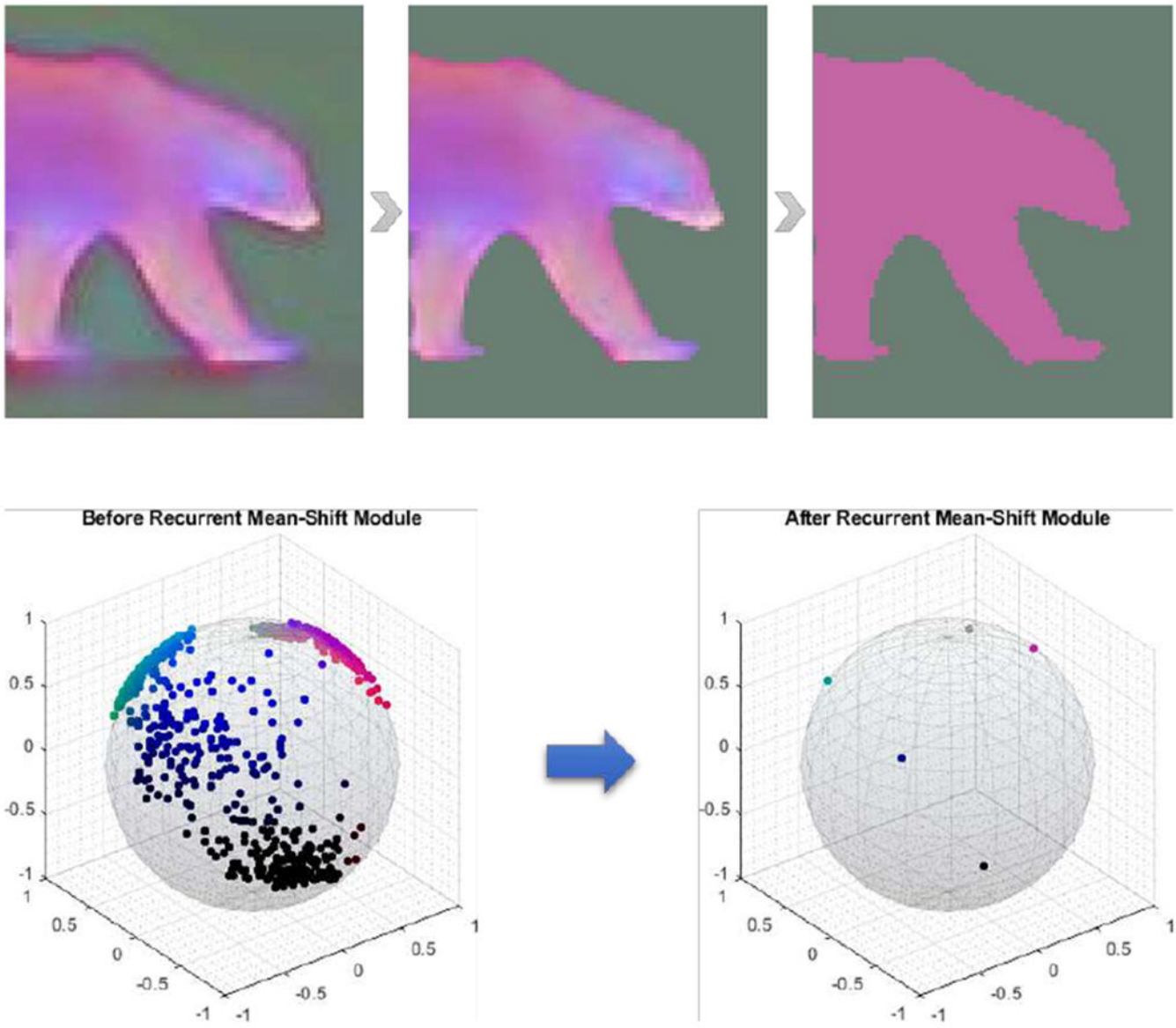

Fig. 6:

Effect of mean-shift clustering module iterations. Top: embedding vectors are reduced to three dimensions for visualization purposes. Bottom: normalized embedding vectors are plotted on a sphere. Embedding vectors are locally smooth. To induce context knowledge from a larger area, embeddings that are similar are collapsed to a single point on the hypersphere.

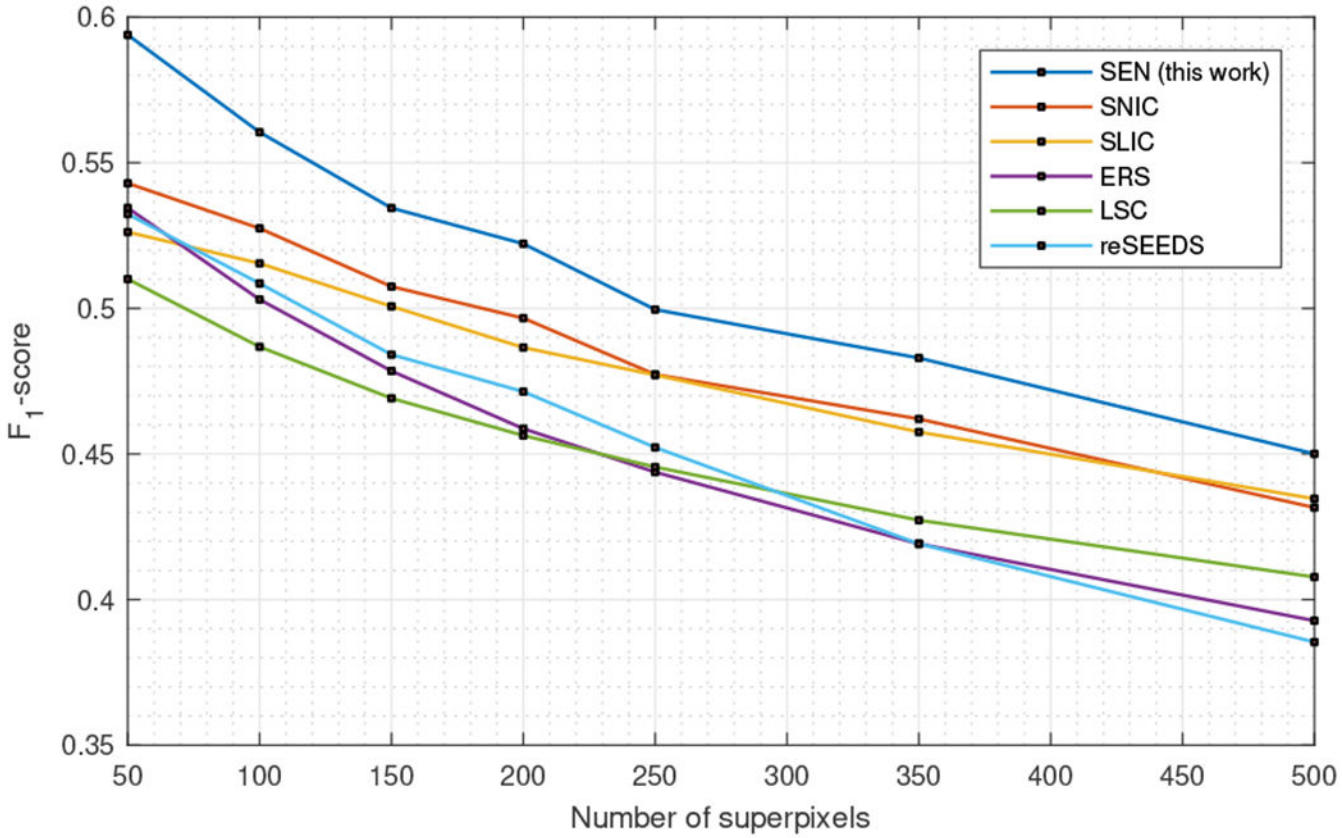

Fig. 7:

Plot of boundary F_1-score eq. 10 as a function of number of superpixels. The F-measure for SEN superpixels is considerably better than SNIC.

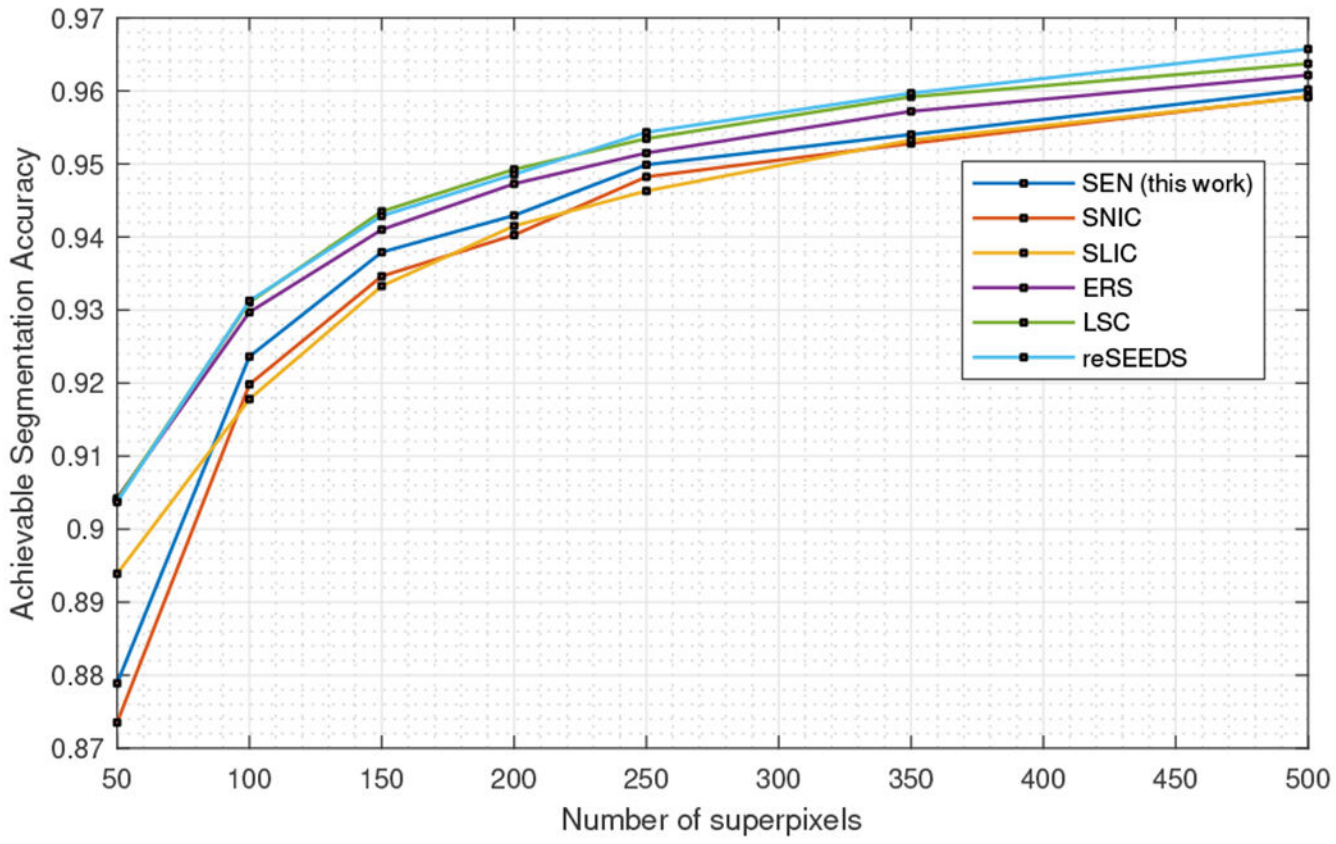

Fig. 8:

Plot of achievable segmentation accuracy eq. 12 as a function of number of superpixels. Using superpixel embedding increases the ASA score. The difference decreases with increase in number of superpixels.

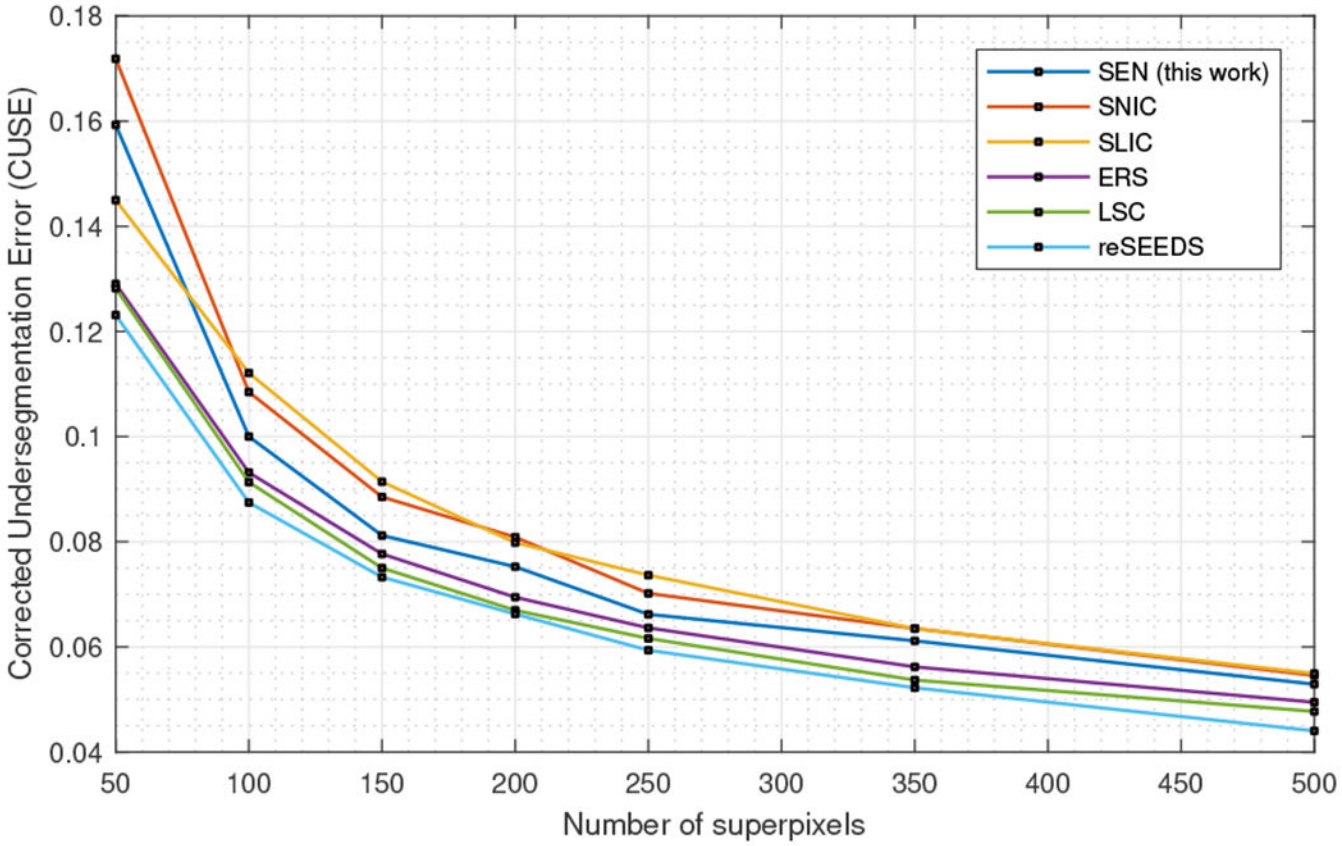

Fig. 9:

Plot of corrected undersegmentation error eq. 11 as a function of number of superpixels. The average CUSE for SEN is better than the best CUSE for SNIC. The difference decreases with increase in number of superpixels.

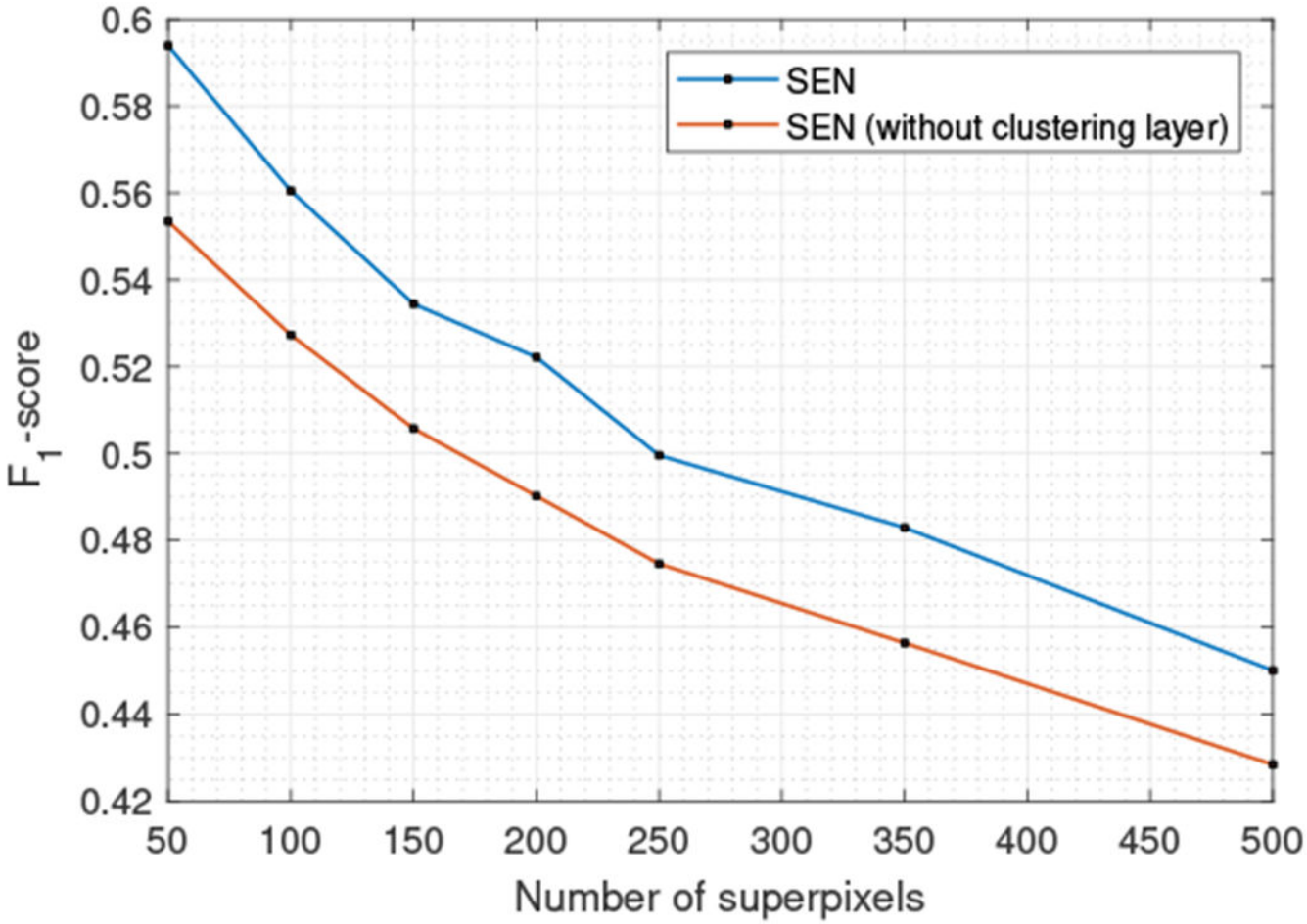

Fig. 10:
Ablation results: SEN boundary F_1-score performance comparison with and without the clustering module.

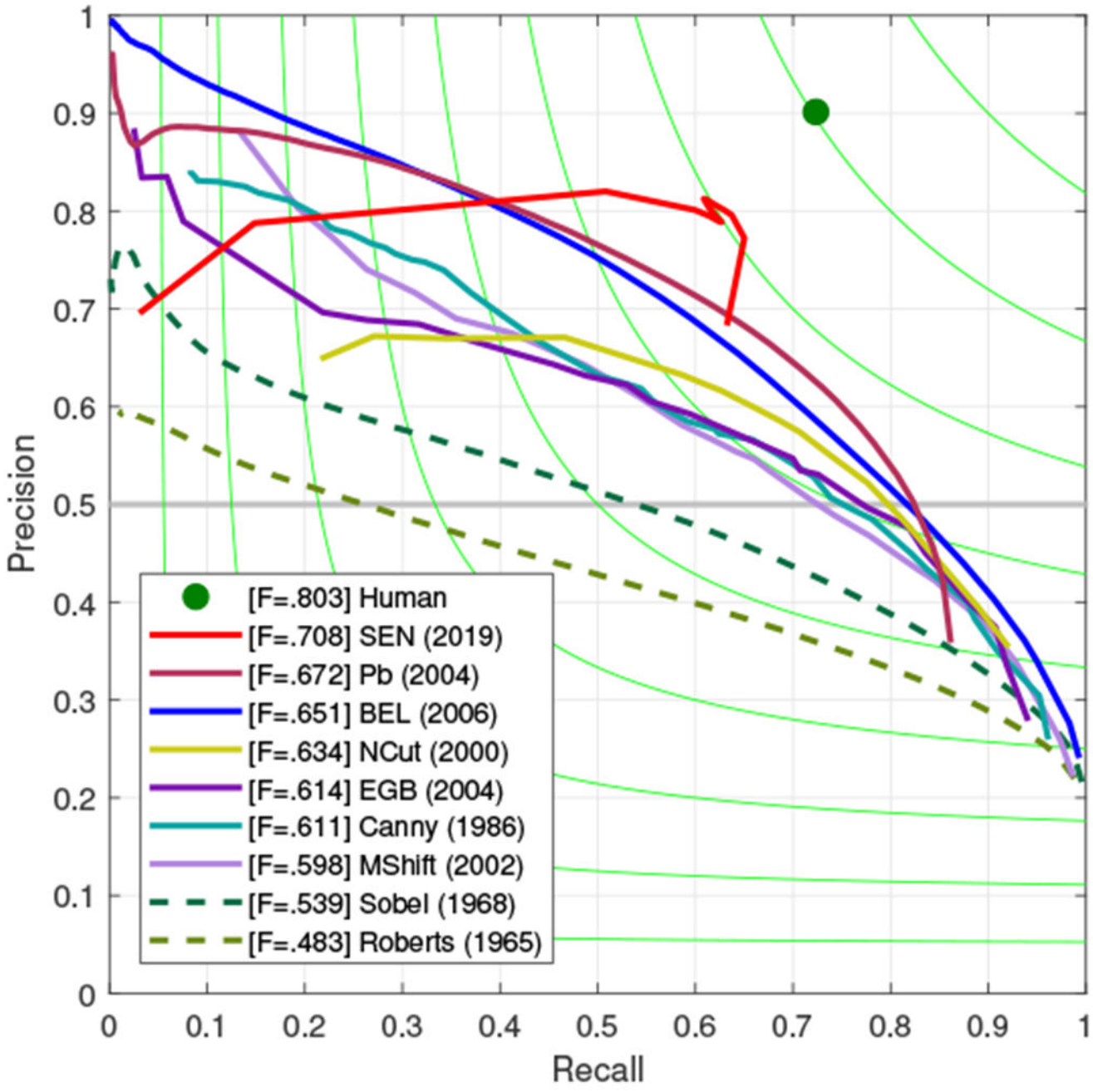

Fig. 11:
Contour detection results on BSDS500. Contours obtained by the proposed method (SEN) outperforms unsupervised edge detection methods.

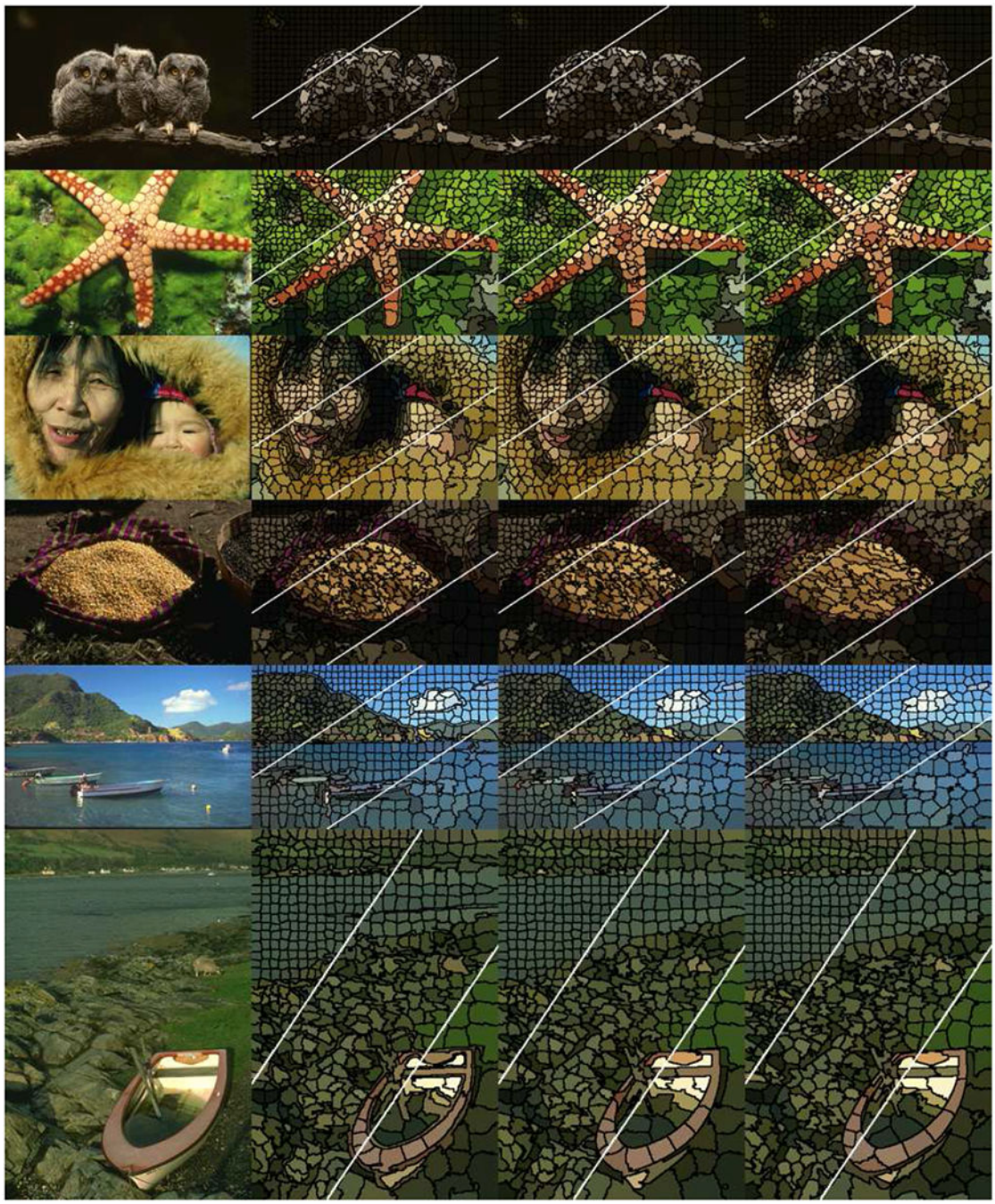

Fig. 12:
Qualitative comparison of superpixels with state-of-the-art methods with BSDS500 images for three different superpixel sizes (50/200/1000) shown as three partitions of the same image. From left: a) Input image b) SEN c) SNIC d) SLIC

TABLE I:

Comparison of time efficiency vs. boundary F_1 score for BSDS500 dataset

Method	Time (s)	F_1 Score
SEN	0.153	**0.559**
SNIC [17]	**0.064**	0.543
SLIC [18]	0.073	0.526
ERS [20]	0.504	0.534
LSC [19]	0.237	0.510
reSEEDS [25]	0.171	0.532