Saliency Enhancement using Superpixel Similarity

Leonardo de Melo Joao¹, Alexandre Xavier Falcão¹∗

Institute of Computer Science, University of Campinas, Campinas, 13083-872, São Paulo, Brazil

Abstract

Saliency Object Detection (SOD) has several applications in image analysis. Deep-learning-based SOD methods are among the most effective, but they may miss foreground parts with similar colors. To circumvent the problem, we introduce a post-processing method, named Saliency Enhancement over Superpixel Similarity (SESS), which executes two operations alternately for saliency completion: object-based superpixel segmentation and superpixel-based saliency estimation. SESS uses an input saliency map to estimate seeds for superpixel delineation and define superpixel queries in foreground and background. A new saliency map results from color similarities between queries and superpixels. The process repeats for a given number of iterations, such that all generated saliency maps are combined into a single one by cellular automata. Finally, post-processed and initial maps are merged using their average values per superpixel. We demonstrate that SESS can consistently and considerably improve the results of three deep-learning-based SOD methods on five image datasets.

Keywords: salient object detection, saliency enhancement, deep-learning, superpixel-based saliency, iterative saliency.

1. Introduction

Saliency Object Detection (SOD) aims to identify the most visually relevant regions within an image. SOD methods have been used in an extensive range of tasks, such as image segmentation [1 2 3], compression [4 5 6], quality assessment [7 8 9], and content-based image retrieval [10 11 12].

Traditional SOD methods can combine heuristics to model object-inspired (top-down) and image-intrinsic (bottom-up) information. Object-inspired models expect that salient objects satisfy specific priors based on domain knowledge – e.g., salient objects in natural images are expected to be centered [13], focused [14], or have vivid colors [15]. Image-intrinsic models provide candidate background and foreground regions used as queries and expect that salient regions be more similar to the foreground than background queries [16 17]. Thus, their results strongly rely on those preselected assumptions.

Recently, deep-learning-based SOD methods have replaced preselected assumptions with examples of salient objects from a given training set. The methods usually rely on the backbone of a Convolutional Neural Network (CNN) trained for image classification. However, there are network architectures that can be easily trained from scratch [18].

Deep SOD methods are among the most effective, but they may miss foreground parts with similar colors (see Figure 1). Such incomplete saliency maps drift away from the results humans expect. Since CNNs are object-inspired models that do not explore image-intrinsic information, we propose to explore superpixels in multiple scales and color similarity to improve saliency maps created by deep SOD methods.

∗afalcao@ic.unicamp.br

Figure 1: (a) Original image (top) and ground-truth segmentation (bottom). (b)-(d) BASNET [19] MSFNet [20], and U²Net [18] saliency maps (top) and their enhanced maps (bottom) using the proposed approach.

For such, we introduce a post-processing method, named Saliency Enhancement over Superpixel Similarity (SESS), which can improve an input saliency map over a few iterations of two alternate operations: object-based superpixel segmentation [21] and superpixel-based saliency estimation [22]. SESS uses an input saliency map to estimate seeds for superpixel delineation and define superpixel queries in foreground and background. A new saliency map is obtained by measuring color similarities between queries and superpixels, such that salient superpixels are expected to have high color similarity with at least one foreground query and low color similarity with most background queries. The process repeats for a given number of iterations, with a decreasing quantity of superpixels (increasing size or scale) per iteration. Although the process is meant to create progressively better saliency maps (Figure 2), we combine all output maps by cellular automata [23]. Finally, post-
Supervised object saliency detection (SOD) methods often rely on hand-crafted heuristics to model a set of characteristics shared by visually salient objects. These methods explore a combination of top-down object-inspired information — characteristics inherent to the object, not its relationship to other image regions — or bottom-up image-intrinsic strategies that use intrinsic image features to estimate saliency based on region similarity. Common top-down approaches expect salient objects centralized [13], composed of vivid colors [23, 15] or focused [14]. Early bottom-up strategies include estimating saliency according to pixel-level similarity to the mean image color [25] or defining a global contrast by comparing all possible pairs of image patches [13].

Later, saliency estimators started using superpixels for better region representation, consistently outperforming pixel and patch-based methods. Most superpixel-based SOD methods use different heuristics to assemble a combination of top-down and bottom-up approaches.

### 2.1. Traditional SOD methods

#### 2.1.1. Overview

Traditional SOD methods often rely on hand-crafted heuristics to model a set of characteristics shared by visually salient objects. These methods explore a combination of top-down object-inspired information — characteristics inherent to the object, not its relationship to other image regions — or bottom-up image-intrinsic strategies that use intrinsic image features to estimate saliency based on region similarity. Common top-down approaches expect salient objects centralized [13], composed of vivid colors [23, 15] or focused [14]. Early bottom-up strategies include estimating saliency according to pixel-level similarity to the mean image color [25] or defining a global contrast by comparing all possible pairs of image patches [13].

Superpixel-graph-based methods [28, 29, 30, 16, 17] estimate saliency by combining superpixel-adjacency contrast with the similarity between every superpixel and probable background and foreground queries. Background queries are usually defined at the image borders, but the methods propose strategies to mitigate error when this assumption is invalid. Such strategies include estimating four saliency maps (one for each image border) so that regions consistently highlighted are taken as salient [16]: using only border regions that share common characteristics amongst themselves [17]; include a weight function for the query regions, so that superpixels that grow close to the image center have a lower weight than the ones connected to the image limits [28]. However, background-based saliency methods often highlight parts of the desired object even when using those error-mitigating strategies. To improve upon background-based saliency, one can use background saliency to define salient regions as foreground queries and then compute a foreground saliency score [16, 17]. Among superpixel-based methods, a recent approach [22] has similarities and differences regarding our post-processing step. Similarly, it adopts alternating executions of object-based superpixel definition and superpixel-based saliency estimation. However, it relies on the superpixel-adjacency graph to estimate the saliency score based on the color similarity between adjacent superpixels and queries. It also allows the incorporation of user-provided domain-specific knowledge and object-inspired information in the saliency model.

A major drawback in all methods above is their reliance on heuristics or domain-specific prior information. We combine saliency estimations from a given object-inspired deep SOD model and the proposed superpixel-based image-intrinsic model to avoid such error-prone strategies. Our superpixel-based image-intrinsic model defines background and foreground queries starting from the saliency map estimated by the deep SOD model, improving those queries along with the iterations. Also, we do not impose spatial constraints as superpixel-graph-based methods do. We simply compute the color similarity between all superpixels and queries.

---

**Figure 2:** SESS saliency enhancement over multiple iterations. (a–c) original image, ground-truth segmentation and U²Net saliency; (d–f) SESS improvement over iterations 1, 3 and 10.
2.2. Deep-learning methods

2.2.1. Multi-layer perceptron (MLP)-based approaches

The usage of deep neural networks for saliency detection has been extensive in the past few years. As presented in a recent survey [31], earlier attempts used MLP-based approaches [42, 43, 44, 45, 46], adapting networks trained for image classification by appending the feature-extraction layers to a pixel-patch classifier. Despite the improvement of these models over heuristic-based traditional methods, they were incapable of providing consistent high spatial accuracy, primarily due to relying on local patch information.

2.2.2. Fully-Convolutional Neural-Networks (FCNN)

FCNN-based methods improved object detection and delineation, becoming the most common network class for visual saliency estimation. Most FCNN models use the pretrained backbone of a CNN for image classification (e.g. DenseNet [27], VGG [38], and Resnet [39]) and a set of strategies to exploit the backbone features in the fully convolutional saliency model. These strategies often explore information from shallow and deep layers, providing methods for aggregating or improving the multiple-scale features. Some methods use multiple atrous convolutions with different sampling rates [40, 41]; Zhang et al. [42] propose a generic framework that integrates multiple feature-scales into multiple resolutions; in [43], an attention-guided recurrent convolutional network is proposed to integrate multi-level features and reintroduce high-level semantic information from deep layers to shallow layers; Zhang et al. [44] presented a bi-direction module to pass information between the shallow and deep layers, and use a pyramid fusing strategy to combine multi-scale information.

The methods described above upsample directly from the low-resolution deep layers back to the full-size input layer, resulting in significant information loss. To reduce the negative impact of drastic upsampling, several methods have adopted the encoder-decoder structure to gradually re-scale the low-resolution features [45, 46, 47, 18, 19, 20]. Each method proposes a different strategy to exploit the multi-scale features in the decoder. As examples, Liu et al. [45] propose a global guidance module based on pyramid pooling to explicitly deliver object-location information in all feature maps of the decoder; Newell et al. [46] present an hourglass-shaped architecture, where multiple branches of convolution-sets are run in parallel at each downsampling and are then gradually fused by performing nearest-neighbor upsampling and element-wise addition; Tang et al. [47] proposed stacking U-shaped networks and densely connect their layers, providing a shared-memory implementation strategy to alleviate the memory-heavy requirements. In recent work, Miao et al. [20] proposes the Auto-Multi-Scale Fusion Network (Auto-MSFNet), which uses Network Architecture Search (NAS) as inspiration to propose an automatic fusion framework instead of trying to elaborate complex human-described strategies for multi-scale fusion. They introduce a new search cell (FusionCell) that receives information from multi-scale features and uses an attention mechanism to select and fuse the most important information.

Another common strategy is to use boundary information to achieve better object delineation. Early on, Luo et al. [48] proposed a new loss that heavily penalizes boundary regions for training an adaptation of VGG-16; Li et al. [49] proposes a two-step methodology that extracts contours from an input image and then extrapolates a saliency-map out of the contours; Su et al. [50] proposes a three-stream methodology with a boundary location stream, an interior perception stream, and a boundary/region transition prediction; and Zhao et al. [51] uses an encoder-decoder network to extract multi-scale salient-object features from a backbone, and uses both the high-resolution backbone features and the encoder-decoder object-location information to assist an edge-feature extraction that guides the delineation for the final saliency estimation. Instead of explicitly or exclusively using boundary information, Qin et al. [19] proposes the Boundary-Aware Segmentation Network (BASNet) with a hybrid loss to represent differences between the ground-truth and the predictions in a pixel, patch, and map level. The loss is a combination of the Binary-Cross-Entropy (pixel-level) [52], the Multi-scale Structural Similarity (patch-level), and the Intersection over Union (map-level). Their strategy consists of a prediction module that outputs seven saliency maps (an upsampled output of each decoder layer) and a residual refinement module that outputs one saliency map at the final decoder layer. All output maps are used when computing the hybrid loss, and both modules are trained in parallel.

All earlier methods require a backbone pretrained in a large classification dataset and often require many training images for fine-tuning. Recently, Qin et al. [18] proposed a nested U-shaped network by the name of U²Net, that has achieved impressive results without requiring a pretrained backbone. Their approach consists of extracting and fusing multiple-scale information throughout all stages of the encoder-decoder by modeling each stage as a U-shaped network. By doing so, they can extract deep information during all network stages without significantly increasing the number of parameters (due to the downscaling inside each inner U-net).

Even though each presented method has specific characteristics and shortcomings, the main goal of this paper is to aggregate image-intrinsic information to the complex object-inspired models provided by the deep saliency methods. In this regard, we selected as baseline three state-of-the-art saliency estimators that apply different strategies to learn their models: U²Net [13], which has the novelty of not requiring a pre-trained backbone; BASNet [19], that provides a higher delineation precision due to its boundary-awareness; and Auto-MSFNet [20], which uses a framework to learn the information fusion strategies that humans commonly define.

3. Saliency Enhancement by Superpixel Similarity (SESS)

SESS aims to improve the results of complex deep-learning-based SOD models by adding image-intrinsic information. For such, SESS combines alternate executions of object-based superpixel segmentation and superpixel-based saliency estimation. The multiple executions of both operations generate en-
hanced saliency maps for integration into a post-processed map, which is subsequently merged with the initial saliency map of the deep SOD model. Figure 3 illustrates the whole process, described in Sections 3.1 and 3.2.

3.1. Saliency Enhancement Module

The saliency enhancement module starts by computing a superpixel segmentation since SESS is a superpixel-based saliency enhancer. For such, we use the Object-based Iterative Spanning Forest algorithm (OISF) [21].

In short, OISF represents an image as a graph, whose pixels are the nodes and the arcs connect 8-adjacent pixels, elects seed pixels based on an input object saliency map, and executes the Image Foresting Transform (IFT) algorithm [55] followed by seed refinement multiple times to obtain a final superpixel segmentation. Each superpixel is then represented as an optimum-path tree rooted at its most closely connected seed.

For the initial seed sampling based on a saliency map, we use OSMOX [56], which allows user control over the ratio of seeds to be placed inside and outside the salient objects with no need to binarize the saliency map. For seed refinement, we set the new seed of each superpixel to be the closest pixel to its geometrical center.

For this setup of OISF, the user has to provide four parameters: $\alpha$ to control the importance of superpixel regularity; $\beta$ to control the importance of boundary adherence; $\gamma$ to provide a balance between saliency information and pixel colors; and $i_o$ the number of iterations to obtain a final superpixel segmentation. Within the method proposed in this paper, we fixed $\alpha = 12$ and $\beta = 0.5$ as suggested in [57], while $\gamma$ and $i_o$ were tuned by grid searching (see Section 4.1).

The only specific implementation change regarding the superpixel segmentation algorithm is the percentage of object seeds required by the seed sampler. Instead of fixing a percentage that would fit most images, we set the number of seeds required by the seed sampler. Instead of fixing a percentage that would fit most images, we set the number of seeds to a minimum of 200 superpixels.

Let $\mathcal{S}$ be a superset of all superpixels, $\mathcal{F} : \mathcal{S} \rightarrow \mathbb{R}^3$ map to every superpixel the mean color of all its pixels in the CIELab color-space. We define two query lists, $\mathcal{Q}_f$ for foreground superpixels, and $\mathcal{Q}_b$ for background ones, where $\mathcal{Q}_f \cup \mathcal{Q}_b = \mathcal{S}$. Take $\psi$ to be the Otsu threshold of the previously computed saliency map, then, the query lists are defined as follows: $S \in \mathcal{Q}_f \leftrightarrow s(S) \geq \psi$, and similarly, $S \in \mathcal{Q}_b \leftrightarrow s(S) < \psi$.

Taking $S, R \in \mathcal{S}$ as superpixels, we define a Gaussian-weighted similarity measure between superpixels as:

$$\text{sim}(S, R) = \exp\left(-\frac{d(F(S), F(R))^2}{\sigma^2}\right), \quad (1)$$

where $d(F(S), F(R))$ is the euclidean distance between the superpixel’s mean colors, and $\sigma^2 = 0.01$ is the variance used to regulate the dispersion of similarity values.

Using the similarity measure and the query lists, we define two saliency scores for each superpixel: one based on foreground queries, and the other on background ones. For the foreground saliency score, a region is deemed to be salient if it shares similar characteristics to at least one other foreground region:

$$s_f(S) = \max_{\forall R \in \mathcal{Q}_f} \{\text{sim}(S, R)\} \quad (2)$$

However, when $S = S_f \in \mathcal{Q}_f$ is a foreground query, its foreground saliency score equals one, which is an out-scaled value as compared to other gaussian-weighted similarities. To keep foreground queries with the highest score but in the same scale of the remaining superpixels’ scores, we update their values to match the highest non-foreground query region’s saliency: $s_f(S_f) = \max_{\forall Q \in \mathcal{Q}_b} (Q)$. By doing so, we allow for a forced exploration of possible foreground regions at each iteration.

Similarly, the background-updated saliency score defines that a superpixel is salient if it has low similarity to most background queries:

$$s_b(S) = 1 - \frac{\sum_{R \in \mathcal{Q}_b, S \neq R} \text{sim}(S, R)}{|\mathcal{Q}_b|} \quad (3)$$

When the background is too complex, the normalized mean difference of all superpixels will be close to fifty percent, resulting in a mostly gray map with little aggregated information (Figure 4). To detect those scenarios, we compute a map-wise distance to fifty percent probability $d_{0.5} = \frac{1}{|\mathcal{S}|} \sum_{S \in \mathcal{S}} (s_b(S) - 0.5)^2 |\mathcal{S}|$. For maps with $d_{0.5} < 0.1$, we set the background saliency score to be the saliency of the previous iteration $s_b(S) = s^{-1}(S)$, where $s^{-1}$ is the final saliency computed in the last iteration and $s^0$ is the initial deep-learning saliency.

The final saliency score for the iteration is the product of both saliencies:

$$s'(S) = s_f(S) \cdot s_b(S) \quad (4)$$

By multiplying both scores, the combined saliency ignores any region that was hardly taken as non-salient by foreground or background saliency and keeps a high score for regions taken as salient in both maps. The new score is saved together with the superpixel representation used to create it. Then, the saliency is fed to the superpixel algorithm to restart the enhancement cycle. We reduce the number of superpixels by 20% at each iteration to a minimum of 200 superpixels.

The result of all iterations is then combined using the method described in Section 3.3.

---

1The OISF method is available at [https://github.com/LIDS-UNICAMP/OISF](https://github.com/LIDS-UNICAMP/OISF)
3.2. Saliency Integration Module

To integrate the saliency maps created throughout the iterations, we use a traditional unsupervised saliency integrator method [23] that models an update function using a Bayesian framework to iteratively update a Cellular Automata whose cells are derived from a stack of the saliency maps.

In short, the saliency maps are stacked to form a three-dimensional grid where the (x,y) coordinates match the saliency map’s (x,y) coordinates and z is the direction in which the different maps are stacked. The pixels are then updated over it iterations in a way that the new pixel saliency depends on how consistently salient its adjacency is. The adjacency used is a cuboid adjacency \( C_p \) around the pixel \( p \), so that a pixel \( q \) is considered adjacent to a \( p \) if \( |x_p - x_q| \leq 1, |y_p - y_q| \leq 1 \); i.e. a 4-adjacency extended to all maps on the z-axis.

For the update rule, the saliency score is represented as log odds:

\[
I(p) = \frac{s(p)}{1-s(p)}
\]

so that the value can be updated by performing subsequent sums. The value changes according to the rule:

\[
s^{t+1}(p) = I(p) + \sum_{q \in C_p} \lambda \delta(q),
\]

where \( \lambda \) is a constant that regulates the update rate, and \( \delta(\cdot) \in \{-1, 1\} \) is a signal function that makes the saliency either increase or decrease at each update depending on whether \( s(q) \) is greater or smaller than the Otsu threshold of the saliency map that originated the cell. Essentially, if a pixel is surrounded by more foreground than background regions, its saliency increase over time by a rate of \( \lambda \); likewise, the saliency decreases for pixels in a non-salient adjacency.

Afterward, the integrated map is run through the last iteration of saliency estimation using Equation 2 and the initial number of superpixels, creating the last color-based saliency score \( s_c \). This last estimation serves two purposes: improve object delineation by relying on the quality of the superpixel algorithm — rather than depending on the cuboid adjacency relation; and eliminate unwanted superpixel leakage that may occur when the superpixel number is reduced (Section 4.2.2).

Lastly, we reintroduce the deep object-inspired model by averaging its saliency values inside the superpixels of the last segmentation, which defines another saliency score for each superpixel:

\[
s_d(S) = \frac{1}{|S|} \sum_{p \in S} s_0(p),
\]

where \( s_0(p) \) is the saliency score provided by the network. The final saliency score is then defined as:

\[
s_f(p) = \max\{s_d(p), s'(p)\}.
\]

Therefore, the final saliency map will highlight salient regions according to the image-intrinsic or the superpixel-delineated object-inspired model.
4. Experiments and Results

4.1. Datasets and experimental setup

Datasets: We used five well-known datasets for comparison among SOD methods. **DUT-OMRON** is composed of 5168 images containing one or two complex foreground objects in a relatively cluttered background; **HKU-IS** contains 4447 images with one or multiple low contrast foreground objects each; **ECSSD** consists of 1000 images with mostly one large salient object per image in a complex background; **ICoSeg** consists of 643 images usually with several foreground objects each; and **SED2** is formed by 100 images with two foreground objects per image.

Parameter tuning: The baselines networks were used as provided by the authors since they have been pretrained on datasets with similar images (e.g., **DUT-OMRON**). For **SESS**, we randomly selected 50 images from each dataset, totaling a training set with 300 images, to optimize its parameters by grid search. The remaining images compose the test sets of each dataset. Grid search was performed with saliency maps from each baseline, creating three sets of parameters (one per baseline). Most parameters were optimized to the same value independently of the baseline, except for the number of iterations \( i \), the number of superpixels \( n_s \), the number of foreground seeds per component \( n_{s,f} \), and the number of OISF iterations \( i_o \). Taking the order U²Net, BASNet, and MSFNet, the parameters were respectively, \( i = \{12, 9, 12\} \), \( n = \{2500, 200, 2500\} \), \( n_{s,f} = \{10, 30, 30\} \), and \( i_o = \{5, 3, 1\} \). The rest of the parameters with fixed values for all methods were optimized to be: \( \gamma = 10.0 \), \( \sigma^2 = 0.01 \), \( \lambda = 0.0001 \), \( t = 3 \), where \( \gamma \) is related to the superpixel segmentation algorithm, \( \sigma^2 \) is the gaussian variance for the graph node similarity, \( \lambda \) and \( t \) are related to the cuboid integration.

Evaluation metrics: We used six measures for quantitative assessment. (1) **Mean Structural-measure**, \( S_m \), which evaluates the structural similarity between a saliency map and its ground-truth; (2) **max F-measure**, \( \max F_{\beta} \), which represents a balanced evaluation of precision and recall for simple threshold segmentation of the saliency maps; (3) **weighted F-measure**, \( F_{\gamma}^w \), which is similar to \( \max F_{\beta} \) but keeps most saliency nuances by not requiring a binary mask; (4) **Mean Absolute Error**, \( MAE \), that provides a pixel-wise difference between the output map and expected segmentation; (5) **mean Enhanced-alignment measure**, \( E_{\gamma}^m \), used to evaluate local and global similarities simultaneously; and (6) **precision-recall curves**, which display precision-recall values between the expected segmentation and saliency maps binarized by varying thresholds.

Post processing: **SESS** can sometimes score a superpixel with close-to-zero saliency values. These small saliencies are not visually perceived by the observer but can significantly improve the local SSIM to change from a full match on the original map to a complete miss on the enhanced map. To deal with these small values, we eliminate regions in the map with saliency lower than half of the Otsu threshold.

4.2. Ablation study

To analyze the impact of some decisions within our proposal, we present ablation studies on (i) the effects of re-introducing the deep model at the end of the last iteration; (ii) re-using the initial number of superpixels at the last iteration. We used the same parameters as the ones reported in Section 4.1 for the ablation studies.

4.2.1. Ablation on reintroducing the deep saliency model

The goal of **SESS** is to improve the complex and robust deep models by adding information more closely related to what a human observer expects. Not re-introducing the deep model would imply creating a color-based model that keeps all the robustness of the deep-learning ones, which is hardly feasible.

An example of why the proposed color-based model is often insufficient can be seen in Figure 6. The object of interest was less uniformly salient: The armband the player is wearing has significantly different colors than the rest of the object, which caused the model not to consider it as part of the foreground. In that example, **SESS** mainly contributed to creating sharper edges.

The downside of re-introducing the deep model is that it reduces the influence of **SESS** on wrongly salient regions. Take Figure 7 as an example: The deep-model wrongly highlighted part of the airplanes’ smoke trail; **SESS** did perceive...
4.2.2. Ablation on using the initial number of superpixels for the last iteration

As the iterations progress and the number of superpixels decreases, the object-representation provided by the superpixels can worsen due to either leakage to the background or under-segmentation of too distinct object parts. As exemplified in Figure 8, the wings of the airplanes were lost in the final saliency map due to superpixel leakage. The model could adequately highlight most of the planes’ wings by increasing the number of superpixels at the last iteration.

4.3. Quantitative comparisons

Table 1 shows that the proposed method can consistently improve $\max F_{\beta}$, $F_{\beta}^n$, MAE and $E_{\psi}^m$ for U²Net [18] and BASNET [19] in all five datasets, and achieve competitive results on $S_m$. On both datasets composed mostly of images containing more than one object (Sed2 and ICoSeg), the results were even more expressive, indicating that SESS can considerably improve the saliency representation of similar objects.

The precision-recall curves show a clear advantage of SESS-enhanced maps over the non-enhanced ones on datasets mainly consisting of multiple objects (sed2 and icoseg). There is a breakpoint where SESS loses precision more rapidly than non-enhanced maps on the other three datasets. We attribute the rapid precision loss to SESS highlighting more parts of non-salient objects deemed partially salient by the deep models (Figure 9). By looking at the $\max F_{\beta}$, we see that by segmenting SESS maps using an adequate threshold, the segmentation...
results are often better than the non-enhanced maps on almost every combination of method and dataset.

The better representation of non-salient objects discussed before also highly impacts the $S_{\text{me}}$. Similar to the example shown in Figure 5, spatially extending the saliency of a non-salient object to other image quadrants drastically affects the quality of the map according to the Mean Structural-measure.

Regarding Auto-MSFNet, apart from the ICoSeg and SED2, SESS could not improve its metrics. By analyzing the precision-recall curves, we identified that Auto-MSFNet has the lowest precision of all three methods. We realized that the AutoMSF-Net creates more frequently partially salient regions on non-salient objects. Both examples used in Figure 9 were taken from the AutoMSF-Net results. Therefore, SESS seems to over-trust the deep model on non-salient regions.

4.4. Qualitative comparisons with non-enhanced maps

To visually evaluate the benefits of enhancing a saliency map using SESS, we compiled images where SESS improved different aspects of object representation (Figure 11). In the first two rows, there are images with minimal salient values in a small part of the desired objects (columns (c) and (g), on the first and second row, respectively). Even though the initial map provides little trust (i.e., small saliency values), SESS created similar resulting maps using all deep models.

In the third and fourth rows, we present examples where SESS captured most intended objects by extending the saliency value to regions with high similarity to the provided saliency. Note, however, that SESS was unable to improve (e), even though (e) is a higher precision map. Because the map is almost binary and most salient regions are taken as background queries, SESS could not highlight the other salient regions.

The bottom four rows present images where SESS was able to complete partially salient objects. Excluding the last row, each row shows simpler objects that the deep models failed to capture fully. In particular, in (e) and (g), SESS could properly include a narrow stem coming out of the traffic light, even though narrow structures can be complex during superpixel delineation. The last image presents a successful case on a difficult challenge due to the radiating shadow coming out of the hole (which makes it hard to delineate adequate borders), especially on (e) where the initial map gave no part of the desired object.

5. Conclusion

We presented a superpixel-based saliency enhancer that takes advantage of a superpixel/saliency enhancement loop to iteratively enhance deep saliency maps, providing a more human-relatable result. Our method estimates background and foreground queries from a saliency map and computes the color similarity between superpixels and queries without spatial constraints for saliency improvement. It exploits multiple scales of superpixel representation and integrates the resulting saliency maps by cellular automata. Experimental results on five public SOD datasets demonstrate that SESS improves object representation, especially in partially salient objects and in images with multiple salient objects with only a few captured by the deep model. Although better superpixel descriptors can be explored, we focused on a simple color descriptor to illustrate how well deep models can be assisted by image-intrinsic information to improve their results.

| Method       | S_{\text{me}} ↑ | maxF_{\beta} ↑ | F_{\gamma} ↑ | MAE ↓ | F_{\psi} ↓ | BLUE ↑ |
|--------------|-----------------|----------------|--------------|-------|------------|--------|
| UNet         | 0.8193          | 0.8140         | 0.7704       | 0.0558| 0.8529     |        |
| UNet+SESS    | 0.8527          | 0.8404         | 0.8067       | 0.0483| 0.8816     |        |
| BASNet       | 0.8555          | 0.8555         | 0.8119       | 0.0514| 0.8881     |        |
| BASNet+SESS  | 0.8643          | 0.8647         | 0.8375       | 0.0470| 0.9037     |        |
| Auto-MSFNet  | 0.8374          | 0.8333         | 0.8032       | 0.0544| 0.8800     |        |
| Auto-MSFNet+SESS | 0.8378    | 0.8518         | 0.8182       | 0.0524| 0.8881     |        |
| ECSSD        | S_{\text{me}} ↑ | maxF_{\beta} ↑ | F_{\gamma} ↑ | MAE ↓ | F_{\psi} ↓ | BLUE ↑ |
| UNet         | 0.9209          | 0.9286         | 0.8993       | 0.0370| 0.9407     |        |
| UNet+SESS    | 0.9182          | 0.9267         | 0.9099       | 0.0339| 0.9547     |        |
| BASNet       | 0.9142          | 0.9262         | 0.8992       | 0.0384| 0.9414     |        |
| BASNet+SESS  | 0.9123          | 0.9267         | 0.9075       | 0.0355| 0.9468     |        |
| Auto-MSFNet  | 0.9067          | 0.9229         | 0.9032       | 0.0387| 0.9427     |        |
| Auto-MSFNet+SESS | 0.9040    | 0.9280         | 0.9026       | 0.0389| 0.9411     |        |
| DUT-JMORON   | S_{\text{me}} ↑ | maxF_{\beta} ↑ | F_{\gamma} ↑ | MAE ↓ | F_{\psi} ↓ | BLUE ↑ |
| UNet         | 0.8444          | 0.7868         | 0.7941       | 0.0541| 0.8627     |        |
| UNet+SESS    | 0.8418          | 0.7895         | 0.7647       | 0.0518| 0.8736     |        |
| BASNet       | 0.8362          | 0.7750         | 0.7465       | 0.0558| 0.8607     |        |
| BASNet+SESS  | 0.8335          | 0.7758         | 0.7556       | 0.0548| 0.8686     |        |
| Auto-MSFNet  | 0.8321          | 0.7748         | 0.7536       | 0.049 | 0.8662     |        |
| Auto-MSFNet+SESS | 0.8323    | 0.7730         | 0.7533       | 0.0491| 0.8680     |        |
| ICoSeg       | S_{\text{me}} ↑ | maxF_{\beta} ↑ | F_{\gamma} ↑ | MAE ↓ | F_{\psi} ↓ | BLUE ↑ |
| UNet         | 0.8727          | 0.8585         | 0.8210       | 0.0476| 0.8972     |        |
| UNet+SESS    | 0.8669          | 0.8706         | 0.8548       | 0.0404| 0.9165     |        |
| BASNet       | 0.8702          | 0.8578         | 0.8217       | 0.0476| 0.8972     |        |
| BASNet+SESS  | 0.8784          | 0.8630         | 0.8397       | 0.0436| 0.9037     |        |
| Auto-MSFNet  | 0.8662          | 0.8515         | 0.8256       | 0.0425| 0.9083     |        |
| Auto-MSFNet+SESS | 0.8703    | 0.8544         | 0.8332       | 0.0417| 0.9095     |        |
| HKU-IS       | S_{\text{me}} ↑ | maxF_{\beta} ↑ | F_{\gamma} ↑ | MAE ↓ | F_{\psi} ↓ | BLUE ↑ |
| UNet         | 0.9183          | 0.9291         | 0.8950       | 0.0311| 0.9453     |        |
| UNet+SESS    | 0.9174          | 0.9292         | 0.8985       | 0.0374| 0.9528     |        |
| BASNet       | 0.9123          | 0.9264         | 0.8977       | 0.0308| 0.9476     |        |
| BASNet+SESS  | 0.9110          | 0.9260         | 0.9075       | 0.0278| 0.9536     |        |
| Auto-MSFNet  | 0.9148          | 0.9297         | 0.9144       | 0.0255| 0.9617     |        |
| Auto-MSFNet+SESS | 0.9150    | 0.9270         | 0.9146       | 0.0255| 0.9603     |        |

Table 1: Quantitative comparison between enhanced and non-enhanced maps in terms of $S_{\text{me}} ↑$, maxF_{\beta} ↑, $F_{\gamma} ↑$, MAE ↓, and $F_{\psi} ↓$. BLUE indicates the best result between the enhanced and non-enhanced maps.

Figure 9: SESS improving the saliency of wrogfully salient object partially highlighted by Auto-MSFNet. (a) original image; (b) label; (c) saliency map by Auto-MSFNet; (d) (c) enhanced by the proposed SESS.
For future work, we want to explore SESS enhanced maps for interactive- and co-segmentation tasks. The goal will be to assist humans to annotate images with fewer interactions, using the highly-trusted human-provided location information to define the precise background and foreground queries, combined with robust deep models to capture finer features. Another characteristic inherent in SESS is the superpixel improvement over time. Although this aspect was not in focus in this paper, future work includes further evaluation of the enhancement loop’s impact on superpixel segmentation and how the final superpixel
map can assist interactive object segmentation.

Acknowledgement

This work was supported by ImmunoCamp, CAPES, and FAPESP2014/12236-1

References

[1] E. Iqbal, A. Niaz, A. A. Memon, U. Asim, K. N. Choi, Saliency-driven active contour model for image segmentation, IEEE Access 8 (2020) 208978–208991.
[2] F. Sun, W. Li, Saliency guided deep network for weakly-supervised image segmentation, Pattern Recognition Letters 120 (2019) 62–68.
[3] H. Bi, H. Tang, G. Yang, H. Shu, J.-L. Dillenseger, Accurate image segmentation using gaussian mixture model with saliency map, Pattern Analysis and Applications 21 (3) (2018) 869–878.
[4] J. Wang, L. de Melo Joao, A. X. Falcão, J. Kosinka, A. C. Telea, Focus-and-context skeleton-based image simplification using saliency maps., in: VISIGRAPP (4: VISAPP), 2021, pp. 45–55.
[5] Y. Patel, S. Appalaraju, R. Mannath, Saliency driven perceptual image compression, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 227–236.
[6] C. Guo, L. Zhang, A novel multiresolution spatiotemporal saliency detection model and its applications in image and video compression, IEEE transactions on image processing 19 (1) (2009) 185–198.
[7] I. Abouelaziz, A. Chetouani, M. El Hassouni, L. J. Latecki, H. Cherifi, 3d visual saliency and convolutional neural network for blind mesh quality assessment, Neural Computing and Applications 32 (21) (2020) 16589–16603.
[8] H. Liu, I. Heynderickx, Towards an efficient model of visual saliency for objective image quality assessment, in: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2012, pp. 1153–1156.
[9] X. Zhou, Y. Zhang, N. Li, X. Wang, Y. Zhou, Y.-S. Ho, Projection invariant feature and visual saliency-based stereoscopic omnidirectional image
[54] G. Máttyus, W. Luo, R. Urtasun, Deeproadmapper: Extracting road topology from aerial images, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 3438–3446.

[55] A. X. Falcão, J. Stolfi, R. A. Lotufo, The image foresting transform: Theory, algorithms, and applications, IEEE Trans. Pattern Anal. Mach. Intell. 26 (2004) 19–29.

[56] F. Belém, L. Melo, S. Guimarães, A. Falcão, The importance of object-based seed sampling for superpixel segmentation, in: Proc. 32nd Conf. Graphics Pattern Images (SIBGRAPI), 2019, pp. 108–115, to appear.

[57] J. E. Vargas-Muñoz, A. S. Chowdhury, E. B. Alexandre, F. L. Galvão, P. A. V. Miranda, A. X. Falcão, An iterative spanning forest framework for superpixel segmentation, IEEE Transactions on Image Processing 28 (7) (2019) 3477–3489.

Alexandre Xavier Falcao is a professor at the Institute of Computing (IC), University of Campinas (UNICAMP). He has worked on machine learning, computer vision, image processing, and data visualization at the IC-UNICAMP with research productivity fellowships from the Brazilian National Council for Scientific and Technological Development (CNPq) since 1998.

Leonardo de Melo Joao is currently pursuing a Ph.D. in Computer Science at University of Campinas (UNICAMP), SP, Brazil. He received a M.Sc degree in Computer Science from the University of Campinas (UNICAMP) in 2020. His research interests include machine learning and image processing, with an emphasis on saliency estimation and image segmentation.