



Iterative Saliency Enhancement over Superpixel Similarity

Leonardo de Melo Joao^{1,2} ^a and Alexandre Xavier Falcao¹  ^b

¹Institute of Computing, State University of Campinas, Campinas, 13083-872, São Paulo, Brazil

²LIGM, Univ. Gustave-Eiffel, Marne-la-Valée, F-77454, France

Keywords: Salient Object Detection, Saliency Enhancement, Deep-Learning, Superpixel-Based Saliency, Iterative Saliency.

Abstract: Saliency Object Detection (SOD) has several applications in image analysis. The methods have evolved from image-intrinsic to object-inspired (deep-learning-based) models. However, when a model fails, there is no alternative to enhance its saliency map. We fill this gap by introducing a hybrid approach, the *Iterative Saliency Enhancement over Superpixel Similarity* (ISESS), that iteratively generates enhanced saliency maps by executing two operations alternately: object-based superpixel segmentation and superpixel-based saliency estimation - cycling operations never exploited. ISESS estimates seeds for superpixel delineation from a given saliency map and defines superpixel queries in the foreground and background. A new saliency map results from color similarities between queries and superpixels at each iteration. The process repeats, and, after a given number of iterations, the generated saliency maps are combined into one by cellular automata. Finally, the resulting map is merged with the initial one by the maximum between their average values per superpixel. We demonstrate that our hybrid model consistently outperforms three state-of-the-art deep-learning-based methods on five image datasets.

1 INTRODUCTION

Saliency Object Detection (SOD) aims to identify the most visually relevant regions within an image. SOD methods have been used in many tasks, such as image segmentation (Iqbal et al., 2020), compression (Wang et al., 2021a), and content-based image retrieval (Al-Azawi, 2021).


Traditional SOD methods 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 often centered (Cheng et al., 2014), focused (Jiang et al., 2013), or have vivid colors (Peng et al., 2016). Image-intrinsic models provide candidate background and foreground regions used as queries and expect that salient regions are more similar to the foreground than background queries (Yang et al., 2013). 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. Instead of modeling assumptions of common salient characteristics, these models learn object-inspired features from abundant annotated data.

Although deep SOD methods are among the most effective, they only rely on object-inspired features learned from the training images. For that, the saliency maps created may drift away from the results humans expect, where foreground parts with similar colors to the ones highlighted are missed (see Figure 1). Currently, there is no alternative to improve said saliency maps and add image-intrinsic characteristics to these models. We fill this gap by proposing a hybrid model named *Iterative Saliency Enhancement over Superpixel Similarity* (ISESS), which explores the color similarity of superpixels in multiple scales to improve the output of deep object-inspired models.

ISESS is meant to improve saliency maps containing genuinely detected salient objects whose parts (superpixels) have similar colors. Although other similarity criteria could be used, its current implementation uses only color similarity. It can improve an input saliency map over a few iterations of two alternate operations: object-based superpixel segmentation (Belém et al., 2019) and superpixel-based saliency estimation. ISESS uses an input saliency

^a  <https://orcid.org/0000-0003-4625-7840>

^b  <https://orcid.org/0000-0002-2914-5380>

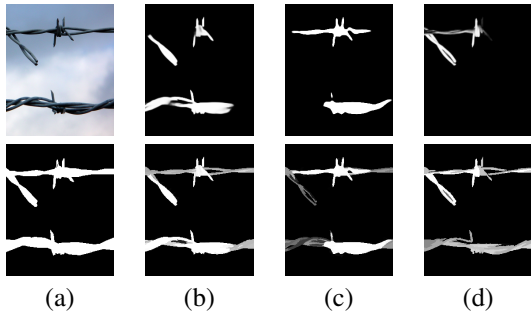


Figure 1: (a) Original image (top) and ground-truth segmentation (bottom). (b)-(d) BASNET (Qin et al., 2019) MSFNet (Miao et al., 2021), and U²Net (Qin et al., 2020) saliency maps (top) and their enhanced maps (bottom).

map to estimate seeds for superpixel delineation and define superpixel queries in the 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 scale) per iteration. Although the process is meant to create progressively better saliency maps (Figure 2), we combine all output maps by cellular automata (Qin et al., 2015). By using the deep model as input, ISESS removes the need for preselected assumptions, providing an image-intrinsic extension of the object-inspired model. Finally, the image-intrinsic and initial (object-inspired) maps are merged by the maximum between their average values per superpixel, creating a hybrid saliency model. It is worth noting that ISESS does not depend on the choice of the object-inspired method.

We demonstrate that our hybrid model can improve three state-of-the-art deep SOD methods (namely Basnet (Qin et al., 2019), U²Net (Qin et al., 2020), and Auto-MSFNet (Miao et al., 2021)) using five well-known datasets. These results are confirmed by several metrics in all datasets, especially when images have multiple salient objects. Qualitative analysis also shows considerably enhanced maps whenever deep models fail to capture salient regions.

Therefore, the contributions of this paper are:

- A hybrid model for saliency object detection that does not rely on preselected assumptions while using object-inspired and multiscale image-intrinsic information.
- A first method to enhance object saliency maps, even when they are generated by deep-learning-based methods, with no need to combine multiple SOD methods (Section 2.2).

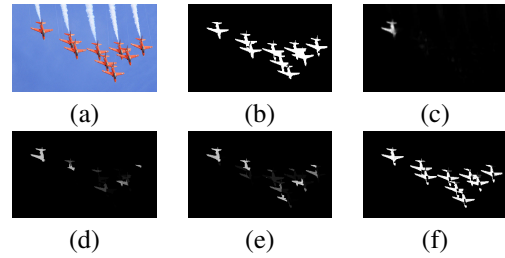


Figure 2: ISESS saliency enhancement over multiple iterations. (a-c) original image, ground-truth segmentation and U²net saliency; (d-f) ISESS improvement over iterations 1, 3 and 10.

- A novel strategy that alternates over time object-based superpixel segmentation and superpixel-based saliency estimation (cycling operations never exploited) for saliency estimation.

Section 2 presents related work and ISESS is detailed in Section 3. Experiments with in-depth analysis of their results are presented in Section 4. We state conclusions and discuss future work in Section 5.

2 RELATED WORK

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. Early bottom-up strategies include modeling saliency according to global contrast by comparing all possible pairs of image patches (Cheng et al., 2014), for instance. 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.

2.1.2 Image-Intrinsic Approaches

Apart from superpixel similarity methods that use global contrast (Jiang et al., 2013), bottom-up approaches (image-intrinsic) require a strategy to select superpixels that represent candidate foreground and background regions — namely, queries. The three

most common query selection strategies are based on *sparsity*, *backgroundness*, or *objectness*. Methods based on low-rank (LR) matrix recovery (Peng et al., 2016) use *sparsity*, and assume an image can be divided into a highly redundant background-likely low-rank matrix and a salient sparse sensory matrix. Superpixel-graph-based methods (Yang et al., 2013) 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. For example, one may use all four image borders and define a region to be salient if it is consistently highlighted (Yang et al., 2013). 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 (Yang et al., 2013).

A major drawback in all methods above is their reliance on heuristics or domain-specific prior information. We propose combining saliency estimations from a given object-inspired deep SOD model and the proposed superpixel-based image-intrinsic model to avoid the heuristic-based approach.

2.2 Saliency Map Aggregation

Multiple SOD methods have been combined to improve incomplete saliency maps (Singh and Kumar, 2020; Li et al., 2018). Unsupervised methods may use min-max operations over the estimated foreground and background regions (Singh and Kumar, 2020). Supervised methods often learn regressors that combine multiple saliency maps into a single saliency score using bootstrap learning (Li et al., 2018).

However, such aggregation strategies create a significant processing overhead due to the execution of multiple SOD methods, and their quality relies on most saliency maps agreeing with respect to the salient object. To our knowledge, our approach is the first that improves incomplete saliency maps without requiring aggregation of multiple SOD methods.

2.3 Deep-Learning Methods

2.3.1 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 (Wang et al., 2021b), ear-

lier attempts used MLP-based approaches (He et al., 2015; Liu and Han, 2016), 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.

2.3.2 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 (Huang et al., 2017), and Resnet (He et al., 2016)) 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. For instance, in (Zhang et al., 2017) a generic framework is proposed for integrating multiple feature-scales into multiple resolutions.

Instead of upsampling directly from the low-resolution deep layers back to the full-size input layer, several methods have adopted the encoder-decoder structure to gradually re-scale the low-resolution features (Liu et al., 2019; Qin et al., 2020; Qin et al., 2019; Miao et al., 2021). Each method proposes a different strategy to exploit the multi-scale features in the decoder. The authors in (Liu et al., 2019) propose a global guidance module based on pyramid pooling to explicitly deliver object-location information in all feature maps of the decoder. In recent work, the **Auto- Multi-Scale Fusion Network (Auto-MSFNet)** is proposed (Miao et al., 2021), using Network Architecture Search (NAS) to create 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.

Additionally, boundary information can be used to achieve better object delineation. Early on, a new loss was proposed (Luo et al., 2017) to heavily penalize boundary regions for training an adaptation of VGG-16; an encoder-decoder network is used in (Zhao et al., 2019) to exploit multi-scale features and assist an edge-feature extraction that guides the delineation for the final saliency estimation. Instead of explicitly or exclusively using boundary information, the **Boundary-Aware Segmentation Network (BASNET)** (Qin et al., 2019) uses a hybrid loss to rep-

resent differences between the ground-truth and the predictions in a pixel, patch, and map leve. 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, the **U²Net** (Qin et al., 2020) was proposed as a nested U-shaped network and 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** (Qin et al., 2020), which has the novelty of not requiring a pre-trained backbone; **BASNet** (Qin et al., 2019), that provides a higher delineation precision due to its boundary-awareness; and **Auto-MSFNet** (Miao et al., 2021), which uses a framework to learn the fusion strategies that humans commonly define.

3 METHODOLOGY

ISESS aims to improve the results of complex deep-learning-based SOD models by adding image-intrinsic information. For such, ISESS combines alternate executions of object-based superpixel segmentation and superpixel-based saliency estimation. The multiple executions of both operations generate enhanced 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 ISESS is a superpixel-based saliency enhancer. For such, we use

the Object-based Iterative Spanning Forest algorithm (OISF) (Belém et al., 2019).

In short, OISF¹ 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 (Falcão et al., 2004) 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 (Belém et al., 2019), 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: α to control the importance of superpixel regularity; β to control the importance of boundary adherence; γ 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 (Vargas-Muñoz et al., 2019), while γ 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 inside salient areas to be $n_{os} = n_s * n_c$, where n_c is the number of connected components found by Otsu binarization of the saliency map resultant from the last iteration, and n_s is a parameter. This way, the number of superpixels depends on how many salient objects the oversegmentation is trying to represent.

Within this paper, let the saliency score of either a pixel or a superpixel be represented as $s(\cdot)$. At the end of each iteration, the saliency score of a superpixel is given by the previously computed pixel-wise saliency map and is taken as the mean saliency score of all pixels inside the superpixel. We do not reuse the saliency of a superpixel directly because the superpixel segmentation changes at each iteration.

Let \mathcal{S} be a superset of all superpixels, $\mathbf{F} : \mathcal{S} \rightarrow \mathbb{R}^m$ map to every superpixel the mean feature vector of all its pixels — in this paper, $m = 3$ because we use the pixel colors in the CIELab color-space as features. We define two query lists, Q_F for foreground super-

¹The OISF method is available at <https://github.com/LIDS-UNICAMP/OISF>.

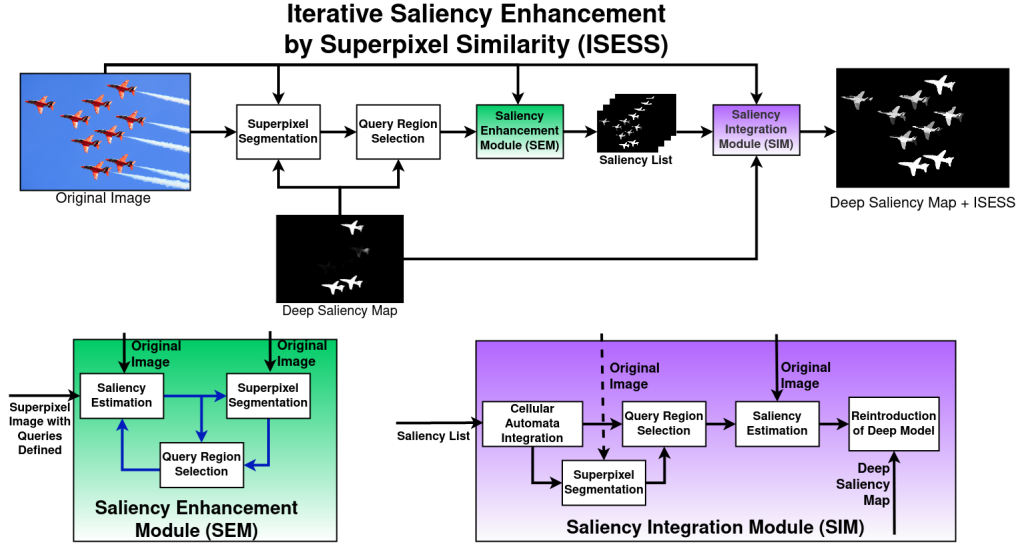


Figure 3: Proposed method. The saliency/superpixel enhancement loop is inside the Saliency Enhancement Module (SEM).

pixels, and Q_B for background ones, where $Q_F \cup Q_B = \mathcal{S}$. Take ψ to be the Otsu threshold of the previously computed saliency map, then, the query lists are defined as follows: $S \in Q_F \leftrightarrow s(S) \geq \psi$, and similarly, $S \in Q_B \leftrightarrow s(S) < \psi$.

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

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

where $\mathbf{d}(\mathbf{F}(S), \mathbf{F}(R))$ is the euclidean distance between the superpixel's mean features, 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:

$$\mathbf{s}_f(S) = \max_{\forall R \in Q_F} \{\mathbf{sim}(S, R)\} \quad (2)$$

However, when $S = S_f \in Q_F$ 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: $\mathbf{s}_f(S_f) = \max_{\forall Q \notin Q_F} \{\mathbf{s}_f(Q)\}$. By doing so, we allow for a forced exploration of possible foreground regions at each iteration.



Figure 4: Image with complex background with little relevant information from background queries. (a) original image; (b) U2Net map; (c) background-based saliency with $d_{0.5} < 0.1$; (d) background-based saliency from last iteration which will substitute (c).

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

$$\mathbf{s}_b(S) = 1 - \frac{\sum_{\forall R \in Q_B, S \neq R} \mathbf{sim}(S, R)}{|Q_B|}. \quad (3)$$

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 $d_{0.5} = \frac{1}{|\mathcal{S}|} \sum_{\forall S \in \mathcal{S}} (\mathbf{s}_b(S) - 0.5)^2 |S|$. For maps with $d_{0.5} < 0.1$, we set the background saliency score to be the saliency of the previous iteration $\mathbf{s}_b(S) = \mathbf{s}^{i-1}(S)$, where \mathbf{s}^{i-1} is the final saliency computed in the last iteration and \mathbf{s}^0 is the initial deep-learning saliency.

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

$$\mathbf{s}^i(S) = \mathbf{s}_f(S) * \mathbf{s}_b(S). \quad (4)$$

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 3.2

3.2 Saliency Integration Module

To integrate the saliency maps created throughout the iterations, we use a traditional unsupervised saliency integrator method (Qin et al., 2015) 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. Note that we are using saliency integration to combine the multiple outputs of our iterative approach, not aggregating different saliency methods.

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 \mathcal{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: $\mathbf{I}^t(p) = \frac{s^t(p)}{1-s^t(p)}$ so that the value can be updated by performing subsequent sums. The value changes according to the rule:

$$s^{t+1}(p) = \mathbf{I}^t(p) + \sum_{\forall q \in \mathcal{C}_p} \lambda \delta^t(q), \quad (5)$$

where λ is a constant that regulates the update rate, and $\delta^t(\cdot) \in \{-1, 1\}$ is a signal function that makes the saliency either increase or decrease at each update depending on whether $s^t(q)$ 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 λ ; 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 4.3.2).

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:

$$s_d(S) = \frac{1}{|S|} \sum_{\forall p \in S} s_0(p), \quad (6)$$

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

$$s_f(p) = \max\{s_d(p), s^i(p)\}. \quad (7)$$

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. The datasets selected with a brief description of their characteristics, as provided by their authors, are: **DUT_OMRON** (Yang et al., 2013), which is composed of 5168 images containing one or two complex foreground objects in a relatively cluttered background; **HKU-IS**, containing 4447 images with one or multiple low contrast foreground objects each; **ECSSD**, consisting of 1000 images with mostly one large salient object per image in a complex background; **ICoSeg**, which contains 643 images usually with several foreground objects each; and **SED2**, 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 **ISESS**, we randomly selected 50 images from each dataset, totalizing 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 , the number of foreground seeds per component n_s , and the number of

OISF iterations i_o . Taking the order U²Net, BASNET, and MSFNet, the parameters were respectively, $i = \{12, 9, 12\}$, $n = \{2500, 200, 2500\}$, $n_s = \{10, 30, 30\}$, 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 γ is related to the superpixel segmentation algorithm, σ^2 is the gaussian variance for the graph node similarity, λ 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_β^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_ψ^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. ISESS can sometimes score a superpixel with close-to-zero saliency values. These small saliencies are not visually perceived by the observer but can significantly impair the presented metrics (Figure 5). In the image presented, the bottom-left quadrants (with a little more than a fourth of the total image weight) had a superpixel slightly salient (a value below 2% of the maximum saliency) by ISESS, which caused 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 Execution Time

ISESS is yet to be optimized for time-sensitive applications. In its current implementation, considering an average among all datasets, ISESS takes 1.1s to run the entire pipeline with one iteration and 686ms for each subsequent one. Out of the time taken, considering the proposed parameters, around 71% is spent (re)computing superpixels, so improving or substituting the superpixel algorithm would provide the highest benefit for reducing execution time.

We are still working on its optimization with parallel computing. However, since ISESS is not using multi-threading, one can simultaneously execute as

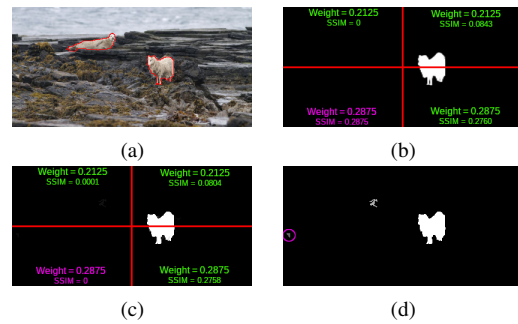


Figure 5: Example of drastic S_m difference between two similar saliency maps. (a) original image with ground-truth objects delineated in red; (b-c) BASNET and BASNET+ISESS saliency maps, respectively, divided into quadrants, with each quadrant weight and Structural Similarity values; (d) histogram-manipulated (c) to show the detail that caused the large difference in similarity value.

many images as threads available.

4.3 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.

4.3.1 Ablation on Reintroducing the Deep Saliency Model

The goal of ISESS 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, ISESS mainly contributed to creating sharper edges.

The downside of re-introducing the deep model is that it reduces the influence of ISESS on wrongfully salient regions. Take Figure 7 as an example: The deep-model wrongfully highlighted part of the airplanes' smoke trail; ISESS removed it entirely; the re-introduction of the deep-model also brought back the error. However, the smoke trail's saliency was reduced thanks to combining the deep model with the

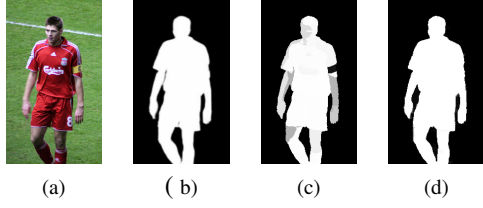


Figure 6: Good example of the re-introduction of the deep-model saliency. (a) original image; (b) U²Net saliency map; (c-d) ISESS without and with the reintroduction of the deep model, respectively.

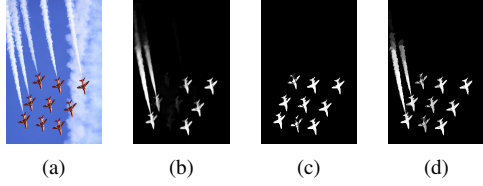


Figure 7: Bad example of the re-introduction of the deep-model. (a) original image; (b) U²Net; (c-d) ISESS without and with the reintroduction of the deep model.

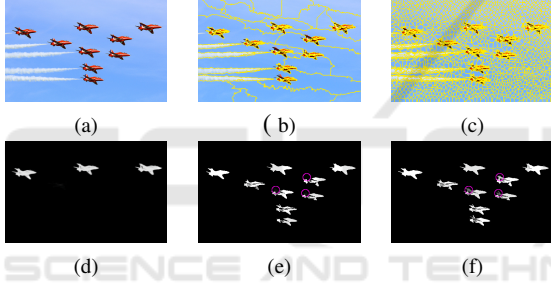


Figure 8: Bad object representation due to the superpixel decrease. (a-c) original image and superpixel segmentations; (d-f) U²Net, U²Net+ISESS considering the maps above. Note part of the object missing (pink circle) due to superpixel leakage.

superpixel segmentation.

4.3.2 Ablation on Superpixels in 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.

4.4 Quantitative Comparisons

Table 1 shows that the proposed method can consistently improve $maxF_{\beta}$, F_{β}^w , MAE and E_{ψ}^m for U²Net

Table 1: Quantitative comparison between **enhanced** and non-enhanced maps in terms of $S_m \uparrow$, $maxF_{\beta} \uparrow$, $F_{\beta}^w \uparrow$, $MAE \downarrow$, and $E_{\psi}^m \uparrow$. **BLUE** indicates the best result between the enhanced and non-enhanced maps.

Sed2	$S_m \uparrow$	$maxF_{\beta} \uparrow$	$F_{\beta}^w \uparrow$	$MAE \downarrow$	$E_{\psi}^m \uparrow$
U ² Net	0.8193	0.8140	0.7704	0.0558	0.8529
U²Net+ISESS	0.8527	0.8404	0.8087	0.0483	0.8816
BASNET	0.8558	0.8565	0.8119	0.0514	0.8881
BASNET+ISESS	0.8643	0.8647	0.8375	0.0470	0.9037
Auto-MSFNet	0.8374	0.8333	0.8032	0.0544	0.8800
Auto-MSFNet+ISESS	0.8378	0.8518	0.8182	0.0524	0.8881
ECSSD	$S_m \uparrow$	$maxF_{\beta} \uparrow$	$F_{\beta}^w \uparrow$	$MAE \downarrow$	$E_{\psi}^m \uparrow$
U ² Net	0.9209	0.9286	0.8993	0.0370	0.9407
U²Net+ISESS	0.9182	0.9297	0.9099	0.0339	0.9467
BASNET	0.9142	0.9262	0.8992	0.0384	0.9414
BASNET+ISESS	0.9123	0.9267	0.9075	0.0355	0.9468
Auto-MSFNet	0.9067	0.9229	0.9032	0.0387	0.9427
Auto-MSFNet+ISESS	0.9069	0.9203	0.9026	0.0389	0.9411
DUT_OMRON	$S_m \uparrow$	$maxF_{\beta} \uparrow$	$F_{\beta}^w \uparrow$	$MAE \downarrow$	$E_{\psi}^m \uparrow$
U ² Net	0.8444	0.7868	0.7491	0.0541	0.8627
U²Net+ISESS	0.8418	0.7895	0.7647	0.0518	0.8736
BASNET	0.8362	0.7750	0.7465	0.0558	0.8607
BASNET+ISESS	0.8335	0.7758	0.7556	0.0548	0.8686
Auto-MSFNet	0.8321	0.7748	0.7536	0.049	0.8692
Auto-MSFNet+ISESS	0.8323	0.7730	0.7533	0.0491	0.8680
ICoSeg	$S_m \uparrow$	$maxF_{\beta} \uparrow$	$F_{\beta}^w \uparrow$	$MAE \downarrow$	$E_{\psi}^m \uparrow$
U ² Net	0.8727	0.8585	0.8210	0.0449	0.8957
U²Net+ISESS	0.8869	0.8769	0.8548	0.0404	0.9165
BASNET	0.8702	0.8578	0.8217	0.0476	0.8972
BASNET+ISESS	0.8784	0.8630	0.8397	0.0436	0.9057
Auto-MSFNet	0.8662	0.8515	0.8256	0.0425	0.9083
Auto-MSFNet+ISESS	0.8703	0.8544	0.8332	0.0417	0.9095
HKU-IS	$S_m \uparrow$	$maxF_{\beta} \uparrow$	$F_{\beta}^w \uparrow$	$MAE \downarrow$	$E_{\psi}^m \uparrow$
U ² Net	0.9183	0.9291	0.8950	0.0311	0.9453
U²Net+ISESS	0.9174	0.9292	0.9085	0.0274	0.9528
BASNET	0.9123	0.9264	0.8977	0.0308	0.9476
BASNET+ISESS	0.9110	0.9262	0.9075	0.0278	0.9536
Auto-MSFNet	0.9148	0.9297	0.9144	0.0255	0.9617
Auto-MSFNet+ISESS	0.9156	0.9270	0.9146	0.0255	0.9603

and BASNET in all five datasets, and achieve similar 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 ISESS can considerably improve the saliency representation of similar objects.

The precision-recall curves show a clear advantage of ISESS-enhanced maps over the non-enhanced ones on datasets mainly consisting of multiple objects (sed2, and icoseg). There is a breakpoint where ISESS loses precision more rapidly than non-enhanced maps on the other three datasets. We attribute the rapid precision loss to ISESS highlighting more parts of non-salient objects deemed partially salient by the deep models (Figure 9). By looking at the $maxF_{\beta}$, we see that by segmenting ISESS 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_m . 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.

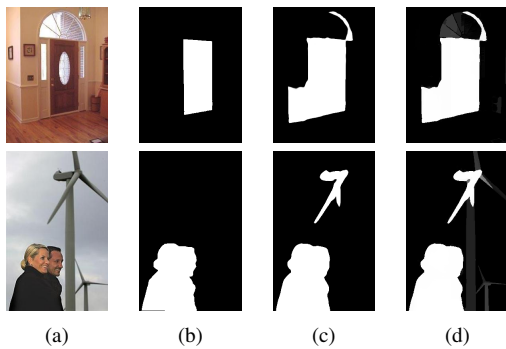


Figure 9: ISESS improving the saliency of wrogfully salient object partially highlighted by Auto-MSFNet. (a) original image; (b) ground-truth; (c) saliency map by Auto-MSFNet; (d) (c) enhanced by the proposed ISESS.

Regarding Auto-MSFNet, apart from the ICoSeg and SED2, ISESS 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, ISESS seems to over-trust the deep model on non-salient regions.

4.5 Qualitative Comparisons with Non-Enhanced Maps

To visually evaluate the benefits of enhancing a saliency map using ISESS, we compiled images where ISESS improved different aspects of object representation (Figure 11). The image in the first row contains minimal salient values in a small part of the desired object (columns (c) and (g)). Even though the initial map provides little trust (*i.e.*, small saliency values), ISESS created similar resulting maps using all deep models.

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

The bottom four rows present images where ISESS 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), ISESS properly included a narrow stem on the traffic light, even though narrow structures can be complex during superpixel delin-

ation. 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.

Since ISESS was meant to improve partially salient objects whose parts (superpixels) have similar colors, it cannot deal with some situations. Figure 12 shows examples of fail cases. In the first two columns, the ground-truth object (pyramid) was not detected by the deep method. ISESS cannot improve the saliency of non-detected objects, which makes it dependent on the map's precision. The other two columns (fish) shows another situation where the map of the detected salient object is incomplete. Since the missing parts have colors too distinct from the highlighted ones, ISESS could not complete the saliency map. However, Figure 11d (second and third rows) show that ISESS can recover the map of missing objects when the highlighted parts have similar colors.

5 CONCLUSION

We presented a hybrid model for saliency enhancement that exploits a loop between superpixel segmentation and saliency enhancement for the first time. ISESS delineates superpixels based on object information, as represented by an input saliency map, and improves the input saliency map by computing feature similarity between superpixels and queries. It exploits multiple scales of superpixel representation and integrates the intermediate saliency maps by cellular automata. Experimental results on five public SOD datasets demonstrate that ISESS can consistently improve object representation, especially in the following cases: (a) partially salient objects and (b) images with multiple salient objects with only a few captured by the deep model. Although better superpixel descriptors can be used for superpixel similarity, we focused on a simple color descriptor to illustrate how well deep models can be assisted by image-intrinsic information to improve their results.

We intend to explore ISESS enhanced maps for interactive- and co-segmentation tasks. The goal will be to assist humans when annotating images with fewer interactions, using the highly-trusted human-provided object location to define the precise background and foreground queries, combined with robust deep models to capture finer features. Another characteristic inherent in ISESS 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 seg-

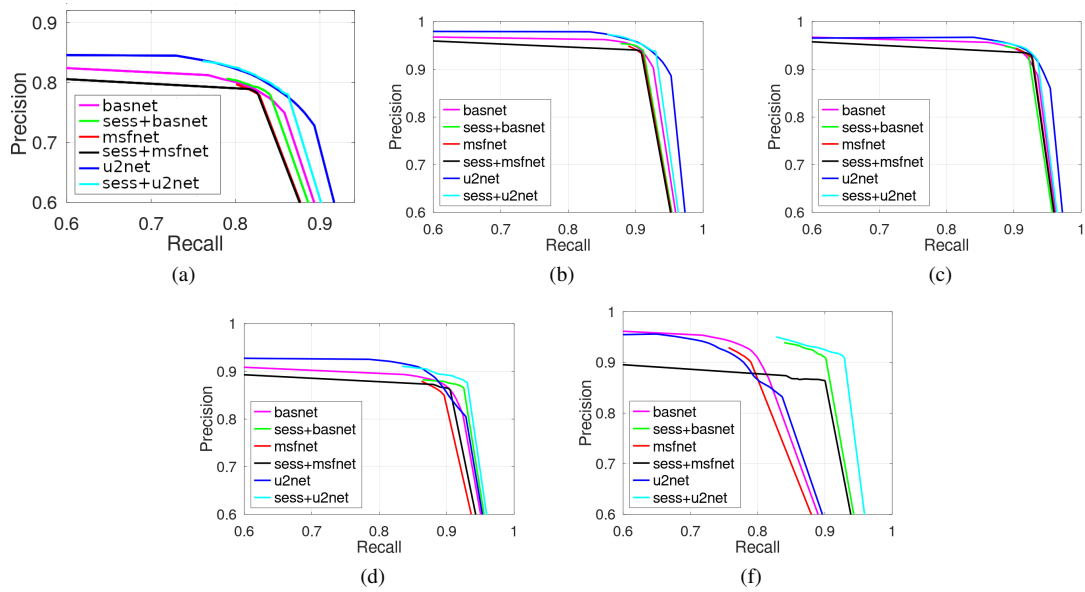


Figure 10: The Precision-Recall curves of all methods with and without enhancement in all datasets. (a) DUT_OMRON; (b) ECSSD; (c) HKU-IS; (d) ICOSEG; (e) SED2.

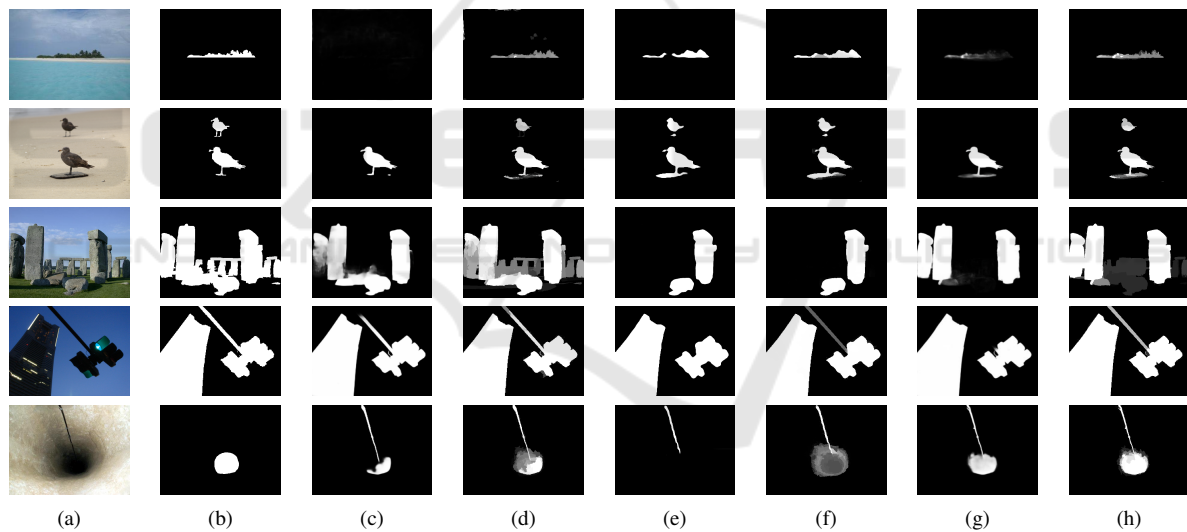


Figure 11: Mural of images with their initial saliency maps and their ISESS enhanced version. (a) original image; (b) ground-truth; (c) BASNET; (d) BASNET+ISESS; (e) Auto-MSFNet; (f) Auto-MSFNet+ISESS; (g) U²Net; (h) U²Net + ISESS

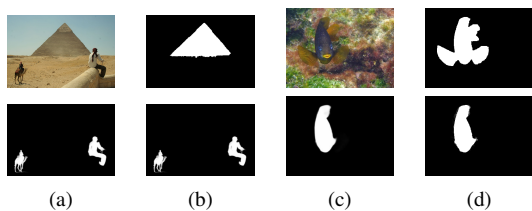


Figure 12: Failed saliency improvements. (a) Original image; (b) ground-truth object; (c) saliency map by U²-Net; and (d) ISESS's output given (c).

mentation and how the final superpixel map can assist interactive object segmentation.

ACKNOWLEDGEMENTS

This work was supported by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) with Finance Code 001, CAPES COFECUB (88887.800167/2022-00), CNPq (303808/2018-7), and FAEPEX.

REFERENCES

- Al-Azawi, M. A. (2021). Saliency-based image retrieval as a refinement to content-based image retrieval. *EL-CVIA: Electronic Letters on Computer Vision and Image Analysis*, 20(1):0001–15.
- ”Belém, F., Guimarães, S. J., and F., Falcão, A. X. (2019). Superpixel segmentation by object-based iterative spanning forest. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 334–341. Springer International Publishing.
- Belém, F., Melo, L., Guimarães, S., and Falcão, A. (2019). The importance of object-based seed sampling for superpixel segmentation. In *Proc. 32nd Conf. Graphics Pattern Images (SIBGRABI)*, pages 108–115. To appear.
- Cheng, M., Mitra, N., Huang, X., Torr, P., and Hu, S. (2014). Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):569–582.
- Falcão, A. X., Stolfi, J., and Lotufo, R. A. (2004). The image foresting transform: Theory, algorithms, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26:19–29.
- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- He, S., Lau, R. W., Liu, W., Huang, Z., and Yang, Q. (2015). Supercnn: A superpixelwise convolutional neural network for salient object detection. *International journal of computer vision*, 115(3):330–344.
- Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.
- Iqbal, E., Niaz, A., Memon, A., Asim, U., and Choi, K. (2020). Saliency-driven active contour model for image segmentation. *IEEE Access*, 8:208978–208991.
- Jiang, P., Ling, H., Yu, J., and Peng, J. (2013). Salient region detection by ufo: Uniqueness, focusness and objectness. In *Proceedings of the IEEE international conference on computer vision*, pages 1976–1983.
- Li, L., Chai, X., Zhao, S., Zheng, S., and Su, S. (2018). Saliency optimization and integration via iterative bootstrap learning. *International Journal of Pattern Recognition and Artificial Intelligence*, 32(09):1859016.
- Liu, J., Hou, Q., Cheng, M., Feng, J., and Jiang, J. (2019). A simple pooling-based design for real-time salient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3917–3926.
- Liu, N. and Han, J. (2016). Dhsnet: Deep hierarchical saliency network for salient object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 678–686.
- Luo, Z., Mishra, A., Achkar, A., Eichel, J., Li, S., and Jodoin, P. (2017). Non-local deep features for salient object detection. In *Proceedings of the IEEE Conference on computer vision and pattern recognition*, pages 6609–6617.
- Miao, Z., Tingwei, L., Yongri, P., ShunYu, Y., and Huchuan, L. (2021). Auto-msfnet: Search multi-scale fusion network for salient object detection. In *ACM Multimedia Conference 2021*, page 667–676.
- Peng, H., Li, B., Ling, H., Hu, W., Xiong, W., and Maybank, S. (2016). Salient object detection via structured matrix decomposition. *IEEE transactions on pattern analysis and machine intelligence*, 39(4):818–832.
- Qin, X., Zhang, Z., Huang, C., Dehghan, M., Zaiane, O. R., and Jagersand, M. (2020). U2-net: Going deeper with nested u-structure for salient object detection. *Pattern Recognition*, 106:107404.
- Qin, X., Zhang, Z., Huang, C., Gao, C., Dehghan, M., and Jagersand, M. (2019). Basnet: Boundary-aware salient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7479–7489.
- Qin, Y., Lu, H., Xu, Y., and Wang, H. (2015). Saliency detection via cellular automata. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 110–119.
- Singh, V. K. and Kumar, N. (2020). Saliency bagging: a novel framework for robust salient object detection. *The Visual Computer*, 36(7):1423–1441.
- Vargas-Muñoz, J. E., Chowdhury, A., Alexandre, E., Galvão, F., Miranda, P., and Falcão, A. (2019). An iterative spanning forest framework for superpixel segmentation. *IEEE Transactions on Image Processing*, 28(7):3477–3489.
- Wang, J., de Melo Joao, L., Falcão, A., Kosinka, J., and Telea, A. (2021a). Focus-and-context skeleton-based image simplification using saliency maps. In *VISIGRAPP (4: VISAPP)*, pages 45–55.
- Wang, W., Lai, Q., Fu, H., Shen, J., Ling, H., and Yang, R. (2021b). Salient object detection in the deep learning era: An in-depth survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Yang, C., Zhang, L., Lu, H., Ruan, X., and Yang, M. (2013). Saliency detection via graph-based manifold ranking. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3166–3173. IEEE.
- Zhang, P., Wang, D., Lu, H., Wang, H., and Ruan, X. (2017). Amulet: Aggregating multi-level convolutional features for salient object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 202–211.
- Zhao, J., Liu, J., Fan, D., Cao, Y., Yang, J., and Cheng, M. (2019). Egnnet: Edge guidance network for salient object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8779–8788.