Boundary-Aware Segmentation Network for Mobile and Web Applications

Xuebin Qin, Deng-Ping Fan, Chenyang Huang, Cyril Diagne, Zichen Zhang,
Adrià Cabeza Sant'Anna, Albert Suàrez, Martin Jagersand, and Ling Shao, *Fellow, IEEE*

Abstract—Although deep models have greatly improved the accuracy and robustness of image segmentation, obtaining segmentation results with highly accurate boundaries and fine structures is still a challenging problem. In this paper, we propose a simple yet powerful Boundary-Aware Segmentation Network (**BASNet**), which comprises a predict-refine architecture and a hybrid loss, for highly accurate image segmentation. The predict-refine architecture consists of a densely supervised encoder-decoder network and a residual refinement module, which are respectively used to predict and refine a segmentation probability map. The hybrid loss is a combination of the binary cross entropy, structural similarity and intersection-over-union losses, which guide the network to learn three-level (*i.e.*, pixel-, patch- and map- level) hierarchy representations. We evaluate our BASNet on two reverse tasks including salient object segmentation, camouflaged object segmentation, showing that it achieves very competitive performance with sharp segmentation boundaries. Importantly, BASNet runs at over 70 fps on a single GPU which benefits many potential real applications. Based on BASNet, we further developed two (close to) commercial applications: **AR COPY & PASTE**, in which BASNet is integrated with augmented reality for "COPYING" and "PASTING" real-world objects, and **OBJECT CUT**, which is a web-based tool for automatic object background removal. Both applications have already drawn huge amount of attention and have important real-world impacts. The code and two applications will be publicly available at: https://github.com/NathanUA/BASNet.

Index Terms—Boundary-aware segmentation, predict-refine architecture, salient object, camouflaged object

1 Introduction

Image segmentation has been studied over many decades using conventional methods, and in the past few years using deep learning. Several different conventional approaches, such as interactive methods, active contour (level-set methods), graph-theoretic approaches, perceptual grouping methods and so on, have been studied for image segmentation over the past decades. Yet automatic methods fail where boundaries are complex. Interactive methods let humans resolve the complex cases. Interactive methods, [99], [114], [8], [91], are usually able to produce accurate and robust results, but with significant time costs. Active contour [82], [48], [1], [118], [6], [7], [124], graph-theoretic [120], [16], [101], [100], [112] and perceptual grouping [47], [20], [73], [2], [21], [97], [92], [88], methods require almost no human interventions so that they are faster than interactive methods. However, they are relatively less robust.

In recent years, to achieve accurate, robust and fast performance, many deep learning models [77] have been developed for image segmentation. Semantic image segmentation [70], [40] is one of the most popular topics, which aims at labeling every pixel in an image, with one of the several predefined class labels. It has been widely used in many applications, such as scene understanding [65], [145], autonomous driving [28], [15], *etc*. The targets in these applications are usually large in size, so most existing methods focus on achieving robustness performance with high regional accuracy. Less attention has been paid to the high spatial accuracy of boundaries and fine structures. However, many other applications, *e.g.* image segmentation/editing [90], [46], [31], [91] and manipulation [45], [76], visual tracking [57], [89], [93], vision guided robot hand manipulation [88] and so on, require highly accurate object boundaries and fine structures.

There are two main challenges in accurate image segmentation: **Firstly**, large-scale features play important roles in classifying pixels since they can provide more semantic information compared with local features. However, large-scale features are usually obtained from deep low-resolution feature maps or by large size kernels and the spatial resolution is sacrificed. Simple upsampling of the low-resolution feature maps to high resolution is not able to recover the fine structures [70]. Thus, many encoder-decoder architectures [98] have been developed for segmenting edges or thin structures. Their skip connections and gradual upsampling operations play important roles in recovering the high resolution probability maps. Additionally, different cascaded

- *Xuebin Qin is with the Department of Computing Science, University of Alberta, Edmonton, AB, Canada, T6G 2R3.(Email: xuebin@ualberta.ca)*
- *Deng-Ping Fan and Ling Shao are with the Inception Institute of Artificial Intelligence (IIAI), Abu Dhabi, UAE. (Email: dengpfan@gmail.com; ling.shao@ieee.org)*
- *Chenyang Huang is with the Department of Computing Science, University of Alberta, Edmonton, AB, Canada. (Email: chuang8@ualberta.ca)*
- *Cyril Diagne is with Init ML (Email: cyril@initml.co)*
- *Zichen Zhang is with the Department of Computing Science, University of Alberta, Edmonton, AB, Canada. (Email: vincent.zhang@ualberta.ca)*
- *Adrià Cabeza Sant'Anna is with the Department of Computer Science, Polytechnic University of Catalonia, BarcelonaTech, Barcelona, Spain. (Email: adriacabezasantanna@gmail.com)*
- *Albert Suàrez is with the Department of Software Engineering, Polytechnic University of Catalonia, BarcelonaTech, Barcelona, Spain. (Email: alsumo95@gmail.com)*
- *Martin Jagersand is with the Department of Computing Science, University of Alberta, Edmonton, AB, Canada. (Email: mj7@ualberta.ca)*
- *A preliminary version of this work has appeared in CVPR [95].*
- *Corresponding author: Deng-Ping Fan.*

Manuscript submitted December 6, 2020; revised xx xx, xx.

or iterative architectures [125], [115], [18], [113] have been introduced to further improve the segmentation accuracy by gradually refining the coarse predictions, which sometimes leads to complicated network architectures and computational bottleneck. **Secondly**, most of the image segmentation models use cross entropy (CE) loss to supervise the training process. CE loss usually gives greater penalties on these seriously erroneous predictions (*e.g.* predict "1" as "0.1" or predict "0" as "0.9"). Therefore, deep models trained with CE loss compromise andes prefer to predict "hard" samples with a non-committed "0.5". In image segmentation tasks, the boundary pixels of targets are usually the hard samples, so this will lead to blurry boundaries in predicted segmentation probability maps. Other losses, such as intersection-over-union (IoU) loss [96], [75], [80], F-score loss [141] and Dice-score loss [27], have also been introduced to image segmentation tasks for handling biased training sets. These are sometimes able to achieve higher (regional) evaluation metrics, *e.g.* IoU, F-score, since their optimization targets are consistent with these metrics. However, they are not specifically designed for capturing fine structures and often produce "biased" results, which tend to emphasize the large structures while neglecting fine details.

To address the above issues, we propose a novel but simple **B**oundary-**A**ware **S**egmentation **Net**work (**BASNet**), which consists of a predict-refine network and a hybrid loss, for highly accurate image segmentation. The predict-refine architecture is designed to predict and refine the predicted probability maps sequentially. It consists of a U-Net-like [98] deeply supervised [56], [123] "heavy" encoder-decoder network and a residual refinement module with "light" encoder-decoder structure. The "heavy" encoder-decoder network transfers the input image to a segmentation probability map, while the "light" refinement module refines the predicted map by learning the residuals between the coarse map and ground truth (GT). In contrast to [85], [43], [18], which iteratively use refinement modules on saliency predictions or intermediate feature maps at multiple scales, our refinement module is used only once on the original scale of the segmentation maps. Overall, our predict-refine architecture is concise and easy to use. The hybrid loss combines the binary cross entropy (BCE) [17], structural similarity (SSIM) [117] and IoU losses [75], to supervise the training process in a three-level hierarchy: pixel-, patch- and map- level. Instead of using *explicit* boundary loss (NLDF+ [72], C2S [61], BANet[104]), we *implicitly* inject the goal of accurate boundary prediction in the hybrid loss, contemplating that it may help reduce spurious error from cross propagating the information learned from the boundaries and other regions on the image (see Fig. 1).

In addition to proposing novel segmentation techniques, developing novel segmentation applications also plays a very important role in advancing the segmentation field. Therefore, we developed two novel BASNet-based applications: **AR COPY & PASTE**[1] and **OBJECT CUT**[2]. AR COPY & PASTE is a mobile app built upon our BASNet model and the Augmented Reality techniques. By using cellphones, it provides a novel interactive user experience where users

1. https://clipdrop.co/
2. https://objectcut.com/

(a) Image (b) GT (c) BASNet

Fig. 1. Sample results of our BASNet on salient object detection (Top) and camouflaged object detection (Bottom).

can "COPY" the real-world targets and "PASTE" them into desktop software . Specifically, AR COPY & PASTE allows users to take a photo of an object using a mobile device. Then the background removed object returned by our remote BASNet server will be shown in the camera view. In this view, the "COPIED" object is overlapped with real scene video stream. Users can move and target the mobile camera at a specific position on the desktop screen. Then tapping the screen of the mobile device will trigger the "PASTE" operation, which transmits the object from the mobile device to the software opened in the desktop. Meanwhile, OBJECT CUT provides a web-based service for automatic image background removal based on our BASNet model. An image can be uploaded from a local machine or through an URL. This application greatly facilitates the background removal for users who have no image editing experience or software. The main contributions can be summarized as:

- We develop a novel boundary-aware image segmentation network, BASNet, which consists of a deeply supervised encoder-decoder and a residual refinement module, and a novel hybrid loss that fuses BCE, SSIM, and IoU to supervise the training process of accurate image segmentation on three levels: pixel-level, patch-level and map-level.
- We conduct thorough evaluations of the proposed method including a comparison with 25 state-of-the-art (SOTA) methods on six widely used public salient object segmentation datasets, a comparison with 16 models on the SOC (Salient Object in Clutter) dataset and a comparison with 13 camouflaged object detection (COD) models on three public COD datasets. BASNet achieves very competitive performance in terms of regional evaluation metrics, while outperforms other models in terms of boundary evaluation metrics.
- We develop two (close to) commercial applications, AR COPY & PASTE and OBJECT CUT, based on our BASNet. These two applications further demonstrate the simplicity, effectiveness and efficiency of our model.

Compared with the CVPR version [95] of this work, the following extensions are made. First, deeper theoretical explanations of the hybrid loss design are added. Second, more comprehensive and thorough experiments on different datasets, including salient objects in clutter (SOC) and COD, are included. Third, two (close to) commercial applications, AR COPY & PASTE and OBJECT CUT, are developed.

2 Related Works

2.1 Traditional Image Segmentation Approaches

Watershed [108], graph cut [52], [53], active contour [82], perceptual grouping [92] as well as the interactive methods based on these approaches mainly rely on well-designed handcrafted features, objective functions and optimization algorithms. Watershed and graph cut approaches segment images based on the regional pixel similarities, so they are less effective in segmenting very fine structures and achieving smooth and accurate segmentation boundaries. Active contour and perceptual grouping methods can be considered as boundary based approaches. Active contour methods represent the 2D segmentation contour by the level set of a 3D function. Instead of directly evolving the 2D contour, this group of approaches evolves the 3D function to find the optimal segmentation contour, which avoids complicated 2D contour splitting and merging issues. Perceptual grouping methods segment images by selecting and grouping subsets of the detected edge fragments or line segments from given images to formulate closed or open contours of the targets to be segmented. However, although these methods are able to produce relatively accurate boundaries, they are very sensitive to noise and local minima, which usually leads to less robust and unreliable performance.

2.2 Patch-wise Deep Models

To improve the robustness and accuracy, deep learning methods have been widely introduced to image segmentation [87]. Early deep methods use existing image classification networks as feature extractors and formulate the image segmentation tasks as patch-wise image pixel (superpixel) [58], [69], [109], [142], [60] classification problems. These models greatly improve the segmentation robustness in some tasks due to the strong fitting capability of deep neural networks. However, they are still not able to produce high spatial accuracy, let alone segmenting fine structures. The main reason is probably that the pixels in patch-wise models are classified independently based on the local features inside each patch and larger-scale spatial contexts are not used.

2.3 FCN and its Variants

With the development of fully convolutional network (FCN) [70], deep convolutional neural networks have become a standard solution for image segmentation problems. Large number of deep convolutional models [77] have been proposed for image segmentation. FCN adapts classification backbones, such as VGG [102], GoogleNet [105], ResNet [35] and DenseNet [40], by discarding the fully connected layers and directly upsampling the output features of certain convolutional layers with specific scales to build a fully convolutional image segmentation model. However, the direct upsampling from low resolution fails in capturing accurate structures. Therefore, The DeepLab family [10], [11], [12] replaces the pooling operations by atrous convolutions to avoid degrading the feature map resolution. Besides, they also introduce a densely connected Conditional Random Field (CRF) to improve the segmentation results. However, applying atrous convolutions on high-resolution maps leads to larger memory costs and CRF usually yields noisy segmentation boundaries. Holistically Edge Detection (HED) [123], RCF [29] and CASENet [130] are proposed to directly segment edges by making full use of the features from both the shallow and deep stages of the image classification backbones. Besides, many variants [59], [50], [37] of FCN have been proposed for salient object detection (binary-class image segmentation) [116]. Most of these works are focusing on either developing novel multi-scale feature aggregation strategies or designing new multi-scale feature extraction modules. Zhang *et al.* (Amulet) [136] developed a generic framework for aggregating multi-level convolutional features of the VGG backbone. Inspired by HED [123], Hou *et al.* (DSS+) [36] introduced short connections to the skip-layer structures of HED to better use the deep layer features. Chen *et al.* (RAS) [13] developed a reverse attention model to iteratively refine the side-outputs from a HED-like architecture. Zhang *et al.* (LFR) [135] designed a symmetrical fully convolutional network which takes images and their reflection as inputs to learn the saliency features from the complementary input spaces. Instead of passing the information with single direction (deep to shallow or shallow to deep), Zhang *et al.* (BMPM) [133] proposed to have the information passed between the shallow and deep layers by a controlled bi-directional passing module.

2.4 Encoder-decoder Architectures

Rather than directly upsampling features from deep layers of the backbones, SegNet [3] and U-Net [98] employ encoder-decoder like structures to gradually up-sample the deep low-resolution feature maps. Combined with skip connections, they are able to recover more details. One of the main characteristics of these models is the symmetrical downsampling and upsampling operations. To reduce the checkerboard artifacts in the prediction, Zhang *et al.* (UCF) [137] reformulated the dropout and developed a hybrid module for the upsampling operation. To better use the features extracted by backbones, Liu *et al.* (PoolNet) [66] constructed the decoder part using their newly developed feature aggregation, pyramid pooling and global guidance modules. In addition, stacked HourglassNet [81], CU-UNet [106], UNet++ [146] and U^2-Net [94] further explore diverse ways of improving the encoder-decoder architectures by cascaded or nested stacking.

2.5 Deep Recurrent Models

Recurrent techniques have been widely applied in image segmentation models. Kuen *et al.* [51] proposed to achieve the completely segmented results by sequentially segmenting image sub-regions using a recurrent framework. Liu *et al.* (PiCANetR) [68] deployed a bidirectional LSTM along the row and column of the feature maps respectively to generate the pixel-wise attention maps for salient object detection. Hu *et al.* (SAC-Net) [38] developed similar strategies to [68] to capture spatial attenuation context for image segmentation. Zhang *et al.* (PAGRN) [138] proposed to transfers global information from deep to shallower layers via a multi-path recurrent connection. Wang *et al.* (RFCN) [111] built a cascaded network by stacking multiple encoder-decoders

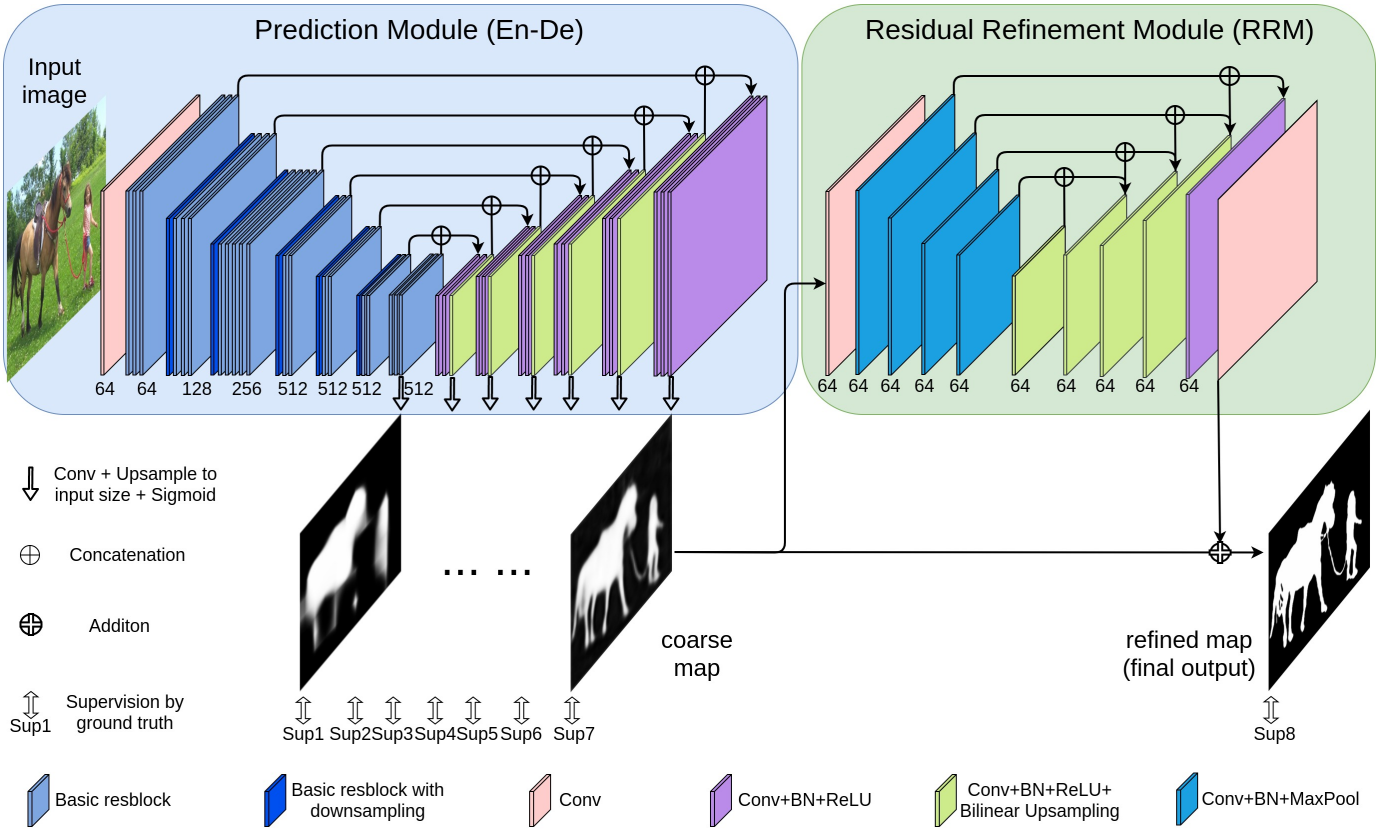


Fig. 2. Architecture of the proposed boundary-aware segmentation network: BASNet. See § 3 for details.

to recurrently correcting the prediction errors of the previous stages. Instead of iteratively refining the segmentation results [111], Hu *et al.* (RADF+) [39] recurrently aggregated and refined multi-layer deep features to achieve accurate segmentation results. However, due to the serial connections between each recurrent step, models using the "recurrent" techniques are relatively less efficient in terms of time costs.

2.6 Deep Coarse-to-Fine Models

This group of models aims at improving the segmentation results by gradually refining the coarse predictions. Lin *et al.* (RefineNet) [63] developed a multi-path refinement segmentation network, which uses long-range residual connections to exploit the information along the down-sampling process. Liu *et al.* (DHSNet) [67] proposed a hierarchical recurrent convolutional neural network (HRCNN), which hierarchically and progressively refines the segmentation results in a coarse-to-fine manner. Wang *et al.* (SRM) [113] developed a multi-stage framework for segmentation map refinement, in which each stage takes the input image and the segmentation maps (lower resolution) from the last stage to produce higher-resolution results. Deng *et al.* (R^3Net+) [18] proposed to alternatively refine the segmentation results based on the shallow, high-resolution and deep low-resolution feature maps. Wang *et al.* (DGRL) [115] developed a global-to-local framework which first localizes the to-be-segmented targets globally and then refines these targets using a local boundary refinement module. The coarse to fine models reduce the probability of overfitting and show promising improvements in accuracy.

2.7 Boundary-assisted Deep Models

Region and boundaries are mutually determined. Therefore, many models introduce boundary information to assist segmentation. Luo *et al.* (NLDF) [72] proposed to supervise a 4×5 grid structure adapted from VGG-16 by fusing the cross entropy and the boundary IoU inspired by Mumford-Shah [79]. Li *et al.* (C2S) [61] tried to recover the regional saliency segmentation from segmented contours. Su *et al.* (BANet) [104] developed a boundary-aware segmentation network with three separate streams: a boundary localization stream, an interior perception stream and a transition compensation stream for boundary, region and boundary/region transition prediction, respectively. Zhao *et al.* (EGNet) [140] proposed an edge guidance network for salient object segmentation by explicitly modeling and fusing complementary region and boundary information. Most of models in this category explicitly use boundary information as either an additional supervision loss or a assisting prediction stream for inferring the region segments.

In this paper, we propose a simple predict-refine architecture which takes advantage of both the encoder-decoder architecture and the coarse-to-fine strategy. Besides, instead of explicitly using boundary loss or additional boundary prediction streams, we design a simple hybrid loss which implicitly describes the dissimilarity between the segmentation prediction and the ground truth at three levels: pixel-, patch- and map-level. The predict-refine architecture together with the hybrid loss provides a simple yet powerful solution for image segmentation and some close to commercial applications.

3 METHODOLOGY

3.1 Overview

Our BASNet architecture consists of two modules as shown in Fig. 2. The prediction module is a U-Net-like densely supervised Encoder-Decoder network [98], which learns to predict segmentation probability maps from input images. The multi-scale Residual Refinement Module (RRM) refines the resulting map of the prediction module by learning the residuals between the coarse map and the GT.

3.2 Prediction Module

Share same spirit of U-Net [98] and SegNet [3], we design our segmentation prediction module as an encoder-decoder fashion, since this kind of architectures is able to capture high-level global contexts and low-level details at the same time. To reduce over-fitting, the last layer of each decoder stage is supervised by the GT, inspired by HED [123] (see Fig. 2). The encoder has an input convolutional layer and six stages comprised of basic res-blocks. The input convolutional layer and the first four stages are adopted from ResNet-34 [35]. The difference is that our input layer has 64 convolutional filters with a size of 3×3 and stride of 1 rather than a size of 7×7 and stride of 2. Additionally, there is no pooling operation after the input layer. This means that the feature maps before the second stage have the same spatial resolution as the input image. This is different from the original ResNet-34, which has a quarter of the resolution in the first feature map. This adaptation enables the network to obtain higher resolution feature maps in earlier layers, while decreasing the overall receptive fields. To achieve the same receptive field as ResNet-34 [35], we add two more stages after the fourth stage of ResNet-34. Both stages consist of three basic res-blocks with 512 filters after a non-overlapping max pooling layer of size 2. To further capture global information, we add a bridge stage between the encoder and decoder. It consists of three convolutional layers with 512 dilated (dilation=2) [129] 3×3 filters. Each of these convolutional layers is followed by a batch normalization [42] and a ReLU activation function [32].

Our decoder is almost symmetrical to the encoder. Each stage consists of three convolution layers followed by a batch normalization and a ReLU activation function. The input of each stage is the concatenated feature maps of the up-sampled output from its previous stage and its corresponding stage in the encoder. To achieve the side-output maps, the multi-channel output of the bridge stage and each decoder stage is fed to a plain 3×3 convolution layer followed by a bilinear upsampling and a sigmoid function. Therefore, given an input image, our prediction module produces seven segmentation probability maps in the training process. Although every predicted map is up-sampled to the same size with the input image, the last one has the highest accuracy and hence is taken as the final output of the prediction module. This output is passed to the refinement module.

3.3 Residual Refinement Module

Refinement Modules (RMs) [43], [18] are usually designed as a residual block [125] that refines the coarse segmentation

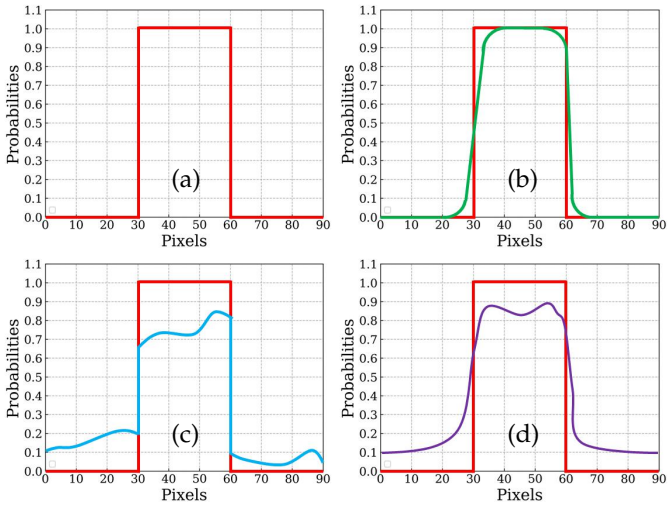


Fig. 3. Illustration of different aspects of coarse prediction in one-dimension: (a) Red: probability plot of GT, (b) Green: probability plot of coarse boundary not aligning with GT, (c) Blue: coarse region having too low probability, (d) Purple: real coarse predictions usually have both (b&c) problems.

maps S_{coarse} by learning the residuals $S_{residual}$ between the coarse maps and the GT, as

$$S_{refined} = S_{coarse} + S_{residual}. \quad (1)$$

Before introducing our refinement module, the term "coarse" has to be determined. Here, "coarse" includes two aspects. One is blurry and noisy boundaries (see the one-dimension illustration in Fig. 3(b)).The other one is the unevenly predicted regional probabilities (see Fig. 3(c)). As shown in Fig. 3(d), real predicted coarse maps usually contain both coarse cases.

The residual refinement module based on local context (RRM_LC), Fig. 4(a), was originally designed for boundary refinement [85]. Since its receptive field is small, Islam *et al.* [43] and Deng *et al.* [18] iteratively or recurrently used it for refining segmentation probability maps on different scales. Wang *et al.* [113] adopted the pyramid pooling module from [34], in which three-scale pyramid pooling features are concatenated. To avoid losing details caused by pooling operations, RRM_MS (Fig. 4(b)) uses convolutions with different kernel sizes and dilations [129], [133] to capture multi-scale contexts. However, these modules are shallow thus hard to capture high-level information for refinement.

To refine inaccuracies in coarse segmentation maps of image reigons and boundaries, we develop a novel residual refinement module. Our RRM employs the residual encoder-decoder architecture, RRM_Ours (see Figs. 2 and 4(c)). Its main architecture is similar but simpler than our prediction module. It contains an input layer, an encoder, a bridge, a decoder and an output layer. Different from the prediction module, both the encoder and decoder have four stages. Each stage only has one convolutional layer. Each layer has 64 filters of size 3×3 followed by a batch normalization and a ReLU activation function. The bridge stage also has a convolutional layer with 64 filters of size 3×3 followed by a batch normalization and ReLU activation. Non-overlapping max pooling is used for downsampling

in the encoder and bilinear interpolation is utilized for upsampling in the decoder. The output of this RM module is used as the final generating segmentation results of BASNet.

3.4 Hybrid Loss

Our training loss is defined as the summation over all outputs:

$$\mathcal{L} = \sum_{k=1}^{K} \alpha_k \ell^{(k)}, \tag{2}$$

where $\ell^{(k)}$ is the loss of the k-th side output, K denotes the total number of the outputs and α_k is the weight of each loss. As described in Sec. 3.2 and Sec. 3.3, our segmentation model is deeply supervised with eight outputs, i.e. $K = 8$, including seven outputs from the prediction module and one output from the refinement module.

To obtain high quality regional segmentation and clear boundaries, we propose to define $\ell^{(k)}$ as a hybrid loss:

$$\ell^{(k)} = \ell^{(k)}_{bce} + \ell^{(k)}_{ssim} + \ell^{(k)}_{iou}, \tag{3}$$

where $\ell^{(k)}_{bce}$, $\ell^{(k)}_{ssim}$, and $\ell^{(k)}_{iou}$ denote BCE loss [17], SSIM loss [117] and IoU loss [75], respectively.

BCE [17] loss is the most widely used loss in binary classification and segmentation. It is defined as:

$$\ell_{bce} = -\sum_{(r,c)} [G(r,c)\log(S(r,c)) + (1-G(r,c))\log(1-S(r,c))], \tag{4}$$

where $G(r,c) \in \{0,1\}$ is the GT label of the pixel (r,c) and $S(r,c)$ is the predicted probability of segmented object.

SSIM [117] was originally devised for image quality assessment. It captures the structural information in an image. Hence, we integrated it into our training loss to learn the structural information of the GT. Let $\mathbf{x} = \{x_j : j = 1, ..., N^2\}$ and $\mathbf{y} = \{y_j : j = 1, ..., N^2\}$ be the pixel values of two corresponding patches (size: $N \times N$) cropped from the predicted probability map S and the binary GT mask G, respectively. The SSIM of $\mathbf{x}$ and $\mathbf{y}$ is defined as:

$$\ell_{ssim} = 1 - \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{5}$$

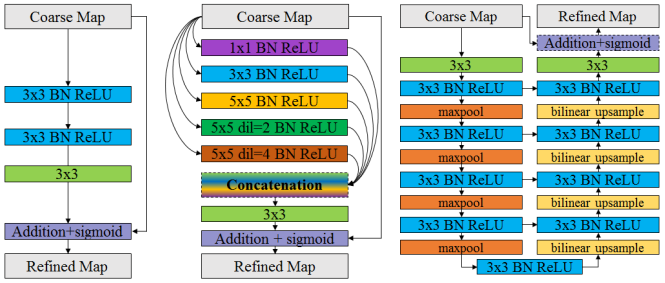
where μ_x, μ_y and σ_x, σ_y are the mean and standard deviations of $\mathbf{x}$ and $\mathbf{y}$ respectively, σ_{xy} is covariance, $C_1 = 0.01^2$ and $C_2 = 0.03^2$ are used to avoid dividing by 0.

IoU was originally proposed for measuring the similarity between two sets [44] and has become a standard evaluation measure for object detection and segmentation. Recently, it has been used as a training loss [96], [75]. To ensure its differentiability, we adopted the IoU loss used in [75]:

$$\ell_{iou} = 1 - \frac{\sum_{r=1}^{H}\sum_{c=1}^{W} S(r,c)G(r,c)}{\sum_{r=1}^{H}\sum_{c=1}^{W} [S(r,c) + G(r,c) - S(r,c)G(r,c)]} \tag{6}$$

where $G(r,c) \in \{0,1\}$ is the GT label of the pixel (r,c) and $S(r,c)$ is the predicted probability of segmented object.

Fig. 5 illustrated the impact of each of the three losses. Fig. 5 (a) and (b) are the input image and its ground truth segmentation mask. It is worth noting that the probability maps in Fig. 5 are generated by fitting a single pair of image and its ground truth (GT) mask. Hence, after a certain number of iterations, all the losses are able to produce perfect results due to over-fitting. Here, we ignore the final fitting results and aim to observe the different characteristics and problems of these losses in the fitting process. The (c), (d), (e) and (f) columns show changes of the intermediate probability maps as the training progresses.



(a) RRM_LC (b) RRM_MS (c) RRM_Ours

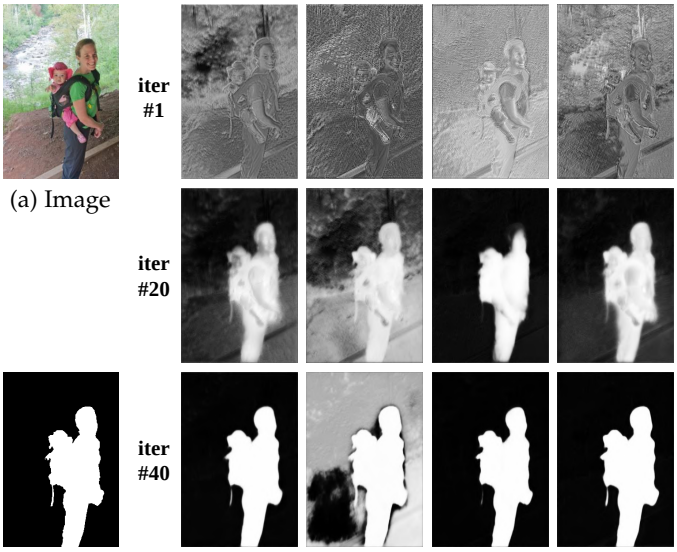
Fig. 4. Illustration of different Residual Refine Modules (RRM): (a) local boundary refinement module RRM_LC; (b) multi-scale refinement module RRM_MS; (c) our encoder-decoder refinement module RRM_Ours.

The BCE loss is computed pixel-wise. It does not consider the labels of the neighborhood and it weights both the foreground and background pixels equally. This helps with the convergence on all pixels and guarantee a relatively good local optima. Since significantly erroneous predictions (predicting 0 as 0.9 or predicting 1 as 0.1) produce large BCE loss, the models trained with BCE loss suppress these errors by giving prediction values around 0.5 around the boundaries, which often leads to blurred boundaries and fine structures, as we can see from the second row of column (c), where the contour of the whole foreground region is blurring, and the third row, in which the cable below the backpack is with low probability values.

The SSIM loss is a patch-level measure, which considers a local neighborhood of each pixel. It assigns higher weights to pixels located in the transitional buffer regions between foregrounds and backgrounds, *e.g.* boundaries, fine structures, so that the loss is higher around the boundary, even when the predicted probabilities on the boundary and the rest of the foreground are the same. It is worth noting that the loss for the background region is similar or sometimes even higher than the foreground region. However, the background loss does not contribute to the training until the prediction of background pixel becomes very close to the GT, where the loss drops rapidly from one to zero. Because μ_y, σ_{xy}, $\mu_x\mu_y$ and σ_y^2 in the SSIM loss (Equ. 5) are all zeros in the background regions, so the SSIM loss can be approximated by:

$$\ell^{bg}_{ssim} = 1 - \frac{C_1 C_2}{(\mu_x^2 + C_1)(\sigma_x^2 + C_2)}. \tag{7}$$

Since $C_1 = 0.01^2$ and $C_2 = 0.03^2$, only if the prediction x is close to zero, the SSIM loss (Equ. 7) will become the dominant term. The second and third rows of column (d) in Fig. 5 illustrate that the model trained with the SSIM loss is able to predict correct results on the foreground region and boundaries while neglecting the background accuracy in the beginning of the training process. This characteristic of the SSIM loss helps the optimization to focus on the boundary and foreground region. As the training progresses, the SSIM loss for the foreground is reduced and the background



(a) Image (b) GT (c) ℓ_{bce} (d) ℓ_{ssim} (e) ℓ_{iou} (f) ℓ_{bsi}

Fig. 5. Intermediate predictions of our BASNet when fitting with different losses.

loss becomes the dominant term. This is helpful since the prediction typically goes close to zero only late in the training process, where BCE loss becomes flat. The SSIM loss ensures that there is still enough gradient to drive the learning process. Hence, the background prediction looks cleaner since the probability is pushed to zero.

The IoU is a map-level measure. Larger ares contribute more to the IoU, so models trained with IoU loss emphasizes more on the large foreground regions and are thus able to produce relatively homogeneous and more confident (whiter) probabilities for these regions. However, these models often produce false negatives on fine structures. As shown in the column (e) of Fig. 5, the human head in the second row and the backpack cord in both the second and third rows are missing.

To take advantage of the above three losses, we combine them together to formulate the hybrid loss. BCE is used to maintain a smooth gradient for all pixels, while IoU is employed to put more focus on the foreground. SSIM is used to encourage the prediction to respect the structure of the original image, by employing a larger loss near the boundaries, as well as further push the backgrounds predictions to zero.

4 Experiments

In this paper, we are focusing on improving the spatial accuracy of segmentation results. Therefore, experiments are conducted on two reverse binary class image segmentation tasks: salient object segmentation [110] and camouflaged object segmentation [25]. Salient object segmentation is a popular task in computer vision, which aims at segmenting the salient regions against their backgrounds. In this task, the targets are usually with high contrast against their backgrounds. However, camouflaged object segmentation is the most challenging one because the camouflaged objects usually have similar appearance to their backgrounds, which means they are difficult to be perceived and segmented. In addition, many of the camouflaged objects have very complex structures and boundaries.

4.1 Implementation and Setting

We implement our network using the publicly available Pytorch 1.4.0 [84]. An 16-core PC with an AMD Threadripper 2950x 3.5 GHz CPU (with 64GB 3000 MHz RAM,) and an RTX Titan GPU (with 24GB memory) is used for both training and testing. During training, each image is first resized to 320×320 and randomly cropped to 288×288. Some of the encoder parameters are initialized from the ResNet-34 model [35]. Other convolutional layers are initialized by Xavier [30]. We use the Adam optimizer [49] to train our network and its hyperparameters are set to the default values, where the initial learning rate lr=1e-4, betas=(0.9, 0.999), eps=1e-8, weight_decay=0. We train the network until the loss converges, without using the validation set. The training loss converges after 400k iterations with a batch size of eight and the whole training process takes about 110 hours. During testing, the input image is resized to 320×320 and fed into the network to obtain its segmentation probability map. Then, the probability map (320×320) is resized back to the original size of the input image. Both resizing processes use bilinear interpolation. The inference for a 320×320 image only takes 0.015s (**70 fps**, different from that reported in our CVPR version [95], in which IO time is included).

4.2 Evaluation Metrics

Five measures are used to evaluate the performance of the proposed model. **(1) Weighted F-measure** F_β^w [74] gives a comprehensive and balanced evaluation on both precision and recall, which is able to better leverage the interpolation, dependency and equal-importance flaw. **(2) Relax boundary F-measure** F_β^b [19] is adopted to quantitatively evaluate the boundary quality of the predicted maps. **(3) Mean absolute error** M [86] reflects the average per-pixel difference between the probability map and the GT. **(4) Mean structural measure** S_α [22] quantizes the structural similarity between the predicted probability map and the GT mask. **(5) Mean enhanced-alignment measure** E_ϕ^m [23] takes both global and local similarity into consideration. Evaluation code: https://github.com/DengPingFan/CODToolbox.

4.3 Experiments on Salient Object Segmentation

4.3.1 Datasets

For salient object segmentation task[3], we train our network using the DUTS-TR [110] dataset, which has 10553 images. Before training, the dataset is augmented by horizontal flipping to 21106 images. For salient object segmentation tasks, we evaluate our method on six commonly used salient object segmentation benchmark datasets: SOD [78], ECSSD [127], DUT-OMRON [128], PASCAL-S [62], HKU-IS [58], DUTS-TE [110]. **DUT-OMRON** has 5,168 images with one or multiple objects. The majority of these objects are structurally complex. **PASCAL-S** was originally

3. The camouflaged object segmentation task use the same augmentation strategies.

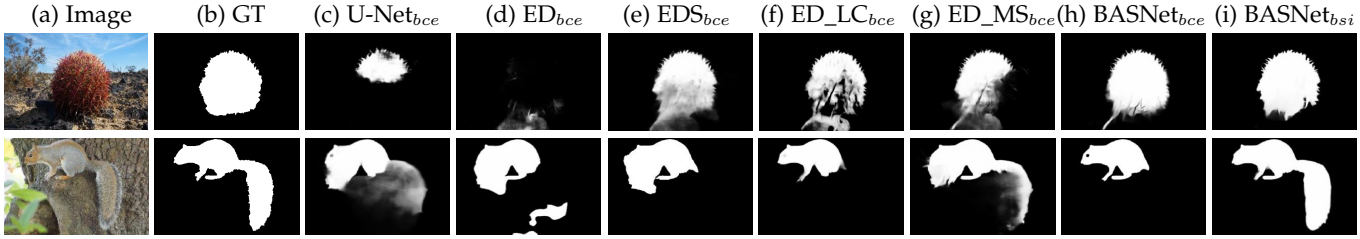


Fig. 6. Qualitative comparison of different configures in the ablation study. The first row show the predicted probability maps of different architectures trained with BCE loss and our BASNet trained with ℓ_{bsi} loss. The second row show the segmentation maps of our proposed prediction-refinement architecture trained with different losses. The corresponding quantitative results can be found in Table 1.

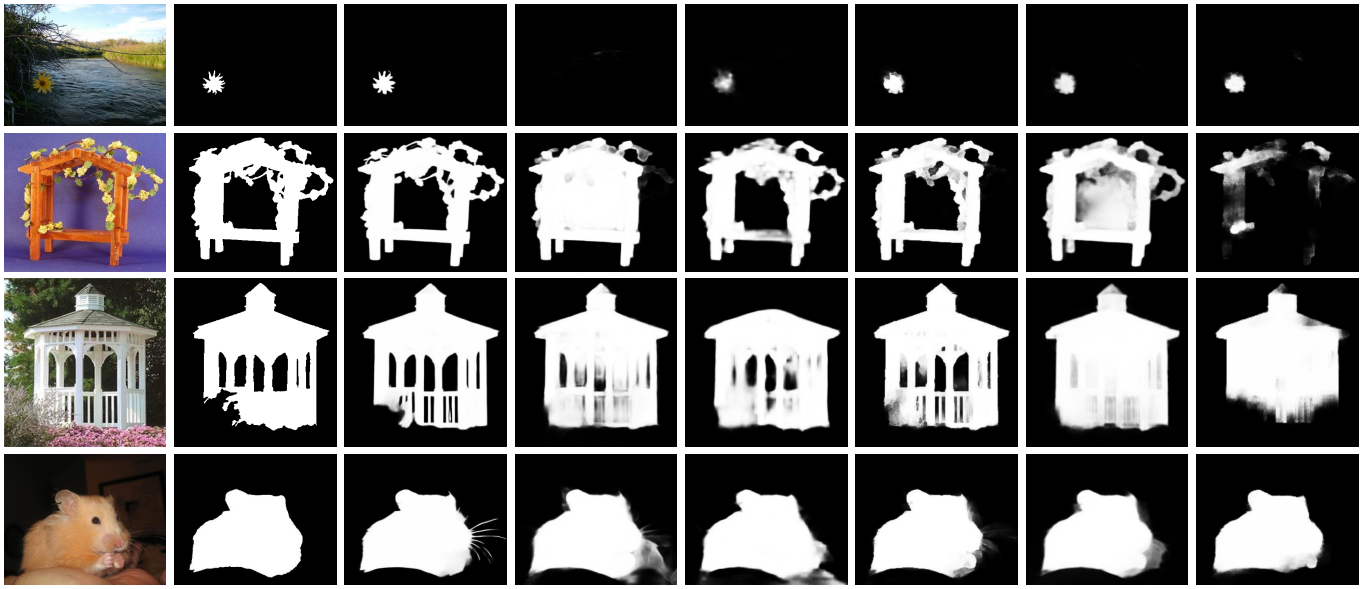


Fig. 7. Qualitative comparison on salient object segmentation datasets.

created for semantic image segmentation and consists of 850 challenging images. **DUTS** is a relatively large salient object segmentation dataset. It has two subsets: DUTS-TR and DUTS-TE. There are 10,553 images in **DUTS-TR** for training and 5,019 images in **DUTS-TE** for testing. In our experiments, DUTS-TR is used for training the model for salient object segmentation. **HKU-IS** contains 4,447 images, many of which contain multiple foreground objects. **ECSSD** contains 1,000 semantically meaningful images. However, the structures of the foreground objects in these images are complex. **SOD** contains 300 very challenging images. These images have either single complicated large foreground objects which overlap with the image boundaries or multiple salient objects with low contrast.

TABLE 1. Ablation study on different architectures (Arch.) and losses: ED: encoder-decoder, EDS: encoder-decoder + side output supervision; ℓ_b, ℓ_s and ℓ_i denote the BCE, SSIM and IoU loss, respectively, $\ell_{bi} = \ell_b + \ell_i$, $\ell_{bs} = \ell_b + \ell_s$, $\ell_{si} = \ell_s + \ell_i$, $\ell_{bsi} = \ell_b + \ell_s + \ell_i$.

Ablation	Configurations	$F_\beta^w \uparrow$	$F_\beta^b \uparrow$	$M \downarrow$	$S_\alpha \uparrow$	$E_\phi^m \uparrow$
Arch.	U-Net [98] + ℓ_b	0.827	0.669	0.064	0.867	0.897
	ED + ℓ_b	0.871	0.786	0.045	0.908	0.923
	EDS + ℓ_b	0.891	0.819	0.041	0.920	0.935
	EDS+RRM_LC + ℓ_b	0.900	0.804	0.038	0.915	0.935
	EDS+RRM_MS + ℓ_b	0.890	0.816	0.041	0.919	0.934
	EDS+RRM_Ours + ℓ_b	0.900	0.827	0.037	0.923	0.943
Loss	EDS+RRM_Ours + ℓ_s	0.886	0.814	0.044	0.904	0.932
	EDS+RRM_Ours + ℓ_i	0.902	0.820	0.037	0.911	0.943
	EDS+RRM_Ours + ℓ_{bs}	0.903	0.823	0.037	0.920	0.942
	EDS+RRM_Ours + ℓ_{bi}	0.909	0.832	0.035	0.921	**0.947**
	EDS+RRM_Ours + ℓ_{si}	0.894	0.812	0.041	0.906	0.938
	EDS+RRM_Ours + ℓ_{bsi}	**0.912**	**0.840**	**0.034**	**0.925**	**0.947**

4.3.2 Ablation Study

In this section, we validate the effectiveness of each key components used in our model. The ablation study is divided into two parts: an architecture ablation and loss ablation. For simplify, the ablation experiments are conducted on the ECSSD dataset. The same hyper-parameters to that described in Sec. 4.1 are used here.

Architecture: To demonstrate the effectiveness of our BASNet, we report quantitative comparison results of our model against other related architectures. We take U-Net [98] as our baseline network. Then we start with our proposed encoder-decoder network and progressively extend it with dense side output supervision and different residual refinement modules, including RRM_LC, RRM_MS and

TABLE 2. Comparison of the proposed method and 25 other methods on three salient object segmentation datasets: DUT-OMRON, DUTS-TE and HKU-IS. ↑ and ↓ indicate the higher the score the better and the lower the score the better, respectively. "*" indicates results post-processed by CRF. **Bold** font denotes the best performance.

Models	DUT-OMRON[128]					DUTS-TE[110]					HKU-IS[58]				
	$F_\beta^w\uparrow$	$F_\beta^b\uparrow$	$M\downarrow$	$S_\alpha\uparrow$	$E_\phi^m\uparrow$	$F_\beta^w\uparrow$	$F_\beta^b\uparrow$	$M\downarrow$	$S_\alpha\uparrow$	$E_\phi^m\uparrow$	$F_\beta^w\uparrow$	$F_\beta^b\uparrow$	$M\downarrow$	$S_\alpha\uparrow$	$E_\phi^m\uparrow$
MDF$_{\text{TIP16}}$	0.565	0.406	0.142	0.721	0.759	0.543	0.447	0.099	0.723	0.764	0.564	0.594	0.129	0.81	0.742
UCF$_{\text{ICCV17}}$	0.573	0.48	0.12	0.76	0.761	0.596	0.518	0.112	0.777	0.776	0.779	0.679	0.062	0.875	0.887
Amulet$_{\text{ICCV17}}$	0.626	0.528	0.098	0.781	0.794	0.658	0.568	0.084	0.796	0.817	0.817	0.716	0.051	0.886	0.910
NLDF*$_{\text{CVPR17}}$	0.634	0.514	0.08	0.77	0.799	0.71	0.591	0.065	0.805	0.851	0.838	0.694	0.048	0.879	0.914
DSS*$_{\text{CVPR17}}$	0.697	0.559	0.063	0.79	0.831	0.755	0.606	0.056	0.812	0.877	0.867	0.706	0.04	0.878	0.925
LFR$_{\text{IJCAI18}}$	0.647	0.508	0.103	0.78	0.799	0.689	0.556	0.083	0.799	0.833	0.861	0.731	0.04	0.905	0.934
C2S$_{\text{ECCV18}}$	0.661	0.565	0.072	0.798	0.823	0.713	0.607	0.062	0.829	0.859	0.829	0.717	0.048	0.883	0.859
RAS$_{\text{ECCV18}}$	0.695	0.615	0.062	0.814	0.844	0.74	0.656	0.059	0.828	0.871	0.843	0.748	0.045	0.887	0.92
RADF*$_{\text{AAAI18}}$	0.723	0.579	0.061	0.815	0.857	0.748	0.608	0.061	0.814	0.869	0.872	0.725	0.039	0.888	0.935
PAGRN$_{\text{CVPR18}}$	0.622	0.582	0.071	0.775	0.772	0.724	0.692	0.055	0.825	0.843	0.82	0.762	0.048	0.887	0.900
BMPM$_{\text{CVPR18}}$	0.681	0.612	0.064	0.809	0.831	0.761	0.699	0.048	0.851	0.883	0.859	0.773	0.039	0.907	0.931
PiCANet$_{\text{CVPR18}}$	0.691	0.643	0.068	0.826	0.833	0.747	0.704	0.054	0.851	0.873	0.847	0.784	0.042	0.906	0.923
MLMS$_{\text{CVPR19}}$	0.681	0.612	0.064	0.809	0.831	0.761	0.699	0.048	0.851	0.883	0.859	0.773	0.039	0.907	0.931
AFNet$_{\text{CVPR19}}$	0.717	0.635	0.057	0.826	0.846	0.785	0.714	0.046	0.855	0.893	0.869	0.772	0.036	0.905	0.935
MSWS$_{\text{CVPR19}}$	0.527	0.362	0.109	0.756	0.729	0.586	0.376	0.908	0.749	0.742	0.685	0.438	0.084	0.818	0.787
R^3**Net***$_{\text{IJCAI18}}$	0.728	0.599	0.063	0.817	0.853	0.763	0.601	0.058	0.817	0.873	0.877	0.74	0.036	0.895	0.939
CapSal$_{\text{CVPR19}}$	0.482	0.396	0.101	0.674	0.659	0.691	0.605	0.072	0.808	0.849	0.782	0.654	0.062	0.85	0.883
SRM$_{\text{ICCV17}}$	0.658	0.523	0.069	0.798	0.808	0.722	0.592	0.058	0.824	0.853	0.835	0.68	0.046	0.887	0.913
DGRL$_{\text{CVPR18}}$	0.697	0.584	0.063	0.810	0.845	0.76	0.656	0.051	0.836	0.887	0.865	0.744	0.037	0.897	0.939
CPD$_{\text{CVPR19}}$	0.719	0.655	**0.056**	0.825	0.847	0.795	0.741	0.043	0.858	0.898	0.875	0.795	0.034	0.905	0.939
PoolNet$_{\text{CVPR19}}$	0.729	0.675	**0.056**	0.836	0.854	0.807	0.765	**0.040**	0.871	0.904	0.881	0.811	0.033	0.917	0.940
BANet$_{\text{ICCV19}}$	0.719	0.611	0.061	0.823	0.861	0.781	0.687	0.046	0.861	0.897	0.869	0.760	0.037	0.902	0.938
EGNet$_{\text{ICCV19}}$	0.728	0.679	**0.056**	0.836	0.853	0.797	0.761	0.043	0.879	0.898	0.875	0.802	0.035	0.910	0.938
MINet$_{\text{CVPR20}}$	0.719	0.640	0.057	0.822	0.846	0.813	0.747	**0.040**	0.875	0.906	0.889	0.799	0.032	0.912	0.944
GateNet$_{\text{ECCV20}}$	0.703	0.625	0.061	0.821	0.840	0.786	0.722	0.045	0.871	0.892	0.872	0.783	0.036	0.910	0.934
BASNet (Ours)	**0.760**	**0.703**	0.057	**0.841**	**0.868**	**0.825**	**0.786**	0.042	**0.881**	**0.907**	**0.900**	**0.821**	**0.030**	**0.918**	**0.948**

TABLE 3. Comparison of the proposed method and 25 other methods on three salient object detection datasets: ECSSD, PASCAL-S and SOD. See Table 2 for details.

Baseline Models	ECSSD[127]					PASCAL-S[62]					SOD[78]				
	$F_\beta^w\uparrow$	$F_\beta^b\uparrow$	$M\downarrow$	$S_\alpha\uparrow$	$E_\phi^m\uparrow$	$F_\beta^w\uparrow$	$F_\beta^b\uparrow$	$M\downarrow$	$S_\alpha\uparrow$	$E_\phi^m\uparrow$	$F_\beta^w\uparrow$	$F_\beta^b\uparrow$	$M\downarrow$	$S_\alpha\uparrow$	$E_\phi^m\uparrow$
MDF$_{\text{TIP16}}$	0.705	0.472	0.105	0.776	0.796	0.589	0.343	0.142	0.696	0.706	0.508	0.311	0.192	0.643	0.607
UCF$_{\text{ICCV17}}$	0.806	0.669	0.069	0.884	0.891	0.694	0.493	0.115	0.805	0.809	0.675	0.471	0.148	0.762	0.773
Amulet$_{\text{ICCV17}}$	0.84	0.711	0.059	0.894	0.909	0.734	0.541	0.100	0.818	0.835	0.677	0.454	0.144	0.753	0.776
NLDF*$_{\text{CVPR17}}$	0.839	0.666	0.063	0.897	0.900	0.737	0.495	0.098	0.798	0.839	0.709	0.475	0.125	0.755	0.777
DSS*$_{\text{CVPR17}}$	0.872	0.696	0.052	0.882	0.918	0.759	0.499	0.093	0.798	0.845	0.710	0.444	0.124	0.743	0.774
LFR$_{\text{IJCAI18}}$	0.858	0.694	0.052	0.897	0.923	0.737	0.499	0.107	0.805	0.835	0.734	0.479	0.123	0.773	0.813
C2S$_{\text{ECCV18}}$	0.851	0.708	0.055	0.893	0.917	0.766	0.543	0.082	0.836	0.864	0.700	0.457	0.124	0.760	0.785
RAS$_{\text{ECCV18}}$	0.857	0.741	0.056	0.893	0.914	0.736	0.560	0.101	0.799	0.830	0.720	0.544	0.124	0.764	0.788
RADF*$_{\text{AAAI18}}$	0.883	0.720	0.049	0.894	0.929	0.755	0.515	0.097	0.802	0.840	0.729	0.476	0.126	0.757	0.801
PAGRN$_{\text{CVPR18}}$	0.834	0.747	0.061	0.889	0.895	0.738	0.594	0.090	0.822	0.830	-	-	-	-	-
BMPM$_{\text{CVPR18}}$	0.871	0.770	0.045	0.911	0.928	0.779	0.617	0.074	0.845	0.872	0.726	0.562	0.108	0.786	0.799
PiCANet$_{\text{CVPR18}}$	0.865	0.784	0.046	0.914	0.924	0.772	0.612	0.078	0.848	0.866	0.722	0.572	0.103	0.789	0.796
MLMS$_{\text{CVPR19}}$	0.871	0.770	0.045	0.911	0.928	0.779	0.62	0.074	0.844	0.875	0.726	0.562	0.108	0.786	0.799
AFNet$_{\text{CVPR19}}$	0.887	0.776	0.042	0.914	0.936	0.798	0.626	0.070	0.849	0.883	0.723	0.545	0.111	0.774	0.79
MSWS$_{\text{CVPR19}}$	0.716	0.411	0.096	0.828	0.791	0.614	0.289	0.133	0.768	0.731	0.573	0.231	0.167	0.700	0.656
R^3**Net***$_{\text{IJCAI18}}$	0.902	0.759	0.040	0.910	0.944	0.761	0.538	0.092	0.807	0.843	0.735	0.431	0.125	0.759	0.796
CapSal$_{\text{CVPR19}}$	0.771	0.574	0.077	0.826	0.849	0.786	0.527	0.073	0.837	0.872	0.597	0.404	0.148	0.695	0.699
SRM$_{\text{ICCV17}}$	0.853	0.672	0.054	0.895	0.913	0.758	0.509	0.084	0.834	0.853	0.670	0.392	0.128	0.741	0.744
DGRL$_{\text{CVPR18}}$	0.883	0.753	0.042	0.906	0.938	0.787	0.569	0.074	0.839	0.877	0.731	0.502	0.106	0.773	0.807
CPD$_{\text{CVPR19}}$	0.898	0.811	0.037	0.918	0.942	0.800	0.639	0.071	0.848	0.878	0.714	0.556	0.112	0.767	0.778
PoolNet$_{\text{CVPR19}}$	0.896	0.813	0.039	0.921	0.940	0.798	0.644	0.075	0.832	0.876	0.759	0.606	0.102	**0.797**	0.818
BANet$_{\text{ICCV19}}$	0.890	0.758	0.041	0.913	0.940	0.792	0.589	0.078	0.840	0.875	0.750	0.589	0.109	0.782	0.813
EGNet$_{\text{ICCV19}}$	0.892	0.814	0.041	0.920	0.936	0.793	0.650	0.077	0.848	0.873	0.737	0.586	0.112	0.784	0.798
MINet$_{\text{CVPR20}}$	0.905	0.805	0.037	0.920	0.943	**0.813**	0.648	**0.065**	0.854	**0.889**	-	-	-	-	-
GateNet$_{\text{ECCV20}}$	0.886	0.782	0.042	0.917	0.933	0.803	0.623	0.068	**0.857**	0.882	-	-	-	-	-
BASNet (Ours)	**0.912**	**0.840**	**0.034**	**0.925**	**0.947**	0.808	**0.674**	0.072	0.847	0.878	**0.762**	**0.640**	**0.102**	0.793	**0.822**

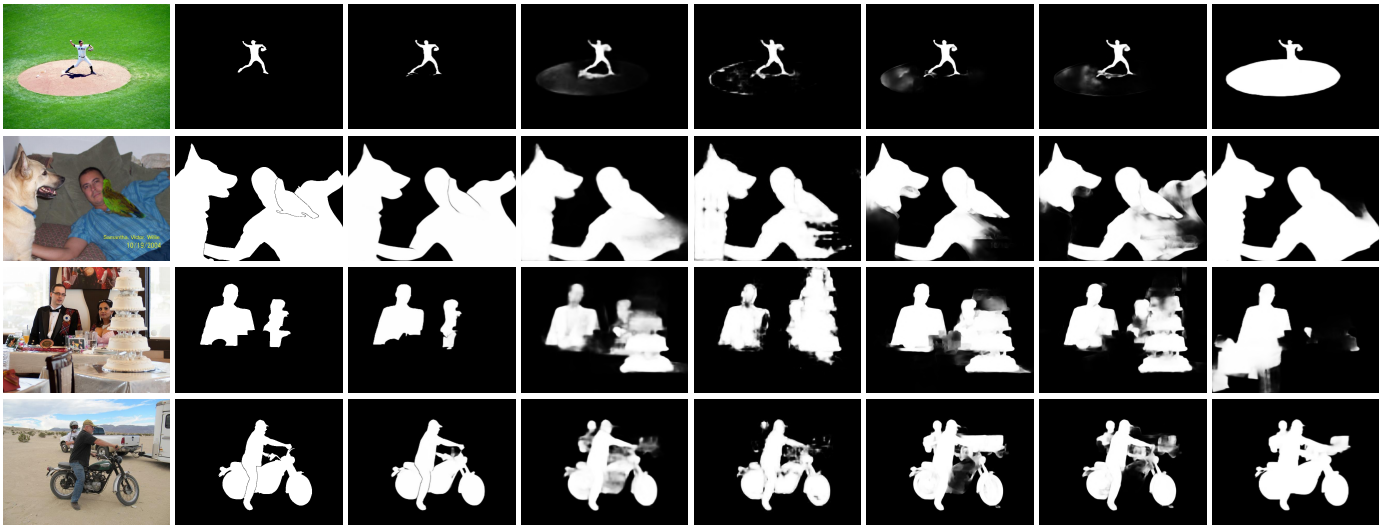
RRM_Ours. The top part of Table 1 and the first row of Fig. 6 illustrate the qualitative and quantitative results of the architecture ablation study, respectively. As we can see, our BASNet architecture achieves the best performance among all configurations.

Loss: To demonstrate the effectiveness of our proposed fusion loss, we conduct a set of experiments over different losses based on our BASNet architecture. The results in Table 1 indicate that the proposed hybrid ℓ_{bsi} loss greatly improves the performance, especially in terms of the boundary quality. It is clear that our hybrid loss achieves superior qualitative results, as shown in the second row of Fig. 6.

4.3.3 Comparison with State-of-the-Arts

We compare our method with 25 state-of-the-art models, including MDF [60], UCF [137], Amulet [136], NLDF [72], DSS [36], LFR [135], C2S [61], RAS [13], RADF [39], PAGRN [138], BMPM [133], PiCANet [68], MLMS [119], AFNet [26], MSWS [131], R^3-Net [18], CapSal [134], SRM [113], DGRL [115], CPD [121], PoolNet [66], BANet [104], EGNet [140], MINet [83] and GateNet [144], on the salient object segmentation task. For fair comparison, we either use the segmentation maps released by the authors or run their publicly available models with their default settings.

Quantitative Evaluation: Tables 2 and 3 provide quantitative comparisons on six salient object segmentation datasets. Our BASNet outperforms other models on the DUT-OMRON, DUTS-TE, HKU-IS, ECSSD and SOD datasets in terms of nearly all metrics, except for the M measures on DUT-OMRON and DUTS-TE and the S_α on SOD. On the PASCAL-S dataset, MINet performs the best in terms of three metrics: F_β^w, MAE and E_ϕ^m. It is worth noting



(a) Image (b) GT (c) BASNet (d) SCRN (e) BANet (f) EGNet (g) PoolNet (h) CPD

Fig. 8. Qualitative comparison on typical samples from the SOC dataset. Images from top to bottom are from attributes SO (Small Object), OV(Out-of-View), OC (Occlusion) and SC (Shape Complexity) respectively.

(a) Image (b) GT (c) BASNet

Fig. 9. Failure cases on salient object segmentation datasets.

that BASNet achieves the highest relax boundary F-measure F_β^b on all of the six datasets, which indicates its strong capability in capturing boundaries and fine structures.

Qualitative Evaluation: Fig. 7 shows the qualitative comparison between our BASNet and 5 other typical models. As we can see, our BASNet is able to handle different challenging cases, such as small target with relatively low contrast (1st row), large object with complicated boundaries (2nd row), object with hollow structures (3rd row) and target with very fine structures (4th row). The third and fourth row of Fig. 7 show inspiring results, in which the segmentation maps predicted by our BASNet contain more details than the GT. These details reveals the possible inconsistency between the labels of training and testing datasets. Although detecting of these details usually leads to the deterioration of the quantitative evaluation scores of our model, it is more practically useful than good scores.

4.3.4 Failure Cases

Fig. 9 shows three typical failure cases of our BASNet on SOD datasets. For instance, the model sometimes fails in very complicated scenarios, in which there seems no salient objects, as show in the first row of Fig. 9. The second row gives an exemplary failure case of "saliency confusing", where the scene contains multiple independent "salient" targets, But only one of them is labeled. Our BASNet sometimes fails in these cases due to lack of the ability of recognizing the tiny saliency differences between multiple connected targets. The recent uncertainty model [132] may be one of the solutions.

4.3.5 Attribute-base Analysis

In addition to the most frequently used salient object segmentation datasets, we also test our model on another dataset, SOC [24]. The SOC dataset contains complicated scenarios, which are more challenging than those in the previous six SOD datasets. Besides, the SOC dataset categorizes images into nine different groups including AC (Appearance Change), BO (Big Object), CL (Clutter), HO (Heterogeneous Object), MB (Motion Blur), OC (Occlusion), OV (Out-of-View), SC (Shape Complexity), and SO (Small Object), according to their attributes. We train our BASNet on both DUTS-TR and the training set (1,800 images with salient objects) of SOC dataset [24] and evaluate their performance on the testing set of SOC-Sal. There are totally 600 images with salient objects in the testing set. Each image may be categorized into one or multiple attributes (*e.g.*AC and BO).

Quantitative Evaluation: Tab. 4 illustrates a comparison between our BASNet and 16 other state-of-the-art models, including Amulet [136], DSS [36], NLDF [72], C2S-Net [61], SRM [113], R3Net [18], BMPM [133], DGRL [115], PiCANet-R (PiC(R)) [68], RANet [14], AFNet [26], CPD [121], PoolNet [66], EGNet [140], BANet [104] and SCRN [122] in terms of attribute-based performance. As we can see, our BASNet achieves obvious improvements against the existing methods. Particularly, our BASNet advances the boundary measure F_β^b by large margins (over 5% and sometimes over 10%) on different attributes.

Qualitative Evaluation: Fig. 8 provides a qualitative comparison of our BASNet and other baseline models. As we can see, BASNet is able to handle different challenges,

TABLE 4. Comparison of the proposed method and other SOTA methods on the SOC test set. ↑ and ↓ indicate the higher score the better and the lower score the better respectively. **Bold** font indicates the best performance. **Avg.** denotes the average of all the attribute-based metric scores.

Attr	Metr.	Amulet [136]	DSS [36]	NLDF [72]	C2SNet [61]	SRM [113]	R3Net [18]	BMPM [133]	DGRL [115]	PiC(R) [68]	RANet [14]	AFNet [26]	CPD [121]	PoolNet [66]	EGNet [140]	BANet [104]	SCRN [122]	Ours (DUTS)	Ours (SOC)
AC	F_β^w ↑	0.620	0.629	0.620	0.647	0.690	0.593	0.680	0.718	0.682	0.603	0.712	0.727	0.713	0.731	0.740	0.724	0.735	**0.792**
	F_β^b ↑	0.448	0.384	0.374	0.408	0.410	0.387	0.531	0.457	0.489	0.448	0.569	0.626	0.578	0.597	0.562	0.588	0.659	**0.696**
	M ↓	0.120	0.113	0.119	0.109	0.096	0.135	0.098	0.081	0.093	0.132	0.084	0.083	0.094	0.085	0.086	0.078	0.087	**0.060**
	S_α ↑	0.752	0.753	0.737	0.755	0.791	0.713	0.780	0.790	0.792	0.708	0.796	0.799	0.795	0.806	0.806	0.809	0.805	**0.831**
	E_ϕ^m ↑	0.791	0.788	0.784	0.807	0.824	0.753	0.815	0.853	0.815	0.765	0.852	0.843	0.846	0.854	0.858	0.849	0.844	**0.885**
BO	F_β^w ↑	0.612	0.614	0.622	0.730	0.667	0.456	0.670	0.786	0.799	0.453	0.741	0.739	0.610	0.585	0.720	0.778	0.747	**0.808**
	F_β^b ↑	0.274	0.213	0.218	0.362	0.274	0.229	0.400	0.392	0.466	0.231	0.450	0.481	0.323	0.319	0.360	0.453	0.519	**0.572**
	M ↓	0.346	0.356	0.354	0.267	0.306	0.445	0.303	0.215	0.200	0.454	0.245	0.257	0.353	0.373	0.271	0.224	0.253	**0.166**
	S_α ↑	0.574	0.561	0.568	0.654	0.614	0.437	0.604	0.684	0.729	0.421	0.658	0.647	0.561	0.528	0.645	0.698	0.666	**0.723**
	E_ϕ^m ↑	0.551	0.537	0.539	0.661	0.616	0.419	0.620	0.725	0.741	0.404	0.698	0.665	0.554	0.528	0.650	0.706	0.677	**0.775**
CL	F_β^w ↑	0.663	0.617	0.614	0.655	0.665	0.546	0.678	0.714	0.692	0.542	0.696	0.724	0.681	0.677	0.726	0.717	0.700	**0.730**
	F_β^b ↑	0.374	0.275	0.292	0.342	0.327	0.315	0.432	0.393	0.420	0.344	0.465	0.553	0.488	0.493	0.461	0.506	0.552	**0.579**
	M ↓	0.141	0.153	0.159	0.144	0.134	0.182	0.123	0.119	0.123	0.188	0.119	0.114	0.134	0.139	0.117	0.113	0.121	**0.110**
	S_α ↑	0.763	0.722	0.713	0.742	0.759	0.659	0.761	0.770	0.787	0.624	0.768	0.773	0.760	0.757	0.784	**0.795**	0.774	0.785
	E_ϕ^m ↑	0.789	0.763	0.764	0.789	0.793	0.710	0.801	0.824	0.794	0.715	0.802	0.821	0.801	0.790	0.824	0.820	0.807	**0.826**
HO	F_β^w ↑	0.688	0.660	0.661	0.668	0.696	0.633	0.684	0.722	0.704	0.626	0.722	0.751	0.739	0.720	0.754	0.743	0.746	**0.764**
	F_β^b ↑	0.465	0.347	0.378	0.398	0.392	0.383	0.496	0.447	0.462	0.425	0.527	0.618	0.570	0.560	0.542	0.577	0.627	**0.639**
	M ↓	0.119	0.124	0.126	0.123	0.115	0.136	0.116	0.104	0.108	0.143	0.103	0.097	0.100	0.106	0.094	0.096	0.099	**0.093**
	S_α ↑	0.791	0.767	0.755	0.768	0.794	0.740	0.781	0.791	0.809	0.713	0.798	0.803	0.815	0.802	0.819	**0.823**	0.809	0.814
	E_ϕ^m ↑	0.810	0.796	0.798	0.805	0.819	0.782	0.813	0.833	0.819	0.777	0.834	0.845	0.846	0.829	**0.850**	0.842	0.843	**0.850**
MB	F_β^w ↑	0.561	0.577	0.551	0.593	0.619	0.489	0.651	0.655	0.637	0.576	0.626	0.679	0.642	0.649	0.672	0.690	0.678	**0.725**
	F_β^b ↑	0.435	0.396	0.397	0.450	0.395	0.348	0.561	0.464	0.520	0.476	0.537	0.619	0.592	0.584	0.539	0.595	0.635	**0.674**
	M ↓	0.142	0.132	0.138	0.128	0.115	0.160	0.105	0.113	0.099	0.139	0.111	0.106	0.121	0.109	0.104	0.100	0.115	**0.072**
	S_α ↑	0.712	0.719	0.685	0.720	0.742	0.657	0.762	0.744	0.775	0.696	0.734	0.754	0.751	0.762	0.764	0.792	0.755	**0.797**
	E_ϕ^m ↑	0.739	0.753	0.740	0.778	0.778	0.697	0.812	0.823	0.813	0.761	0.762	0.804	0.779	0.789	0.803	0.817	0.805	**0.836**
OC	F_β^w ↑	0.607	0.595	0.593	0.622	0.630	0.520	0.644	0.659	0.638	0.527	0.680	0.672	0.659	0.658	0.678	0.673	0.672	**0.707**
	F_β^b ↑	0.395	0.310	0.335	0.382	0.343	0.323	0.456	0.396	0.439	0.382	0.503	0.545	0.510	0.505	0.466	0.514	0.573	**0.601**
	M ↓	0.143	0.144	0.149	0.130	0.129	0.168	0.119	0.116	0.119	0.169	0.109	0.115	0.119	0.121	0.112	0.111	0.115	**0.101**
	S_α ↑	0.735	0.718	0.709	0.738	0.749	0.653	0.752	0.747	0.765	0.641	0.771	0.750	0.756	0.754	0.765	0.775	0.760	**0.780**
	E_ϕ^m ↑	0.763	0.760	0.755	0.784	0.780	0.706	0.800	0.808	0.784	0.718	0.820	0.810	0.801	0.798	0.809	0.800	0.806	**0.829**
OV	F_β^w ↑	0.637	0.622	0.616	0.671	0.682	0.527	0.701	0.733	0.721	0.529	0.723	0.721	0.697	0.707	**0.752**	0.723	0.730	0.749
	F_β^b ↑	0.405	0.311	0.339	0.420	0.368	0.336	0.494	0.434	0.490	0.383	0.524	0.592	0.526	0.541	0.509	0.545	0.617	**0.630**
	M ↓	0.173	0.180	0.184	0.159	0.150	0.216	0.136	0.125	0.127	0.217	0.129	0.134	0.148	0.146	0.119	0.126	0.132	**0.114**
	S_α ↑	0.721	0.700	0.688	0.728	0.745	0.624	0.751	0.762	**0.781**	0.611	0.761	0.748	0.747	0.752	0.779	0.774	0.764	**0.781**
	E_ϕ^m ↑	0.751	0.737	0.736	0.790	0.779	0.663	0.807	0.828	0.810	0.664	0.817	0.803	0.795	0.802	**0.835**	0.808	0.809	0.828
SC	F_β^w ↑	0.608	0.599	0.593	0.611	0.638	0.550	0.677	0.669	0.627	0.594	0.696	0.708	0.695	0.678	0.706	0.691	0.728	**0.746**
	F_β^b ↑	0.481	0.407	0.414	0.433	0.423	0.427	0.561	0.455	0.492	0.504	0.572	0.627	0.613	0.597	0.562	0.603	0.654	**0.672**
	M ↓	0.098	0.098	0.101	0.100	0.090	0.114	0.081	0.087	0.093	0.110	0.076	0.080	0.075	0.083	0.078	0.078	0.074	**0.072**
	S_α ↑	0.768	0.761	0.745	0.756	0.783	0.716	0.799	0.772	0.784	0.724	0.808	0.793	0.807	0.793	0.807	0.809	0.812	**0.820**
	E_ϕ^m ↑	0.794	0.799	0.788	0.806	0.814	0.765	0.841	0.837	0.799	0.792	0.854	0.858	0.856	0.844	0.851	0.843	0.861	**0.872**
SO	F_β^w ↑	0.523	0.524	0.526	0.531	0.561	0.487	0.567	0.602	0.566	0.518	0.596	0.623	0.626	0.594	0.621	0.614	0.634	**0.684**
	F_β^b ↑	0.386	0.325	0.341	0.353	0.334	0.342	0.442	0.382	0.417	0.412	0.468	0.533	0.523	0.494	0.457	0.506	0.551	**0.612**
	M ↓	0.119	0.109	0.115	0.116	0.099	0.118	0.096	0.092	0.095	0.113	0.089	0.091	0.087	0.098	0.090	0.082	0.092	**0.075**
	S_α ↑	0.718	0.713	0.703	0.706	0.737	0.682	0.732	0.736	0.748	0.682	0.746	0.745	0.768	0.749	0.755	0.767	0.758	**0.787**
	E_ϕ^m ↑	0.745	0.756	0.747	0.752	0.769	0.732	0.780	0.802	0.766	0.759	0.792	0.804	0.814	0.784	0.801	0.797	0.800	**0.835**
Avg.	F_β^w ↑	0.613	0.604	0.600	0.636	0.650	0.533	0.661	0.695	0.674	0.552	0.688	0.705	0.674	0.667	0.708	0.706	0.708	**0.745**
	F_β^b ↑	0.407	0.330	0.343	0.394	0.363	0.343	0.486	0.424	0.466	0.401	0.513	0.577	0.525	0.521	0.495	0.543	0.599	**0.631**
	M ↓	0.156	0.157	0.161	0.142	0.137	0.186	0.131	0.117	0.117	0.185	0.118	0.120	0.137	0.140	0.119	0.112	0.121	**0.096**
	S_α ↑	0.726	0.713	0.700	0.730	0.746	0.653	0.747	0.755	0.774	0.647	0.760	0.757	0.751	0.745	0.769	0.782	0.767	**0.791**
	E_ϕ^m ↑	0.748	0.743	0.739	0.775	0.775	0.692	0.788	0.815	0.793	0.706	0.803	0.806	0.788	0.780	0.809	0.809	0.806	**0.837**

including small objects (1st row), out-of-view objects (2nd row), occluded targets (3rd row) and objects with complicated shapes (4th row).

4.4 Experiments on Camouflaged Object Segmentation

To further evaluate the performance of the proposed BASNet, we also tested it on the camouflaged object segmentation (COS) task [126], [55], [25]. Compared with salient object segmentation, COS is a relatively newer and more challenging task. Because the contrast between the camouflaged targets and their backgrounds is sometimes extremely low. Besides, the targets usually have similar color and texture to their backgrounds. In addition, their shape or structure of these targets can sometimes be very complex.

4.4.1 Datasets

We test our model on the CHAMELEON [103], CAMO-Test [54] and COD10K-Test datasets [25]. CHAMELEON [103] contains 76 images taken by independent photographers. These images are marked as good examples of camouflaged animals by the photographers. CAMO [54] contains both camouflaged and non-camouflaged subsets. We use the camouflaged subset, which comprises two further subsets: CAMO-Train (1,000 images) and CAMO-Test (250 images). COD10K [25] is currently the largest camouflaged object detection dataset. It comprises 10,000 images of 78 object categories in various natural scenes. There are 5,066 images densely labeled with accurate (matting-level) binary masks. COD10K consists of 3,040 images for training (COD10K-Train) and 2,026 images for testing (COD10K-Test). For fair

TABLE 5. Comparison of the proposed method and 13 other methods on three camouflaged object segmentation datasets: CHAMELEON, CAMO-Test and COD10K-Test. ↑ and ↓ indicate the higher score the better and the lower the score the better, respectively. **Bold** font indicates the best performance.

Baseline Models	CHAMELEON[103]					CAMO-Test[54]					COD10K-Test[25]				
	$F_\beta^w \uparrow$	$F_\beta^b \uparrow$	$M \downarrow$	$S_\alpha \uparrow$	$E_\phi^m \uparrow$	$F_\beta^w \uparrow$	$F_\beta^b \uparrow$	$M \downarrow$	$S_\alpha \uparrow$	$E_\phi^m \uparrow$	$F_\beta^w \uparrow$	$F_\beta^b \uparrow$	$M \downarrow$	$S_\alpha \uparrow$	$E_\phi^m \uparrow$
FPN$_{\text{CVPR17}}$	0.590	0.246	0.075	0.794	0.784	0.483	0.232	0.131	0.684	0.677	0.411	0.195	0.075	0.697	0.692
MaskRCNN$_{\text{CVPR17}}$	0.518	0.128	0.099	0.643	0.778	0.430	0.117	0.151	0.574	0.715	0.402	0.110	0.081	0.613	0.748
PSPNet$_{\text{CVPR17}}$	0.555	0.207	0.085	0.773	0.756	0.455	0.191	0.139	0.663	0.659	0.377	0.166	0.080	0.678	0.681
UNet++$_{\text{DLMIA18}}$	0.501	0.246	0.094	0.695	0.763	0.392	0.232	0.149	0.599	0.654	0.350	0.195	0.086	0.623	0.674
PiCANet$_{\text{CVPR18}}$	0.536	0.200	0.084	0.769	0.749	0.356	0.166	0.155	0.609	0.584	0.322	0.173	0.083	0.649	0.643
MSRCN$_{\text{CVPR19}}$	0.443	0.074	0.091	0.637	0.686	0.454	0.128	0.133	0.618	0.669	0.419	0.101	0.073	0.641	0.706
PFANet$_{\text{CVPR19}}$	0.378	0.096	0.139	0.679	0.648	0.391	0.130	0.169	0.659	0.622	0.286	0.107	0.118	0.636	0.618
HTC$_{\text{CVPR2019}}$	0.204	0.071	0.129	0.517	0.489	0.174	0.076	0.172	0.477	0.442	0.221	0.099	0.088	0.548	0.520
PoolNet$_{\text{CVPR2019}}$	0.555	0.151	0.079	0.777	0.779	0.494	0.155	0.128	0.703	0.698	0.416	0.126	0.07	0.705	0.713
ANet-SRM$_{\text{CVIU19}}$	-	-	-	-	-	0.484	0.217	0.126	0.682	0.686	-	-	-	-	-
CPD$_{\text{CVPR2019}}$	0.706	0.383	0.052	0.853	0.868	0.550	0.306	0.115	0.726	0.730	0.508	0.286	0.059	0.747	0.771
EGNet$_{\text{ICCV19}}$	0.702	0.289	0.050	0.848	0.871	0.583	0.264	0.104	0.732	0.768	0.509	0.209	0.056	0.737	0.779
SINet$_{\text{CVPR20}}$	0.740	0.410	0.044	0.869	0.893	0.606	0.334	0.100	**0.752**	0.772	0.551	0.311	0.051	0.771	0.809
BASNet (Ours)	**0.866**	**0.650**	**0.022**	**0.914**	**0.954**	**0.646**	**0.420**	**0.096**	0.749	**0.796**	**0.677**	**0.546**	**0.038**	**0.802**	**0.855**

(a) Image (b) GT (c) BASNet

Fig. 10. Failure cases on camouflaged object segmentation task. The first row shows the typical false negative artifacts. The second row illustrates the false positive phenomenon.

comparison, we use the same training sets as SINet [25].

4.4.2 Comparison with State-of-the-Arts

To validate the performance of the proposed BASNet on the camouflaged object segmentation task, we compared BASNet against 13 sate-of-the-art models, including FPN[64], MaskRCNN[33], PSPNet[139], UNet++[146], PiCANet[68], MSRCN[41], PFANet[143], HTC[9], PoolNet[66], ANet-SRM[54], CPD[121], EGNet[140] and SINet[25]. For fair comparison, the results of different models are either provided by the authors or obtained by re-training the model with the default settings with same training data.

Quantitative Evaluation: The quantitative evaluation results are illustrated in Table 5. As we can see that our BASNet achieves the best performance in nearly all metrics with great advantages. SINet is the second best model. EGNet and CPD are competitive with each other and can be ranked as the third and fourth best. Our BASNet improves the weighted F-measure $F_\beta^w \uparrow$ with large margins (12.6%, 4.0% and 12.6% on CHAMELEON, CAMO-Test and COD10K-Test respectively). Particularly, our BASNet outperforms the second best model SINet by 24.0%, 8.6% and 23.5% in terms of the relax boundary F-measure $F_\beta^b \uparrow$ on CHAMELEON, CAMO-Test and COD10K-Test datasets. This reveals the effectiveness and accuracy of our BASNet in capturing boundaries and fine structures. In terms of the $M \downarrow$, our BASNet reduces the metric by 50.0%, 4.0% and 25.5% on the three datasets, respectively. For the structure measures, $S_\alpha \uparrow$, the improvements of our BASNet against the second best model are also significant (4.5% and 3.1% on CHAMELEON and COD10K-Test datasets) but there is a 0.3% S_α decrease on CAMO-Test. Compared with SINet, $E_\phi^m \uparrow$ takes both local and global structure similarity into consideration. As we can see, our BASNet achieves even greater improvements (6.1%, 2.4% and 4.6% on the three datasets, respectively) against the second best model in $E_\phi^m \uparrow$ than in $S_\alpha \uparrow$.

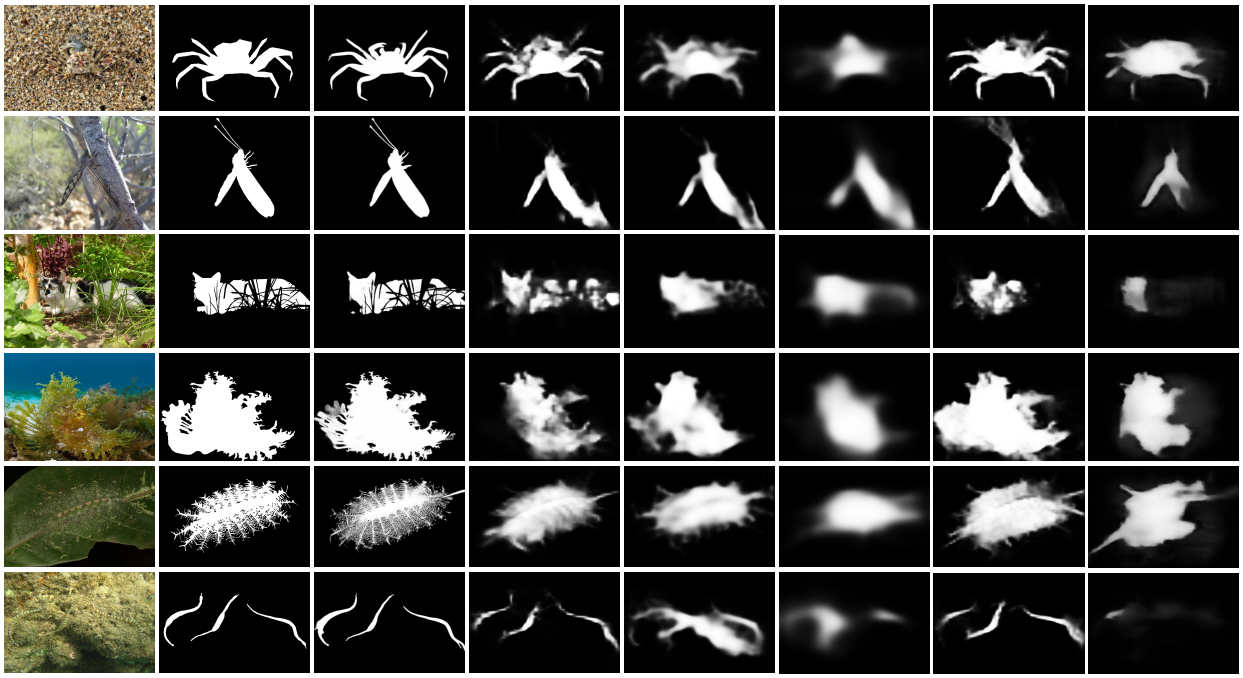
Qualitative Evaluation: Qualitative comparisons against several of the baseline models are illustrated in Fig. 11. As we can see, our BASNet (the 3rd column) is able to handle different types of challenging camouflaged cases including complex foreground targets with low contrast (the 1st row), targets with very thin foreground structures (the 2nd and 5th row), targets occluded by fine objects (the 3rd row), targets with complicated boundaries (the 4th row), targets with extremely complex hollow structures (the 5th row), multiple objects with low contrast (the 6th row), *etc*. Compared with the results of other models, the results of our BASNet demonstrates its excellent abilities of perceiving fine structures and complicated boundaries, which also explains why our BASNet is able to achieve such high boundary evaluation scores $F_\beta^b \uparrow$ on camouflage object segmentation datasets (see Table 5).

4.4.3 Failure Cases

Although our BASNet outperforms other camouflaged object segmentation (COS) models and rarely produces completely incorrect results, there are still some false negative (the 1st row in Figure 10) and false positive predictions (the 2nd row in Fig. 10) in many of the COS cases. It is worth noting that other models usually have the same or even worse results on these challenging cases. Although these failure cases may not have a huge impact on evaluation metrics, they will somehow limit the applications and degrade the user experiences.

5 Applications

Thanks to the high accuracy, fast speed and simple architecture of our network, we developed two real-world applications based on BASNet: **AR COPY & PASTE** and **OBJECT CUT**. These two applications further demonstrate the effectiveness and efficiency of our BASNet.



(a) Image (b) GT (c) BASNet (d) SINet (e) EGNet (f) PoolNet (g) CPD (h) PiCANet

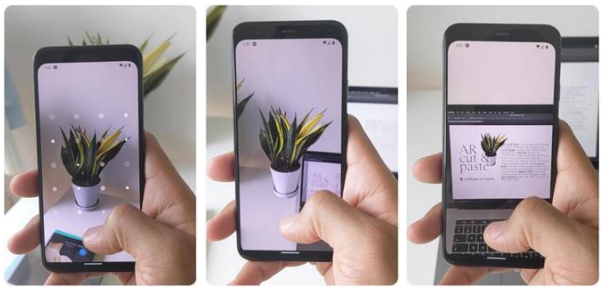
Fig. 11. Qualitative comparison on camouflaged object segmentation datasets. See § 4.4.2 for details.

5.1 Application I: AR COPY & PASTE

Introduced in HCI by Larry Tesler in the early 70s [107], cut/copy-paste has become essential to many applications of modern computing. In this section, we explore how BASNet can be used to apply this principle to the mobile phone camera and seamlessly transfer visual elements between the physical space and the digital space. AR COPY & PASTE is a prototype that we built upon our BASNet to conveniently capture real-world items using the camera of a mobile phone (*e.g.* objects, people, drawings, schematics, *etc*), automatically remove the background, and insert the result in an editing software on the computer by pointing the camera at it, as shown in Fig. 12. Specifically, AR COPY & PASTE first removes the background of the photo and only shows the foreground salient target on the mobile phone screen. Then users can "paste" the segmented target by moving the cellphone to point the mobile camera at a specific location of a document opened on a computer. The whole process of AR COPY & PASTE makes it seem like the real-world target is "copied" and "pasted" into a document, which provides a novel and inspiring interactive experience. A demonstration video[4] of the prototype has been released along with the source code.[5] Both have received world-wide attention (millions of views for the video, tens of thousands of Github stars for the source code). Hundreds of thousands of people have subscribed to the private beta.

5.1.1 Workflow of AR COPY & PASTE

From the perspective of users, our AR COPY & PASTE only consists of two main steps: "copy" and "paste", as shown in Fig. 13. **(1) Copy.** The first step consists in pointing the mobile camera at a subject and tapping the screen to take a picture. BASNet is then used to hide all the pixels that are not part of the main foreground subject. The remaining pixels keep floating on top of the camera and provide a preview of the paste result. Compared to other methods like image segmentation [77], BASNet can produce very accurate segmentation results with sharp and high-quality boundaries, which is essential in many image composition workflows. **(2) Paste.** The second step consists of pointing the mobile phone at a specific location on the computer screen and tapping to "paste" the "copied" subject. SIFT [71] (implemented in OpenCV [4]) is used to find the corresponding computer screen coordinates targeted by the center of the mobile phone camera. The image containing the background removed target is finally imported in an



(a) Copy (b) Move (c) Paste

Fig. 12. Screenshots from the video demonstration. (a) **Copy**: Point and tap to "copy" the object by masking its background out using BASNet. (b) **Move**: Move the mobile phone, where the "copied" object is shown, to target at the computer screen at a specific location. (c) **Paste**: Tap to "paste" the "copied" object into the current document.

4. https://twitter.com/cyrildiagne/status/1256916982764646402
5. https://github.com/cyrildiagne/ar-cutpaste

TABLE 6. Number of operations of different methods for image capturing and masking out.

Methods	Number of steps (on mobile)	Number of steps (on desktop)
Cross-platform (Android v11 and macOS v10.15)	6 ① Take a photo, ② Tap the thumbnail, ③ Tap share, ④ Tap more, ⑤ Select Bluetooth, ⑥ Tap the destination on a device.	5 ① Click "Open" on the Bluetooth notification, ② Export image, ③ Select destination software, ④ Use "Select Object" Tool, ⑤ Apply mask.
Constructor-specific (iOS v13 and macOS v10.15)	5 ① Take a photo, ② Tap the thumbnail, ③ Tap share, ④ Tap "AirDrop", ⑤ Tap the destination on a device.	5 ① Click "Open" on the AirDrop notification, ② Export image, ③ Select destination software, ④ Use "Select Object" Tool, ⑤ Apply mask.
AR COPY & PASTE (Ours)	2 ① Take a photo, ② Tap toward the destination software.	0

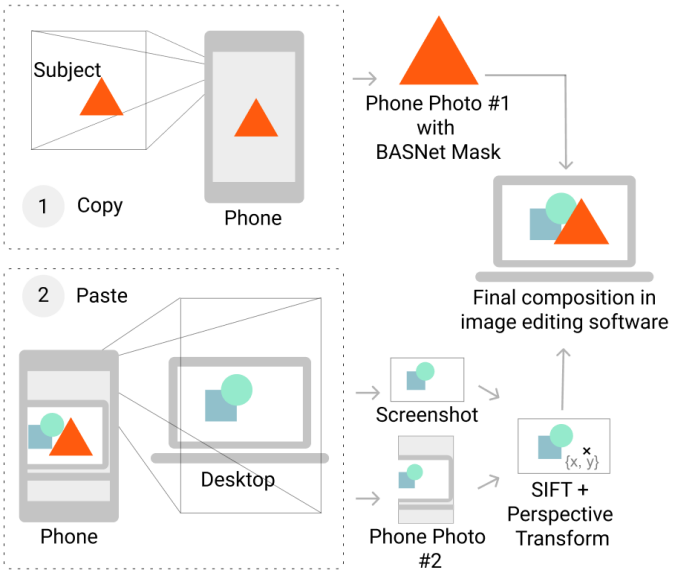


Fig. 13. Schematic of the AR COPY & PASTE flow.

image editing software at the computed screen coordinates to create the final composition.

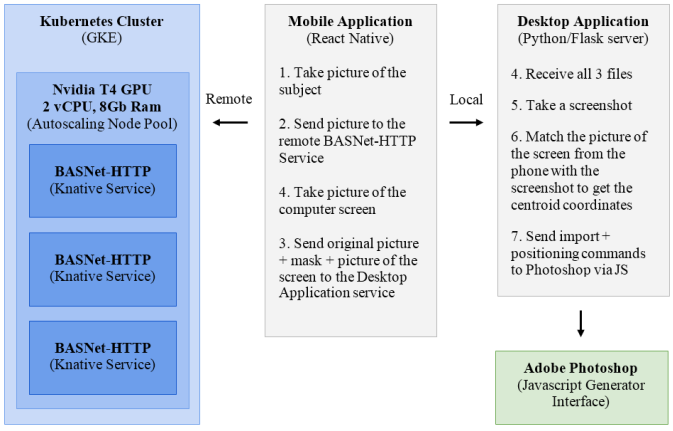


Fig. 14. Overall pipeline of the AR COPY & PASTE.

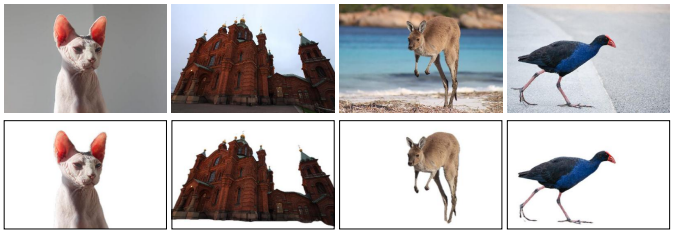


Fig. 15. Sample results given by the OBJECT CUT API: the first row shows the input images and the second row shows the background removed results.

5.1.2 *Implementation Details*

Fig. 14 illustrates the overall implementation pipeline of the AR COPY & PASTE prototype, which consists of three main parts: Kubernetes cluster, mobile application and desktop application. The AR COPY & PASTE prototype was built using our BASNet model trained on DUTS-TR [110]. To make sure that it runs smoothly on mobile device, it has been wrapped as an HTTP service/container image[6] so that it can be deployed easily on remote GPUs using Kubernetes [7]. Hence, photos taken by mobile devices in AR COPY & PASTE are sent to the remote server to obtain their segmentation masks. The desktop application is a python HTTP server which takes three files from the mobile application as input (original picture, BASNet mask, photo of the screen) and runs SIFT feature matching and perspective transformation based on the photo of the screen and the screenshot to determine the destination coordinates. Finally, the desktop application is responsible for sending javascript commands to desktop apps like Photoshop[8] in order to import the image into the document at the right position.

5.1.3 *Comparison with Other Methods*

Our AR COPY & PASTE prototype applies BASNet in a novel human-computer-interaction setting, which makes the process easier and faster than other methods (two operations for our method versus 10 or 11 operations for other methods). Table 6 illustrates examples of typical user flow

6. https://github.com/cyrildiagne/basnet-http
7. https://github.com/kubernetes/kubernetes
8. https://www.photoshop.com/en

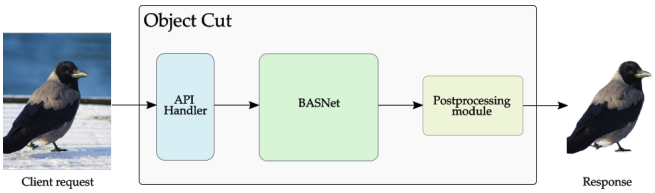


Fig. 16. OBJECT CUT pipeline.

to clip and import an object from a mobile devices to an desktop image editing software, such as Adobe Photoshop. As we can see, our prototype greatly reduces the numbers of operations and simplifies the process. Besides, our AR COPY & PASTE allows users to delegate some of the lower-level decisions (how visible each pixel should be), and focus on the higher-level objectives (how do they want the object to look). Removing these tasks lowers the barrier to entry (there is no need to learn how to paint masks in an image editing software), saves a significant amount of time, and ultimately leads to better end results by removing the cognitive load of the low-level tasks [5].

5.2 Application II: OBJECT CUT

OBJECT CUT is an online image background removal service that uses BASNet. Removing the background from an image is a common operation in the daily work of professional photographers and image editors. This process is usually a repeatable and manual task that requires a lot of effort and time. However, thanks to BASNet, one of the most robust and fastest performing deep learning models for image segmentation, OBJECT CUT was able to turn it into an easy and automatic process. The program was built as an API to make it as easy as possible to integrate. APIs, also known as Application Programming Interfaces, are already commonly used to integrate different types of solutions to improve systems without actually knowing what is happening inside. For instance, RESTful APIs are a standard in the software engineering field for designing and specifying APIs. Making it substantially easier to adapt desired APIs to specific workflows.

Our system is based on three well-distinguished parts, as shown in Fig. 16: 1) the API Handler, responsible for receiving and validating clients requests, downloading and preprocessing input images and sending those to the model; 2) BASNet, responsible for performing salient object detection. 3) Once the output from the network is generated, the postprocessing module applies an unsharp masking algorithm and morphological operations to improve the quality of the output. Afterward, OBJECT CUT uploads the cropped image to the Google Cloud Storage and returns its public URL to the client. This is structured as-is in order to isolate different parts, such as the BASNet component, removing all the API parameter validations as well as image download and upload processes, as much as possible. In this scenario, OBJECT CUT maximizes the operations running on the BASNet thread. The whole stack from the API is running under Docker containers, all managed by the cloud native application proxy called Traefik. The usage of Traefik here allows us to have a production-ready deployment making easy, from the containers' perspective, to communicate with other processes. In addition, we have a Load Balancer system in place to enable each of the components to scale more easily. The source code for this pipeline can be found under the OBJECT CUT GitHub repository: https://github.com/AlbertSuarez/object-cut.

To ensure easy integration, it is publicly available at RapidAPI[9], the biggest API marketplace in the world, and it has been effectively utilized by people and companies from 86 different countries around the globe, including China, United States, Canada, India, Spain, Russia, *etc*. OBJECT CUT was born to power up the designing and image editing process for the people who work with images daily. Integrating the OBJECT CUT API removes the necessity of understanding the complex inner workings behind it and automates the process of removing the background from images in a matter of seconds. See examples in Fig. 15.

6 Conclusion

In this paper, we proposed a novel end-to-end boundary-aware model, BASNet, and a hybrid fusion loss for accurate image segmentation. The proposed BASNet is a predict-refine architecture, which consists of two components: a prediction network and a refinement module. Combined with the hybrid loss, BASNet is able to capture both large-scale and fine structures, *e.g.* thin regions, holes, and produce segmentation probability maps with highly accurate boundaries. Experimental results on six salient object segmentation datasets, one salient object in clutter dataset and three camouflaged object segmentation datasets demonstrate that our model achieves very competitive performance in terms of both region-based and boundary-aware measures against other models. Additionally, the proposed network architecture is modular. It can be easily extended or adapted to other tasks by replacing either the prediction network or the refinement module. The two (close to) commercial applications, AR COPY & PASTE and OBJECT CUT, based on our BASNet not only prove the effectiveness and efficiency of our model but also provide two practical tools for reducing the workload in real-world production scenarios. The world-wide impacts of these two applications indicates the huge demand for highly accurate segmentation approaches, which motivates us to explore more accurate and reliable segmentation models.

References

[1] Amir A Amini, Terry E Weymouth, and Ramesh C Jain. Using dynamic programming for solving variational problems in vision. *TPAMI*, (9):855–867, 1990.
[2] Arnon Amir and Michael Lindenbaum. A generic grouping algorithm and its quantitative analysis. *TPAMI*, 20(2):168–185, 1998.
[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *TPAMI*, (12):2481–2495, 2017.
[4] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.
[5] Shan Carter and Michael Nielsen. Using artificial intelligence to augment human intelligence. *Distill*, 2(12):e9, 2017.
[6] Vicent Caselles, Francine Catté, Tomeu Coll, and Françoise Dibos. A geometric model for active contours in image processing. *Numerische mathematik*, 66(1):1–31, 1993.

9. https://rapidapi.com/objectcut.api/api/background-removal

[7] Vicent Caselles, Ron Kimmel, and Guillermo Sapiro. Geodesic active contours. *IJCV*, 22(1):61–79, 1997.

[8] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In *CVPR*, pages 4485–4493, 2017.

[9] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *CVPR*, pages 4974–4983, 2019.

[10] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. 2015.

[11] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 40(4):834–848, 2017.

[12] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[13] Shuhan Chen, Xiuli Tan, Ben Wang, and Xuelong Hu. Reverse attention for salient object detection. In *ECCV*, pages 236–252, 2018.

[14] Shuhan Chen, Xiuli Tan, Ben Wang, Huchuan Lu, Xuelong Hu, and Yun Fu. Reverse attention-based residual network for salient object detection. *TIP*, 29:3763–3776, 2020.

[15] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, pages 3213–3223, 2016.

[16] Ingemar J Cox, Satish B Rao, and Yu Zhong. " ratio regions": a technique for image segmentation. In *ICPR*, volume 2, pages 557–564, 1996.

[17] Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals OR*, 134(1):19–67, 2005.

[18] Zijun Deng, Xiaowei Hu, Lei Zhu, Xuemiao Xu, Jing Qin, Guoqiang Han, and Pheng-Ann Heng. R3net: Recurrent residual refinement network for saliency detection. pages 684–690. IJCAI, 2018.

[19] Marc Ehrig and Jérôme Euzenat. Relaxed precision and recall for ontology matching. In *Proc. K-Cap 2005 workshop on Integrating ontology*, pages 25–32. No commercial editor., 2005.

[20] James Elder and Steven Zucker. The effect of contour closure on the rapid discrimination of two-dimensional shapes. *Vision Research*, 33(7):981–991, 1993.

[21] James H. Elder, Amnon Krupnik, and Leigh A. Johnston. Contour grouping with prior models. *TPAMI*, 25(6):661–674, 2003.

[22] Deng-Ping Fan, Ming-Ming Cheng, Yun Liu, Tao Li, and Ali Borji. Structure-measure: A new way to evaluate foreground maps. In *ICCV*, pages 4558–4567, 2017.

[23] Deng-Ping Fan, Cheng Gong, Yang Cao, Bo Ren, Ming-Ming Cheng, and Ali Borji. Enhanced-alignment measure for binary foreground map evaluation. In *IJCAI*, pages 698–704, 2018.

[24] Deng-Ping Fan, Ming-Ming Cheng, Jiang-Jiang Liu, Shang-Hua Gao, Qibin Hou, and Ali Borji. Salient objects in clutter: Bringing salient object detection to the foreground. In *ECCV*, 2018.

[25] Deng-Ping Fan, Ge-Peng Ji, Guolei Sun, Ming-Ming Cheng, Jianbing Shen, and Ling Shao. Camouflaged object detection. In *CVPR*, pages 2777–2787, 2020.

[26] Mengyang Feng, Huchuan Lu, and Errui Ding. Attentive feedback network for boundary-aware salient object detection. In *CVPR*, pages 1623–1632, 2019.

[27] Lucas Fidon, Wenqi Li, Luis C. Herrera, Jinendra Ekanayake, Neil Kitchen, Sébastien Ourselin, and Tom Vercauteren. Generalised wasserstein dice score for imbalanced multi-class segmentation using holistic convolutional networks. In *MICCAI-W*, pages 64–76, 2017.

[28] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *IJRR*, 32(11):1231–1237, 2013.

[29] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014.

[30] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, pages 249–256, 2010.

[31] Stas Goferman, Lihi Zelnik-Manor, and Ayellet Tal. Context-aware saliency detection. *TPAMI*, 34(10):1915–1926, 2012.

[32] Richard HR Hahnloser and H Sebastian Seung. Permitted and forbidden sets in symmetric threshold-linear networks. In *NIPS*, pages 217–223, 2001.

[33] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, pages 2961–2969, 2017.

[34] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, pages 346–361. Springer, 2014.

[35] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[36] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip Torr. Deeply supervised salient object detection with short connections. In *CVPR*, pages 5300–5309, 2017.

[37] Ping Hu, Bing Shuai, Jun Liu, and Gang Wang. Deep level sets for salient object detection. In *CVPR*, volume 1, page 2, 2017.

[38] Xiaowei Hu, Chi-Wing Fu, Lei Zhu, Tianyu Wang, and Pheng-Ann Heng. Sac-net: Spatial attenuation context for salient object detection. *TCSVT*, 2020.

[39] Xiaowei Hu, Lei Zhu, Jing Qin, Chi-Wing Fu, and Pheng-Ann Heng. Recurrently aggregating deep features for salient object detection. In *AAAI*, pages 6943–6950, 2018.

[40] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, pages 2261–2269, 2017.

[41] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask scoring r-cnn. In *CVPR*, pages 6409–6418, 2019.

[42] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML*, pages 448–456, 2015.

[43] Md Amirul Islam, Mahmoud Kalash, Mrigank Rochan, Neil DB Bruce, and Yang Wang. Salient object detection using a context-aware refinement network. In *BMVC*, 2017.

[44] Paul Jaccard. The distribution of the flora in the alpine zone. 1. *New phytologist*, 11(2):37–50, 1912.

[45] Martin Jägersand. Saliency maps and attention selection in scale and spatial coordinates: An information theoretic approach. In *ICCV*, pages 195–202, 1995.

[46] Timor Kadir and Michael Brady. Saliency, scale and image description. *International Journal of Computer Vision*, 45(2):83–105, 2001.

[47] G. Kanizsa. *Organization in Vision*. New York: Praeger 1979.

[48] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *IJCV*, 1(4):321–331, 1988.

[49] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[50] Srinivas SS Kruthiventi, Vennela Gudisa, Jaley H Dholakiya, and R Venkatesh Babu. Saliency unified: A deep architecture for simultaneous eye fixation prediction and salient object segmentation. In *CVPR*, pages 5781–5790, 2016.

[51] Jason Kuen, Zhenhua Wang, and Gang Wang. Recurrent attentional networks for saliency detection. In *CVPR*, pages 3668–3677, 2016.

[52] M Pawan Kumar, PHS Ton, and Andrew Zisserman. Obj cut. In *CVPR*, volume 1, pages 18–25, 2005.

[53] Lubor Ladicky, Chris Russell, Pushmeet Kohli, and Philip HS Torr. Graph cut based inference with co-occurrence statistics. In *ECCV*, pages 239–253, 2010.

[54] Trung-Nghia Le, Tam V Nguyen, Zhongliang Nie, Minh-Triet Tran, and Akihiro Sugimoto. Anabranch network for camouflaged object segmentation. *CVIU*, 184:45–56, 2019.

[55] Trung-Nghia Le, Vuong Nguyen, Cong Le, Tan-Cong Nguyen, Minh-Triet Tran, and Tam V Nguyen. Camoufinder: Finding camouflaged instances in images. In *AAAI*, 2021.

[56] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial Intelligence and Statistics*, pages 562–570, 2015.

[57] Hyemin Lee and Daijin Kim. Salient region-based online object tracking. In *WACV*, pages 1170–1177, 2018.

[58] Guanbin Li and Yizhou Yu. Visual saliency based on multiscale deep features. In *CVPR*, pages 5455–5463, 2015.

[59] Guanbin Li and Yizhou Yu. Deep contrast learning for salient object detection. In *CVPR*, pages 478–487, 2016.

[60] Guanbin Li and Yizhou Yu. Visual saliency detection based on multiscale deep cnn features. *TIP*, 25(11):5012–5024, 2016.

[61] Xin Li, Fan Yang, Hong Cheng, Wei Liu, and Dinggang Shen. Contour knowledge transfer for salient object detection. In *ECCV*, pages 370–385, 2018.

[62] Yin Li, Xiaodi Hou, Christof Koch, James M Rehg, and Alan L Yuille. The secrets of salient object segmentation. In *CVPR*, pages 280–287, 2014.

[63] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid.

Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *CVPR*, pages 1925–1934, 2017.
[64] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017.
[65] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014.
[66] Jiang-Jiang Liu, Qibin Hou, Ming-Ming Cheng, Jiashi Feng, and Jianmin Jiang. A simple pooling-based design for real-time salient object detection. In *CVPR*, pages 3917–3926, 2019.
[67] Nian Liu and Junwei Han. Dhsnet: Deep hierarchical saliency network for salient object detection. In *CVPR*, pages 678–686, 2016.
[68] Nian Liu, Junwei Han, and Ming-Hsuan Yang. Picanet: Learning pixel-wise contextual attention for saliency detection. In *CVPR*, pages 3089–3098, 2018.
[69] Nian Liu, Junwei Han, Dingwen Zhang, Shifeng Wen, and Tianming Liu. Predicting eye fixations using convolutional neural networks. In *CVPR*, pages 362–370, 2015.
[70] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015.
[71] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.
[72] Zhiming Luo, Akshaya Mishra, Andrew Achkar, Justin Eichel, Shaozi Li, and Pierre-Marc Jodoin. Non-local deep features for salient object detection. In *CVPR*, pages 6593–6601, 2017.
[73] Shyjan Mahamud, Lance R. Williams, Karvel K. Thornber, and Kanglin Xu. Segmentation of multiple salient closed contours from real images. *TPAMI*, 25(4):433–444, 2003.
[74] Ran Margolin, Lihi Zelnik-Manor, and Ayellet Tal. How to evaluate foreground maps. *CVPR*, pages 248–255, 2014.
[75] Gellért Máttyus, Wenjie Luo, and Raquel Urtasun. Deeproadmapper: Extracting road topology from aerial images. In *ICCV*, pages 3458–3466, 2017.
[76] Roey Mechrez, Eli Shechtman, and Lihi Zelnik-Manor. Saliency driven image manipulation. In *WACV*, pages 1368–1376, 2018.
[77] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *arXiv preprint arXiv:2001.05566*, 2020.
[78] Vida Movahedi and James H Elder. Design and perceptual validation of performance measures for salient object segmentation. In *CVPR-W*, pages 49–56, 2010.
[79] David Mumford and Jayant Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *CPAM*, 42(5):577–685, 1989.
[80] Gattigorla Nagendar, Digvijay Singh, Vineeth N. Balasubramanian, and C. V. Jawahar. Neuro-iou: Learning a surrogate loss for semantic segmentation. In *BMVC*, page 278, 2018.
[81] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, pages 483–499, 2016.
[82] Stanley Osher and James A Sethian. Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations. *Journal of computational physics*, 79(1):12–49, 1988.
[83] Youwei Pang, Xiaoqi Zhao, Lihe Zhang, and Huchuan Lu. Multi-scale interactive network for salient object detection. In *CVPR*, pages 9413–9422, 2020.
[84] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.
[85] Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters—improve semantic segmentation by global convolutional network. In *CVPR*, pages 1743–1751, 2017.
[86] Federico Perazzi, Philipp Krähenbühl, Yael Pritch, and Alexander Hornung. Saliency filters: Contrast based filtering for salient region detection. In *CVPR*, pages 733–740, 2012.
[87] Xuebin Qin. Visual salient object detection: Interactive, unsupervised and supervised methods. *Doctoral dissertation*, 2020.
[88] Xuebin Qin, Shida He, Camilo Perez Quintero, Abhineet Singh, Masood Dehghan, and Martin Jagersand. Real-time salient closed boundary tracking via line segments perceptual grouping. In *IROS*, pages 4284–4289, 2017.
[89] Xuebin Qin, Shida He, Camilo Perez Quintero, Abhineet Singh, Masood Dehghan, and Martin Jägersand. Real-time salient closed boundary tracking via line segments perceptual grouping. In *IROS*, pages 4284–4289, 2017.
[90] Xuebin Qin, Shida He, Xiucheng Yang, Masood Dehghan, Qiming Qin, and Martin Jagersand. Accurate outline extraction of individual building from very high-resolution optical images. *GRSL*, (99):1–5, 2018.
[91] Xuebin Qin, Shida He, Zichen Zhang, Masood Dehghan, and Martin Jagersand. Bylabel: A boundary based semi-automatic image annotation tool. In *WACV*, pages 1804–1813, 2018.
[92] Xuebin Qin, Shida He, Zichen Vincent Zhang, and Masood Dehghan. Real-time salient closed boundary tracking using perceptual grouping and shape priors. In *BMVC*, 2017.
[93] Xuebin Qin, Shida He, Zichen Vincent Zhang, Masood Dehghan, and Martin Jägersand. Real-time salient closed boundary tracking using perceptual grouping and shape priors. In *BMVC*, 2017.
[94] Xuebin Qin, Zichen Zhang, Chenyang Huang, Masood Dehghan, Osmar R Zaiane, and Martin Jagersand. U2-net: Going deeper with nested u-structure for salient object detection. *PR*, 106:107404, 2020.
[95] Xuebin Qin, Zichen Zhang, Chenyang Huang, Chao Gao, Masood Dehghan, and Martin Jagersand. Basnet: Boundary-aware salient object detection. In *CVPR*, pages 7479–7489, 2019.
[96] Md Atiqur Rahman and Yang Wang. Optimizing intersection-over-union in deep neural networks for image segmentation. In *ISVC*, pages 234–244. Springer, 2016.
[97] Xiaofeng Ren, Charless C. Fowlkes, and Jitendra Malik. Scale-invariant contour completion using conditional random fields. In *ICCV*, pages 1214–1221, 2005.
[98] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241. Springer, 2015.
[99] Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Labelme: A database and web-based tool for image annotation. *IJCV*, 77(1-3):157–173, 2008.
[100] Sudeep Sarkar and Padmanabhan Soundararajan. Supervised learning of large perceptual organization: Graph spectral partitioning and learning automata. *TPAMI*, 22(5):504–525, 2000.
[101] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Departmental Papers (CIS)*, page 107, 2000.
[102] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015.
[103] Przemysław Skurowski, Hassan Abdulameer, Jakub Błaszczyk, Tomasz Depta, Adam Kornacki, and Przemysław Kozieł. Animal camouflage analysis: Chameleon database. 2018.
[104] Jinming Su, Jia Li, Yu Zhang, Changqun Xia, and Yonghong Tian. Selectivity or invariance: Boundary-aware salient object detection. In *ICCV*, pages 3799–3808, 2019.
[105] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015.
[106] Zhiqiang Tang, Xi Peng, Shijie Geng, Lingfei Wu, Shaoting Zhang, and Dimitris Metaxas. Quantized densely connected u-nets for efficient landmark localization. In *ECCV*, 2018.
[107] Larry Tesler. A personal history of modeless text editing and cut/copy-paste. *interactions*, 19(4):70–75, 2012.
[108] Luc Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *TPAMI*, (6):583–598, 1991.
[109] Lijun Wang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang. Deep networks for saliency detection via local estimation and global search. In *CVPR*, pages 3183–3192, 2015.
[110] Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan. Learning to detect salient objects with image-level supervision. In *CVPR*, pages 136–145, 2017.
[111] Linzhao Wang, Lijun Wang, Huchuan Lu, Pingping Zhang, and Xiang Ruan. Salient object detection with recurrent fully convolutional networks. *TPAMI*, 2018.
[112] Song Wang and Jeffrey Mark Siskind. Image segmentation with ratio cut. *TPAMI*, 25(6):675–690, 2003.
[113] Tiantian Wang, Ali Borji, Lihe Zhang, Pingping Zhang, and Huchuan Lu. A stagewise refinement model for detecting salient objects in images. In *ICCV*, pages 4039–4048, 2017.
[114] Tinghuai Wang, Bo Han, and John P. Collomosse. Touchcut: Fast image and video segmentation using single-touch interaction. *CVIU*, 120:14–30, 2014.
[115] Tiantian Wang, Lihe Zhang, Shuo Wang, Huchuan Lu, Gang Yang, Xiang Ruan, and Ali Borji. Detect globally, refine locally: A novel approach to saliency detection. In *CVPR*, pages 3127–3135, 2018.

[116] Wenguan Wang, Qiuxia Lai, Huazhu Fu, Jianbing Shen, Haibin Ling, and Ruigang Yang. Salient object detection in the deep learning era: An in-depth survey. *arXiv preprint arXiv:1904.09146*, 2019.
[117] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *ACSSC*, volume 2, pages 1398–1402, 2003.
[118] Donna J. Williams and Mubarak Shah. A fast algorithm for active contours. In *ICCV*, pages 592–595, 1990.
[119] Runmin Wu, Mengyang Feng, Wenlong Guan, Dong Wang, Huchuan Lu, and Errui Ding. A mutual learning method for salient object detection with intertwined multi-supervision. In *CVPR*, pages 8150–8159, 2019.
[120] Zhenyu Wu and Richard Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *TPAMI*, (11):1101–1113, 1993.
[121] Zhe Wu, Li Su, and Qingming Huang. Cascaded partial decoder for fast and accurate salient object detection. In *CVPR*, pages 3907–3916, 2019.
[122] Zhe Wu, Li Su, and Qingming Huang. Stacked cross refinement network for edge-aware salient object detection. In *ICCV*, pages 7263–7272, 2019.
[123] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *ICCV*, pages 1395–1403, 2015.
[124] Chenyang Xu, Jerry L Prince, et al. Snakes, shapes, and gradient vector flow. *TIP*, 7(3):359–369, 1998.
[125] Ning Xu, Brian Price, Scott Cohen, and Thomas Huang. Deep image matting. In *CVPR*, pages 2970–2979, 2017.
[126] Jinnan Yan, Trung-Nghia Le, Khanh-Duy Nguyen, Minh-Triet Tran, Thanh-Toan Do, and Tam V Nguyen. Mirrornet: Bio-inspired adversarial attack for camouflaged object segmentation. *arXiv preprint arXiv:2007.12881*, 2020.
[127] Qiong Yan, Li Xu, Jianping Shi, and Jiaya Jia. Hierarchical saliency detection. In *CVPR*, pages 1155–1162, 2013.
[128] Chuan Yang, Lihe Zhang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang. Saliency detection via graph-based manifold ranking. In *CVPR*, pages 3166–3173, 2013.
[129] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *ICLR*, 2016.
[130] Zhiding Yu, Chen Feng, Ming-Yu Liu, and Srikumar Ramalingam. Casenet: Deep category-aware semantic edge detection. In *CVPR*, pages 5964–5973, 2017.
[131] Yu Zeng, Yunzhi Zhuge, Huchuan Lu, Lihe Zhang, Mingyang Qian, and Yizhou Yu. Multi-source weak supervision for saliency detection. In *CVPR*, pages 6074–6083, 2019.
[132] Jing Zhang, Deng-Ping Fan, Yuchao Dai, Saeed Anwar, Fatemeh Saleh, Sadegh Aliakbarian, and Nick Barnes. Uncertainty inspired rgb-d saliency detection. *CVPR*, pages 8579–8588, 2020.
[133] Lu Zhang, Ju Dai, Huchuan Lu, You He, and Gang Wang. A bi-directional message passing model for salient object detection. In *CVPR*, pages 1741–1750, 2018.
[134] Lu Zhang, Jianming Zhang, Zhe Lin, Huchuan Lu, and You He. Capsal: Leveraging captioning to boost semantics for salient object detection. In *CVPR*, pages 6024–6033, 2019.
[135] Pingping Zhang, Wei Liu, Huchuan Lu, and Chunhua Shen. Salient object detection by lossless feature reflection. In *IJCAI*, pages 1149–1155, 2018.
[136] Pingping Zhang, Dong Wang, Huchuan Lu, Hongyu Wang, and Xiang Ruan. Amulet: Aggregating multi-level convolutional features for salient object detection. In *ICCV*, pages 202–211, 2017.
[137] Pingping Zhang, Dong Wang, Huchuan Lu, Hongyu Wang, and Baocai Yin. Learning uncertain convolutional features for accurate saliency detection. In *ICCV*, pages 212–221, 2017.
[138] Xiaoning Zhang, Tiantian Wang, Jinqing Qi, Huchuan Lu, and Gang Wang. Progressive attention guided recurrent network for salient object detection. In *CVPR*, pages 714–722, 2018.
[139] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, pages 2881–2890, 2017.
[140] Jia-Xing Zhao, Jiang-Jiang Liu, Deng-Ping Fan, Yang Cao, Jufeng Yang, and Ming-Ming Cheng. Egnet: Edge guidance network for salient object detection. In *ICCV*, pages 8779–8788, 2019.
[141] Kai Zhao, Shanghua Gao, Qibin Hou, Dandan Li, and Ming-Ming Cheng. Optimizing the f-measure for threshold-free salient object detection. *CoRR*, abs/1805.07567, 2018.
[142] Rui Zhao, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Saliency detection by multi-context deep learning. In *CVPR*, pages 1265–1274, 2015.
[143] Ting Zhao and Xiangqian Wu. Pyramid feature attention network for saliency detection. In *CVPR*, pages 3085–3094, 2019.
[144] Xiaoqi Zhao, Youwei Pang, Lihe Zhang, Huchuan Lu, and Lei Zhang. Suppress and balance: A simple gated network for salient object detection. 2020.
[145] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017.
[146] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. In *DLMIA*, pages 3–11. 2018.

Xuebin QIN obtained his PhD degree from the University of Alberta, Edmonton, Canada, in 2020. Since March, 2020, He is a Postdoctoral Fellow in the Department of Computing Science and the Department of Radiology and Diagnostic Imaging, University of Alberta, Canada. His research interests include highly accurate image segmentation, salient object detection, image labeling and detection. He has published about 10 papers in vision and robotics conferences such as CVPR, BMVC, ICPR, WACV, IROS, etc.

Deng-Ping FAN received his PhD degree from the Nankai University in 2019. He joined Inception Institute of Artificial Intelligence (IIAI) in 2019. He has published about 25 top journal and conference papers such as CVPR, ICCV, ECCV, etc. His research interests include computer vision and visual attention, especially on RGB salient object detection (SOD), RGB-D SOD, Video SOD, Co-SOD. He won the Best Paper Finalist Award at IEEE CVPR 2019, the Best Paper Award Nominee at IEEE CVPR 2020.

Chenyang Huang obtained his M.Sc. degree from the University of Alberta, Edmonton, Canada, in 2019. He is currently pursuing a Ph.D. degree in the Department of Computing Science of the same university. His research is mainly focusing on deep learning, natural language processing, and computer vision. He has publications on some prestigious conferences such as NAACL and CVPR.

Cyril Diagne is a designer and coder and a co-founder of Init ML, a company that brings machine learning to production through practical uses, such as ClipDrop. Cyril is a former Professor and Head of Media & Interaction Design at ECAL (Lausanne University of Arts & Design, Switzerland) where he continues to give regular workshops. In 2015, he started a residency at Google Arts & Culture, where he helped kickstart the Google Arts Experiments initiative and created multiple machine learning projects such as the viral phenomenon Art Selfie.

Zichen Zhang is a Ph.D. student in Statistical Machine Learning at the University of Alberta. He obtained his M.Sc degrees from Dalhousie University and the University of Alberta and B.E degree from Huazhong University of Science and Technology. He's interested in machine learning and its applications in robotics perception and control.

Adrià Cabeza Sant'Anna is a computer engineer who graduated at Universitat Politècnica de Catalunya, BarcelonaTech in Computer Science. His current position is Deep Learning Engineer at restb.ai, a Computer Vision company for Real Estate based in Barcelona. Previously, he worked as an Algorithmic Methods of Data Mining grader assistant in the Department of Computer Science at Aalto University, Helsinki. He was the president of the Student Representatives Association at Barcelona School of Informatics. His research interests include machine learning, computer vision, generative models, and data mining. He has co-developed ObjectCut and participates in the organization of Hack-UPC, the biggest student-run hackathon in Europe, located at Barcelona School of Informatics.

Albert Suàrez is a software engineer who graduated at Universitat Politècnica de Catalunya, BarcelonaTech in Software Engineering. His current position is Principal Software Engineer at restb.ai, a Computer Vision company for Real Estate based in Barcelona, Spain. He was the co-director of the biggest student-run hackathon in Europe, called HackUPC, located at Barcelona School of Informatics.

Martin Jagersand's research interests are in Robotics, Computer Vision, and Graphics, especially vision guided motion control and vision-based human-robot interfaces. He studied physics at Chalmers Sweden (MSc 1991). He was awarded a Fulbright fellowship for graduate studies in the USA. He studied Computer Science at the Univ. of Rochester, NY (MSc 1994, PhD 1997). He held an NSF CISE postdoc fellowship, at Yale University, and then was a research faculty in the Engineering Research Center for Surgical Systems and Technology at Johns Hopkins University. He is now a faculty member at the University of Alberta.

Ling Shao is currently the CEO and the Chief Scientist of the Inception Institute of Artificial Intelligence, Abu Dhabi, United Arab Emirates. He is also the Executive Vice President and a Provost of the Mohamed bin Zayed University of Artificial Intelligence. His current research interests include computer vision, machine learning, and medical imaging. Dr. Shao is a fellow of IAPR, IET, and BCS. He is an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and several other top journals.