BASNet: Boundary-Aware Salient Object Detection

Xuebin Qin, Zichen Zhang, Chenyang Huang, Chao Gao, Masood Dehghan and Martin Jagersand
University of Alberta, Canada
{xuebin,vincent.zhang,chuang8,cgao3,masood1,mj7}@ualberta.ca

Abstract

Deep Convolutional Neural Networks have been adopted for salient object detection and achieved the state-of-the-art performance. Most of the previous works however focus on region accuracy but not on the boundary quality. In this paper, we propose a predict-refine architecture, BASNet, and a new hybrid loss for Boundary-Aware Salient object detection. Specifically, the architecture is composed of a densely supervised Encoder-Decoder network and a residual refinement module, which are respectively in charge of saliency prediction and saliency map refinement. The hybrid loss guides the network to learn the transformation between the input image and the ground truth in a three-level hierarchy – pixel-, patch- and map- level – by fusing Binary Cross Entropy (BCE), Structural SIMilarity (SSIM) and Intersection-over-Union (IoU) losses. Equipped with the hybrid loss, the proposed predict-refine architecture is able to effectively segment the salient object regions and accurately predict the fine structures with clear boundaries. Experimental results on six public datasets show that our method outperforms the state-of-the-art methods both in terms of regional and boundary evaluation measures. Our method runs at over 25 fps on a single GPU. The code is available at: https://github.com/NathanUA/BASNet

1. Introduction

The human vision system has an effective attention mechanism for choosing the most important information from visual scenes. Computer vision aims at modeling this mechanism in two research branches: eye-fixation detection [20] and salient object detection [3]. Our work focuses on the second branch and aims at accurately segmenting the pixels of salient objects in an input image. The results have immediate applications in e.g. image segmentation/editing [53, 25, 11, 54] and salient object detection [5]. Our method runs at over 25 fps on a single GPU. The code is available at: https://github.com/NathanUA/BASNet

Figure 1. Sample result of our method (BASNet) compared to PiCANetR [39]. Column (a) shows the input image, zoom-in view of ground truth (GT) and the boundary map, respectively. (b), (c) and (d) are results of ours, PiCANetR and PiCANetRC (PiCANetR with CRF [27] post-processing). For each method, the three rows respectively show the predicted saliency map, the zoom-in view of saliency map and the zoom-in view of boundary map.

There are two main challenges in accurate salient object detection: (i) the saliency is mainly defined over the global contrast of the whole image rather than local or pixel-wise features. To achieve accurate results, the developed saliency detection methods have to understand the global meaning of the whole image as well as the detailed structures of the objects [6]. To address this problem, networks that aggregate multi-level deep features are needed; (ii) Most of the salient object detection methods use Cross Entropy (CE) as their training loss. But models trained with CE loss usually have low confidence in differentiating boundary pixels, leading to blurry boundaries. Other losses such as Intersection over Union (IoU) loss [56, 42, 47], F-measure loss [78] and Dice-score loss [8] were proposed for biased training sets but they are not specifically designed for capturing fine
to traditional methods, their predicted saliency maps are still defective in fine structures and/or boundaries (see Figs. 1(c), 1(d)).
structures.

To address the above challenges, we propose a novel Boundary-Aware network, namely BASNet, for Salient object detection, which achieves accurate salient object segmentation with high quality boundaries (see Fig. 1(b)). (i) To capture both global (coarse) and local (fine) contexts, a new predict-refine network is proposed. It assembles a U-Net-like [57] deeply supervised [31, 67] Encoder-Decoder network with a novel residual refinement module. The Encoder-Decoder network transfers the input image to a probability map, while the refinement module refines the predicted map by learning the residuals between the coarse saliency map and ground truth (see Fig. 2). In contrast to [50, 22, 6], which use refinement modules iteratively on saliency predictions or intermediate feature maps at multiple scales, our module is used only once on the original scale for saliency prediction. (ii) To obtain high confidence saliency map and clear boundary, we propose a hybrid loss that combines Binary Cross Entropy (BCE) [5], Structural SIMilarity (SSIM) [60] and IoU losses [42], which are expected to learn from ground truth information in pixel-, patch- and map- level, respectively. Rather than using explicit boundary losses (NLDF+ [41], C2S [36]), 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 on the boundary and the other regions on the image.

The main contributions of this work are:

- A novel boundary-aware salient object detection network: BASNet, which consists of a deeply supervised encoder-decoder and a residual refinement module,
- A novel hybrid loss that fuses BCE, SSIM and IoU to supervise the training process of accurate salient object prediction on three levels: pixel-level, patch-level and map-level,
- A thorough evaluation of the proposed method that includes comparison with 15 state-of-the-art methods on six widely used public datasets. Our method achieves state-of-the-art results in terms of both regional and boundary evaluation measures.

2. Related Works

Traditional Methods: Early methods detect salient objects by searching for pixels according to a predefined saliency measure computed based on handcrafted features [69, 80, 60, 71]. Borji et al. provide a comprehensive survey in [3].

Patch-wise Deep Methods: Encouraged by the advancement on image classification of Deep CNNs [28, 59], early deep salient object detection methods search for salient objects by classifying image pixels or super pixels into salient or non-salient classes based on the local image patches extracted from single or multiple scales [33, 40, 61, 79, 35]. These methods usually generate coarse outputs because spatial information are lost in the fully connected layers.

FCN-based Methods: Salient object detection methods based on FCN [34, 29] achieve significant improvement compared with patch-wise deep methods, presumably because FCN is able to capture richer spatial and multi-scale information. Zhang et al. (UCF) [75] developed a reformulated dropout and a hybrid upsampling module to reduce the checkerboard artifacts of deconvolution operators as well as aggregating multi-level convolutional features in (Amulet) [74] for saliency detection. Hu et al. [18] proposed to learn a Level Set [48] function to output accurate boundaries and compact saliency. Luo et al. [41] designed a network (NLDF+) with a 4×5 grid structure to combine local and global information and used a fusing loss of cross entropy and boundary IoU inspired by Mumford-Shah [46]. Hou et al. (DSS+) [17] adopted Holistically-Nested Edge Detector (HED) [67] by introducing short connections to its skip-layers for saliency prediction. Chen et al. (RAS) [4] adopted HED by refining its side-output iteratively using a reverse attention model. Zhang et al. (LFR) [73] predicted saliency with clear boundaries by proposing a sibling architecture and a structural loss function. Zhang et al. (BMPM) [72] proposed a controlled bi-directional passing of features between shallow and deep layers to obtain accurate predictions.

Deep Recurrent and attention Methods: Kuen et al. [30] proposed a recurrent network to iteratively perform refinement on selected image sub-regions. Zhang et al. (PA-GRN) [76] developed a recurrent saliency detection model that transfers global information from the deep layer to shallower layers by a multi-path recurrent connection. Hu et al. (RADF+) [19] recurrently concatenated multi-layer deep features for saliency object detection. Wang et al. (RFCN) [63] designed a recurrent FCN for saliency detection by iteratively correcting prediction errors. Liu et al. (PiCANet) [39] predicted the pixel-wise attention maps by a contextual attention network and then incorporated it with U-Net architecture to detect salient objects.

Coarse to Fine Deep Methods: To capture finer structures and more accurate boundaries, numerous refinement strategies have been proposed. Liu et al. [38] proposed a deep hierarchical saliency network which learns various global structured saliency cues first and then progressively refine the details of saliency maps. Wang et al. (SRM) [64] proposed to capture global context information with a pyramid pooling module and a multi-stage refinement mechanism for saliency maps refinement. Inspired by [50], Amirul et al. [22] proposed an encoder-decoder network that utilizes a refinement unit to recurrently refine the saliency maps from low resolution to high resolution. Deng
et al. (R²Net+) [6] developed a recurrent residual refinement network for saliency maps refinement by incorporating shallow and deep layers’ features alternately. Wang et al. (DGRL) [65] proposed to localize salient objects globally and then refine them by a local boundary refinement module. Although these methods raise the bar of salient object detection greatly, there is still a large room for improvement in terms of the fine structure segment quality and boundary recovery accuracy.

3. BASNet

This section starts with the architecture overview of our proposed predict-refine model, BASNet. We describe the prediction module first in Sec. 3.2 followed by the details of our newly designed residual refinement module in Sec. 3.3. The formulation of our novel hybrid loss is presented in Sec. 3.4.

3.1. Overview of Network Architecture

The proposed BASNet consists of two modules as shown in Fig. 2. The prediction module is a U-Net-like densely supervised Encoder-Decoder network [57], which learns to predict saliency map from input images. The multi-scale Residual Refinement Module (RRM) refines the resulting saliency map of the prediction module by learning the residuals between the saliency map and the ground truth.

3.2. Predict Module

Inspired by U-Net [57] and SegNet [2], we design our salient object prediction module as an Encoder-Decoder network because 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 ground truth inspired by HED [67] (see Fig. 2). The encoder part has an input convolution layer and six stages comprised of basic res-blocks. The input convolution layer and the first four stages are adopted from ResNet-34 [16]. The difference is that our input layer has 64 convolution filters with size of 3×3 and stride of 1 rather than size of 7×7 and stride of 2. Additionally, there is no pooling operation after the input layer. That means 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 quarter scale resolution in the first feature map. This adaptation enables the network to obtain higher resolution feature maps in earlier layers, while it also decreases the overall receptive fields. To achieve the same receptive field as ResNet-34 [16], 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 the decoder. It consists of three convolution layers with 512 dilated (dilation=2) 3×3 filters. Each of these convolution layers is followed by a batch normalization [21] and a ReLU activation function [13].

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 upsampled output from its previous stage and its corresponding stage in the encoder. To achieve the side-output saliency 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 a input image, our predict module produces seven saliency maps in the training process. Al-
though every saliency map is upsampled 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 predict module. This output is passed to the refinement module.

### 3.3. Refine Module

Refinement Module (RM) \[22\] [6] is usually designed as a residual block which refines the predicted coarse saliency maps \(S_{\text{coarse}}\) by learning the residuals \(S_{\text{residual}}\) between the saliency maps and the ground truth as

\[
S_{\text{refined}} = S_{\text{coarse}} + S_{\text{residual}}. \tag{1}
\]

Before introducing our refinement module, we have to define the term “coarse”. Here, “coarse” includes two aspects. One is the blurry and noisy boundaries (see its one-dimension (1D) illustration in Fig. 3(b)). The other one is the unevenly predicted regional probabilities (see Fig. 3(c)). The real predicted coarse saliency maps usually contain both coarse cases (see Fig. 3(d)).

Residual refinement module based on local context (RRM\_LC), Fig. 4(a), was originally proposed for boundary refinement \[50\]. Since its receptive field is small, Islam et al. \[22\] and Deng et al. \[6\] iteratively or recurrently use it for refining saliency maps on different scales. Wang et al. \[64\] adopted the pyramid pooling module from \[15\], 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 \[70, 72\] to capture multi-scale contexts. However, these modules are shallow thus hard to capture high level information for refinement.

To refine both region and boundary drawbacks in coarse saliency maps, 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 to our predict module. It contains an input layer, an encoder, a bridge, a decoder and an output layer. Different from the predict module, both encoder and decoder have four stages. Each stage only has one convolution layer. Each layer has 64 filters of size 3 \(\times\) 3 followed by a batch normalization and a ReLU activation function. The bridge stage also has a convolution layer with 64 filters of size 3 \(\times\) 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 the upsampling in the decoder. The output of this RM module is the final resulting saliency map of our model.

### 3.4. Hybrid Loss

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

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

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

\[
\ell^{(k)} = \ell^{(k)}_{\text{bce}} + \ell^{(k)}_{\text{ssim}} + \ell^{(k)}_{\text{iou}}. \tag{3}
\]

where \(\ell^{(k)}_{\text{bce}}, \ell^{(k)}_{\text{ssim}}\) and \(\ell^{(k)}_{\text{iou}}\) denote BCE loss \[5\], SSIM loss \[66\] and IoU loss \[42\], respectively.

BCE \[5\] loss is the most widely used loss in binary classification and segmentation. It is defined as:

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

where \(G(r,c) \in \{0, 1\}\) is the ground truth label of the pixel \((r, c)\) and \(S(r, c)\) is the predicted probability of being salient object.

SSIM is originally proposed for image quality assessment \[66\]. It captures the structural information in an image. Hence, we integrated it into our training loss to learn
the structural information of the salient object ground truth. Let \( x = \{x_j : j = 1, \ldots, N^2\} \) and \( y = \{y_j : j = 1, \ldots, N^2\} \) be the pixel values of two corresponding patches (size: \( N \times N \)) cropped from the predicted probability map \( S \) and the binary ground truth mask \( G \) respectively, the SSIM of \( x \) and \( y \) is defined as

\[
\ell_{\text{ssim}} = 1 - \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}
\]

where \( \mu_x, \mu_y \) and \( \sigma_x, \sigma_y \) are the mean and standard deviations of \( x \) and \( y \) respectively, \( \sigma_{xy} \) is their covariance, \( C_1 = 0.01^2 \) and \( C_2 = 0.03^2 \) are used to avoid dividing by zero.

IoU is originally proposed for measuring the similarity of two sets [23] and then used as a standard evaluation measure for object detection and segmentation. Recently, it has been used as the training loss [56, 42]. To ensure its differentiability, we adopted the IoU loss used in [42]:

\[
\ell_{\text{iou}} = 1 - \frac{\sum_{r=1}^{M} \sum_{c=1}^{W} S(r,c)G(r,c)}{\sum_{r=1}^{M} \sum_{c=1}^{W} [S(r,c)+G(r,c)−S(r,c)G(r,c)]}
\]

where \( G(r, c) \in \{0, 1\} \) is the ground truth label of the pixel \((r, c)\) and \( S(r,c) \) is the predicted probability of being salient object.

We illustrate the impact of each of the three losses in Fig. 5. These heatmaps show change of the loss at each pixel as the training progresses. The three rows correspond to the BCE loss, SSIM loss and IoU loss, respectively. The three columns represent different stages of the training process. BCE loss is pixel-wise. It does not consider the labels of the neighborhood and it weights both the foreground and background pixels equally. It helps with the convergence on all pixels.

SSIM loss is a patch-level measure, which considers a local neighborhood of each pixel. It assigns higher weights to the boundary, i.e., the loss is higher around the boundary, even when the predicted probabilities on the boundary and the rest of the foreground are the same. In the beginning of training, the loss along the boundary is the largest (see second row of Fig. 5). It helps the optimization to focus on the boundary. As the training progresses, the SSIM loss of the foreground reduces and the background loss becomes the dominant term. However, the background loss does not contribute to the training until when the prediction of background pixel becomes very close to the ground truth, where the loss drops rapidly from one to zero. 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’s still enough gradient to drive the learning process. The background prediction looks cleaner since the probability is pushed to zero.

![Figure 5. Illustration of the impact of the losses. \( \hat{P}_{fg} \) and \( \hat{P}_{bg} \) denote the predicted probability of the foreground and background, respectively.](image)

IoU is a map-level measure. But we plot the per-pixel IoU following Eq. (6) for illustration purpose. As the confidence of the network prediction of the foreground grows, the loss of the foreground reduces eventually to zero. When combining these three losses, we utilize BCE to maintain a smooth gradient for all pixels, while using IoU to give more focus on the foreground. SSIM is used to encourage that the prediction respects the structure of the original image, by a larger loss near the boundary.

### 4. Experimental Results

#### 4.1. Datasets

We evaluated our method on six frequently used benchmark datasets: SOD [45], ECSSD [68], DUT-OMRON [69], PASCAL-S [37], HKU-IS [33], DUTS [62]. SOD contains 300 images which are originally designed for image segmentation. These images are very challenging since most of them contains multiple salient objects either with low contrast or overlapping with the image boundary. ECSSD contains 1000 semantically meaningful but structurally complex images. DUT-OMRON has 5168 images with one or two objects in each images. Most of the foreground objects are structurally complex. PASCAL-S consists of 850 images with cluttered backgrounds and complex foreground objects. HKU-IS contains 4447 images. Most of them have more than one connected or disconnected foreground objects. DUTS is currently the largest saliency detection dataset. It is comprised of two subsets: DUTS-TR and DUTS-TE. DUTS-TR contains 10553 images designed for training and DUTS-TE has 5019 images for testing.
4.2. Implementation and Experimental Setup

We train our network using the DUTS-TR dataset, which has 10553 images. Before training, the dataset is augmented by horizontal flipping to 21106 images. During training, each image is first resized to 256×256 and randomly cropped to 224×224. Part of the encoder parameters are initialized from the ResNet-34 model [16]. Other convolutional layers are initialized by Xavier [10]. We utilize the Adam optimizer [26] to train our network and its hyper parameters are set to the default values, where the initial learning rate \( lr=1e^{-3}, betas=(0.9, 0.999), \) \( eps=1e^{-8}, \) weight\_decay=0. We train the network until the loss converges without using validation set. The training loss converges after 400k iterations with a batch size of 8 and the whole training process takes about 125 hours. During testing, the input image is resized to 256×256 and fed into the network to obtain its saliency map. Then, the saliency map (256×256) is resized back to the original size of the input image. Both the resizing processes use bilinear interpolation.

We implement our network based on the publicly available framework: Pytorch 0.4.0 [49]. An eight-core PC with an AMD Ryzen 1800x 3.5 GHz CPU (with 32GB RAM) and a GTX 1080ti GPU (with 11GB memory) is used for both training and testing. The inference for a 256×256 image only takes 0.040s (25 fps). The source code will be released.

4.3. Evaluation Metrics

We use four measures to evaluate our method: Precision-Recall (PR) curve, F-measure, Mean Absolute Error (MAE) and relaxed F-measure of boundary \( \text{relax}F_{\beta} \).

PR curve is a standard way of evaluating the predicted saliency probability maps. The precision and recall of a saliency map are computed by comparing the binarized saliency map against the ground truth mask. Each binarization threshold results in a pair of average precision and recall of a saliency map against the ground truth mask. Each binarized saliency probability maps. The precision and recall of a saliency map are computed by comparing the binarized saliency map against the ground truth mask. Each binarization threshold results in a pair of average precision and recall of a saliency map against the ground truth mask.

Then, to have a comprehensive measure on both precision and recall, \( F_{\beta} \) is computed based on each pair of precision and recall as:

\[
F_{\beta} = \frac{(1+\beta^2) \times \text{Precision} \times \text{Recall}}{\beta^2 \times \text{Precision} + \text{Recall}} \tag{7}
\]

where \( \beta^2 \) is set to 0.3 to weight precision more than recall [11]. The maximum \( F_{\beta} \) \( (\max F_{\beta}) \) of each dataset is reported in this paper.

\( \text{MAE} \) [11] denotes the average absolute per-pixel difference between a predicted saliency map and its ground truth mask. Given a saliency map, its \( \text{MAE} \) is defined as:

\[
\text{MAE} = \frac{1}{H \times W} \sum_{r=1}^{H} \sum_{c=1}^{W} |S(r, c) - G(r, c)| \tag{8}
\]

where \( S \) and \( G \) are saliency probability map and its ground truth respectively, \( H \) and \( W \) represents the height and width of the saliency map and \( (r, c) \) denotes the pixel coordinates. For a dataset, its \( \text{MAE} \) is the average \( \text{MAE} \) of all the saliency maps.

Additionally, we adopt the relaxed F-measure \( \text{relax}F_{\beta} \) [7] to quantitatively evaluate boundaries. Given a saliency map \( S \), we first convert it to a binary mask \( S_{bw} \) using a threshold of 0.5. Then, we obtain the mask of its one pixel wide boundary by conducting an XOR \( (S_{bw}, S_{er}) \) operation where \( S_{er} \) is the eroded binary mask [14] of \( S_{bw} \). The same method is used to get the boundaries of ground truth mask. The relaxed boundary precision \( \text{relaxPrecision} \) is then defined as the fraction of predicted boundary pixels within a range of \( \rho \) pixels from ground truth boundary pixels. The relaxed boundary recall \( \text{relaxRecall} \) measures the fraction of ground truth boundary pixels that are within \( \rho \) pixels of predicted boundary pixels. In our experiments, we set the slack parameter \( \rho \) to 3 similar to the previous studies [44] [58] [77]. The relaxed boundary F-measure \( \text{relax}F_{\beta} \) of each predicted saliency map is computed using equation (7), in which \( \text{Precision} \) and \( \text{Recall} \) are replaced by \( \text{relaxPrecision} \) and \( \text{relaxRecall} \). For each dataset, we report its average \( \text{relax}F_{\beta} \) of all predicted saliency maps.

### 4.4. Ablation Study

In this section, we validate the effectiveness of each key components used in our model. The ablation study contains two parts: architecture ablation and loss ablation. The ablation experiments are conducted on the ECSSD dataset.

**Architecture ablation:** To prove the effectiveness of our BASNet, we report the quantitative comparison results of our model against other related architectures. We take U-Net [57] as our baseline network. Then we start with our proposed Encoder-Decoder network and progressively extend it with densely side output supervision and different residual refinement modules including RRM LC, RRM MS.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Configurations</th>
<th>( \text{max}F_{\beta} )</th>
<th>( \text{relax}F_{\beta} )</th>
<th>( \text{MAE} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline U-Net</td>
<td>( \ell_{bce} )</td>
<td>0.896</td>
<td>0.669</td>
<td>0.066</td>
</tr>
<tr>
<td>En-Dec + ( \ell_{bce} )</td>
<td>0.929</td>
<td>0.767</td>
<td>0.047</td>
<td></td>
</tr>
<tr>
<td>En-Dec+Sup + ( \ell_{bce} )</td>
<td>0.934</td>
<td>0.805</td>
<td>0.040</td>
<td></td>
</tr>
<tr>
<td>En-Dec+Sup+RRM LC + ( \ell_{bce} )</td>
<td>0.936</td>
<td>0.803</td>
<td>0.040</td>
<td></td>
</tr>
<tr>
<td>En-Dec+Sup+RRM MS + ( \ell_{bce} )</td>
<td>0.935</td>
<td>0.804</td>
<td>0.042</td>
<td></td>
</tr>
<tr>
<td>En-Dec+Sup+RRM Ours + ( \ell_{bce} )</td>
<td>0.937</td>
<td>0.806</td>
<td>0.042</td>
<td></td>
</tr>
<tr>
<td>En-Dec+Sup+RRM Ours + ( \ell_{ssim} )</td>
<td>0.924</td>
<td>0.808</td>
<td>0.042</td>
<td></td>
</tr>
<tr>
<td>En-Dec+Sup+RRM Ours + ( \ell_{iou} )</td>
<td>0.933</td>
<td>0.795</td>
<td>0.039</td>
<td></td>
</tr>
<tr>
<td>En-Dec+Sup+RRM Ours + ( \ell_{bs} )</td>
<td>0.940</td>
<td>0.815</td>
<td>0.040</td>
<td></td>
</tr>
<tr>
<td>En-Dec+Sup+RRM Ours + ( \ell_{bs} )</td>
<td>0.940</td>
<td>0.813</td>
<td>0.038</td>
<td></td>
</tr>
<tr>
<td>En-Dec+Sup+RRM Ours + ( \ell_{bsi} )</td>
<td>0.942</td>
<td>0.826</td>
<td>0.037</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Ablation study on different architectures and losses: En-Dec: Encoder-Decoder, Sup: side output supervision; \( \ell_{bce} = \ell_{bce} + \ell_{iou}, \ell_{bs} = \ell_{bce} + \ell_{ssim}, \ell_{bsi} = \ell_{bce} + \ell_{ssim} + \ell_{iou}. \)
Figure 6. Illustration of PR curves (the first row) and F-measure curves (the second row) on the five largest dataset.

![Figure 6](image)

Figure 7. Sample results trained with our BASNet on different losses.

![Figure 7](image)

and RRM_Ours. Table 1 illustrates the results of this ablation study. As we can see, our BASNet architecture achieves the best performance among these configurations.

**Loss ablation:** 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 signifies that our proposed hybrid $\ell_{bsi}$ loss greatly improves the performance, especially for the boundary quality. To further illustrate the qualitative effect of losses, results of our BASNet trained with different losses are shown in Fig. 7. It is clear that the proposed hybrid loss achieves superior qualitative results.

### 4.5. Comparison with State-of-the-arts

We compare our method with 15 state-of-the-art models, PiCANetR [39], BMPM [72], R^2Net [5], PAGRN [76], RADF [19], DGRL [65], RAS [4], C2S [50], LFR [73], DSS [17], NLDF [41], SRM [64], Amulet [74], UCF [75], MDF [35]. For fair comparison, we either use saliency maps provided by the authors or run their released models.

**Quantitative evaluation:** To evaluate the quality of segmented salient objects, we show the precision-recall curves (PR) and the F-measure curves for the five largest datasets in Fig. 6. In addition, Table 2 summarizes the maximum region-based F-measure ($\max F_{\beta}$), the relaxed boundary F-measure ($\relax F_{\beta}$) and the MAE measure for all datasets. As we can see, our method outperforms the state-of-the-arts in terms of both regional and boundary measures. Particularly, our method improves $\relax F_{\beta}$ by 4.1%, 5.1%, 6.2%, 6.2%, 3.4%, 5.9% on SOD, ECSSD, DUT-OMRON, PASCAL-S, HKU-IS and DUTS-TE datasets, respectively.

**Qualitative evaluation:** To further illustrate the superior performance of our method, Fig. 8 shows the qualitative comparison of the results with other top seven methods. We can see that our method is able to accurately segment salient objects under various challenging scenarios, including images with low contrast (1st and 2nd rows), fine structures (3rd and 4th rows), large object touching image boundaries (5th and 6th rows), complex object boundaries (7th and 8th rows), cluttered foreground and background (last two rows). We would like to emphasize that the saliency probability maps produced by our method (without CRF) are more uniform than that of others. In addition, the object boundaries of our results are more clear and sharper than others. More quantitative and qualitative comparison results are provided in the supplementary material.

### 5. Conclusion

In this paper, we proposed a novel end-to-end boundary-aware model, BASNet, and a hybrid fusing loss for accurate salient object detection. 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 salient object detection maps with clear boundaries. Experimental results on six datasets demonstrate that our model outperforms other 15 state-of-the-art methods in terms of both region-based and boundary-aware measures. Additionally, our proposed network architecture is modular. It can be easily extended or adapted to other tasks by replacing either the predicting network or the refinement module.
Table 2. Comparison of the proposed method and other 15 methods on six datasets in terms of the maximum F-measure $maxF_{\beta}$ (larger is better), the relaxed boundary F-measure $relaxF_{\beta}$ (larger is better) and the $MAE$ (Smaller is better). Red, Green, and Blue indicate the best, second best and third best performance. "+" means the results are achieved with post-processing by CRF. "DT". "MK", "MB" are training dataset DUTS-TR, MSRA10K, MSRA-B respectively. “M30K” used in C2S is an extended dataset from MSRA10K.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours</td>
<td>ResNet-34</td>
<td>DT 10553</td>
<td>0.858</td>
<td>0.690</td>
<td>0.114</td>
<td>0.746</td>
<td>0.555</td>
<td>0.167</td>
</tr>
<tr>
<td>FCANetR</td>
<td>ResNet-50</td>
<td>DT 10553</td>
<td>0.856</td>
<td>0.528</td>
<td>0.104</td>
<td>0.735</td>
<td>0.575</td>
<td>0.046</td>
</tr>
<tr>
<td>BMPM</td>
<td>VGG-16</td>
<td>DT 10553</td>
<td>0.856</td>
<td>0.562</td>
<td>0.108</td>
<td>0.728</td>
<td>0.670</td>
<td>0.045</td>
</tr>
<tr>
<td>R3Net+</td>
<td>ResNet-16</td>
<td>MK 10800</td>
<td>0.850</td>
<td>0.431</td>
<td>0.125</td>
<td>0.724</td>
<td>0.599</td>
<td>0.063</td>
</tr>
<tr>
<td>PAGRN</td>
<td>VGG-19</td>
<td>DT 10553</td>
<td>0.852</td>
<td>0.474</td>
<td>0.061</td>
<td>0.731</td>
<td>0.582</td>
<td>0.071</td>
</tr>
<tr>
<td>RADF+</td>
<td>VGG-16</td>
<td>MK 10000</td>
<td>0.833</td>
<td>0.476</td>
<td>0.126</td>
<td>0.723</td>
<td>0.579</td>
<td>0.061</td>
</tr>
<tr>
<td>DORL+</td>
<td>ResNet-50</td>
<td>DT 10553</td>
<td>0.848</td>
<td>0.502</td>
<td>0.106</td>
<td>0.725</td>
<td>0.573</td>
<td>0.064</td>
</tr>
<tr>
<td>RAS+</td>
<td>VGG-16</td>
<td>MB 2500</td>
<td>0.851</td>
<td>0.544</td>
<td>0.124</td>
<td>0.721</td>
<td>0.741</td>
<td>0.056</td>
</tr>
<tr>
<td>C2S</td>
<td>VGG-16</td>
<td>M10K 30000</td>
<td>0.823</td>
<td>0.457</td>
<td>0.124</td>
<td>0.710</td>
<td>0.788</td>
<td>0.055</td>
</tr>
<tr>
<td>LPR+</td>
<td>VGG-16</td>
<td>MK 10000</td>
<td>0.828</td>
<td>0.479</td>
<td>0.125</td>
<td>0.711</td>
<td>0.694</td>
<td>0.052</td>
</tr>
<tr>
<td>DSS+</td>
<td>VGG-16</td>
<td>MB 2500</td>
<td>0.846</td>
<td>0.444</td>
<td>0.124</td>
<td>0.721</td>
<td>0.696</td>
<td>0.052</td>
</tr>
<tr>
<td>NLDF+</td>
<td>VGG-16</td>
<td>MB 2500</td>
<td>0.841</td>
<td>0.475</td>
<td>0.125</td>
<td>0.705</td>
<td>0.666</td>
<td>0.063</td>
</tr>
<tr>
<td>SRM+</td>
<td>ResNet-50</td>
<td>DT 10553</td>
<td>0.843</td>
<td>0.392</td>
<td>0.128</td>
<td>0.717</td>
<td>0.672</td>
<td>0.054</td>
</tr>
<tr>
<td>Amulet</td>
<td>VGG-16</td>
<td>MK 10000</td>
<td>0.798</td>
<td>0.454</td>
<td>0.144</td>
<td>0.715</td>
<td>0.711</td>
<td>0.059</td>
</tr>
<tr>
<td>UCF</td>
<td>VGG-16</td>
<td>MK 10000</td>
<td>0.808</td>
<td>0.471</td>
<td>0.148</td>
<td>0.703</td>
<td>0.669</td>
<td>0.069</td>
</tr>
<tr>
<td>MDF+</td>
<td>R-CNN+</td>
<td>MB 2500</td>
<td>0.746</td>
<td>0.311</td>
<td>0.192</td>
<td>0.682</td>
<td>0.472</td>
<td>0.105</td>
</tr>
</tbody>
</table>

Acknowledgements. This project was partially supported by Alberta Innovates Graduate Student Scholarship (AITF). We would like to thank Dr. Yiming Qian for his constructive suggestions and Dr. Dana Cobzas for her GPU resources.

Figure 8. Qualitative comparison of the proposed method with seven other methods. Each sample occupies two rows. The second row of each sample is the zoom-in view.
References


