

A Mutual Learning Method for Salient Object Detection with intertwined Multi-Supervision

Runmin Wu¹, Mengyang Feng¹, Wenlong Guan¹, Dong Wang^{1*}, Huchuan Lu¹, Errui Ding²

¹Dalian University of Technology,

²Department of Computer Vision Technology (VIS), Baidu Inc.

josephinerabbit@mail.dlut.edu.cn, mengyangfeng@mail.dlut.edu.cn, guanwenlong@mail.dlut.edu.cn,
wdice@dlut.edu.cn, lhchuan@dlut.edu.cn, dingerrui@baidu.com

Abstract

Though deep learning techniques have made great progress in salient object detection recently, the predicted saliency maps still suffer from incomplete predictions due to the internal complexity of objects and inaccurate boundaries caused by strides in convolution and pooling operations. To alleviate these issues, we propose to train saliency detection networks by exploiting the supervision from not only salient object detection, but also foreground contour detection and edge detection. First, we leverage salient object detection and foreground contour detection tasks in an intertwined manner to generate saliency maps with uniform highlight. Second, the foreground contour and edge detection tasks guide each other simultaneously, thereby leading to precise foreground contour prediction and reducing the local noises for edge prediction. In addition, we develop a novel mutual learning module (MLM) which serves as the building block of our method. Each MLM consists of multiple network branches trained in a mutual learning manner, which improves the performance by a large margin. Extensive experiments on seven challenging datasets demonstrate that the proposed method has delivered state-of-the-art results in both salient object detection and edge detection.

1. Introduction

Salient object detection aims to segment the most distinctive object regions in a given image, acting as an important pre-processing step for various vision tasks, including image captioning [8], visual tracking [3], visual question answering [15], and person re-identification [35].

By integrating multi-scale features, previous deep learning based methods [32, 10, 16, 22] are capable of detecting the salient objects from a global and coarse view. Though good performance has been achieved, they still suffer from

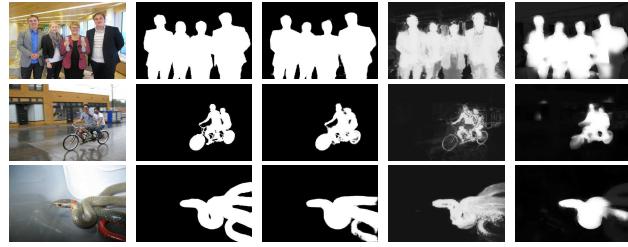


Figure 1. Visual comparison of different CNN based methods.



Figure 2. The examples of our two inputs and three supervisions. E and E-gt are collected from edge detection dataset. S and S-gt are collected from salient object detection dataset, and FC-gt is the foreground contour extracted from S-gt.

two major drawbacks shown in Figure 1: 1) the entire salient objects can hardly be uniformly highlighted due to their complex internal structures; 2) the predictions around object contours are inaccurate due to the information loss caused by strided convolution and pooling operations.

To address the above-mentioned issues, we present a new training strategy and a network design which simultaneously leverage three tasks, including salient object detection, foreground contour detection and edge detection. We feed two images into our network, one for salient object detection and another for edge detection, with three kinds of corresponding supervisions. The two input images are collected from salient object detection and edge detection data-sets respectively. For description convenience, we use E and E-gt to denote the input image and ground truth from the dataset for edge detection, and S and S-gt for the input image and ground truth for salient object detection. Ad-

*Corresponding Author: Dr. Wang

ditionally, the foreground contour ground truth, denoted as FC-gt, is extracted by the Canny [19] operator on the salient object masks (S-gt). The two input images and three supervisions are shown in Figure 2.

First, to generate the saliency maps entirely with uniform highlight, we employ salient object detection and foreground contour detection in an intertwined manner. These two tasks are highly correlated since both demanding accurate foreground detection. Nevertheless, they are also different from each other in the scenes that saliency detection involves dense labeling (i.e., “filling” the internal of object regions) and is more likely to be affected by the internal complexity of salient objects, giving rise to uneven foreground highlight. In contrast, the foreground contour can be “extracted” based on low-level cues, such as edges and textures, given the rough location of foreground objects. Therefore, the contour detection task is more robust to objects’ internal structures, but may be misled by rich edge informations around object contours.

In the proposed intertwined strategy, the two tasks are interlaced at different blocks of the network, forcing the network to learn “filling” and “extracting” the foreground contour alternatively. As a consequence, the network can benefit from the strengths of both tasks and overcome their defects, leading to more entirely highlighted salient regions.

Second, to alleviate the blur boundary issue in the predicted saliency maps, we propose to improve foreground contour detection with auxiliary supervision from edge detection task. To this end, an edge module is designed and jointly trained with our backbone network. Since the edge module takes as input the saliency features from our backbone network, the semantic information encoded in these features can effectively suppress noisy local edges. Meanwhile, the edge features extracted by the edge module serve as additional input to foreground contour detection, ensuring more accurate detection results with low-level cues.

Third, to further improve performance, we propose a novel Mutual Learning Module (MLM) inspired by the success of Deep Mutual Learning (DML) [34]. A MLM is built on top of each block of our backbone network and comprises of multiple subnetworks which are trained in the peer-teaching strategy by a mimicry loss, yielding an additional performance gain.

Our main contributions can be summarized as follows.

- We propose to train deep networks for saliency detection using multi-task intertwined supervision, where foreground contour detection is effectively leveraged to render accurate saliency detection.
- We employ the foreground contour detection and edge detection task to guide each other to generate more accurate foreground contour and reduce the noise in edge detection simultaneously.

- We design a novel network architecture named Mutual Learning Module, which can better leverage the correlation of multiple tasks and significantly improves saliency detection accuracy.

We compare our method with 15 state-of-the-art salient object detection on 7 challenging datasets and 6 popular edge detection methods on BSD500 [1]. The results show that the proposed algorithm performs much better than other competing saliency models, meanwhile, it achieves comparable edge detection performance with much faster speed.

2. Related Work

Effectively utilizing spatial details and semantic information is a crucial factor in achieving the state-of-the-art performance on salient object detection. Most existing methods use *skip-connection* or *recurrent architecture* to integrate hierarchical features from convolution Neural Networks (CNNs). They can roughly detect the targets but cannot uniformly highlight the entire objects, and also suffer from the blur boundary. To generate clear boundaries, some methods attempt to introduce extra edge information to the saliency network.

2.1. Integrating Hierarchical Features

2.1.1 Skip-connection

In HED [27], the authors propose to build *skip-connections* to exploit multi-scale deep features for edge detection. The edge detection is an easier task since it does not rely too much on high-level information. On the contrary, the salient object detection is in great need of semantic features. Thus directly introducing *skip-connections* into salient object detection is unsatisfactory. A promoted version called *short-connections*, proposed in DSS [10], resolves this issue by linking the deeper layers towards shallower ones and skipping the middle ones. Another work by SRM [23] proposes a stage-wise refinement model and a pyramid pooling module to integrate both local and global context information for saliency prediction. In this way, the multi-scale feature maps can assist to locate salient targets and recover local details more effectively.

2.1.2 Recurrent Architecture

A Recurrent Fully convolution Network (RFCN) [22] is proposed to incorporate saliency prior knowledge for more accurate inference. It can refine the saliency map by remedying its previous errors through the recurrent architecture. A recurrent convolution layer (RCL) is introduced in DHS [16] to progressively recover image details of saliency maps through integrating local context information.

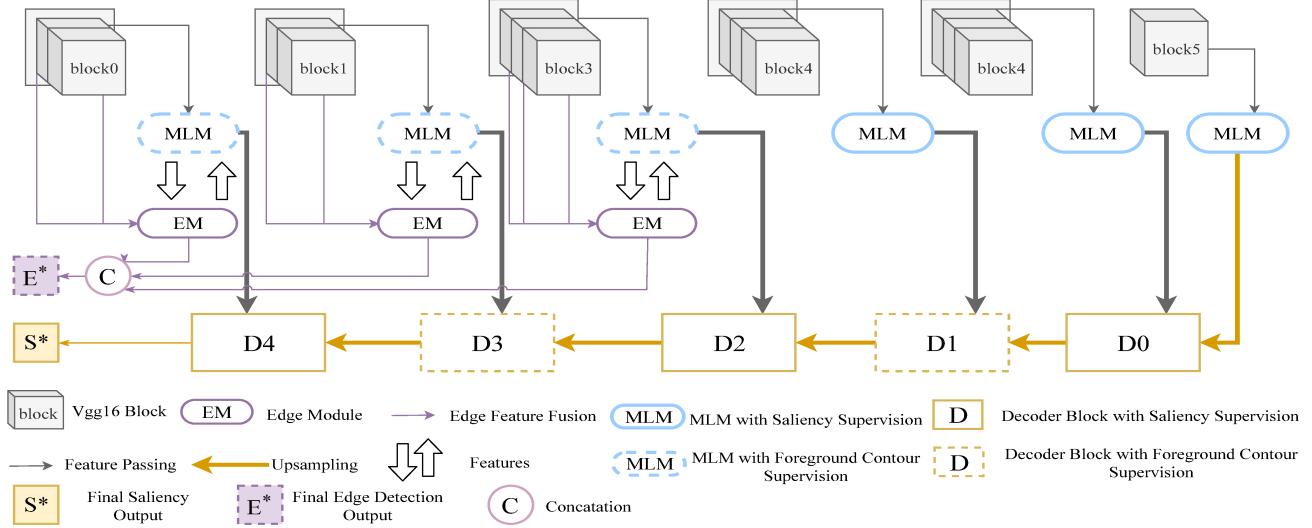


Figure 3. The overall framework consists of the VGG-16 (block0~block5), 6 mutual learning modules (MLMs), 3 edge modules (EMs) and a decoder (D0-D4). We employ deep supervision at each MLM, EM and decoder block. Here, the module with dashed frame denotes the block supervised by FC-gt and the module with solid frame supervised by S-gt or E-gt.

Most of these methods attempt to find a way to fuse multi-scale features for better recognizing salient and non-salient regions. However, supervised by S-gt only, they still fail to generate a binary map as the ground truth. Because the pixels inside the saliency region could be complex due to various cues (*e.g.*, illumination and color), the lower level features are more susceptible by these factors, which makes each pixel has different responses at different scale. As results, the fusion strategy could not capture the entire object. In this paper, we explore to learn the saliency detection and foreground contour in an intertwined manner, which enables our network to recognize the overall shape of the salient targets and uniformly assign foreground labels to the entire objects.

2.2. Exploiting Edge Information

There are some recent attempts exploiting extra edge information for saliency detection to generate prediction maps with clear boundary. In [30], the authors make a new dense label of three categories (saliency objects, saliency objects' boundaries and background) to emphasize the accuracy of saliency boundary detection. Moreover, they introduce extra hand-craft edge features as a complementary to preserve edge information effectively. In [25] and [26], edge knowledge is used for detecting region proposals. They first exploit a pre-trained edge detection model to detect objects' edge. Then based on the detected edge, they segment the input image into edge regions (similar to super-pixels), and generate saliency score map in every region via a mask-based Fast R-CNN. With the edge-region based method, the models succeed to preserve the objects' boundary in

saliency detection.

However, all these existing edge-based methods only utilize prior edge knowledge to help saliency task. In contrast, our model utilizes the edge detection and foreground contour detection to guide each other simultaneously, which demonstrates that the foreground contour detection also benefits the edge detection. As results, our method achieves comparable performance while running faster than other edge detection methods.

3. Proposed Method

3.1. Architecture Overview

Our network follows the encoder-decoder architecture and consists of four main components, which are the VGG [20] backbone, mutual learning module (MLM), edge module (EM) and decoder blocks. The overview of our network is shown in Figure 3.

The encoder part consists of a VGG-16 backbone, six mutual learning modules and three edge modules. For the VGG-16 backbone, we discard the layers after pool15 and denote the remaining blocks (conv1 ~ conv5, pool15) as block0~block5. We build six MLMs on top of each blocks to extract the foreground contour features and saliency features. Additionally, we apply an EM at every block to extract edge features, and each EM is connected with all convolution layers in corresponding VGG block. We use a residual architecture to transfer features between the EM and the corresponding MLM.

As for the decoder, we employ a deeply supervised framework similar to the U-net for fusing multi-scale features from MLMs and generating predictions. The five de-

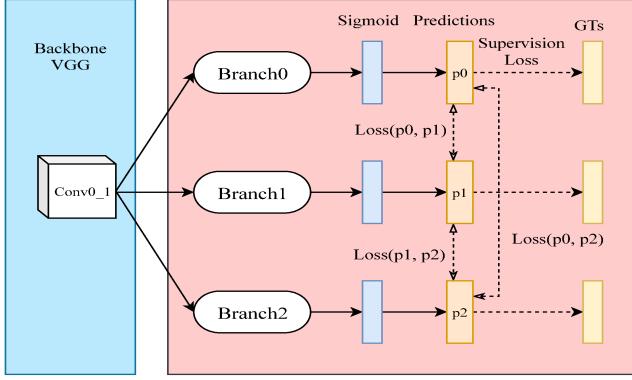


Figure 4. The example of Mutual Learning Module. Each student branch is supervised by ground truths and a L2-based mimicry loss is used to match with every other branch.

coder blocks (D0~D4) share the similar structure. Each of the decoder block aims to fuse the features from MLM and the upsampled features from the previous block. It merges the input features and outputs upsampled features to the next block by a deconvolution layer. In addition, for deep supervision, we employ a convolution layer to produce a predicted map at each block.

3.1.1 Mutual Learning Module

The Mutual Learning Module (MLM) aims at improving the performance of saliency detection and foreground contour detection inspired by the success of Deep Mutual Learning [34] (DML). The DML method sets several student networks for a same task and utilizes their predictions as sub-supervisions for each other. Each student network is a complete model trained in a mutual manner, but can work independently.

In contrast, each student network in our MLM is a simple sub-network consisted of three consecutive layers, which is aimed at extracting features and generating a prediction, as the Figure 4 illustrated. We set larger convolution kernels with dilation for the deeper MLMs to capture fine global information. The L2 distance is employed for mimicry loss in our model instead of the KL distance used in DML.

Let v_{ij} denote the features in j -th convolution layer of the block i , and ϕ_s^{ik} represent the function of the k -th student branch. The prediction A_s^{ik} can be given as:

$$\begin{cases} A_s^{ik} = \text{Sigmoid}(\phi_s^{ik}(v_{i1})), & \text{for } i = 0, 1 \\ A_s^{ik} = \text{Sigmoid}(\phi_s^{ik}(v_{i2})), & \text{for } i = 2, 3, 4 \end{cases} \quad (1)$$

where *Sigmoid* is the activate function, and the 0-based indexed mode is applied.

By learning from each other, MLM provides a softer loss for each sub-network, which leads to the parameters of each student networks converging to a better local minimal. Notice that several student branch networks are trained in a

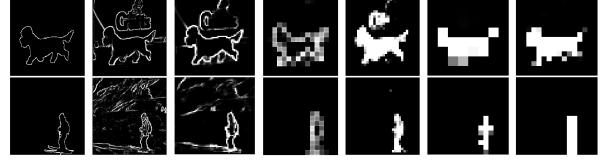


Figure 5. The S-E (left) is the saliency boundary labels extracted by Canny from existing saliency segmentation ground truth. The block0 to block5 show the saliency outputs from corresponding side out branches.

MLM, but we only use one of those student branches for testing by random.

Concretely, we apply the shallower three MLMs for foreground contour detection with the supervision of FC-gt based on the observation in [9]. The deeper three MLMs are supervised by S-gt for saliency detection. The predictions in every MLM are shown in Figure 5.

3.1.2 Edge Module

In [9], the authors visualize neurons' receptive fields in a pre-trained VGG-19 model before and after fine-tuning for the saliency detection task. The result shows that the responses of neurons at first three pooling layers hardly change after fine-tuning for saliency task. What's more, even fine-tuning the network for saliency detection, neurons' responses at each of the first three convolution layers still keep lots of similarities with edge patterns. It indicates that the first three blocks in the pre-trained VGG are suitable to capture both edge information and saliency information at the same time. Based on this, we employ an extra edge module (EM) on each first three VGG blocks for edge detection task and helping foreground contour detection in MLM.

As Figure 6 shows, every convolution layer in one block is connected to another convolution layer for extracting rich edge features, and features from different layers are merged to generate an edge probability map.

Specifically, for the input image E , every EM produces an edge probability map A_e^i , and the outputs are collected for merging into final edge prediction E^* . For the input image S , the EM only provides edge feature maps a_e^i for the MLM. The two modules are connected in a residual manner, which is designed to reduce the noise in edge features for foreground contour detection.

For the i -th block, denoting the function of EM as ψ^i , the output feature map a_e^i and edge probability map A_e^i can be given by:

$$\begin{cases} a_e^i = \psi^i(v_{i0}, v_{i1}), & \text{for } i = 0, 1 \\ a_e^i = \psi^i(v_{i0}, v_{i1}, v_{i2}), & \text{for } i = 2 \\ A_e^i = \text{Sigmoid}(a_e^i) \end{cases} \quad (2)$$

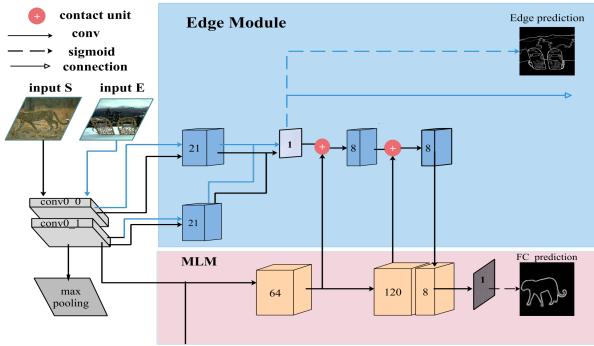


Figure 6. The example of EM and the connections with MLM, we only show block0 for simplicity. Notice that there are two input images, S and E, for edge detection and foreground contour detection simultaneously. The information stream of E is shown as blue line, and the information stream of S is shown in black line. Numbers on the boxes represent the channel numbers.

Receiving the information from EM, the MLM is capable to generate precise contour. In the meanwhile, the shared basic blocks provide semantic information from foreground contour detection and thereby helping the edge detection to ignore the useless local edges. We show more validations in Section 4.5.

3.2. Intertwined Supervision Method

In previous MLMs, we employ the FC-gt as supervision at the shallower three MLMs for keeping fine detailed contour information and the S-gt as supervision at the deeper three modules to focus on semantic information.

We further apply two tasks alternatively at different decoder blocks in an intertwined manner. To be more specific, we set S-gt for supervision at D0, D2, D4, and FC-gt at D1, D3. Each decoder block is aimed at fusing the features from the previous block and corresponding MLM, and then transmitting the features to the next block.

As illustrated in Figure 3, the block D1 takes in the high-level semantic information and transmits it into foreground contour features under the supervision of FC-gt's, where the object's internal noise is discarded and the extracted contour features become cleaner.

The D2 block works in a ‘filling’ scheme, taking in the contour features and then transmits them into saliency features. It requires to retrieve the internal information based on the contour features, which compels the D2 to produce uniform predicted score for each pixel inside the contour, like a ‘filling’ process.

Then the relative clean semantic information from the D2, and the low-level contour information from a shallow MLM, is send to the D3 block. The D3 block under the su-

pervision of FC-gt's aims at keeping the precise foreground contour and eliminating excess noise from the low-level contour information with the high-level semantic knowledge.

The last block, D4 works similarly with the D2, under the supervisions of S-gt's, trying to filling the saliency map based on the foreground information.

With the supervisions of FC-gt and S-gt in this intertwined manner, our method succeeds to generate saliency predictions with entire uniform highlight and keep fine foreground contours simultaneously.

3.3. Training Strategy

3.3.1 Loss Function

In the deep supervision strategy, we resize the three kinds of ground truths for adapting the resolution of each stage. For simplicity, we treat salient object detection and foreground contour detection as saliency tasks and denote S-gt along with FC-gt as \hat{S} and E-gt as \hat{E} . Our total loss is consisted of two parts, *i.e.* the loss \mathcal{L}_{Enc} in encoder and the one \mathcal{L}_{Dec} in decoder.

Mathematically, the \mathcal{L}_{Enc} can be given as,

$$\mathcal{L}_{Enc} = \theta_s \mathcal{L}_S + \theta_e \mathcal{L}_E + \theta_m \mathcal{L}_{mimicry}, \quad (3)$$

where the \mathcal{L}_S , \mathcal{L}_E , and $\mathcal{L}_{mimicry}$ are the loss functions for saliency tasks, edge detection and the mimicry loss in MLMs, respectively, and the θ 's are their weights, which are set as 0.7, 0.2, 0.1.

Specifically, we employ the binary cross-entropy loss l_{bce} for L_S and L_E . They can be wrote as,

$$\mathcal{L}_S = \sum_{i=0}^5 r_s^i l_{bce}(A_s^i, \hat{S}), \quad (4)$$

$$\mathcal{L}_E = \sum_{i=0}^2 r_e^i l_{bce}(A_e^i, \hat{E}) + l_{bce}(E^*, \hat{E}), \quad (5)$$

where r_s^i and r_e^i represent the weights at i -th block. In our experiments, the r_s^i are simply set as 0.2 except the the r_s^5 which is set as 0.5. For the mimicry loss, we employ the MSE loss function $l_{mse}(a, b) = (a - b)^2$. Considering the six MLMs with k student branches, the total mimicry loss can be given as,

$$\mathcal{L}_{mimicry} = \frac{1}{2} \sum_{i=0}^5 \sum_{n=0}^K \sum_{m=0}^K r_{mlm}^i l_{mse}(A_s^{i_n}, A_s^{i_m}) \quad (6)$$

where r_{mlm}^i denotes the weight at the i -th block and K denotes the number of the student branches and all the r_{mlm}^i are set as 1.

As for the decoder, the weight at the i -th decoder block is denoted as r_{dec}^i . Then \mathcal{L}_{Dec} can be given as,

$$\mathcal{L}_{Dec} = \sum_{i=0}^4 r_{dec}^i l_{bce}(D^i, \hat{S}) \quad (7)$$

where the r_{dec}^i are set as 0.5 except the r_{dec}^4 is set as 1.

3.3.2 Training Steps

We alternatively train our encoder and decoder networks and the training steps are as follows,

Step.1 We first train the encoder network, *i.e.* the backbone network with MLMs and EMs, with \mathcal{L}_{Enc} .

Step.2 We fix the encoder and feed features in MLMs into the decoder network then train the decoder network alone with \mathcal{L}_{Dec} .

Step.3 After training the decoder, we then fix the decoder and fine-tune the encoder using the supervision of decoder, *i.e.* \mathcal{L}_{Dec} , to optimize the parameters of encoder.

We repeat the steps above iteratively, and in each training step we stay for 10 epochs before switch to the next.

4. Experiments

4.1. Datasets

For the saliency detection task, we use the training set of DUTS [21] and evaluate our algorithm on the the test set of the dataset and other six popular datasets, ECSSD [28], DUT- OMRON [29], SOD [18], and HKU-IS [12]. DUTS is the biggest released dataset containing 10,553 images for training and 5,019 images for testing. Both training and test sets contain very complex scenarios with high content variety. ECSSD contains 1,000 natural and complex images with pixel-accurate ground truth annotations. The images are manually selected from the Internet. DUT-OMRON has more challenging images with 5,168 images. All images are resized so as to the maximal dimension is 400 pixels long. SOD has 300 images contained multiple, low-contrast objects. HKU-IS has 4,447 images which are selected by meeting at least one of the following three criteria, *i.e.* multiple salient objects with overlapping, objects touching the image boundary and low color contrast. PASCAL-S dataset is part of the PASCAL VOC [6] dataset and contains 850 image. SOC dataset is a latest dataset which contains 6000 images including 3000 images with salient objects and 3000 images with non-salient objects from more than 80 daily object categories, besides, these salient objects have challenging attributes such as motion blur, occlusion and cluttered background. We only use the valid set for testing to evaluate the robustness of our method.

Regarding the edge detection task, we use BSD500 [1] to train and test. The BSD500 dataset contains 200 training, 100 validation and 200 test images.

4.2. Implementation Details

All experiments are conducted using a single Nvidia GTX TITAN X GPU. A pretrained VGG-16 model is used to initialize the convolution layers in the backbone network. The parameters in other convolution layers are randomly initialized. All training and test images are resized to 256×256 before being fed into the network. We use the ‘Adam’ method with the weight decay 0.005. The learning rate is set to 0.0004 for the encoder and 0.0001 for the decoder. Our source codes are available on the website: <https://github.com/JosephineRabbit/MLMSNet>.

4.3. Evaluation Metrics

For the salient object detection task, we use four effective metrics to evaluate the performance, including the precision-recall curves, F-measure, Mean Absolute Error [2] and S-measure [7]. We first compute pairs of precision and recall values by thresholding the predicted saliency map for the use of plotting PR curves. Besides, F-measure value is used as another metric, which is computed on every binarized saliency map and is averaged on the whole dataset. The threshold is determined to be twice the mean saliency value. The S-measure is calculated by a region-aware structural similarity S_r and a object-aware structural similarity S_o : $S_m = \alpha * S_o + (1 - \alpha) * S_r$ where $\alpha \in [0, 1]$. We set $\alpha = 0.5$ as recommended in [7].

As for the edge detection task, we evaluate the edge probability map using F-measure of both Optimal Dataset Scale (ODS) and Optimal Image Scale (OIS).

4.4. Performance Comparison

We compare our method with 14 state-of-the-art saliency detection methods, including BMPM [31], DGRL [24], PAGR [33], RAS [4], PiCANet [17], R3Net [5], MSRNet[11], SRM [23], Amulet [32], DSS [10], DHS [16], DCL [13], RFCN [22] and DS [14]. We compare our edge detection results with some popular algorithms in Table 2 such as gPb-UCM [1] as HED [27]. When evaluating for edge detection, only the first three blocks of our backbone network and corresponding EMs are utilized. Therefore, our edge detection network is a relative simple and tiny architecture compared with other edge detection models.

Quantitative Evaluation. For the saliency detection task, we compare the proposed method with the others in terms of MAE scores, mean F-measure scores and S-measure scores, which are shown in Table 1. The proposed method performs favorably against other counterparts among all datasets over these evaluation metrics.

As for the edge detection task, the results are illustrated in Table 2. Our edge detection network achieves around $3 \times$ faster than other methods due to its lightweight, and achieves a comparable accuracy.

Table 1. Quantitative evaluations. The best three scores are shown in red, green and black, respectively.

	DUTS			PASCAL-S			ECSSD			SOD			HKU-IS			OMRON		
	F_β	MAE	S_m															
Ours	0.802	0.045	0.856	0.838	0.069	0.849	0.914	0.038	0.911	0.811	0.106	0.780	0.893	0.034	0.901	0.742	0.056	0.817
BMPM	0.751	0.049	0.861	0.769	0.074	0.845	0.869	0.045	0.911	0.763	0.107	0.787	0.871	0.039	0.907	0.692	0.064	0.809
DGRL	0.768	0.050	0.841	0.825	0.072	0.836	0.903	0.041	0.903	0.799	0.104	0.771	0.890	0.036	0.895	0.733	0.062	0.806
PAGR	0.788	0.056	0.837	0.807	0.093	0.818	0.894	0.061	0.889	-	-	-	0.886	0.048	0.887	0.711	0.071	0.805
RAS	0.755	0.060	0.839	0.785	0.104	0.795	0.889	0.056	0.893	0.799	0.124	0.764	0.871	0.045	0.887	0.713	0.062	0.814
PiCANet	0.755	0.054	0.861	0.801	0.077	0.850	0.884	0.047	0.914	0.791	0.102	0.791	0.870	0.042	0.906	0.713	0.062	0.814
R3Net	0.802	0.045	0.829	0.807	0.097	0.800	0.917	0.046	0.900	0.789	0.136	0.732	0.905	0.038	0.891	0.756	0.061	0.815
MSRNet	0.708	0.061	0.840	0.744	0.081	0.840	0.839	0.054	0.896	0.741	0.113	0.779	0.868	0.036	0.912	0.676	0.073	0.808
SRM	0.757	0.059	0.834	0.801	0.085	0.832	0.892	0.054	0.895	0.800	0.127	0.742	0.874	0.046	0.887	0.707	0.069	0.797
Amulet	0.676	0.085	0.803	0.768	0.098	0.820	0.870	0.059	0.894	0.755	0.141	0.758	0.839	0.054	0.883	0.647	0.098	0.780
DSS	0.724	0.067	0.817	0.804	0.796	0.797	0.901	0.052	0.882	0.795	0.121	0.751	0.895	0.041	0.879	0.729	0.066	788
DHS	0.724	0.067	0.817	0.779	0.094	0.807	0.872	0.059	0.884	0.774	0.128	0.750	0.855	0.053	0.870	-	-	-
DCL	0.714	0.149	0.735	0.714	0.125	0.754	0.829	0.088	0.828	0.741	0.141	0.735	0.853	0.072	0.819	0.684	0.097	0.713
RFCN	0.712	0.091	0.792	0.751	0.118	0.808	0.834	0.107	0.852	0.751	0.170	0.730	0.835	0.079	0.858	0.627	0.111	0.774
DS	0.633	0.090	0.793	0.669	0.176	0.739	0.826	0.122	0.821	0.698	0.190	0.712	0.788	0.080	0.852	0.603	120	750

Table 2. The comparison of edge results with some wide-used methods on BSDS500 dataset. † means GPU time. The leading results are shown in **bold**.

Method	ODS	OIS	FPS
gPb-UCM [1]	.729	.755	1/240
HED [27]	.788	.808	30†
Ours(baseline)	.736	.751	102 †
Ours(joint learning)	.769	.780	102 †

Table 3. Ablation analysis of each component. The best results are shown in **bold**.

	DUTS		PASCAL-S		ECSSD	
	F_β	MAE	F_β	MAE	F_β	MAE
Baseline	0.695	0.073	0.774	0.092	0.861	0.062
MLM	0.714	0.066	0.809	0.083	0.866	0.049
MLM + FC	0.790	0.046	0.827	0.071	0.909	0.038
MLM + FC + EM	0.802	0.045	0.838	0.069	0.914	0.038
FC + EM	0.789	0.049	0.781	0.077	0.902	0.043

Visual Comparison. We visualize some example saliency maps of our method in Figure 7. From which we can see that our method can produce more accurate results with clear boundary and entire uniform highlight.

4.5. Ablation Study

4.5.1 Saliency Detection

First, to demonstrate the impact of MLMs and EMs and the power of the intertwined supervision strategy, we train four models for comparison as follows: (1) baseline: a network with simple encoder-decoder architecture as proposed using

only S-gt for supervision, and removes the EMs and redundant student branches in MLMs; and (2) +MLM: adding three student branches in each MLM; and (3) +FC: applying S-gt and FC-gt for supervision in intertwined manner; and (4) +EM: the proposed whole network with S-gt and FC-gt for supervision in intertwined manner and employing the EM for edge detection simultaneously. The comparison is conducted across three datasets (DUTS, PASCAL-S, ECSSD) and the results are shown in Table 3. We can observe that our intertwined supervision strategy (+FC) contributes most to the overall performance.

In addition, we calculate the average score of the highlight pixel in saliency maps (ASHP). Given a saliency map \hat{S} , the ASHP could be given as:

$$ASHP(\hat{S}) = \frac{1}{N_s} \sum_{x=1}^W \sum_{y=1}^H |\hat{S}(x, y)|, \quad (8)$$

where N_s is the number of pixels which satisfies $\hat{S}(x, y) > 0$. Moreover, we calculate the accuracy of foreground contour predictions (AFCP). Given a foreground contour prediction map F , the AFCP can be calculated by

$$AFCP = \frac{N_c}{N_f}, \quad (9)$$

where N_c is the numbers of pixels which satisfy $|F(x, y) - \hat{S}(x, y)| < 60$ and $\hat{S}(x, y) > 200$, and N_f is the numbers of pixels which satisfy $\hat{S}(x, y) > 200$.

From the results in Figure 8, we can observe (1) the MLMs could improve the low accuracy of baseline model into a competitive scale; (2) the higher ASHP of +FC

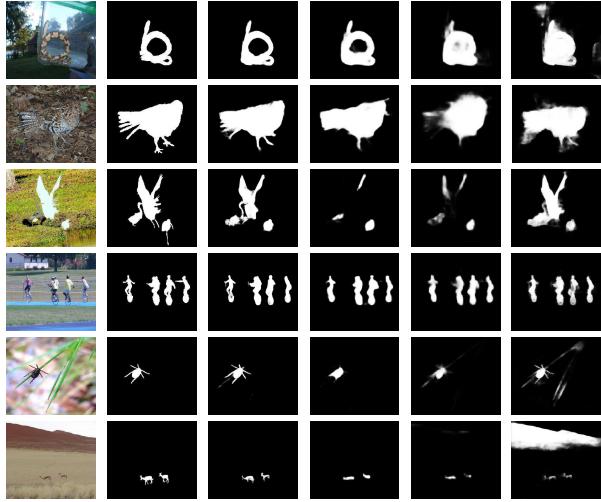


Figure 7. Performance comparison of the proposed method with the latest state-of-the-art methods. The rows 1~2 show the comparison in edge complicated objects. The rows 3~4 show the comparison in multiple objects. The rows 5~6 show the comparison in small objects.

Table 4. The cost of MLM with three student sub-networks. The Original stands for our baseline network with FC and EMs and the training time is the time cost per epoch in DUTS training dataset.

Cost	Original	+MLM
Training time	1334.548s	1897.216s
Test time	13fps	13fps
GPU memory	7775M	7984M

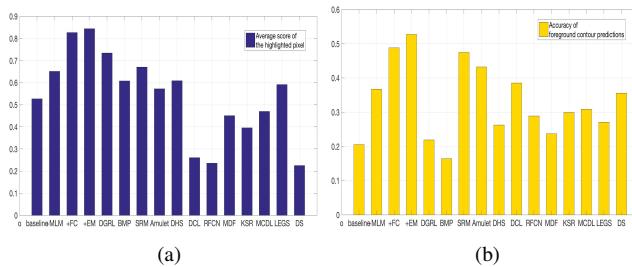


Figure 8. Ablation Study on Saliency Detection Task and Foreground Contour Detection Task. (a) shows the comparison in ASHP and (b) shows the comparison in AFCP.

demonstrates that the intertwined supervision contributes to generating predictions with uniform highlight; and (3) the improvement in AFCP of +EM illustrates that the EMs benefit to generating more accurate foreground contour.

Second, we test the time and GPU cost brought by the MLMs, which consists of three sub-network in our experiments. The results are shown in Table 4. We can observe that the MLMs cost a little resources.

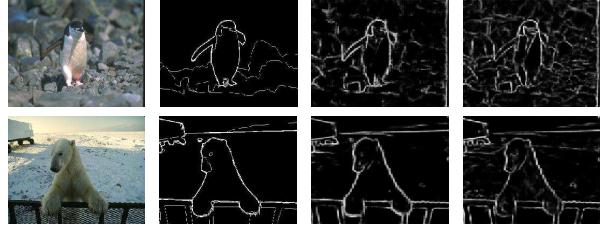


Figure 9. Ablation Study on Edge Detection Task. (a) and (b) are test images and labels from BSD500 and (c) shows the results by jointly learning EM and MLM (EM+MLM). (d) shows the results with EM only.

4.5.2 Edge Detection

To investigate the effectiveness of the joint learning method for edge detection, we evaluate the overall performance between using edge-only supervision and training with saliency tasks. The compared results are shown in Figure 9 and Table 2. We can find that the edge detection network enables to capture better semantic information by joint learning with saliency tasks. This strategy suppresses the noises of redundant local details and brings significant improvement on the edge performance.

5. Conclusion

In this paper, we propose a multi-task algorithm for salient object detection, foreground contour detection and edge detection. We employ an intertwined supervision strategy for salient object detection and foreground contour detection, which encourages the network to produce high predicted score on the entire target objects. What's more, we utilize edge detection and saliency detection to guide each other and both tasks gain benefits. In addition, we propose a mutual learning module (MLM) to allow the network parameters to converge to a better local minimal thereby improving the performance.

In this way, our model is capable to generate predictions with uniform highlighted region inside the salient objects and accurate boundaries. The experiments demonstrate that these mechanisms can result in more accurate saliency maps over a variety of images, in the meanwhile, our model is capable of detecting the satisfactory edges much faster.

Acknowledgements. This work is supported by the Natural Science Foundation of China under Grant 61725202, 61829102, 61872056 and 61751212. This work is also supported by the Fundamental Research Funds for the Central Universities under Grant DUT18JC30.

References

- [1] P. Arbelaez, M. Maire, C. C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011.
- [2] A. Borji, M. Cheng, H. Jiang, and J. Li. Salient object detection: A benchmark. In *IEEE Transactions on Image Processing*, volume 24, pages 5706–5722, 2015.
- [3] A. Borji, S. Frintrop, D. N. Sihite, and L. Itti. Adaptive object tracking by learning background context. In *CVPR Workshops*, pages 23–30, 2012.
- [4] S. Chen, X. Tan, B. Wang, and X. Hu. Reverse attention for salient object detection. In *European Conference on Computer Vision*, pages 236–252, 2018.
- [5] Z. Deng, X. Hu, L. Zhu, X. Xu, J. Qin, G. Han, and P. Heng. R³net: Recurrent residual refinement network for saliency detection. In *International Joint Conference on Artificial Intelligence*, pages 684–690, 2018.
- [6] M. Everingham, L. J. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.
- [7] D. Fan, M. Cheng, Y. Liu, T. Li, and A. Borji. Structure-measure: A new way to evaluate foreground maps. In *International Conference on Computer Vision*, pages 4558–4567, 2017.
- [8] H. Fang, S. Gupta, F. N. Iandola, R. K. Srivastava, L. Deng, P. Dollár, J. Gao, X. He, M. Mitchell, J. C. Platt, C. L. Zitnick, and G. Zweig. From captions to visual concepts and back. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1473–1482, 2015.
- [9] S. He and N. Pugeault. Deep saliency: What is learnt by a deep network about saliency? *abs/1801.04261*, 2018.
- [10] Q. Hou, M. Cheng, X. Hu, A. Borji, Z. Tu, and P. H. S. Torr. Deeply supervised salient object detection with short connections. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5300–5309, 2017.
- [11] G. Li, Y. Xie, L. Lin, and Y. Yu. Instance-level salient object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 247–256, 2017.
- [12] G. Li and Y. Yu. Visual saliency based on multiscale deep features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5455–5463, 2015.
- [13] G. Li and Y. Yu. Deep contrast learning for salient object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 478–487, 2016.
- [14] X. Li, L. Zhao, L. Wei, M.-H. Yang, F. Wu, Y. Zhuang, H. Ling, and J. Wang. Deepsaliency: Multi-task deep neural network model for salient object detection. In *IEEE Transactions on Image Processing*, volume 25, pages 3919–3930, 2016.
- [15] Y. Lin, Z. Pang, D. Wang, and Y. Zhuang. Task-driven visual saliency and attention-based visual question answering. *Computing Research Repository*, *abs/1702.06700*, 2017.
- [16] N. Liu and J. Han. Dhsnet: Deep hierarchical saliency network for salient object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 678–686, 2016.
- [17] N. Liu, J. Han, and M.-H. Yang. Picanet: Learning pixel-wise contextual attention for saliency detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3089–3098, 2018.
- [18] V. Movahedi and J. H. Elder. Design and perceptual validation of performance measures for salient object segmentation. In *Computer Vision and Pattern Recognition Workshops*, pages 49–56, 2010.
- [19] A. D. Sappa and F. Dornaika. An edge-based approach to motion detection. In *Computational Science*, pages 563–570, 2006.
- [20] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.
- [21] L. Wang, H. Lu, Y. Wang, M. Feng, D. Wang, B. Yin, and X. Ruan. Learning to detect salient objects with image-level supervision. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3796–3805, 2017.
- [22] L. Wang, L. Wang, H. Lu, P. Zhang, and X. Ruan. Saliency detection with recurrent fully convolutional networks. In *European Conference on Computer Vision*, pages 825–841, 2016.
- [23] T. Wang, A. Borji, L. Zhang, P. Zhang, and H. Lu. A stagewise refinement model for detecting salient objects in images. In *International Conference on Computer Vision*, pages 4039–4048, 2017.
- [24] T. Wang, L. Zhang, S. Wang, H. Lu, G. Yang, X. Ruan, and A. Borji. Detect globally , refine locally : A novel approach to saliency detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 27–35, 2018.
- [25] X. Wang, H. Ma, and X. Chen. Salient object detection via fast r-cnn and low-level cues. In *International Conference on Image Processing*, pages 1042–1046, 2016.
- [26] X. Wang, H. Ma, X. Chen, and S. You. Edge preserving and multi-scale contextual neural network for salient object detection. In *IEEE Transactions on Image Processing*, volume 27, pages 121–134, 2018.
- [27] S. Xie and Z. Tu. Holistically-nested edge detection. *International Journal of Computer Vision*, 125(1-3):3–18, 2017.
- [28] Q. Yan, L. Xu, J. Shi, and J. Jia. Hierarchical saliency detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1155–1162, 2013.
- [29] C. Yang, L. Zhang, H. Lu, X. Ruan, and M. Yang. Saliency detection via graph-based manifold ranking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3166–3173, 2013.
- [30] J. Zhang, Y. Dai, F. M. Porikli, and M. He. Deep edge-aware saliency detection. In *Computing Research Repository*, volume *abs/1708.04366*, 2017.
- [31] L. Zhang, J. Dai, H. Lu, Y. He, and G. Wang. A bi-directional message passing model for salient object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1741–1750, 2018.
- [32] P. Zhang, D. Wang, H. Lu, H. Wang, and X. Ruan. Amulet: Aggregating multi-level convolutional features for salient

- object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 202–211, 2017.
- [33] X. Zhang, T. Wang, J. Qi, H. Lu, and G. Wang. Progressive attention guided recurrent network for salient object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 714–722, 2018.
- [34] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu. Deep mutual learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328, 2018.
- [35] R. Zhao, W. Ouyang, and X. Wang. Person re-identification by saliency learning. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 39, pages 356–370, 2017.