Bidirectional Multiscale Refinement Network for Crisp Edge Detection

Shi-shui Bao¹, You-rui Huang¹⁺², Guang-yu Xu¹

1 School of Computer Science and Engineering, Anhui University of Science and Technology, Huainan, China
2 Anhui Science and Technology University, Bengbu, China

Corresponding author: You-rui Huang(byr628@163.com).
Guang-yu Xu(Xgy761220@163.com).

This work was supported in part by the National Natural Science Foundation of China under Grant (61471004).

ABSTRACT

With the development of deep convolution neural networks (CNNs), contour detection has made great progress. Some contour detectors based on CNNs have better performance than human beings on standard benchmarks. However, it is easier for CNNs to learn the similar features of adjacent pixels, and the number of background pixels and edge pixels in the input training sample is highly imbalanced. Therefore, the prediction edge by the edge detector based on CNNs is thick and requires post-processing to obtain crisp edges. Accordingly, we introduce a novel parallel attention model and a novel loss function that combines cross-entropy and dice loss through the use of adaptive coefficients, and propose a novel bidirectional multiscale refinement network (BMRN) that stacks multiple refinement modules in order to achieve richer feature representation. The experimental results show that our method has better performance than the state-of-the-art on BSDS500 (ODS F-score of 0.828), NYUDv2 depth datasets (ODS F-score of 0.778) and Multi-Cue dataset (ODS F-score 0.905(0.002)).

INDEX TERMS: Edge detection, Refinement network, Bidirectional, Deep learning

I. INTRODUCTION

Contour is an important feature of images, and accurate edge detection is a basic task of machine vision and image processing. In the early years of image technology, most users obtained the target edge by looking for the gray intensity mutation in the image. However, most users currently obtain the semantic meaningful object contour, because it is the basis of object recognition, image segmentation, and epic flow algorithms. With the development of deep learning, CNN has become the main method of edge detection, such as N²-field[26], Deep Contour[11], Deep Edge[4], HED[12], and RCF[13]. These make full use of the hierarchical feature learning ability of neural networks and obtain optimal F-score performance in benchmark datasets such as BSDS500 and NYUDv2.

Although CNNs have achieved better accuracy in detecting target contours than traditional classical algorithms, their prediction is much thicker than the classical method. Fig 1 shows the contour maps that were obtained by the Canny operator and the richer convolution features (RCF) detector, which are not processed by non-maximum suppression. It can be found from the previous literature that deep learning methods pay more attention to the precision and recall of contour, and less attention to the crispness of boundary (precisely localizing edge pixels) than traditional methods. CED[14] added a residual enhancement module in HED[12] and used sub-pixel convolution to recover image resolution step by step, which allowed the decoding process to integrate all features of image coding as much as possible, and improved the crispness of contour. LPCB[15] achieved greater accuracy of contour by improving the loss function. BDCN[42] trained the neural network by using multiscale features. Inspired by LPCB[15], BDCN[42], LRC[47], RCF[13], and CBAM[32], a bidirectional multiscale refined network (BMRN) is proposed. The BMRN model uses a multi-scale training method to train on the enhanced Berkeley segmentation dataset (BSDS500), the NYUDv2 depth dataset, and Multi-Cue in order to further improve its generalization performance.

FIGURE 1. (a) is the original image of the aircraft in BSDS500; (b) is ground truth; (c) is the result of Canny; and (d) is the output of the HED detection method. None of the output results are post-processing.

II. RELATED WORK

For the task of edge detection, the early literature used the gray information mutation between the edge and background of the object and used local gradient to detect the target contour. For example, the Prewitt operator [1] and the Canny operator [2] both use derivative to detect contour. In recent literature, Tao Fang et al. proposed BAR[45], which is based on the bilateral asymmetric receptive field mechanism of the visual pathway. Qing Zhang et al. proposed DIDY[46], a biologically inspired model for contour detection based on binocular disparity and receptive field dynamics. With the evolution of machine learning, the whole and local information of the image is fully considered.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
Edge detection is regarded as a pixel-level classification problem, with the pixels classified by extracting the local features of the image. Martin et al. proposed the Pb [21] by using the channel characteristics of image brightness (BG), texture (CG), color (TG), and the logistic regression algorithm. Arbeláez et al. proposed gPb [3], which gives each pixel a global probability by calculating global characteristics on the basis of multi-scale Pb [21]. Subsequently, Arbeláez et al. constructed the image Gaussian pyramid and extracted the contour features at different scales to achieve contour extraction and proposed the multi-scale combinatorial grouping (MCG) algorithm. In recent years, with the success of deep CNNs in the field of computer vision, deep CNNs are increasingly being applied to contour detection. Inspired by FCN [43], Xie et al. proposed HED, the first end-to-end contour detection algorithm [12]. The network structure of the algorithm is divided into coding and decoding networks, and the entire network is trained directly end-to-end by the backpropagation algorithm. On the basis of HED [12], Liu et al. proposed RCF [13], which compresses the features of each convolution layer with the same resolution, thereby greatly improving the network’s ability to express image features. Also on the basis of HED [12], Wang et al. added a residual enhancement module and used sub-pixel convolution to gradually restore the image resolution, so that the decoding process could synthesize all the features of image coding as much as possible. They proposed CED [14], which greatly improved the accuracy of contour.

Because the CNN-based edge detection method has the ability to learn features automatically, the robustness of the algorithm is improved to a certain extent. However, these models of end-to-end edge detection, such as RCF [13] and CED [14], use the VGG16 [44] traditional classification network as the backbone coding network, making it easier to produce similar responses of adjacent pixels. At the same time, the pooling layer in the network structure further reduces the image resolution, and the edge pixels and non-edge pixels in the training samples are not balanced. These factors make it difficult to guarantee both the precision and accuracy of the output contour map. This type of thick and fuzzy edge contour may be a disadvantage in some visual tasks, such as recent optical flow methods, which require accurate and crisp edge input to interpolate sparse matching results.

In view of the above problems, this paper proposes a bidirectional multiscale refinement network based on BDCN [42]. The three main contributions of this paper are:

1. To improve the performance of the network model in terms of obtaining the local details of the contour, we propose a novel parallel attention model, which can encode the spatial context information of a large range of feature images into local features.
2. We propose a novel refinement block, which can automatically obtain multiscale feature weights in the end-to-end training process, so that multiscale features can be reconstructed.
3. We propose an adaptive weight loss function, which combines cross-entropy and dice loss functions and minimizes the distance between image and ground truth from the image level to the pixel level.

III. PROPOSED METHOD

In this section, the specific structure of the proposed method is described. First, the formula reasoning of the network model is introduced, and the overall structure of the network model is described. In order to obtain a crisper contour, the loss function is improved, and a new adaptive loss function is created by fusing the cross-entropy and dice loss functions.

A. FORMULA

\( (X, Y) \) is used as a pair of samples in the training dataset, where \( X \) represents the input original image and \( Y \) is the corresponding real label image. According to the described object scale, \( Y \) is decomposed into \( S \) binary edge images, and the formula is:

\[
Y = \sum_{s=1}^{S} Y_s
\]

(1)

\( Y_s \) represents the edge image corresponding to scale \( s \).

\( Y_s \) is described as two complementary label edge images in BCDN: one to ignore the edges with larger scale than \( s \), and the other to ignore the edges with smaller scale. The two complementary label images are defined by:

\[
y_{s}^{2d} = Y - \sum_{i<s} p_{i}^{2d}
\]

(2)

\[
y_{s}^{d2} = Y - \sum_{i>s} p_{i}^{d2}
\]

(3)

In the formula, \( s2d \) represents information propagated from the shallow layers to the deep layers, and \( d2s \) represents information propagated from the deep layers to the shallow layers. Therefore, \( Y_s \) is defined as \( Y_s = y_{s}^{2d} + y_{s}^{d2} \).

The predicted values of edge \( p_{s}^{2d} \) and \( p_{s}^{d2} \) are approximately \( y_{s}^{2d} \) and \( y_{s}^{d2} \) at scale \( s \), respectively, and their sum is approximately \( Y_s \). The formula is:

\[
p_{s}^{2d} + p_{s}^{d2} - 2Y - \sum_{i<s} p_{i}^{2d} - \sum_{i>s} p_{i}^{d2}
\]

(4)

Based on the above formula, the edge prediction results are interpolated by \( p_{s}^{2d} + p_{s}^{d2} \) at scale \( s \). To make full use of the image multiscale features to train the neural network, we propose a bidirectional multiscale refinement network model, whose specific structure is described in Section B.

B. NETWORK STRUCTURE

The overall structure of the proposed network architecture is shown in Fig 2. The proposed network is composed of multiple ID Blocks and Refinement Blocks, each of which is trained by layer-specific supervisions inferred by a bi-directional cascade structure, as shown in Fig. 2(a). The optimized VGG16 [44] is used in the backbone network, which removes three full connection layers and the final pooling layer to decrease the numbers of parameters and remove the unclear features due to too small a scale. The 13 convolution layers of VGG16 [44] are divided into five modules (ID Block). A \( 2 \times 2 \) max pooling layer is used between each ID Block module to increase the receptive field layer by layer. Richer multiscale hierarchical features can be obtained in the horizontal direction of the network structure. The output feature maps with different resolutions by different ID Blocks are input to the corresponding Refinement Block in a cascade manner, as shown in Fig. 2(b). This structure can integrate the multi-scale features generated by the coding process, and allows us to make full use of the multi-resolution features of the side output to conduct deep supervised learning in the training process. Previous studies have shown that this type of structure does a
good job of capturing hierarchical features and produces semantic edge contours through fusion.

FIGURE 2. The architecture of BMRN. (a) is the overall architecture of BMRN. For illustration, we did not show the final result that the predictions of refine blocks are fused. (b) is the detailed architecture of BMRN. For illustration, we only show 3 ID Blocks and 2 Refinement Blocks, and the cascade from shallow to deep. It is similar from deep to shallow.

FIGURE 3. ID Block architecture. (a) represents the structure diagram of ID Block 1 and 2, where C represents the number of channels, 64 and 128, respectively. (b) represents the structure diagram of ID Block 3, 4, and 5. Channel number C is 256, 512, and 1024, respectively. SEM is the dilated convolution module. PAM represents the proposed parallel attention model.

ID Blocks and Refinement Blocks are the basic module of the entire network model. We use the attention mechanism to improve the ID Blocks of BDCN [42], so that it is more conducive to extract the local details of the contours on the basis of retaining the SEM module. The input feature image is propagated from shallow layers to deep layers along the main coding network, and the predicted edges $p_2^{d2}$ and $p_2^{s2d}$ of the output are generated after each intermediate Refinement Block. $p_2^{d2}$ and $p_2^{s2d}$, $p_3^{d2}$ and $p_3^{s2d}$ are used as training samples for bidirectional supervised training of the network model. Finally, all the side outputs of the middle layer are fused to calculate the final edge contour. This network structure does not increase the horizontal depth of the network expands the network vertically. On the contrary, it increases the nonlinearity of the model, reduces the correlation of adjacent pixels, and is more conducive to the extraction of crisp edges.

1) PARALLEL ATTENTION MODEL (PAM)

Inspired by CBAM [32], SENet [34], and DANet [33], we observe that each channel of the high-level feature map can be regarded as a specific class response, and that different semantic responses are related to each other. The channel attention model can enhance the expression of specific semantic features and improve the most important features in the channel feature map. The position attention model, which is a part of parallel attention model, can encode the spatial context information of a wide range of feature images into localness features, so that the semantic part of the feature map is enhanced. Our goal is to detect the crisp contour, and to do so we need to obtain more details of the local contour. Based on this, we propose a novel parallel attention model that uses the position attention module in DANet [33] and the SE module in SENet [34]. Fig 4 shows the details of the proposed attention model, which can be embedded into some network models as an entire module. The SE module and the position attention module can be connected in parallel and in series in a network structure. The series connection can be further divided into two parts, according to the sequence of the two modules, with either the SE module or position attention module in the front. For the above connection, the ablation experiment shows that parallel connection can better extract the local features of the contour. For a further trade-off of the proportion of the position attention and SE attention modules, the feature values processed by position attention and SE attention are multiplied by factors $\alpha$ and $\beta$, and then pixel level summation is performed. At the beginning of training, $\alpha$ and $\beta$ can be given an initial value, and then the optimal solution can be achieved gradually in the training process.

FIGURE 4. PAM model structure diagram.

As shown in Fig 4, the calculation process of the position attention module is shown in formulas (5) and (6).

$$S_{ij} = \frac{\exp(h_i, c_j)}{\sum_{i'} \exp(h_i, c_j)}$$  \hspace{1cm} (5)

In formula (5), we first feed feature map $X \in R^{C \times H \times W}$ into a convolution layer to generate two new feature maps, B and C, and transform their dimensions to $\{B, C\} \in R^{C \times N}$, where $N = H \times W$ represents the number of pixels. Matrix multiplication is performed between the transpose of B and C, and then the spatial attention feature map is obtained by the softmax function calculation.

$$\hat{X}_j = \sum_{i=1}^{N}(s_{ij}D_i) + X_j$$  \hspace{1cm} (6)

In formula (6), we feed feature X into a convolution layer to generate a new feature map D, and its dimension is transformed into D $\in R^{C \times N}$. Matrix multiplication is performed between the transpose of D and S, and its dimension is transformed into $R^{C \times H \times W}$. Finally, perform an element-wise sum operation with the features X to obtain the final output $\hat{X} \in R^{C \times H \times W}$.

As shown in Fig 4, the calculation process of SE module is shown in formulas (7), (8), and (9).

$$z_c = F_{sq}(X_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} X_c(i, j)$$  \hspace{1cm} (7)

In formula (7), $X \in R^{C \times H \times W}$ represents the input image, and the feature map of the c-th channel of input image X is processed by average pooling layer to obtain $z_c$.

$$s = F_{ec}(z, W) = \sigma(g(z, W)) = \sigma(W \delta(W, z))$$  \hspace{1cm} (8)

where $s = [s_1, s_2, \ldots, s_C], W_1 \in R^{C \times C}$ and $W_2 \in R^{C \times N}, \delta$ denotes the Relu function, and $\sigma$ the activation function.

$$\hat{X}_c = F_{scale}(s_c, X_c) = s_c X_c$$  \hspace{1cm} (9)

Formula (9) represents the c-th channel feature map of output image $\hat{X}$, which is obtained by multiplying scalar $s_c$ with the c-th channel feature map of input image $X_c$.

The final formula definition of parallel attention model is:

$$X' = \alpha \times SE(X) + \beta \times M_p(X)$$  \hspace{1cm} (10)

In formula (10), SE represents the squeeze and exception module, and $M_p$ represents the position attention module. $\alpha$ and $\beta$ are given an initial value at the beginning of training (the empirical value is set to 0.01 and 1.1, respectively), and the optimal solution is obtained in the training process.

2) REFINEMENT BLOCK

Inspired by DRC [41], in order to better integrate multiscale features, we propose the refinement Block to transform the original network model. The main idea is to use the middle edge feature map to aggregate the edge evidence of edges. The structure is shown in Fig 5(a), including middle edge convolution layer, sub-pixel convolution, and element-wise add layer. As shown in Fig 5(b), the weighted convolution layer uses two identical feature maps as inputs, one adopting $1 \times 1$
convolution following ReLU layer and another adopting a sigmoid function activated trainable parameter \( \alpha \). Through the weight convolution layer, different feature weights can be automatically learned in the end-to-end training model, which is more effective than manually specifying weights. According to the different resolution of each ID Block output feature map, we use sub-pixel convolution to transform a low-resolution feature map into one of high resolution. We use sub-pixel convolution rather than up-sample, because up-sample did not restore the spatial details of pixels by using a bilinear interpolation algorithm, which makes the edges of the detected objects more blurred. Sub-pixel convolution is a standard convolution layer, which rearranges the feature values, called phase shift. This method helps to eliminate the blocking effect in the image super-resolution task and keeps the computational cost low. Sub-pixel convolution is highly effective for accurate edge location. Finally, we adopt an element-wise add layer to fuse the outputs of multiple weighted convolution layers, because the element-wise add layer has fewer training parameters than a \( 1 \times 1 \) convolution layer after concatenating the outputs of multiple weighted convolution layers. The output channel is set to the dimension of the smallest one in the same refinement block.

**FIGURE 5.** (a) Structure diagram of the proposed refinement module. (b) Structure diagram of the weighted convolution layer.

### C. ADAPTIVE WEIGHT LOSS FUNCTION

Loss function is an important part of end-to-end CNN, because it directly affects the prediction results. In CNN-based edge detection, the design of loss function needs to pay attention to three things: the imbalance between positive and negative samples; the combination of multiple losses; and the accuracy of test results. Based on these three considerations, we fuse cross-entropy and dice coefficient by adaptive weight, which improves the performance of edge detection. Due to the high imbalance between the edge and non-edge pixels of the input image in contour detection, HED [12] uses the weighted cross-entropy loss function to solve this problem, so that the end-to-end CNN can be trained. The expression of weighted cross-entropy loss function is:

\[
L(W, w) = -\gamma \sum_{j \in E} \log Pr(y_j = 1|X; W, w) - (1 - \gamma) \sum_{j \in N} \log Pr(y_j = 0|X; W, w)
\]

(11)

In formula (11), \( Y_e \) represents edge pixels, \( Y_n \) represents non-edge pixels, \( \gamma = |Y_e|/|Y| \), and \( 1 - \gamma = |Y_n|/|Y| \). \( X \) represents the input image, and \( Pr(y_j|X; W, w) \) is the classification probability of pixel \( y_j \) through the softmax function.

**FIGURE 6.** A simple test on the class-balance weight \( \gamma \). From left to right: (a) Original map of polar bear in BSDS500 dataset. (b) Detection result using normal cross-entropy function, i.e., \( \gamma = 0.5 \). (c) Detection result using weighted cross-entropy loss function.

In LPCB [15], Deng et al. found that the edge detected by the weight cross-entropy loss function is relatively thick, while the normal cross-entropy loss function did not detect the complete edge, but it could obtain details of the edge. The experimental results are shown in Fig 6. To learn the crisp contour, Deng et al. proposed the dice coefficient loss function, the loss function is given by:

\[
L_D(P, Y) = \text{Dist}(P, Y) = \frac{\sum_{i=1}^{N} p_i^2 + \sum_{i=1}^{N} y_i^2}{2 \sum_{i=1}^{N} p_i y_i}
\]

(12)

In formula (12), \( p_i \) and \( y_i \) represent the corresponding values of prediction map \( P \) and ground truth \( Y \) at the \( i \)-th pixel, respectively. The dice loss can be used to measure the approximation of two sets of image pixels, and it is not necessary to consider the imbalance between edge pixels and non-edge pixels. Dice loss is considered as the measure function of image-level similarity, while cross-entropy loss focuses on pixel-level difference. Therefore, the fusion of these two loss functions can minimize the distance between images and labels from the image level to the pixel level. In LPCB [15], it is necessary to manually set two weight values of loss function. Because these are empirical values obtained through many experiments, it is easy for errors to occur. To solve this problem, according to the principle of Nash equilibrium, take \( \lambda L_D(P, Y) + \theta L_C(P, Y) \) as one side of the game, and then construct a term \( \log (1 + \frac{1}{x}) \) opposite to the original cost function as the other side of the game. In order for the network model to achieve end-to-end training, and finally obtain the weight coefficient \( \lambda \) and \( \theta \) value to achieve Nash equilibrium, we propose an adaptive weight loss function. The formula is:

\[
L(P, Y) = \lambda L_D(P, Y) + \theta L_C(P, Y) + \log \left(1 + \frac{1}{\lambda \theta}ight)
\]

(13)

In formula (13), \( L_D(P, G) \) is equation (12), \( L_C(P, Y) \) represents the normal cross-entropy loss \( L_C(P, Y) = -\sum_i^N \left(g_j \log p_j + (1 - g_j) \log (1 - \log p_j) \right) \), and \( N \) is the total pixel number of input images. In the training process, the larger values of \( \lambda \) and \( \theta \) increase the contribution of \( \lambda L(D, G) + \theta L_C(W, w) \), while the smaller values of \( \lambda \) and \( \theta \) decrease its contribution. The last term \( \log (1 + \frac{1}{\lambda \theta}) \) is the regularization term of parameters \( \lambda \) and \( \theta \).

All ID Blocks and Refinement Blocks are trained with two layer-specific side supervisions in our network. In addition, we fuse the intermediate edge predictions with a fusion layer as the final result. Therefore, BMRN is trained with three types of loss. We define the total loss \( L_{\text{final}} \) as,

\[
L_{\text{final}} = \omega_{\text{side}} \cdot L_{\text{side}} + \omega_{\text{fuse}} \cdot L_{\text{fuse}}(P, Y)
\]

(14)

\[
L_{\text{side}} = \sum_{s=1}^{S} L\left(P_s^{d2s}, Y_s^{d2s}\right) + L\left(P_s^{s2d}, Y_s^{s2d}\right)
\]

(15)

where \( \omega_{\text{side}} \) and \( \omega_{\text{fuse}} \) are weights for the side loss and fusion loss, respectively. \( L\left(P_s^{d2s}, Y_s^{d2s}\right) \), \( L\left(P_s^{s2d}, Y_s^{s2d}\right) \) and \( L_{\text{fuse}}(P, Y) \) all adopt the formula (13). \( P \) represents the final edge prediction, \( Y = \sum_{s=1}^{S} P_s^{d2s} + \sum_{i=1}^{S} P_i^{s2d} \). \( Y \) represents the ground truth.

**IV. EXPERIMENT**

In this section, we introduce the dataset and hyperparameter settings and discuss the use of ablation experiments to verify the proposed method. The ablation experiment results show that the subsystems of the proposed model affect the performance of the entire network. In addition, further
comparisons with state-of-the-art methods verified that our method has better performance in detecting crisp contour.

A. EXPERIMENT DETAILS

1) DATASET ENHANCEMENT

In the experiments, the widely used BSDS500 [3], NYUDv2 [6] depth, and Multi-Cue edge/boundary [21] datasets were used to train and evaluate our model. The BSDS500 consists of 500 image pairs, the training set consists of 200 image-label pairs, the verification set consists of 100 image-label pairs, and the test set consists of 200 image-label pairs. NYUDv2 consists of 1449 image pairs of RGB and depth images. The training set includes 381 image-label pairs, the verification set includes 414 image-label pairs, and the test set includes 654 image-label pairs. To avoid insufficient training and poor robustness due to the lack of training images, the training set on the BSDS500 dataset was randomly scaled by 0.75–1.25 times, and the image-label pairs were rotated by 16 different angles and flipped at the same time, so that the number of image-label pairs in the training set reached more than 10,000. On the NYUDv2 dataset, image label pairs were rotated at four angles (0°, 90°, 180°, and 270°) and flipped at each angle. Multi-Cue contains 100 natural scenes. Each scene has two frame sequences, one obtained from the left view and one from the right view. The last frame of the left view sequence is labeled as edge and boundary. According to HED [12] and RCF [13], we randomly divided 100 labeled scene images, 80 for training and 20 for testing. Three experiments were carried out independently, and the average results of three experiments were used as the final output.

2) HYPERPARAMETERS

In this paper, we used the PyTorch 1.20 framework to implement the proposed neural network model. The network model uses SGD as the optimizer, and the backbone network uses the VGG16 [44] model pretrained on the ImageNet dataset. Other hyperparameters are set as follows: min batch size (10), global learning rate 1e-4, and momentum (0.9). In the BSDS500 and NYUDv2 datasets, the iterative training was 40,000 times, and the learning rate decreased 10 times every 10,000 times. We trained 2,000 and 4,000 iterations for Multi-Cue edge/boundary, \( \omega_{\text{side}} \) and \( \omega_{\text{fuse}} \) were set to 0.5 and 1.1, respectively, and all experiments were performed on a Nvidia GeForce RTX 2070 video card with 8 GB of RAM.

Based on the previous work, the standard non-maximum suppression was performed to obtain the final edge. To evaluate the performance of different methods, we used average precision (AP) and F-measure, the latter at both optimal dataset scale (ODS) and optimal image scale (OIS) [30,31]. To make the edge prediction value match the ground truth annotations, the maximum distances on BSDS500 and NYUDv2 were set to 0.0075 and 0.011, respectively [14].

B. ABLATION EXPERIMENTS

We performed ablation experiments to analyze the performance of each subsystem in our method in the BSDS500 dataset. We used the BSDS500 training set and validation set to train the network model and evaluated the performance of the model on the test set. Table 1 shows the influence of different attention modules in the proposed network model on performance. It can be seen that the proposed PAM has better performance than previous attention models. In addition, we applied the proposed loss function to HED [12], CED [14], and the proposed network model BMRN for comparative experiments. The results are summarized in Table 2, which shows that the proposed loss function can achieve better output results.

Table 3 shows the ablation experimental results of BDCN combined with the proposed modules. It can be seen from the experimental results that each proposed module improves the performance of BDCN model, and the experimental performance of our algorithm is the best. Compared with BDCN model, our method improves the values of ODS, OIS and AP by 2.73%, 1.82% and 3.31%, respectively. During the ablation experiment, other parameters had the same settings.

Table 1. Quantitative study of the influence of different attention models on the performance of the proposed network.

| Method         | ODS   | OIS   | AP    |
|----------------|-------|-------|-------|
| BMRN+CBAM      | 0.792 | 0.801 | 0.841 |
| BMRN+DAE       | 0.802 | 0.805 | 0.843 |
| BMRN+SE        | 0.791 | 0.794 | 0.836 |
| BMRN+PAM       | 0.828 | 0.841 | 0.875 |

Table 2. Performance comparison of the proposed loss function on different models.

| Loss Function       | ODS   | OIS   | AP    |
|---------------------|-------|-------|-------|
| HED+Proposed Loss   | 0.791 | 0.809 | 0.843 |
| CED+Proposed Loss   | 0.804 | 0.825 | 0.873 |
| BDCN+Proposed Loss  | 0.821 | 0.834 | 0.867 |
| Ours                | 0.828 | 0.841 | 0.875 |

Table 3. Comparison of ablation experimental results between BDCN and the proposed module on bsds500 validation set.

| Method                | ODS   | OIS   | AP    |
|-----------------------|-------|-------|-------|
| BDCN                  | 0.806 | 0.826 | 0.847 |
| BDCN+PAM              | 0.814 | 0.831 | 0.854 |
| BDCN+ Refine Block    | 0.818 | 0.830 | 0.856 |
| BDCN+ Proposed Loss   | 0.821 | 0.834 | 0.867 |
| Ours                  | 0.828 | 0.841 | 0.875 |

C. COMPARISON WITH OTHER METHODS

1) PERFORMANCE COMPARISON ON BSDS500

The proposed method was compared with several non-deep learning methods and deep learning-based contour detection methods, including N4-field[26], DeepContour[11], DeepEdge [4], HED[12], RCF[13], CED[14], LRC[47], TIN[49], and PDiNet[50], and traditional contour detection methods, including Canny [1], gPb [3], SE [8], and PMI [18]. The results are summarized in Table 4 and Fig 7. The proposed method achieved better results (ODS = 0.828, OIS = 0.841, AP = 0.875)
than the previous best method CED [14] and also exceeded the human standard on the BSDS500 dataset (ODS = 0.803). It can be seen from the qualitative results in Fig 8 that our method is thinner and crisper than the output of HED [12] and CED [14]. Note that in Table 4, the suffix of the network model is MS, which means multi-scale fusion is adopted, and others use single scale. In single-scale edge detection, we input an original image into the network model, and then the edge probability map is output. In order to improve the quality of prediction edge, we created image pyramids to detect multi-scale edges, and each of these images is separately input to the network model. All prediction edge probability maps are resampled to the original image size using bilinear interpolation. Finally, these prediction maps are averaged to obtain a final prediction map. Trade off accuracy and speed, we adopted three scales of 0.5, 1.0, and 2.0.

Table 4. Performance comparison between our method and other methods on BSDS500 test set. † indicates the cited GPU speeds. ‡ indicates the speeds with our implementations based on a NVIDIA RTX 2070 GPU.

| Method      | ODS  | OIS  | AP   | FPS  |
|-------------|------|------|------|------|
| Canny       | 0.611| 0.676| 0.520| 28   |
| gPb         | 0.729| 0.755| 0.745| 1/240|
| SE          | 0.743| 0.764| 0.800| 2.5  |
| PMI         | 0.741| 0.769| 0.799| -    |
| N^2-fields  | 0.753| 0.772| 0.802| 1/5† |
| DeepEdge    | 0.753| 0.772| 0.787| 1/1000†|
| DeepContour | 0.757| 0.776| 0.790| 1/30† |
| HED         | 0.788| 0.808| 0.840| 30†  |
| RCF         | 0.806| 0.823| -    | 30†  |
| RCF-MS      | 0.811| 0.830| -    | 8†   |
| CED         | 0.803| 0.820| 0.871| 30†  |
| CED-MS      | 0.815| 0.833| 0.889| 10†  |
| LPCB        | 0.800| 0.816| -    | 30†  |
| LPCB-MS     | 0.815| 0.834| -    | 10†  |
| BDCN        | 0.806| 0.826| 0.847| 35‡  |
| BDCN-MS     | 0.828| 0.844| 0.890| 12‡  |
| LRC         | 0.802| 0.821| 0.830| -    |
| LRC+MS      | 0.816| 0.836| 0.864| -    |
| TIN1        | 0.749| 0.772| -    | 30†  |
| TIN2        | 0.772| 0.795| -    | 30†  |
| PidNet      | 0.807| 0.823| -    | 78‡  |
| BMRN        | 0.828| 0.841| 0.875| 33‡  |
| BMRN-MS     | 0.832| 0.846| 0.891| 11‡  |

Table 5. Quantitative comparison between the proposed method and other methods on NYUDv2. † indicates the cited GPU speeds. ‡ indicates the speeds with our implementations based on a NVIDIA RTX 2070 GPU.

| Method       | ODS  | OIS  | AP   | FPS  |
|--------------|------|------|------|------|
| gPb-UCM      | 0.632| 0.661| -    | 1/365|
| gPb-NG       | 0.687| 0.716| 0.629| 1/375|
| SE           | 0.695| 0.708| 0.679| 5    |
| HED          | RGB  | 0.720| 0.734| 0.734| 20†  |
|              | HHA  | 0.682| 0.695| 0.702| 20†  |
|              | RGB-HHA| 0.746| 0.761| 0.786| 10†  |
| RCF          | RGB  | 0.729| 0.742| -    | 20†  |
|              | HHA  | 0.705| 0.715| -    | 20†  |
|              | RGB-HHA| 0.757| 0.771| -    | 10†  |
| AMH-Net-ResNet50 | RGB      | 0.744| 0.758| 0.765| -    |
|              | HHA  | 0.716| 0.729| 0.734| -    |
|              | RGB-HHA| 0.771| 0.786| 0.802| -    |
| LPCB         | RGB  | 0.739| 0.754| -    | 30†  |
|              | HHA  | 0.707| 0.719| -    | 30†  |
|              | RGB-HHA| 0.762| 0.778| -    | 15†  |
| BDCN         | RGB  | 0.748| 0.763| 0.770| 25‡  |
|              | HHA  | 0.707| 0.719| 0.731| 25‡  |
|              | RGB-HHA| 0.765| 0.781| 0.813| 13‡  |

FIGURE 7. Precision-Recall curves of different methods on BSDS500. Our method achieves the best result (F = 0.832).

FIGURE 8. Qualitative comparison of detection methods on BSDS500. From left to right: (a) is original image, (b) is ground truth, (c) is the detection result of HED, (d) is the detection result of CED, and (e) is the detection result of our method. All results were not processed by non-maximum suppression.

2) PERFORMANCE COMPARISON ON NYUDV2
The NYUDv2 dataset contains two sub-datasets: RGB dataset and HHA dataset. Experiments were carried out on all two sub-datasets, and the final result of RGB-HHA was obtained by averaging the prediction results of the RGB and HHA training models. The proposed method was compared with several non-deep learning methods, including gPb-UCM [3], gPb-NG [45], and SE [8], and deep learning methods, including HED [12], RCF [13], AMH-Net-Resnet [41], LPCB [15], BDCN [42], LRC [47], TIN [49], and PidNet[50]. All experiments were based on single-scale inputs. The results of the quantitative comparison experiments are summarized in Table 5, from which we see that the RGB-HHA version performs better than the RGB and HHA data training versions in terms of ODS, OIS, and AP. It can also be seen that COB-ResNet50 [19] achieves the best results, because it uses UCM [3] in the post-processing, which has better performance for contour detection. Fig 9 shows the precision-recall curve of the different methods. The qualitative results of our method and CED [14] are summarized in Fig 10. Compared with CED [14], our prediction results obtain crisper edges, which proves the effectiveness of our method.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
The article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3146339, IEEE Access

**Table 6.** Quantitative comparison between the proposed method and recent methods on Multi-Cue. ‡ indicates the speeds with our implementations based on a NVIDIA RTX 2070 GPU.

| Cat      | Method       | ODS         | OIS         | AP          | FPS          |
|----------|--------------|-------------|-------------|-------------|--------------|
| Boundary | Human        | 0.760(0.017) | -           | -           | -            |
|          | Multicue     | 0.720(0.014) | -           | -           | -            |
|          | HED          | 0.814(0.011) | 0.822(0.008) | 0.869(0.015) | 12‡          |
|          | RCF          | 0.817(0.004) | 0.825(0.005) | -           | 10‡          |
|          | RCF-MS       | 0.825(0.008) | 0.836(0.007) | -           | 7‡           |
|          | BDCN         | 0.836(0.001) | 0.846(0.003) | 0.893(0.001) | 7‡           |
|          | BDCN-MS      | 0.838(0.004) | 0.853(0.009) | 0.906(0.005) | 5‡           |
|          | PiDiNet      | 0.818(0.003) | 0.830(0.005) | -           | 11‡          |
|          | BMRN         | 0.848(0.002) | 0.857(0.004) | 0.898(0.002) | 6‡           |
|          | BMRN-MS      | **0.852(0.003)** | **0.864(0.005)** | **0.918(0.006)** | **5‡**        |

| Edge     | Human        | 0.750(0.024) | -           | -           | -            |
|          | Multicue     | 0.830(0.002) | -           | -           | -            |
|          | HED          | 0.851(0.014) | 0.864(0.011) | -           | 11‡          |
|          | RCF          | 0.857(0.004) | 0.862(0.004) | -           | 9‡           |
|          | RCF-MS       | 0.860(0.005) | 0.864(0.004) | -           | 7‡           |
|          | BDCN         | 0.891(0.001) | 0.898(0.002) | 0.935(0.002) | 6‡           |
|          | BDCN-MS      | 0.894(0.002) | 0.901(0.004) | 0.941(0.005) | 5‡           |
|          | PiDiNet      | 0.855(0.007) | 0.860(0.005) | -           | 10‡          |
|          | BMRN         | 0.905(0.002) | 0.912(0.003) | 0.942(0.001) | 5‡           |
|          | BMRN-MS      | **0.912(0.003)** | **0.918(0.005)** | **0.952(0.004)** | **4‡**        |

To further demonstrate its advantages, we evaluated the performance of the proposed method and HED [12], RCF [13], and BDCN [42] in their respective intermediate output layers. The experiment was carried out on BSDS500. The F-measure values of the middle output layer prediction edge of different methods are summarized in Table 7. It can be observed that the prediction results of our method are consistently higher than HED [12], RCF [13], and BDCN [42].

**Table 7.** Performance (ODS) of each layer in our method, HED, RCF, and BDCN on the BSDS500 test set.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
| Layer ID     | HED  | RCF  | BDCN | Ours |
|------------|------|------|------|------|
| Layer-One  | 0.595| 0.595| 0.727| 0.731|
| Layer-Two  | 0.697| 0.710| 0.762| 0.768|
| Layer-Three| 0.750| 0.766| 0.771| 0.774|
| Layer-Four | 0.748| 0.761| 0.802| 0.812|
| Layer-Five | 0.637| 0.758| 0.815| 0.818|
| Fuse       | 0.790| 0.805| 0.820| 0.824|

V. CONCLUSION

In this paper, we analyze the reason that CNN-based edge detectors do not produce crisp edge, and propose a novel model that largely improves the localness performance of CNN-based edge detectors. The proposed network model can make each middle layer focus on a specific scale through a bidirectional multiscale refinement structure, and train the network model through specific supervision in each middle layer. In order to accurately locate the edge pixels, we propose a refinement block and parallel attention model, and embed them into the entire model structure. At the same time, to solve the problem of a significant imbalance between edge and non-edge pixels, an adaptive weight loss function is proposed. Independent experiments were carried out on the BSDSS500, NYUD-v2, and Multi-Cue datasets, and the results verify that the proposed method has better performance than the state-of-the-art.

REFERENCES

[1] J. M. PREWITT, Object enhancement and extraction[J]. Picture Processing and Psychopictorics,1970,10(1):15-19
[2] J. Canny, "A Computational approach to edge detection," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-8, no. 6, pp. 679-698, Nov. 1986, doi: 10.1109/TPAMI.1986.4767851.
[3] P. Arbeláez, M. Maire, C. Fowlkes and J. Malik, "Contour detection and hierarchical image segmentation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 5, pp. 898-916, May 2011, doi: 10.1109/TPAMI.2010.161.
[4] G. Bertasius, J. Shi and L. Torresani, "DeepEdge: A multi-scale bifurcated deep network for top-down contour detection," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 4380-4389, doi: 10.1109/CVPR.2015.729067.
[5] G. Bertasius, J. Shi and L. Torresani, "High-for-Low and Low-for-High: Efficient boundary detection from deep object features and its applications to High-Level vision," 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 504-512, doi: 10.1109/ICCV.2015.65.
[6] N. Silberman, D. Hoiem, P. Kohli and R. Fergus, Indoor segmentation and support inference from RGBD images. In: Fitzgibbon A., Lazebnik S., Perona P., Sato Y., Schmid C. (eds) Computer Vision – ECCV 2012. ECCV 2012. Lecture Notes in Computer Science, vol 7576. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-33715-4_54
[7] Y. Zhang, Q. Xian, X. Tan, J. Han and Y. Tang, "Sketch-based image retrieval by salient contour reinforcement," in IEEE Transactions on Multimedia, vol. 18, no. 8, pp. 1604-1615, Aug. 2016, doi: 10.1109/TMM.2016.2568138.
[8] P. Dollár, C. L. Zitnick, "Structured forests for fast edge detection," 2013 IEEE International Conference on
[16] P.O. Pinheiro, T.Y. Lin, R. Collobert and Dollár P. (2016) Learning to refine object segments. In: Leibe B., Matas J., Sebe N., Welling M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, vol 9905. Springer, Cham. https://doi.org/10.1007/978-3-319-46448-0_5

[17] W. Shi et al., "Real-Time single image and video Super-Resolution using an efficient Sub-Pixel convolutional neural network," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1874-1883, doi: 10.1109/CVPR.2016.207.

[18] P. Isola, D. Zoran, D. Krishnan and Adelson E.H. Crisp boundary detection using pointwise mutual information. In: Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8691. Springer, Cham. https://doi.org/10.1007/978-3-319-10578-9_52

[19] K. Maninis, J. Pont-Tuset, P. Arbeláez and L. Van Gool, "Convolutional oriented boundaries: From image segmentation to High-Level tasks," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 4, pp. 819-833, 1 April 2018, doi: 10.1109/TPAMI.2017.2700300.

[20] K. Yang, C. Li and Y. Li, "Multifeature-Based surround inhibition improves contour detection in natural images," in IEEE Transactions on Image Processing, vol. 23, no. 12, pp. 5020-5032, Dec. 2014, doi: 10.1109/TIP.2014.2361210.

[21] D. R. Martin, C. C. Fowlkes and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 5, pp. 530-549, May 2004, doi: 10.1109/TPAMI.2004.1273918.

[22] S. Nowozin, C. Lampert, Structured learning and prediction in computer vision. Now Foundations and Trends, 2011.

[23] X. Ren, Multi-scale improves boundary detection in natural images. In: Forsyth D., Torr P., Zisserman A. (eds) Computer Vision – ECCV 2008. ECCV 2008. Lecture Notes in Computer Science, vol 5303. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-88690-7_40

[24] Y. Li, M. Paluri, J. M. Rehg and P. Dollar, "Unsupervised learning of edges," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1619-1627, doi: 10.1109/CVPR.2016.179.

[25] Z. Zhang, F. Xing, X. Shi and L. Yang, "SemiContour: A Semi-Supervised learning approach for contour detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 251-259, doi: 10.1109/CVPR.2016.34.

[26] Y. Ganin, V. Lempitsky, "N4-fields: Neural network nearest neighbor fields for image transforms," in Asian Conference on Computer Vision, 2014, pp. 536-551: Springer.

[27] J. Yang, B. Price, S. Cohen, H. Lee and M. Yang, "Object contour detection with a fully convolutional Encoder-Decoder network," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 193-202, doi: 10.1109/CVPR.2016.28.

[28] R. Mottaghi et al., "The role of context for object detection and semantic segmentation in the wild," 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 891-898, doi: 10.1109/CVPR.2014.119.

[29] N. Murthy, V. Singh, T. Chen, R. Mannmatha and D. Comaniciu, "Deep decision network for Multi-class image classification," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2240-2248, doi: 10.1109/CVPR.2016.246.

[30] J. Pont-Tuset, F. Marques, "Supervised evaluation of image segmentation and object proposal techniques," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, no. 7, pp. 1465-1478, 1 July 2016, doi: 10.1109/TPAMI.2015.2481406.

[31] J. Pont-Tuset and F. Marques, "Measures and Meta-Measures for the supervised evaluation of image segmentation," 2015 IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1231-2138, doi: 10.1109/CVPR.2013.277.

[32] S. Woo, J. Park, J.Y. Lee and Kweon I.S. (2018) CBAM: Convolutional block attention module. In: Ferrari V., Hebert M., Sminchisescu C., Weiss Y. (eds) Computer Vision – ECCV 2018. ECCV 2018. Lecture Notes in Computer Science, vol 11211. Springer, Cham. https://doi.org/10.1007/978-3-030-01234-2_1

[33] Jun Fu, Jing Liu, Haisie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, Hanqing Lu. Dual attention network for scene segmentation. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 3146-3154

[34] Jie Hu, Li Shen and Gang Sun. Squeeze-and-Excitation networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 7132-7141

[35] J. Fu, H. Zheng and T. Mei, "Look closer to see better: Recurrent attention convolutional neural network for Fine-Grained image recognition," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4476-4484, doi: 10.1109/CVPR.2017.476.

[36] P. Krahnenbühl and V. Koltun, "Learning to propose objects," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1574-1582, doi: 10.1109/CVPR.2015.7987675.

[37] J. Kivinen, C. Williams and N. Heess. Visual boundary prediction:A deep neural prediction network and quality dissection. Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, PMLR 33:512-521, 2014.

[38] C. Peng, X. Zhang, G. Yu, G. Luo and J. Sun, "Large kernel matters — improve semantic segmentation by global convolutional network," 2017 IEEE Conference on
Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1743-1751, doi: 10.1109/CVPR.2017.189.

[39] A.G. Roy, N. Navab, C. Wachinger, Concurrent spatial and channel 'Squeeze & Excitation' in fully convolutional networks. In: Frangi A., Schnabel J., Davatzikos C., Alberola-López C., Fichtinger G. (eds) Medical Image Computing and Computer Assisted Intervention – MICCAI 2018. MICCAI 2018. Lecture Notes in Computer Science, vol 11070. Springer, Cham. https://doi.org/10.1007/978-3-030-09292-1_48

[40] Dan Xu, Wanli Ouyang, Xavier Alameda-Pineda, Elisa Ricci, Xiaogang Wang, Nicu Sebe. Learning Deep structured Multi-Scale features using Attention-Gated CRFs for contour prediction. In NIPS, pages 3964–3973, 2017.

[41] Y. J. Cao, C. Lin and Y. J. Li, "Learning crisp boundaries using deep refinement network and adaptive weighting loss," in IEEE Transactions on Multimedia, vol. 23, pp. 761-771, 2021, doi: 10.1109/TMM.2020.2987685.

[42] H. Jianzhong, Z. Shiliang, Y. Ming, S. Yanhu and H. Tiejun. Bi-Directional cascade network for perceptual edge detection. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 3828-3837

[43] J. Long, E. Shelhamer and T. Darrell, "Fully convolutional networks for semantic segmentation," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3431-3440, doi: 10.1109/CVPR.2015.7298965.

[44] K. Simonyan, A. Zisserman. Very deep convolutional networks for Large-Scale image recognition. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015

[45] S. Gupta, P. Arbeláez and J. Malik, "Perceptual organization and recognition of indoor scenes from RGB-D images," 2013 IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 564-571, doi: 10.1109/CVPR.2013.79.

[46] Fang, T., Fan, Y. & Wu, W. Salient contour detection on the basis of the mechanism of bilateral asymmetric receptive fields. SIViP 14, 1461–1469(2020). https://doi.org/10.1007/s11176-020-01689-1

[47] Q.Zhang, C.Lin, F.Z.Li. Application of binocular disparity and receptive field dynamics: A biologically-inspired model for contour detection. Pattern Recognit. 110: 107657 (2021), doi: https://doi.org/10.1016/j.patcog.2020.107657

[48] C.Lin, L.H.Cui, F.z.Li, Y.J.Cao, Lateral refinement network for contour detection, Neurocomputing, Volume 409,2020, Pages 361-371, https://doi.org/10.1016/j.neucom.2020.06.069.

[49] J. K. Wibisono and H. -M. Hang, "Traditional Method Inspired Deep Neural Network For Edge Detection," 2020 IEEE International Conference on Image Processing (ICIP), 2020, pp. 678-682, doi: 10.1109/ICIP40778.2020.9190982.

[50] Z.Su, W.Z.Liu, Z.TYu and D.W.Hu, et al. Pixel Difference Networks for Efficient Edge Detection, ArXiv Preprint ArXiv:2108.07009, 2021.

Shishui Bao received a master's degree and is currently pursuing a Ph.D. degree at the AnHui University of Science and Technology. His research interests include image processing, deep convolution neural networks and artificial intelligence.

Yourui Huang was born in 1971. He has a Ph.D. degree and he is a Professor with AnHui University of Science and Technology, Huai nan City, Anhui Province, China. His research interests include intelligent information process and wireless sensor networks.

Guangyu Xu was born in 1975. He received the master’s degree in the AnHui University of Science and Technology and Ph.D. degrees in computer science from the HeFei University of Technology. His research interests include digital image processing and artificial intelligence.