Deep Texture-Aware Features for Camouflaged Object Detection

Jingjing Ren¹,*, Xiaowei Hu²,*, Lei Zhu², Xuemiao Xu¹,†, Yangyang Xu¹,
Weiming Wang³, Zijun Deng¹, and Pheng-Ann Heng²
¹ South China University of Technology, ² The Chinese University of Hong Kong,
³ The Open University of Hong Kong

Abstract

Camouflaged object detection is a challenging task that aims to identify objects having similar texture to the surroundings. This paper presents to amplify the subtle texture difference between camouflaged objects and the background for camouflaged object detection by formulating multiple texture-aware refinement modules to learn the texture-aware features in a deep convolutional neural network. The texture-aware refinement module computes the covariance matrices of feature responses to extract the texture information, designs an affinity loss to learn a set of parameter maps that help to separate the texture between camouflaged objects and the background, and adopts a boundary-consistency loss to explore the object detail structures. We evaluate our network on the benchmark dataset for camouflaged object detection both qualitatively and quantitatively. Experimental results show that our approach outperforms various state-of-the-art methods by a large margin.

1. Introduction

In nature, animals try to conceal themselves by adapting the texture of their bodies to the texture of the surroundings, which helps them avoid being recognized by predators. This strategy is easily used to deceive the visual perceptual system [34] and current vision algorithms may fail to distinguish camouflaged objects from the background.

Hence, camouflaged object detection [7] has been a great challenge and solving this problem could benefit a lot of applications in computer vision, such as polyp segmentation [8], lung infection segmentation [9, 36], photo-realistic blending [11], and recreational art [4].

To solve this problem, Fan et al. [7] collects the first large-scale dataset for camouflaged object detection. The dataset includes 10,000 images with over 78 object categories in various natural scenes, which make it possible to apply deep learning algorithms to learn to recognize camouflaged objects from large data. Camouflaged objects usually have similar texture to the surroundings, and deep learning algorithms designed for generic object detection [28, 27, 12, 2, 40, 29, 3] and salient object detection [42, 41] generally do not perform well to detect camouflaged objects in such difficult situations. Recently, algorithms for camouflaged object detection based on deep neural networks have been proposed. SINet [7] first adopts a search module to find the candidate regions of camouflaged objects and then uses an identification module to precisely detect camouflaged objects. ANet [19] first leverages a classification network to identify whether the image contains camouflaged objects or not, and then adopts a fully convolutional network for camouflaged object segmentation. However, these algorithms may still misunderstand camouflaged objects as the background due to their similar texture.

Texture refers to a particular way how visual primitives are organized spatially in natural images [7]. Essentially, there are subtle differences in the texture between camouflaged objects and the background. As shown in Figure 1 (a), the texture of the fish involves a combination of dense and small white particles and brown regions while the texture of background has the combination of white and brown regions. Based on this observation, we present to amplify the texture difference between camouflaged objects and the background by learning texture-aware features from the deep neural network, thus improving the performance of camouflaged object detection; see Figure 1 (c)-(e).

To achieve this, we design the texture-aware refinement module (TARM) in a deep neural network, where we first compute the covariance matrices of feature responses to extract the texture information from the convolutional features, and then learn a set of affinity functions to amplify the texture difference between camouflaged objects and the background. Moreover, we design a boundary-consistency loss in TARM to improve the segmentation quality by revisiting the image patches across boundaries on high-resolution feature maps. After that, we adopt multiple...
TARMs in a deep network (TANet) to learn deep texture-aware features in different layers and predict a detection map per layer for camouflaged object detection. Finally, we qualitatively and quantitatively compare our approach with 13 state-of-the-art methods designed for camouflaged object detection, salient object detection, and semantic segmentation on the benchmark dataset, showing the superiority of our network.

We summarize the contributions of this work as follow:

• First, we design a novel texture-aware refinement module (TARM) to amplify the texture difference between camouflaged objects and the background, which significantly enhances camouflaged object recognition.

• Second, we design a boundary-consistency loss to enhance detail information across boundaries without extra computation overhead in testing.

• Third, we evaluate our network on the benchmark dataset and compare it with 13 state-of-the-art methods on camouflaged object detection, salient object detection, and semantic segmentation. Qualitative and quantitative results show that our approach outperforms previous methods by a large margin.

2. Related Work

Camouflaged object detection. Early work for camouflaged object detection (COD) adopted various hand-crafted features, e.g., color [15, 31], convex intensity [35], edge [31], and texture [1, 17]. Recently, deep convolution neural network achieves great success with the help of large-scale camouflaged object datasets [32, 19, 7]. SINet [7] found the candidate regions of camouflaged objects by a search module and precisely detected camouflaged objects through an identification module. ANet [19] first identified the image that contains camouflaged objects through a classification network and then adopted a fully convolutional network for camouflaged object segmentation. These methods, however, do not consider the subtle texture difference between camouflaged objects and the background, and may fail to detect camouflaged objects in complex situations.

Salient object detection and semantic segmentation. Salient object detection (SOD) predicts a binary mask to indicate the saliency regions while semantic segmentation (SS) aims to generate masks with category labels to identify the image regions with different classes. The deep-learning-based methods designed for SOD [21, 26, 37, 41, 42] and SS [12, 43, 40, 2, 14] can be employed for camouflaged object detection by retraining them on camouflaged object datasets, and we compare our network with these methods for camouflaged object detection on the benchmark dataset; see Section 4.
3. Methodology

Figure 2 shows the overall network architecture (TANet) with texture-aware refinement modules (TARM) for camouflaged object detection. Given the input image, we adopt the feature extractor to extract the feature maps with multiple resolutions, and then use the residual refine blocks (RRB) to refine the feature maps at different layers to enhance the fine details and remove the background noises.

We ignore to refine the feature maps at the first layer due to the large memory footprint. Next, we present the texture-aware refinement module (TARM) to learn the texture-aware features, which help to improve the visibility of camouflaged objects. Lastly, we predict the binary masks to indicate the camouflaged objects at each layer by adding the supervision signals at multiple layers.

In the following subsections, we will elaborate on the texture-aware refinement module (Section 3.1) in detail, present the training and testing strategies in Section 3.2, and visualize the learned texture-aware features in Section 3.3.

3.1. Texture-Aware Refinement Module

As shown in Figure 1, the camouflaged objects have similar texture with the surroundings. However, there still exists subtle texture difference between the camouflaged objects and the background. Hence, we present a texture-aware refinement module to extract texture information and amplify the texture difference of camouflaged objects and background, thus improving the performance for camouflaged object detection.

3.1.1 Architecture

Figure 3 shows the architecture of the proposed texture-aware refinement module. First, it takes a feature map with the resolution $C \times H \times W$ as the input, and then use multiple convolution operations with $1 \times 1$ kernel to obtain multiple kinds of feature maps and each with the size of $C_1 \times H \times W$. Note that $C_1$ is smaller than $C$ for the computational efficiency and these feature maps are used to learn multiple aspects of textures in the following operations. Next, we compute the co-variance matrix among the feature channels at each position to capture the correlations between different responses on the convolutional features. The co-variance matrix among features measures the co-occurrence of features, describes the combination of features, and is used to represent texture information.

As shown in Figure 3(b), for each pixel $f^k_m$ on the feature map $f^k \in C_1 \times H \times W$, we compute its covariance matrix as the inner product between $f^k_m$ and $f^k_m^T$. Since the covariance matrix $(C_1 \times C_1)$ has the property of diagonal symmetry, we just adopt the upper triangle of this matrix to represent the texture feature, and reshape the result into a feature vector. We perform the same operations for each pixel on the feature map and concatenate the results to obtain $g^k \in C_1 \times (C_1 + 1) \times H \times W$, which contain the texture information. Then, we fuse all the covariance matrices computed from different feature maps by a $3 \times 3$ convolution. After that, we adopt two sets of $3 \times 3$ convolution from the texture features to learn two parameter maps.
Figure 4: The schematic illustration of how boundary-consistency loss works. Affinity loss is computed within small patches across boundary, which is used to enhance detail information.

\[ \gamma \in C' \times H \times W \] and \[ \beta \in C' \times H \times W \] (\( C' \) denotes the channel number) which are used to amplify the texture difference of camouflaged objects and their surroundings by adjusting the texture of input features \( f_{\text{in}} \) \cite{13, 24}. Finally, we obtain the output feature \( f_{\text{out}} \) by:

\[
f_{\text{out}} = \text{conv}(\gamma f'_{\text{in}} - \mu(f'_{\text{in}}) + \beta) + f_{\text{in}},
\]

where \( f'_{\text{in}} \) is obtained by applying a \( 3 \times 3 \) convolution on \( f_{\text{in}} \). \( \mu(f'_{\text{in}}) \) and \( \sigma(f'_{\text{in}}) \) are the mean and variance of \( f'_{\text{in}} \), which are used to normalize the feature map. Finally, we add the original feature map with the feature maps refined by the above operations as the output of our texture-aware refinement module.

### 3.1.2 Loss Function

**Affinity loss.** To make the parameter maps \( \gamma \) and \( \beta \) capture the texture difference between the camouflaged objects and the background, we adopt the affinity loss \cite{39} on \( \gamma \) and \( \beta \) to explicitly amplify the difference between their texture features. As shown in Figure 3 (c), we first use the pooling operations to downsample the parameter map and then compute the affinity matrix \( A_{h_{m,n}} \) at the position \( m,n \):

\[
A_{h_{m,n}} = \frac{h_m^T h_n}{\| h_m \|_2 \| h_n \|_2},
\]

where \( h_m \) and \( h_n \) are the parameter vectors of the downsampled map. \( A^h \) is the result matrix that captures the pairwise texture similarity. Next, we calculate the ground truth affinity matrix \( A^{gt} \) by:

\[
A^{gt}_{m,n} = 2 \times 1_{(C_m=C_n)} - 1,
\]

where \( 1 \) is a indicator, which is equal to one when the labels \( (C_m, C_n) \) of positions \( m \) and \( n \) are the same, otherwise it is equal to zero.

In natural images, camouflaged objects usually occupy small regions, and we formulate the affinity loss as follow to solve the class imbalance problem:

\[
L_{aff} = \sum_m \sum_n w_m w_n d(A_{h_{m,n}}, A^{gt}_{m,n}),
\]

where \( w_m = 1 - \frac{N_{C_m}}{H'W'} \) and \( w_n = 1 - \frac{N_{C_n}}{H'W'} \); \( N_{C_m} \) and \( N_{C_n} \) are the numbers of pixels that have the same class label with the pixel \( m \) and \( n \); \( H' \) and \( W' \) are the height and width of the parameter map. From this loss function, we can see that the parameter map will learn to maximize the texture difference.

**Boundary-Consistency Loss.** The convolutional features contain highly semantic features but tend to produce blurry boundaries between the camouflaged objects and the background due to the small resolutions of the parameter maps. To solve this issue, we present a boundary-consistency loss \( L_{edge} \) to improve the boundary quality by revisiting the prediction results across boundary regions:

\[
L_{edge} = \sum_{\forall i,b_i \in B} \sum_{m,n \in b_i} d(A_{h_{m,n}}, A^{gt}_{m,n}),
\]

Figure 5: Visual comparison results of our method with and without boundary-consistency loss.
where $b_i$ is the $i$-th image patch in $B$, which contains all the image patches that across boundary, as shown in Figure 4. These image patches are selected when they contain the pixels that belong to different categories. Note that unlike affinity loss, the $L_{edge}$ is performed on parameter maps without downsampling operations and the parameter maps with higher resolutions help to provide more detailed information for boundary prediction. The additional memory consumption during training is reasonable since we only consider affinity relationship within small patches and no extra computational time is introduced in testing process. The comparison results in Figure 5 show that our method with the boundary-consistency loss better preserves the detailed structures of camouflaged objects.

3.2. Training and Testing Strategies

The overall loss function was defined as

$$L = \lambda_0 L_{seg} + \lambda_1 L_{aff} + \lambda_2 L_{edge},$$

(6)

where $L_{seg}$ was the binary cross entropy loss used for camouflaged segmentation. We empirically set $\lambda_0 = \lambda_1 = 1$ and $\lambda_2 = 10$ to balance the numerical magnitude. We implemented our network in Pytorch and used ResNeXt50 [38] as the backbone network, which was pre-trained on ImageNet [30]. We used stochastic gradient descent (SGD) with the poly learning strategy [22] to optimize the network by setting the initial learning rate as 0.001 and decay power as 0.9. The training process was stopped after 30 epochs. It took about two hours to train the network on a 1080Ti GPU with the batch size of 16. The input images were resized as $384 \times 384$. In testing, we adopted the prediction mask with the highest resolution as the final result. We used 0.03s to process an image with the size of $384 \times 384$.

3.3. Visualization

To visualize the learned texture-aware features, we adopt a decoder [18] to reconstruct the images using the feature maps before and after refined by texture-aware refinement module (TARM); see Section 3.1. Note that during the visualization process, the weights in our TANet are fixed and we only trained the newly added decoder to reconstruct the original images. Figures 1&6 show the visualization results, where after adopting our TARM module to refine features, the texture differences between camouflaged objects and background are clearly amplified, which helps to improve the overall performance of camouflaged object detection.
Table 1: Comparison with state-of-the-art methods for camouflaged object detection on three benchmark datasets.

| Method               | Year | CHAMELEON | CAMO-Test | COD10K-Test |
|----------------------|------|-----------|-----------|-------------|
|                      |      | $S_\alpha$ | $E_\phi$ | $E_\phi$ $P_\beta$ | $M$ | $S_\alpha$ | $E_\phi$ | $E_\phi$ $P_\beta$ | $M$ | $S_\alpha$ | $E_\phi$ | $E_\phi$ $P_\beta$ | $M$ |
| FPN [20]             | 2017 | 0.794     | 0.783    | 0.590 | 0.075       | 0.684 | 0.677 | 0.483 | 0.131 | 0.697 | 0.691 | 0.411 | 0.075 |
| MaskRCNN [12]        | 2017 | 0.643     | 0.778    | 0.518 | 0.099       | 0.574 | 0.715 | 0.430 | 0.151 | 0.613 | 0.748 | 0.402 | 0.080 |
| PSPNet [40]          | 2017 | 0.773     | 0.758    | 0.555 | 0.085       | 0.663 | 0.659 | 0.455 | 0.139 | 0.678 | 0.680 | 0.377 | 0.080 |
| UNet++ [43]          | 2018 | 0.695     | 0.762    | 0.501 | 0.094       | 0.599 | 0.653 | 0.392 | 0.149 | 0.623 | 0.672 | 0.350 | 0.086 |
| PiCANet [21]         | 2018 | 0.769     | 0.749    | 0.536 | 0.085       | 0.609 | 0.584 | 0.356 | 0.156 | 0.649 | 0.643 | 0.322 | 0.090 |
| MSRCNN [14]          | 2019 | 0.637     | 0.686    | 0.443 | 0.091       | 0.617 | 0.669 | 0.454 | 0.133 | 0.641 | 0.706 | 0.419 | 0.073 |
| BASNet [26]          | 2019 | 0.687     | 0.721    | 0.474 | 0.118       | 0.618 | 0.661 | 0.413 | 0.159 | 0.634 | 0.678 | 0.365 | 0.105 |
| PFA-Net [42]         | 2019 | 0.679     | 0.648    | 0.378 | 0.144       | 0.659 | 0.622 | 0.391 | 0.172 | 0.636 | 0.618 | 0.286 | 0.128 |
| CPD [37]             | 2019 | 0.853     | 0.866    | 0.706 | 0.052       | 0.726 | 0.729 | 0.550 | 0.115 | 0.747 | 0.770 | 0.508 | 0.059 |
| HTC [2]              | 2019 | 0.517     | 0.489    | 0.204 | 0.129       | 0.476 | 0.442 | 0.174 | 0.172 | 0.548 | 0.520 | 0.221 | 0.088 |
| EGNet [41]           | 2019 | 0.848     | 0.870    | 0.702 | 0.050       | 0.732 | 0.768 | 0.583 | 0.104 | 0.737 | 0.779 | 0.509 | 0.056 |
| ANet-SRM [19]        | 2019 | 0.881     | 0.873    | 0.685 | 0.048       | 0.682 | 0.685 | 0.484 | 0.126 | 0.731 | 0.771 | 0.551 | 0.051 |
| SINet [7]            | 2020 | 0.869     | 0.891    | 0.740 | 0.044       | 0.785 | 0.771 | 0.606 | 0.100 | 0.771 | 0.806 | 0.551 | 0.051 |
| TANet v1             | -    | 0.881     | 0.907    | 0.773 | 0.039       | 0.778 | 0.813 | 0.659 | 0.089 | 0.794 | 0.838 | 0.613 | 0.043 |
| TANet (ours)         | -    | 0.888     | 0.911    | 0.786 | 0.036       | 0.793 | 0.834 | 0.690 | 0.083 | 0.803 | 0.848 | 0.629 | 0.041 |

Table 2: Ablation study results. Here, we compare the quantitative results of our full pipeline and baseline networks on three benchmark datasets.

| Method     | CHAMELEON | CAMO-Test | COD10K-Test |
|------------|-----------|-----------|-------------|
|            | $S_\alpha$ | $E_\phi$ | $P_\beta$ | $M$ | $S_\alpha$ | $E_\phi$ | $P_\beta$ | $M$ | $S_\alpha$ | $E_\phi$ | $P_\beta$ | $M$ |
| $M_1$ basic [20] | 0.856 | 0.866 | 0.710 | 0.050 | 0.763 | 0.783 | 0.621 | 0.097 | 0.772 | 0.797 | 0.557 | 0.049 |
| $M_2$ basic+RRB   | 0.862 | 0.885 | 0.733 | 0.046 | 0.774 | 0.807 | 0.649 | 0.091 | 0.782 | 0.813 | 0.583 | 0.046 |
| $M_3$ basic+RRB+TARM w/o BCL | 0.879 | 0.909 | 0.767 | 0.039 | 0.782 | 0.820 | 0.675 | 0.087 | 0.796 | 0.840 | 0.618 | 0.042 |
| Ours TANet       | 0.888 | 0.911 | 0.786 | 0.036 | 0.793 | 0.834 | 0.690 | 0.083 | 0.803 | 0.848 | 0.629 | 0.041 |

4. Experimental Results

4.1. Benchmark Protocols

We employ three widely-used COD benchmark datasets to test each COD method. COD10K [7] is the largest annotated dataset with 3,040 training images and 2,026 testing images. CAMO [19] includes 1,000 training images and 250 testing images while CHAMELEON [32] consists of 76 images. The same training dataset used by the most recent COD method [7] is employed to train our network for fair comparisons. The training set consists of 3,040 training images of COD10K (3,040 images), 600 training images of CPD1K, and 1,000 training images of CAMO. We test different COD methods on the testing set (COD10K-Test) of COD10K, the testing set (CAMO-Test) of CAMO, and the whole CHAMELEON for their results. We shall release our code, the trained network model, and the predicted COD maps of our method upon the publication of this work.

4.2. Evaluation Metrics

To conduct quantitative comparisons, we employ four common metrics to test different COD methods. They are Structure-measure [5] ($S_\alpha$), Enhanced-measure [6] ($E_\phi$), weight-F-measure [23] ($F_\beta$), and MAE [25]; see SINet [7] for the definitions of the four evaluation metrics. Overall, a better COD method has larger $S_\alpha$, $E_\phi$, and $F_\beta$ scores, but a smaller MAE score.

4.2.1 Comparison with SOTA Methods

We compare our method against 13 cutting-edge methods, including (1) FPN [20], (2) MaskRCNN [12], (3) PSPNet [40], (4) UNet++ [43], (5) PiCANet [21], (6) MSRCNN [14], (7) BASNet [26], (8) PFA-Net [42], (9) CPD [37], (10) HTC [2], (11) EGNet [41], (12) ANet-SRM [19], and (13) SINet [7]. Note that SINet has reported the quantitative results and released camouflaged object maps predicted by all compared COD methods. Hence, we use these public results for conducting fair comparisons.

Quantitative comparisons. Table 1 summarizes the quantitative results of different COD methods on three benchmark datasets. Apparently, SINet, as a dedicated COD method, has a superior performance on four metrics over other semantic segmentation methods and saliency detectors on three COD benchmark datasets. Compared to SINet, our method has larger $S_\alpha$, $E_\phi$, and $F_\beta$ scores, but a smaller $M$ score, which demonstrates that our method can more accurately identify camouflaged objects. Specifically, our method has a 3.98% improvement on the average $S_\alpha$, a
5.21% improvement on the average $E_\phi$, a 11.41% improvement on the average $F_\beta$, and a 18.26% improvement on the average $M$ on the three benchmark datasets. We also retrain our method by taking ResNet-50 as the backbone network, and report the results “TANet v1” in Table 1, where our method still achieves the best performance.

Visual comparisons. Figure 7 visually compares the COD maps produced by our network and compared methods. Apparently, all compared methods tend to include many non-camouflaged regions or neglect parts of camouflaged objects in their COD maps. On the contrary, our method can more accurately detect camouflaged objects, and our results (see Figure 7 (c)) are most consistent with the ground truths shown in Figure 7 (b). More visual comparisons can be found in the supplementary material.

4.3. Ablation Analysis

We conduct ablation study experiments to verify the effectiveness of TARMs, and the boundary-consistency loss; see Figure 3. The first baseline (“basic”) equals to remove all RRBs and all TARMs from our network. The second baseline (“basic+RRB”) is to add RRBs into “basic” to merge features at the adjacent CNN layers. It equals to remove all TARMs from our network. The third baseline (“basic+RRB+TARM w/o BCL”) is to add TARMs without boundary-consistency loss into the second baseline. Compared to the four baselines, our method is equal to add...
boundary-consistency loss of TARMs into the third baseline.

Figure 8 shows the visual comparison results of camouflaged object detection maps produced by our full pipeline and other baseline methods, demonstrating that our full pipeline better identifies camouflaged objects than others. Table 2 compares the results of our network and four baseline networks on three benchmark datasets, i.e., CHAMELEON, CAMO-Test, and COD10K-Test.

Effectiveness of RRBs. From the results of Table 2, “basic+RRB” has better metric results than “basic”, indicating RRBs blocks help our network detect camouflaged objects.

Effectiveness of the affinity loss of TARMs w/o BCL. As shown in Table 2, “basic+RRB+TARM w/o BCL” has larger $S_\alpha$, $E_\phi$, $F_w^{\beta}$ scores and a smaller $M$ score than “basic+RRB” on three benchmark datasets, demonstrating that it enables our network to better capture the texture difference between camouflaged objects and the background and thus benefits camouflaged object detection.

Effectiveness of the boundary-consistency loss of TARMs. The superior metric results of Our method over “basic+RRB+TARM w/o BCL” on the three benchmark datasets indicate that computing the boundary-consistency loss in TARMs can further improve the camouflaged object detection accuracy of our network by enhancing the consistency across boundaries.

5. Conclusion

This paper designs a novel deep network architecture for camouflaged object detection by learning deep texture-aware features. Our key idea is to amplify the texture difference between camouflaged objects and their surroundings, thus benefiting to identify camouflaged objects from the background. In our network, we design a texture-aware refinement module (TARM) that computes the covariance matrices of feature responses among feature channels to represent the texture structures and adopt the affinity loss to learn a set of parameter maps that perform a linear transformation of the convolutional features to separate the texture between camouflaged objects and the background. A
boundary-consistency loss is further proposed to learn the object details. In this way, we can obtain deep texture-aware features and formulate the TANet by embedding multiple TARMs in a deep neural network for the task. In the end, we evaluate our method on the benchmark dataset, compare it with various state-of-the-art methods, and show the superiority of our method both qualitatively and quantitatively. In the future, we will explore the potential of our network for generic image segmentation in more complex environments, especially for objects having similar color to the background.

References

[1] Nagappa U. Bhajantri and P. Nagabhushan. Camouflage defect identification: a novel approach. In *International Conference on Information Technology*, pages 145–148, 2006. 2

[2] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4974–4983, 2019. 1, 2, 6, 7

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 834–848, 2017. 1

[4] Hung-Kuo Chu, Wei-Hsin Hsu, Niloy J. Mitra, Daniel Cohen-Or, Tien-Tsin Wong, and Tong-Yee Lee. Camouflage images. *ACM Trans. Graph.*, 29(4):51–1, 2010. 1

[5] Deng-Ping Fan, Ming-Ming Cheng, Yun Liu, Tao Li, and Ali Borji. Structure-measure: A new way to evaluate foreground maps. In *IEEE International Conference on Computer Vision*, pages 4548–4557, 2017. 6

[6] Deng-Ping Fan, Cheng Gong, Yang Cao, Bo Ren, Ming-Ming Cheng, and Ali Borji. Enhanced-alignment measure for binary foreground map evaluation. *arXiv preprint arXiv:1805.10421*, 2018. 6

[7] Deng-Ping Fan, Ge-Peng Ji, Guolei Sun, Ming-Ming Cheng, Jianbing Shen, and Ling Shao. Camouflaged object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2777–2787, 2020. 1, 2, 6, 7

[8] Deng-Ping Fan, Ge-Peng Ji, Tao Zhou, Geng Chen, Huazhu Fu, Jianbing Shen, and Ling Shao. Planet: Parallel reverse attention network for polyp segmentation. In *International Conference on Medical Image Computing and Computer Assisted Intervention*, pages 263–273. Springer, 2020. 1

[9] Deng-Ping Fan, Tao Zhou, Ge-Peng Ji, Yi Zhou, Geng Chen, Huazhu Fu, Jianbing Shen, and Ling Shao. Inf-net: Automatic covid-19 lung infection segmentation from ct images. *IEEE Transactions on Medical Imaging*. 2020. 1

[10] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016. 3

[11] Shiming Ge, Xin Jin, Qiting Ye, Zhao Luo, and Qiang Li. Image editing by object-aware optimal boundary searching and mixed-domain composition. *Computational Visual Media*, 4(1):71–82, 2018. 1

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *IEEE International Conference on Computer Vision*, pages 2961–2969, 2017. 1, 2, 6

[13] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *IEEE International Conference on Computer Vision*, pages 1501–1510, 2017. 4

[14] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask scoring r-cnn. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6409–6418, 2019. 2, 6

[15] Iván Huerta, Daniel Rowe, Mikhail Mozherov, and Jordi González. Improving background subtraction based on a casualty of colour-motion segmentation problems. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 475–482, 2007. 2

[16] Levent Karacan, Erkut Erdem, and Aytuk Erdem. Structure-preserving image smoothing via region covariances. *ACM Transactions on Graphics*, 32(6):1–11, 2013. 3

[17] Ch Kavitha, B Prabhakara Rao, and A. Govardhan. An efficient content based image retrieval using color and texture of image sub-blocks. *International Journal of Engineering Science and Technology*, 3(2):1060–1068, 2011. 2

[18] Junho Kim, Minjae Kim, Hyeonwoo Kang, and Kwanghee Lee. U-gat-it: unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation. *arXiv preprint arXiv:1907.10830*, 2019. 5

[19] Trung-Nghia Le, Tam V Nguyen, Zhongliang Nie, Minh-Triet Tran, and Akihiro Sugimoto. Anabranch network for camouflaged object segmentation. *Computer Vision and Image Understanding*, pages 45–56, 2019. 1, 2, 6

[20] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017. 6

[21] Nian Liu, Junwei Han, and Ming-Hsuan Yang. Picanet: Learning pixel-wise contextual attention for saliency detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3089–3098, 2018. 2, 6

[22] Wei Liu, Andrew Rabinovich, and Alexander C. Berg. Parsenet: Looking wider to see better. *arXiv preprint arXiv:1506.04579*, 2015. 5

[23] Ram Margolin, Lihi Zelnik-Manor, and Ayellet Tal. How to evaluate foreground maps? In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2014. 6

[24] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2337–2346, 2019. 4

[25] Federico Perazzi, Philipp Krähenbühl, Yael Pritch, and Alexander Hornung. Saliency filters: Contrast based filtering
for salient region detection. In IEEE Conference on Computer Vision and Pattern Recognition, pages 733–740, 2012.

[26] Xuebin Qin, Zichen Zhang, Chenyang Huang, Chao Gao, Masood Dehghan, and Martin Jagersand. Basnet: Boundary-aware salient object detection. In IEEE Conference on Computer Vision and Pattern Recognition, pages 7479–7489, 2019.

[27] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In CVPR, pages 779–788, 2016.

[28] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In Conference on Neural Information Processing Systems, pages 91–99, 2015.

[29] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical Image Computing and Computer Assisted Intervention, pages 234–241, 2015.

[30] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. International Journal of Computer Vision, 115(3):211–252, 2015.

[31] P Siricharoen, S Aramvith, TH Chalidabhongse, and S Siddhichai. Robust outdoor human segmentation based on color-based statistical approach and edge combination. In International Conference on Green Circuits and Systems, pages 463–468, 2010.

[32] P. Skurowski, H. Abdulaemeer, J. Błaszczyk, T. Depta, A. Kornacki, and P. Kozieł. Animal camouflage analysis: Chameleon database. Unpublished Manuscript, 2018.

[33] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. Autoint: Automatic feature interaction learning via self-attentive neural networks. In ACM International Conference on Information and Knowledge Management, pages 1161–1170, 2019.

[34] Martin Stevens and Sami Merilaita. Animal camouflage: current issues and new perspectives. Philosophical Transactions of the Royal Society B: Biological Sciences, 364(1516):423–427, 2009.

[35] Ariel Tankus and Yehezkel Yeshurun. Convexity-based visual camouflage breaking. Computer Vision and Image Understanding, pages 208–237, 2001.

[36] Yu-Huan Wu, Shang-Hua Gao, Jie Mei, Jun Xu, Deng-Ping Fan, Chao-Wei Zhao, and Ming-Ming Cheng. Jcs: An explainable covid-19 diagnosis system by joint classification and segmentation. arXiv preprint arXiv:2004.07054, 2020.

[37] Zhe Wu, Li Su, and Qingming Huang. Cascaded partial decoder for fast and accurate salient object detection. In IEEE Conference on Computer Vision and Pattern Recognition, pages 3907–3916, 2019.

[38] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In IEEE Conference on Computer Vision and Pattern Recognition, pages 1492–1500, 2017.

[39] Changqian Yu, Jingbo Wang, Changxin Gao, Gang Yu, Chunhua Shen, and Nong Sang. Context prior for scene segmentation. In IEEE Conference on Computer Vision and Pattern Recognition, pages 12416–12425, 2020.

[40] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In IEEE Conference on Computer Vision and Pattern Recognition, pages 2881–2890, 2017.

[41] Jia-Xing Zhao, Jiang-Jiang Liu, Deng-Ping Fan, Yang Cao, Jufeng Yang, and Ping-Ming Cheng. Egnet: Edge guidance network for salient object detection. In IEEE International Conference on Computer Vision, pages 8779–8788, 2019.

[42] Ting Zhao and Xiangqian Wu. Pyramid feature attention network for saliency detection. In IEEE Conference on Computer Vision and Pattern Recognition, pages 3085–3094, 2019.

[43] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. In Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support, pages 3–11. Springer, 2018.