Locate, Refine and Restore: A Progressive Enhancement Network for Camouflaged Object Detection

Xiaofei Li\textsuperscript{1}, Jiaxin Yang\textsuperscript{1}, Shuohao Li\textsuperscript{1}, Jun Lei\textsuperscript{1}, Jun Zhang\textsuperscript{1}\textsuperscript{*} and Dong Chen\textsuperscript{2}

\textsuperscript{1}Laboratory for Big Data and Decision, National University of Defense Technology, China
\textsuperscript{2}Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, China

\{xf, lishuohao, leijun1987, zhangjun1975, dongchen\}@nudt.edu.cn, yangjiaxin0313@163.com

Abstract

Camouflaged Object Detection (COD) aims to segment objects that blend in with their surroundings. Most existing methods mainly tackle this issue by a single-stage framework, which tends to degrade performance in the face of small objects, low-contrast objects and objects with diverse appearances. In this paper, we propose a novel Progressive Enhancement Network (PENet) for COD by imitating the human visual detection system, which follows a three-stage detection process: locate objects, refine textures and restore boundary. Specifically, our PENet contains three key modules, i.e., the object location module (OLM), the group attention module (GAM) and the context feature restoration module (CFRM). The OLM is designed to position the object globally, the GAM is developed to refine both high-level semantic and low-level texture feature representation, and the CFRM is leveraged to effectively aggregate multi-level features for progressively restoring the clear boundary. Extensive results demonstrate that our PENet significantly outperforms 32 state-of-the-art methods on four widely used benchmark datasets.

1 Introduction

Camouflage is a widespread biological phenomenon in nature that helps certain organisms hide in the surroundings to protect themselves from predators [Cuthill \textit{et al.}, 2005]. In practice, camouflaged objects usually conceal themselves by imitating the appearance, colors, or patterns of the environment and the disruptive coloration [Price \textit{et al.}, 2019], making them difficult to be found. Based on this strategy, human beings began to study bionic disguise things according to their own ideas for the purpose of camouflage. For example, in military combat soldiers achieve stealth by wearing camouflage clothing made of special materials, hunters use disguised sounds in the forest to trap animals farmers build scarecrows in the fields to repel birds. Recently, camouflaged object detection (COD) has attracted an increasing research interest from the computer vision community, and a lot of potential applications have been developed in different fields, such as medical diagnosis (e.g., polyp segmentation, lung infection segmentation), agriculture (e.g., locust detection to prevent invasion), industry (surface defect detection), security and surveillance (e.g., search and rescue work), animal conservation (e.g., species discovery), and art (e.g., recreational art).

However, COD is an extremely challenging task due to the camouflaged objects making themselves “perfectly” assimilate into their surroundings by means of the materials, coloration or illumination. As shown in Fig.1, due to low contrast and the appearance of expressive diversity, the texture of the object is similar to the surroundings, it is very challenging to discover it. To address this problem, early traditional camouflaged object detectors attempt to extract discriminative hand-crafted low-level features rely on the manual visual feature, such as color, 3D convexity and appearance texture. In recent years, numerous algorithms for COD based on deep neural networks have been proposed, which can si-
multaneously extract low-level texture features and high-level semantic information. Despite good performance achievements, there is still a large place for improvement. First, the idea of most existing approaches is to use ASPP [Chen et al., 2017] modules in a single-stage manner to extract the context and then decode the segmentation results using some simple fusion methods. This method does not take into account the localization of the camouflaged object, so there is often inaccuracy in segmenting the object. Second, although there are some two-stage approaches, they usually consider only the predicted camouflaged objects from coarse to fine, without considering also the object positioning, texture enhancement and boundary restoration.

Previous biological studies [Hall et al., 2013] have shown that the human eye will first locate the general position of an object when viewing it, then focus on the main area of the object to find details, and then further fuse and refine the boundary until the object is completely detected from the background. Inspired by this human behavior and considering the shortcomings of these methods, we aim to address the COD issue: How to accurately locate the object and refine the textures to restore clear boundary under complex scenes?

To this end, we develop a novel bio-inspired framework, termed progressive enhancement network (PENet), which significantly improves the existing camouflaged object segmentation performance. Our PENet consists of three key modules, i.e., the object location module (OLM), the group attention module (GAM) and the context feature restoration module (CFRM), to accurately locate the camouflaged object, refine the textures and restore the boundary in a progressively enhanced manner, respectively. Specifically, the OLM consists of a global attention component and a local attention component to mimic the human detection process by first locating the target globally. The GAM is designed as a group attention module that contains channel attention and spatial attention to focus on detailed features at different scales. The CFRM uses attention-guided fusion of texture features between different layers to achieve the boundary-reduction.

In summary, the main contributions of this paper can be summarized as follows:

- We present a new bio-inspired framework called progressive enhancement network (PENet) for COD, which greatly improves the performance of COD in a progressively enhanced manner to locate objects, enhance texture features and restore boundary.
- We propose an object location module (OLM) to infer the initial position of the camouflaged objects, which can effectively extract global information and local features so that the location of the camouflaged object can be accurately determined. We also design a group attention module (GAM) to refine textures, and a context feature restoration module (CFRM) to restore clear boundary.
- Extensive experiments on four benchmark datasets demonstrate that our PENet achieves the state-of-the-art performance of COD. Qualitative and quantitative results demonstrate the effectiveness of our method.

## 2 Related Work

### 2.1 Camouflaged Object Detection

Different from salient object detection that aims to detect and segments the most compelling objects in the image, the purpose of camouflaged object detection is to find objects that closely resemble their surroundings. [Le et al., 2019] proposed an anabranched network, which leverages both classification and segmentation tasks. [Fan et al., 2020a] proposed a Search Identification Network to address this challenge by first roughly searching for camouflaged objects, and then segmenting the objects by a recognition module. [Sun et al., 2021] proposed C2F-Net for COD, which considers global contextual information to integrate multi-level features. [Jia et al., 2022] proposed an iterative refinement framework, coined SegMaR, which integrates Segment, Magnify and Revolve in a multi-stage detection fashion.

### 2.2 Object Location

Accurate object localization is important for computer vision tasks, and it often affects the performance of these tasks. In order to solve the co-localization problem, [Gokberk Cinbis et al., 2014] proposed a multiplicative multi-instance learning procedure to iteratively train the detector, which prevents training from prematurely locking onto erroneous object locations. [Bazzani et al., 2016] proposed a self-taught object localization method that localizes objects by identifying the regions causing the maximal activations. [Zhang et al., 2018a] proposed an Adversarial Complementary Learning approach for discovering entire objects of interest by two adversarial classifiers.

### 2.3 Multi-Scale Feature Refinement

As the CNNs become deeper, the detailed features may be diluted. To make more efficient use of texture features, one solution is to aggregate multi-scale information. [Chen et al., 2017] proposed a spatial pyramid pooling to robustly segment objects at multiple scales by an incoming convolutional feature layer with filters at multiple sampling rates and effective fields-of-view. [Pang et al., 2020] presented a multi-scale interactive network for salient object detection, which embeds a self-interaction module in each decoder unit in order to obtain more effective multi-scale features from the integrated features. [Zhu et al., 2021] proposed an interactive guidance framework to interactively refine multi-level features of camouflaged object detection and texture detection.

### 2.4 Context-Aware Feature Learning

The contextual information plays a crucial role in enhancing feature representation for many computer vision tasks. [Hu et al., 2018] proposed a network for shadow detection by analyzing spatial context in a direction-aware manner. [Chen et al., 2020] used some progressive context-aware Feature Interweaved Aggregation (FIA) modules to integrate low-level appearance features, high-level semantic features and global contextual features. [Mei et al., 2020] explored abundant contextual cues with a large-field contextual feature integration (LCFI) module for robust glass detection. [Dai et al., 2021] proposed a multi-scale channel attention module to fuse features given at different scales.
3 Proposed Method

3.1 Overall Architecture

The overall framework of our PENet is shown in Fig.2. Specifically, for an input RGB image $I$ with size $H \times W$, we adopt Res2Net-50 [Gao et al., 2019] as the backbone to extract its multi-level features from the input image, denoted as $f_i$ ($i = 1, 2, ..., 5$). Then, we use three object location modules (OLMs) to locate the potential camouflaged objects. Next, we feed these features with their initial position to three convolutional blocks and group attention modules (GAMs) to refine the details. Finally, we leverage three context feature restoration modules (CFRMs) to restore clear and complete objects.

3.2 Initial Object Location

In the locating stage, we design an object location module (OLM) to initially locate the potential location of the camouflaged object. As shown in Fig.3, it consists of a global block and a local block. The global block is implemented in a non-local way to capture long-range dependencies for enhancing the contextual semantic representation from a global perspective. In contrast, the local block is designed to extract the local information by using several convolutional layers. These two blocks explore potential object regions in a complementary way.

Specifically, we first leverage the receptive fields block (RFB) [Liu et al., 2018] structure to enlarge the receptive field, obtain four feature maps $B, C, D$ and $E$ through four $1 \times 1$ convolutional layers, where $\{B, C, D, E\} \in \mathbb{R}^{C \times H \times W}$. Then we reshape them separately for $\mathbb{R}^{C \times N}$. Next, we multiply the transpose of $B$ by the $C$ matrix, and perform a softmax layer to calculate the global spatial attention maps $g_{sa_{ij}} \in \mathbb{R}^{N \times N}$. Consequently, we use matrix multiplication operation to get the semantic-enhanced global feature $f_g \in \mathbb{R}^{C \times H \times W}$. The process can be depicted as follows:

$$
g_{sa_{ij}} = \frac{\exp(B_{i} \cdot C_{j})}{\sum_{i=1}^{N} \exp(B_{i} \cdot C_{j})} \tag{1}$$

$$
f_g^i = \eta \sum_{j=1}^{N} (g_{sa_{ij}} \cdot D_{j}) + f_{i} \tag{2}$$

where $g_{sa_{ij}}$ denotes the $j^{th}$ position's impact on the $i^{th}$ position. $\eta$ is initialized as 0 and gradually learns more weight.

In the local block, we adopt a set of $1 \times 1$ convolutional layers, BatchNorm2d and ReLU to obtain the local features $f_l \in \mathbb{R}^{C \times H \times W}$. Then, we perform the element-wise addition operation on global feature $f_g$ and local features $f_l$ to obtain the aggregated features $f_a$. This process can be described as follows:

$$
f_l = ReLU(Conv_{1 \times 1}(ReLU(BN(Conv_{1 \times 1}(E)))))) \tag{3}$$
\[ f_a = ReLU(Conv_{3x3}(f_g + f_i)) \]  

(4)

where \(Conv_{1x1}\) indicates 1 \(\times\) 1 convolutional layers. \(BN\) means the BatchNorm2d and \(Conv_{3x3}\) indicates 3 \(\times\) 3 convolutional layers.

### 3.3 Texture Detail Refinement

Since the obtained initial prediction in the localization stage is coarse and contains irrelevant noise, we need to further refine the texture features. In the refinement stage, we develop a group attention pyramid network to refine the details for more effective feature representation. It contains three convolutional blocks and three group attention modules (GAM).

Specifically, as shown in Fig.2, we first use a holistic attention (HA) module [Wu et al., 2019a] to merge the coarse prediction and feature maps to highlight the whole object region. Then, we use three convolutional blocks (i.e., \(f'_3, f'_i\) and \(f'_g\)) to get the the bottom-up features, and then construct a pyramid network from them and the features of \(f_2\). In each level of the top-down pathway of the pyramid network, we design a group attention module (GAM) to mining multi-scale features. As shown in Fig 4, we first use the RFB structure to enlarge the receptive field, then the global features \(f_g \in \mathbb{R}^{C \times H \times W}\) and the attention features \(f_a \in \mathbb{R}^{C \times H \times W}\) are splitted into \(M\) fixed groups along the channel dimension. After that, the splitted features \(\{f_g^m\}_{m=1}^M \in \mathbb{R}^{C/M \times H \times W}\) and \(\{f_a^m\}_{m=1}^M \in \mathbb{R}^{C/M \times H \times W}\) are obtained. Consequently, we can get the regrouped features \(f \in \mathbb{R}^{C \times H \times W}\) by a concatenation operation:

\[ f = Concat(f_g^1, f_a^1, ..., f_g^m, f_a^m, ..., f_g^M, f_a^M) \]  

(5)

where \(Concat()\) denotes the concatenation operation.

To enhance the texture features, we apply spatial attention and channel attention to the regrouped features separately. For the spatial attention, we use a deconvolution layer with one output channel and a 3 \(\times\) 3 kernel size to extract spatial information, and then we use a sigmoid function to normalize it to the range of (0,1). The process can be expressed as follows:

\[ SA = \sigma(Deconv(f)) \]  

(6)

where \(\sigma\) represents the sigmoid function. \(Deconv\) refers to the 3 \(\times\) 3 deconvolution layer. \(SA\) represents the spatial attention feature.

In the channel attention, we use a global average pooling (GAP) and two 1 \(\times\) 1 convolutional layers to reduce dimension. Then a ReLU and a sigmoid function is applied to gained the channel attention:

\[ CA = \sigma(Conv_{1x1}(ReLU(Conv_{1x1}(GAP(f))))) \]  

(7)

where \(GAP\) represents the global average pooling. \(Conv_{1x1}\) indicates 1 \(\times\) 1 convolutional layers. \(\sigma\) represents the sigmoid function. \(CA\) is the channel attention feature. Finally, we perform the addition operation on the spatial attention feature and the channel attention feature, then we can get the weight feature \(f_t\), which serves as the input attention map for the next GAM:

\[ f_t = Up(f \cdot (SA + CA)) \]  

(8)

where \(Up\) represents the up-sample operation.

### 3.4 Context Boundary Restoration

As we all know, the context features contain rich semantic information, the fusion of context features is critical for restoring complete camouflaged objects. Therefore, we propose a context feature restoration module (CFRM) to aggregate the rich context features for improving the performance of COD.

Specifically, as shown in Fig.5, the CFRM consists of three branches (i.e., low-level features, global features and high-level features). First, we add a 3 \(\times\) 3 convolutional layer to each of them. Then, the global features are used to guide the high-level features and the low-level features respectively through a concatenation operation. Next, global average pooling is used for the low-level features and the high-level features to get the pooled features \(f'_{low}\) and \(f'_{high}\). Subsequently, the concatenation operation and a convolutional layer are adopted to gain the next global features \(f'_g\):

\[ f'^{GAP}_{low} = GAP(Concat(Conv_{3x3}(f_{low}), Up(Conv_{3x3}(f_g))) \]  

(9)

\[ f'^{GAP}_{high} = GAP(Concat(Conv_{3x3}(f_{high}), Conv_{3x3}(f_g))) \]  

(10)

\[ f'_g = Conv_{3x3}(Concat(f'^{GAP}_{low}, f'^{GAP}_{high})) \]  

(11)

Simultaneously, in the low-level features and the high-level features, we leverage the element-wise addition operation to augment the missing context features. Then, another set of 3 \(\times\) 3 convolutional layers are added to each of the two branches for obtaining the enhanced features \(f'_{low}\) and \(f'_{high}\).
Finally, a concatenation operation and a $3 \times 3$ convolutional layer are applied to fuse these context features. This process can be formulated as:

$$f'_{\text{low}} = \text{Conv}_{3\times3}(f_{\text{low}}^{\text{GAP}} + \text{Conv}_{3\times3}(f_{\text{low}}))$$ (12)

$$f'_{\text{high}} = \text{Conv}_{3\times3}(f_{\text{high}}^{\text{GAP}} + \text{Conv}_{3\times3}(f_{\text{high}}))$$ (13)

$$f' = \text{Conv}_{3\times3}($$

In addition, high-level segmentation features contain rich semantic information and low-level texture features have a lot of boundary details. In order to obtain clear and whole object boundary, we introduce edge supervision. In particular, we fuse the high-level features with the low-level features and supervise them using edge maps. Specifically, we perform a residual channel attention block (RCAB) [Zhang et al., 2018b] in the high-level features (i.e., $f'_{\text{high}}$) as the guiding information, and perform an element-wise multiplication operation with the low-level features (i.e., $f'_{\text{low}}$), followed by two $3 \times 3$ convolutional layers, a normalization layers and a ReLU layers to obtain the fused edge map $f_{\text{edge}}$. This process can be formulated as:

$$f_{\text{edge}} = \text{CBR}(\text{CBR}(\text{RCAB}(f'_{\text{low}}) + f'_{\text{high}}))$$ (15)

where $\text{CBR}$ denotes a $3 \times 3$ convolutional layer, BatchNorm2d and ReLU.

### 3.5 Loss Function

The binary cross entropy (BCE) loss and the intersection-over-union (IoU) loss are widely used in various image segmentation tasks. However, these losses treat each pixel equally and cannot distinguish the differences between pixels. In this paper, we use the weighted binary cross-entropy loss ($\text{wBCE}$) and IoU loss ($\text{wIoU}$), which can calculate the difference between the center pixel and its surrounding environment, and pay more attention on hard pixels to enhance the model generalization. In particular, we use the Consistency-Enhanced Loss (CEL) as an assistant, which consider the inter-pixel relationships and highlight the entire camouflaged region. To sum up, the loss function of our model is defined as follows:

$$L = L_{\text{BCE}}^W + L_{\text{IoU}}^W + \lambda L_{\text{CEL}}$$ (16)

where $\lambda$ is a hyperparameter, it is set to 1 for balancing the contributions of the three losses.

In addition, we use the dice loss ($L_{\text{dice}}$) [Xie et al., 2020] to address the strong imbalance between positive and negative samples. At last, the total loss can be formulated as:

$$L_{\text{total}} = L(P_{\text{loc}}, G) + L(P_{\text{res}}, G) + L_{\text{dice}}(P_{\text{edge}}, E)$$ (17)

where $P_{\text{loc}}$ is the predicted location map, $P_{\text{res}}$ is the predicted restoration map, $G$ is the ground-truth map and $E$ is the edge map.

### 4 Experiments

#### 4.1 Datasets

We employ four widely-used COD benchmark datasets to evaluate our method, including: CHAMELEON [Skurowski et al., 2018], CAMO [Le et al., 2019], COD10K [Fan et al., 2020a] and NC4K [Lv et al., 2021]. Following the previous work [Fan et al., 2021a], we use the combination of the train sets from CAMO and COD10K (4,040 images) as the training set, and evaluate on the rest ones.

#### 4.2 Evaluation Metrics

Conventionally, we adopt four popular and standard metrics to evaluate the performance of our method: structure-measure ($S_o$) [Fan et al., 2017], E-measure ($E_o$) [Fan et al., 2021b], weighted F-measure ($F_o$) [Margolin et al., 2014] and mean absolute error ($M$) [Perazzi et al., 2012].

#### 4.3 Implementation Details

We implement our model with PyTorch and adopt ResNet-50 [Gao et al., 2019] pre-trained on ImageNet as our backbone. We resize all the input images and ground-truths to $352 \times 352$ for both training and testing. During training, we set the batch size to 36 and use Adam algorithm to optimize the network parameters with a learning rate of $1e^{-4}$, and decay it by 0.1 every 30 epochs. We apply random horizontal flipping, random cropping and random rotating to augment the training data. The training and testing processes are conducted on an NVIDIA Tesla V100 GPU (with 32GB memory) and Intel(R) Xeon(R) Gold 6240, 2.60 GHz CPU device.

#### 4.4 Comparison with State-of-the-Art Methods

We compare our PENet with 32 state-of-the-art COD methods, including CPD [Wu et al., 2019a], PoolNet [Liu et al., 2019], EGNNet [Zhao et al., 2019], SCRNet [Wu et al., 2019b], F3Net [Wei et al., 2020], CSNet [Gao et al., 2020], SSAL [Zhang et al., 2020b], UCNet [Zhang et al., 2020a], MINet [Pang et al., 2020], ITSD [Zhou et al., 2020], PraNet [Fan et al., 2020b], VST [Liu et al., 2021], RCSB [Ke and Tsubono, 2022], SINet [Fan et al., 2020a], R-MOL [Zhai et al., 2021], TINet [Zhu et al., 2021], UGTR [Yang et al., 2021], PFNet [Mei et al., 2021], SLSR [Lv et al., 2021], UJSC [Li et al., 2021], D2CNet [Wang et al., 2021], SINet-V2 [Fan et al., 2021a], C2F-Net [Sun et al., 2021], BgNet [Chen et al., 2022], SegMaR-1 [Jia et al., 2022], BSA-Net [Zhu et al., 2022], BGNNet [Sun et al., 2022], ER-RNet [Ji et al., 2022], CubeNet [Zhu et al., 2022], ZoomNet [Pang et al., 2022], C2F-Net-V2 [Chen et al., 2022a] and FBNNet [Lin et al., 2023]. For a fair comparison, we obtain the
results of these methods from the authors, ZoomNet [Pang et al., 2022], or obtained by running the publicly available codes with well-trained models.

**Quantitative Evaluation**

Table 1 reports the detailed comparison results of PENet against other 32 state-of-the-art methods on four benchmark datasets. It can be seen that our proposed method consistently and significantly surpasses all the previous methods with a large margin on all four standard metrics. For example, compared with the simultaneously localize, segment and rank network SLSR [Lv et al., 2021], our PENet improves the $F_{\text{\tiny{IOU}}}^+$ by 7.6%, 4.0%, 7.3% and 5.3% on the four dataset, respectively. It is worth mentioning that the results of our method is competitive with the C2F-Net [Sun et al., 2021] that uses the strategy of context-aware cross-level fusion, which boosts the $S_o$ by 3.1%, 2.0% and 2.8% on the CAMO, CHAMELEON and COD10K dataset, respectively. In addition, our method surpasses UJSC [Li et al., 2021] which even introduces extra SOD data for training.

**Qualitative Evaluation**

We provide several typical examples in Figure 6, which visually shows the qualitative results of our PENet with other cutting-edge methods. It can be seen that most compared methods tend to detect some irrelevant surroundings or neglect some regions of camouflaged objects (e.g., the 2-nd and 3-rd rows). By contrast, the detection results of our PENet are more accurate and closest to the ground-truth annotations, in particular on the 5-th row). These results visually demonstrate the superior performance of our method.

**4.5 Ablation Study**

In order to validate the effectiveness of each key module, we design a series of controlled experiments and present the results in Table 2.

**Effectiveness of OLM.** In Table 2, compared with the basic model (a), we can see that (b) outperforms (a) by 6.0%, 6.5%, 9.2% and 7.3% in terms of $F_{\text{\tiny{IOU}}}^+$ on the four dataset, respectively. This demonstrates that the OLM is effective for object localization and plays a key role in achieving high performance for COD tasks.

**Effectiveness of GAM.** From (b) and (c) in Table 2, we can see that our proposed GAM further improves the metric.
Table 2: Ablation analyses on the four datasets. Ver. = Version. “B” denotes removing all OLMs, GAMs and CFRMs, which just use the pre-trained models by a simple concatenation. “w/o R” means without RCAB. “w/o E” means without edge supervision.

<table>
<thead>
<tr>
<th>Ver.</th>
<th>Method</th>
<th>CAMO-Test</th>
<th>CHAMELEON</th>
<th>COD10K-Test</th>
<th>NC4K</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>S_P</td>
<td>F_M</td>
<td>M_L</td>
<td>E_S</td>
<td>S_P</td>
</tr>
<tr>
<td>(a)</td>
<td>B</td>
<td>.790</td>
<td>.687</td>
<td>.083</td>
<td>.848</td>
</tr>
<tr>
<td>(b)</td>
<td>B+OLM</td>
<td>.818</td>
<td>.747</td>
<td>.070</td>
<td>.880</td>
</tr>
<tr>
<td>(c)</td>
<td>B+OLM+GAM</td>
<td>.820</td>
<td>.754</td>
<td>.069</td>
<td>.879</td>
</tr>
<tr>
<td>(d)</td>
<td>B+OLM+GAM+CFRM w/o R</td>
<td>.823</td>
<td>.766</td>
<td>.066</td>
<td>.886</td>
</tr>
<tr>
<td>(e)</td>
<td>B+OLM+GAM+CFRM w/o E</td>
<td>.822</td>
<td>.762</td>
<td>.067</td>
<td>.883</td>
</tr>
<tr>
<td>(f)</td>
<td>B+OLM+GAM+CFRM (Ours)</td>
<td>.828</td>
<td>.771</td>
<td>.063</td>
<td>.890</td>
</tr>
</tbody>
</table>

5 Conclusion

In this paper, we are committed to addressing the challenges of accurate COD. We develop a “locate-refine-restore” strategy to gradually restore clear and complete camouflaged objects, which helps to improve the understanding and judgment of camouflaged objects. Specifically, we first propose an object location module (OLM) to initially locate the camouflaged region. Then, we design a group attention module (GAM) to enhance the texture feature representation. Finally, we introduce a context feature restoration module (CFRM) to restore the clear boundary by fusing the context features. We conduct extensive experiments on four benchmark datasets using four widely used evaluation metrics, which illustrates that our method can achieve state-of-the-art performance.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (NSFC) [grant number 62101571], Natural Science Foundation of Hunan [grant number 2021JJ40685], and Hunan Provincial Innovation Foundation for Postgraduate [grant number QL20220018].
References


[Lv et al., 2021] Yunqiu Lv, Jing Zhang, Yuchao Dai, Aixuan Li, Bowen Liu, Nick Barnes, and Deng-Ping Fan. Si-


