





Article

Camouflage Object Segmentation Using an Optimized Deep-Learning Approach

Muhammad Kamran ¹, Saeed Ur Rehman ^{1,*} , Talha Meraj ¹ , Khalid A. Alnowibet ²  and Hafiz Tayyab Rauf ^{3,*} 

¹ Department of Computer Science, COMSATS University Islamabad, Wah Campus, Rawalpindi 47040, Pakistan
² Statistics and Operations Research Department, College of Science, King Saud University, Riyadh 11451, Saudi Arabia
³ Centre for Smart Systems, AI and Cybersecurity, Staffordshire University, Stoke-on-Trent ST4 2DE, UK
* Correspondence: srehan@ciitwah.edu.pk (S.U.R.); hafiztayyabrauf093@gmail.com (H.T.R.)

Abstract: Camouflage objects hide information physically based on the feature matching of the texture or boundary line within the background. Texture matching and similarities between the camouflage objects and surrounding maps make differentiation difficult with generic and salient objects, thus making camouflage object detection (COD) more challenging. The existing techniques perform well. However, the challenging nature of camouflage objects demands more accuracy in detection and segmentation. To overcome this challenge, an optimized modular framework for COD tasks, named Optimize Global Refinement (OGR), is presented. This framework comprises a parallelism approach in feature extraction for the enhancement of learned parameters and globally refined feature maps for the abstraction of all intuitive feature sets at each extraction block's outcome. Additionally, an optimized local best feature node-based rule is proposed to reduce the complexity of the proposed model. In light of the baseline experiments, OGR was applied and evaluated on a benchmark. The publicly available datasets were outperformed by achieving state-of-the-art structural similarity of 94%, 93%, and 96% for the Kvasir-SEG, COD10K, and Camouflaged Object (CAMO) datasets, respectively. The OGR is generalized and can be integrated into real-time applications for future development.

Keywords: semantic segmentation; global refinement; camouflage objects; graph fusion; edge enhancement; boundary guidance; graph convolutional network; vision transformer

MSC: 68T07

Citation: Kamran, M.; Rehman, S.U.; Meraj, T.; Alnowibet, K.A.; Rauf, H.T. Camouflage Object Segmentation Using an Optimized Deep-Learning Approach. *Mathematics* **2022**, *10*, 4219. <https://doi.org/10.3390/math10224219>

Academic Editors: Alvaro Figueira and Francesco Renna

Received: 9 October 2022

Accepted: 9 November 2022

Published: 11 November 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In the exploration of different kinds of objects by visual observation and the process of differentiating them, most objects are efficiently observed and found easily and are thus classified as generic [1], while some hide their features [2]. Biologists declared that those objects hide their features, are difficult to recognize by a human visual sensor, and become “camouflaged”. These camouflage objects, as shown in Figure 1, use the natural phenomenon of hiding features in objects using the same combination pattern of color, structure, and material to its surroundings, making its visibility and differentiation difficult—for example, a polar bear on ice, a red bee on the red carpet, an owl on a tree branch, etc. This hiding technique deceives the observer from clearly defining and exploring these objects. Therefore, it requires more boundary information about the objects to be recognized and similarities between them and their background. Some animals also take advantage of natural camouflage [3] and change their body color and structure to match their surroundings to prevent recognition by their predator. A high level of object structure and boundary information is required to find and explore these camouflage objects using computing devices as a COD [4] task. Thus, COD is a more challenging task in its data samples and techniques than salient object detection [5]. Although different applied computer vision tasks, e.g., semantic analysis, data processing, face object detection and recognition [6], and

high-level understanding of image [7] and video segmentation [8], etc., helped with deep learning interpolation to solve the COD challenges.

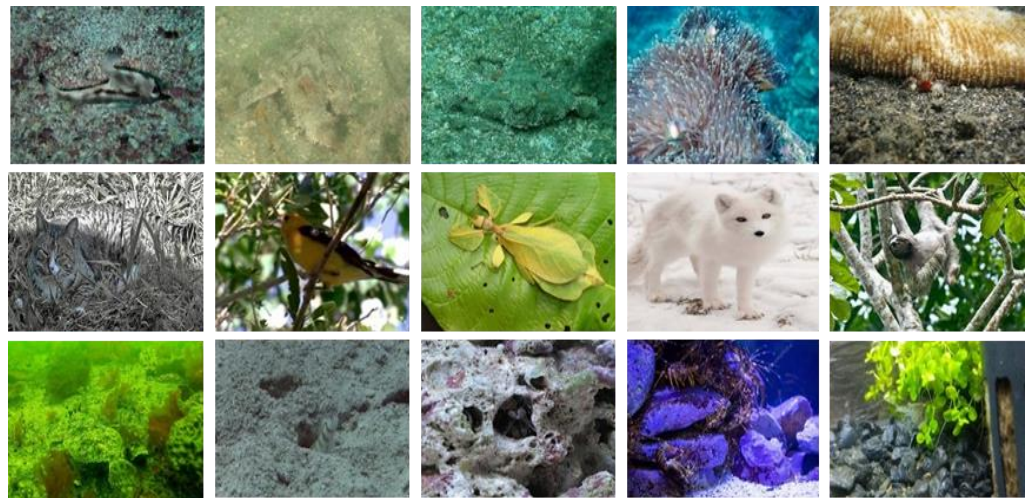


Figure 1. Example of different camouflage objects [4].

There is still a particular area requiring further improvement regarding different groups of camouflage objects and techniques. Sometimes camouflage objects consist of multiple instances that do not have a completely singular and relevant boundary. Thus, these independent shapes are difficult to understand as a single object, e.g., the chameleon backed by a leaf as shown in Figure 2, etc.

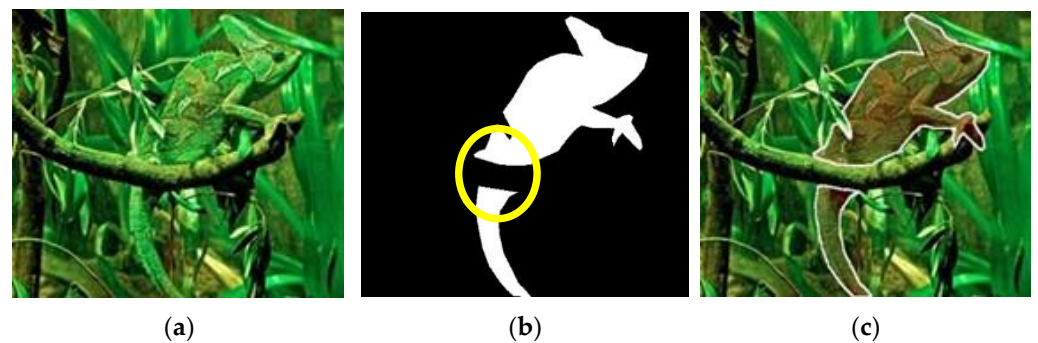


Figure 2. Occlusion in camouflage objects: (a) original image (The circle shows that the particular object sitting on the branch has occlusion); (b) ground truth label; (c) overlaid.

These multiple instances may differ from the other, and the complete object visualization using relevant and singular body information becomes difficult. Unlike salient object detection [5], in camouflage, the object boundary is unclear, and different separate fragments make an object more challenging. These fragments of objects need to be addressed due to the separate regions of camouflage objects in an image.

To overcome COD dataset challenges [4], many computer vision scientists and biologists have worked extensively on the goal of high-resolution image dataset collection and preparing annotations for each sample regarding each class of the camouflaged object at the object level, as shown above in Figure 2.

In [4], a significant COD10K dataset [4] was presented along with the deep learning model for a COD task. Afterward, the scientists found the flexibility to explore this dataset with multiple machine learning and deep learning methods to compare it with the methods using the generic object and the salient object dataset. In the same year, the parallel attention model [9] was proposed to speed up its performance. Recently, a modified instance-level dataset for camouflage objects was presented, and multiple prescribed methods of instance segmentation were used as frameworks. Therefore, the need for experiments on camouflage

objects with high-resolution image data in such a challenging task requires the design and development of a versatile, comprehensive, and optimized approach.

As shown in Figure 3, the proposed framework contains an optimized solution using a deep learning technique with a complete training flow as described in Algorithm 1, and the optimization rule is explored in Algorithm 2. This framework follows the modular approach toward a COD task and is adopted on multiple dataset samples to ensure it outperforms previous approaches. The following contributions comprised the model's components focusing on the optimal solution. Thus, the proposed framework provides the solution for COD using three modules or sub-frameworks, i.e., global refinement, optimizer, and parallel convolution.

Algorithm 1. Optimize, Parallel Refinement.

Input:

1. Sample collection in a combined list of samples I and its target $GT [I]$
2. Hypothesis M with parameters w initialized with random or zero initializer.
3. Optimized process O states the member of the hypothesis function. $O \in M (f(x))$

Output:

1. for $U \leftarrow i$ to E do
 2. for $L \leftarrow l$ to length I_{train} do
 3. $y_{pred}^{train} \leftarrow M(I_{train}, G_{i-1}), O^p (M_i^p) \Rightarrow i>0 \mid ! G[\phi]$
 4. $e_i^{train} = y_i^{train} - x_i^{true}$
 5. end for
 6. $e_i^{train} / L \leq e_{i-1}^{train} \wedge G_{i-1} \mid G_b$
 7. e_i^{train} minima.
 8. Monitor var
 9. $y_{pred}^{train} \leftrightarrow GT [I], Met [f, e, s]$
 10. e_i^{train} Backpropagate
 11. G_i . Update
 12. end for
 13. In validation do
 14. for $K \leftarrow k$ to length I_{test} do
 15. $y_{pred}^{test} \leftarrow h(I_{test}, G_i)$
 16. $e_k^{test} = y_k^{test} - x_k^{true}$
 17. $\frac{e_k^{test}}{L} \leq e_{i-1}^{test}$
 18. end for
 19. end for
-

Algorithm 2. Optimization Framework

Input:

1. Initialize parameters. $W, N, epochs, etc.$

Output:

1. for $i \leftarrow 1$ to E do
 2. for $M \leftarrow 1$ to length I_{train} do
 3. Convert the system into binary distribution.
 4. Preprocess the algorithm's parameters in all dimensions.
 5. Calculate fitness using the upper and lower boundary of the sample space.
 6. Calculate fitness by reducing error and the performance measure.
 7. Update all the dimensions of all the sample space
 8. end for
 9. end for
-

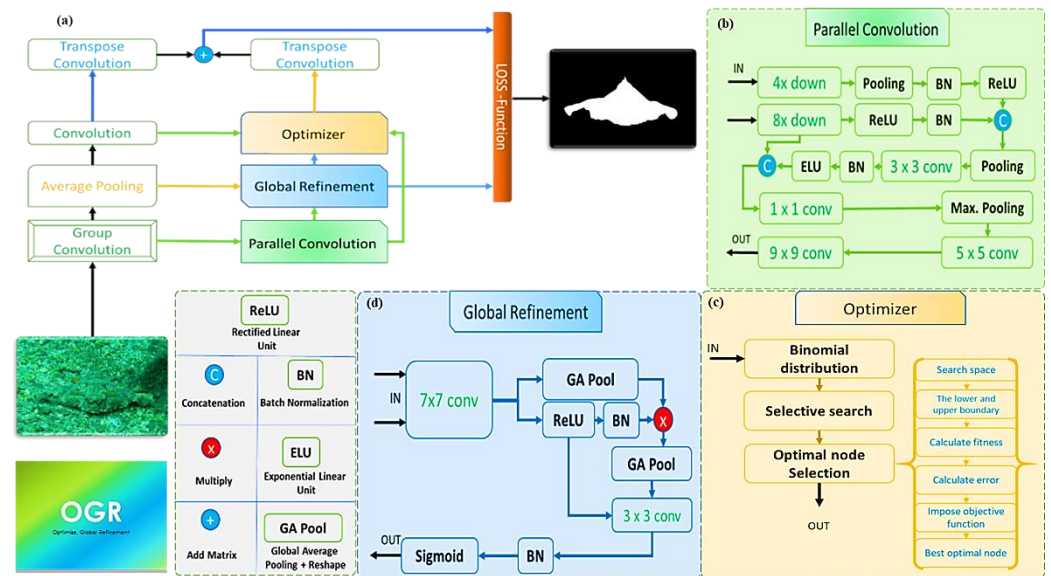


Figure 3. Optimize, Global Refinement Proposed Architecture: (a) Proposed Model; (b) Global Refinement; (c) Optimizer; (d) Parallel Convolution.

The Global Refinement Framework is proposed to utilize the ratio-dependent fusion of global features for the best alignment of the localized objects on the background map. A Parallel Convolution Framework is proposed to extract the features in parallel to make the global optimization from the parallel fused features for the best boundary construction at a low level with the least number of channels. The Optimizer Framework is proposed for the best feature channel selection and regularization. Furthermore, it helps to select the best feature matrices for the linear minimization of the loss function.

This article is composed of the following sections: Section 1 states the introduction of the domain and the problem statement that results in the proposed solution—the brief history of the domain is followed by the progressive improvement in the solution. Section 2 relates to the data, the previous methodology, and its related work. Section 3, Section 4, and Section 5 describe the proposed methodology using the mathematical model. Section 6 comprises the experiment and its results. Section 7 discusses future work and conclusions.

2. Materials and Methods

Over a decade, object detection and segmentation techniques were encouraged due to their frequent application to many real-life problems, mainly focused on the image and its pixel-wise labeling. In generic objects, labeling and prediction are comparatively easy tasks due to the discreet texture of the object and clear boundary lines to obtain the best annotations. The image’s resolution is not required to be higher due to the vividness and clarity between the object and the background. However, for the COD task, the findings of the dataset are not only the depiction of the problem, but the solution faces challenging conditions in terms of the camouflage. The algorithm trained on the generic objects fails on the salient object due to the fewer features that were found, and failed to detect the camouflaged object due to having faint information of the object boundary. Therefore, the dataset and the model design are to be explored to better understand this domain and the problem statement.

This domain has a very limited collection of datasets; due to its challenging nature, the images are difficult to find and prepare, i.e., image collection and GT annotations, respectively. The CAMO dataset [10] consists of 1250 images of ecological camouflage, and the Kvasir-SEG [11] has images of medical camouflage, consisting of 1000 images. While the COD10K dataset has the natural camouflage, consisting of 10,000 images.

Many techniques in deep learning and image processing were proposed and applied specifically for camouflaged objects, and the special attention of computer vision researchers was attained. These types of objects are often hard to be recognized by humans themselves due to

the complexity of their pixels and the boundary line of the distinct surface area contained within the image pixels. To express the existence of the camouflaged objects, different technologies and techniques have been promoted, as shown in Figure 4, not only to predict these features from the known data but to apply them to the unseen data. This task is classified as object detection and segmentation using specific dataset samples and algorithmic approaches.

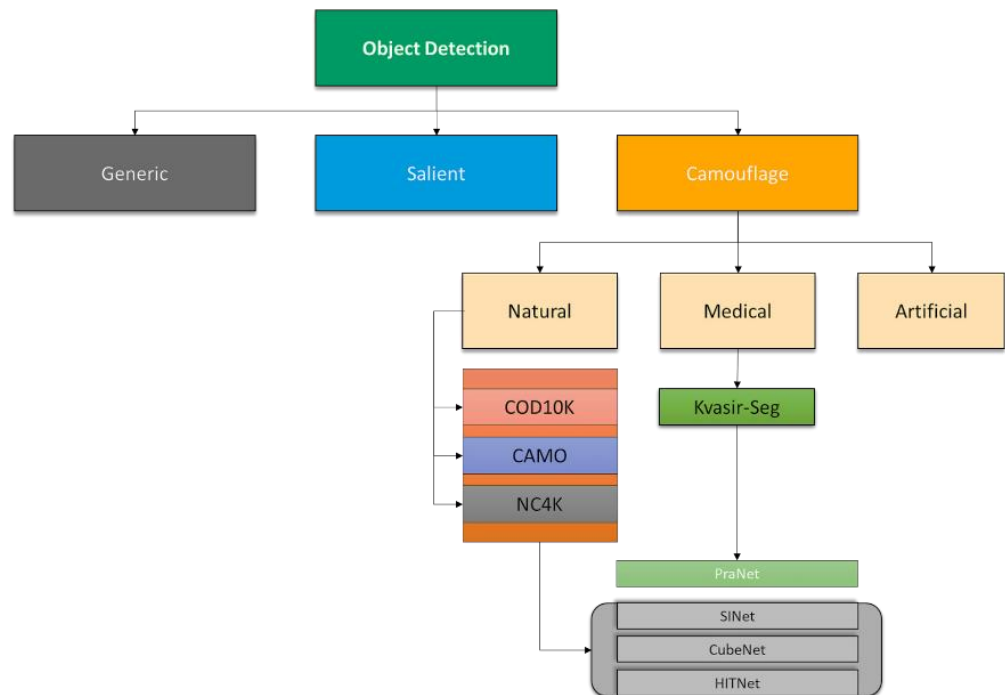


Figure 4. Object Detection Organogram.

2.1. Search and Identification

This method [4] is biologically inspired by the living predator in which, initially, a required object is searched out using the biological boundary space, having sensitive information, then the object is identified as a target for the predator. The search module uses a convolutional neural network-based inference and encloses and reduces the feature dimension of the network stream. The identification module used the block for object identification concerning the multiple evaluation functions. For this purpose, searching the identification network has resnet-50 architecture followed by the receptive field (RF) component.

The extracted features are divided for the identification module through multiple up-sampling and down-sampling serial units to be passed into the RELU activation function. The identification module has a partial decoder component (PDC) that performs element-wise multiplication to decrease the gap between the same-level features in the dual stream [12]. Hence, the designed model was accomplished with the achievement of dense feature extraction.

According to the latest research, the medical COD was explored in polyp segmentation [13], as shown in Figure 4, when a parallel reverse attention-based network [9] was utilized. Like an ordinary semantic segmentation network, it has two modules of parallel decoding and reverses the attention mechanism. The parallel decoder takes the input from a skipped serial connection of the down-sampled stream in the network, while the reverse attention is the featured mask multiplication and reversed matrix pixel decomposition followed by an up-sampled stream. The actual size is achieved after a series of up-sampled streams followed by a reverse attention module.

2.2. Boundary Guidance and Edge Enhancement

A denoised model [14] for the COD involves the removal of noise and its effects from the camouflage map in the form of uncertainty and is validated in the noisy ground truth.

The noisy labeling leads to the prediction of uncertain conditions for all types of objects and is not limited to camouflage objects. Camouflage has the particular scope to be cleaned, and handles the input and predicted weights more carefully for semantic inheritance. To achieve semantic inheritance and boundary guidance [15], as shown in Figure 5, the high-level features fused with the low-level features. These features were extracted from the independent layer and transformed into a residual channel attention block with the serial feature link to the module for each connection. Two predictions with the feature maps 3–5 and 2, with 6–8 repeating the above-mentioned process, computed the \hat{y}^{ini} and \hat{y}^{ref} , respectively, in the 0–1 range, knowing that the \hat{y}^{ref} is the final predictor. A UNet-based model [16] was proposed using the leaky ReLU [17] activation function called CODNet [18]. It uses both the predictor of \hat{y}^{ini} and \hat{y}^{ref} for the concatenation with the input to create a feature map as a confidence map. This confidence map is followed by the prediction of COD and ground truth of camouflage y .

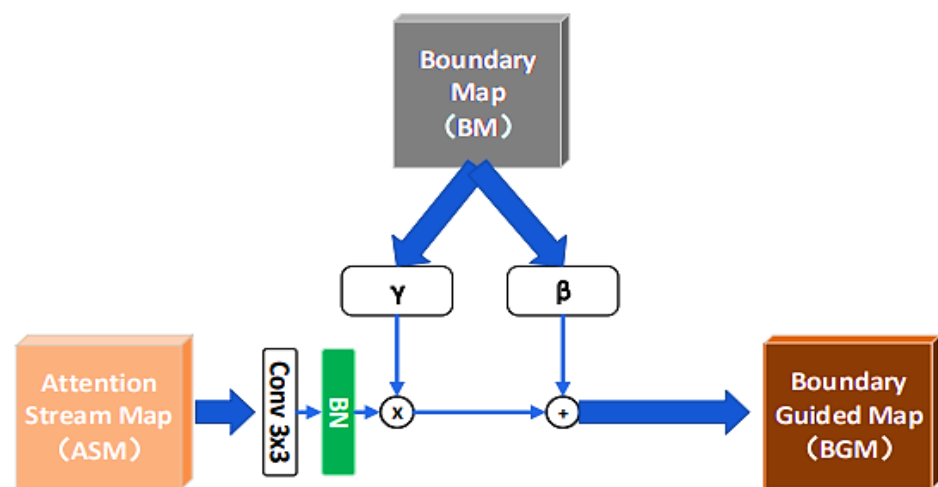


Figure 5. Boundary Guided Module [15].

The final prediction is the dynamic uncertainty precision, composed of the pixel-by-pixel distance between the actual ground truth and the confidence map of the predictor of CODNet that has two input branches. These inputs are used to adapt pixel-wise multiplication and concatenating two convolution operations following the residual block focusing on the foreground map instead of the background map. The prior edge gains the shape of the 64 channels used for the recalibration unit, and this unit is used for the refinement of the input in the decoder block instead of the fusion mechanism.

Edge-enhancement-based [19] COD tasks differentiate the traditional model approaches by modifying the sample data. The purpose is to enhance the edge information with different feature engineering techniques. A sub-edge decoder (SED) [20] is used in between the two square fusion decoders (SFD). SED uses both features, i.e., the original feature map and the fused feature map. Like most of the COD tasks, this model also uses the resnet-50 model and borrows its residual behavior to transfer to the new design. Second SFD is treated as the final camouflage map for the interaction of the loss function.

Like other COD algorithms [4,9,18], the loss function remains cooperative with the binary cross-entropy and IoU. This edge enhancement further undergoes feature-based optimization with respect to the object. The binary pixel-wise classification algorithms have the potential to be estimated by the edge dataset. The notable COD10K dataset [4] contains the edge information for all the camouflage objects and instances. These samples are to be used with edge-based classification or boundary regression using the values of the edge.

2.3. Vision Transformer in COD

The latest research found the transformer-based model's [21] interaction with the computer vision tasks led to the basic encoder and decoder network in semantic or instance

segmentation to the transformer module. The process modified the basic design of the feed-forward neural network with the refinement of feedback [22] that was attained from the loss function after the complete input inferences. The feed-forward block utilized the transformer model, and the training went iteratively backward, followed by the refinement of the effect of the loss function. A pyramid vision transformer [23] was used for the feature extraction and fed to the feedback block in which the layer is concatenated with the input. These feedback blocks were followed by some basic blocks.

The basic block consists of the convolutional layer in series with the multiplication of the fully connected layer. This output is provided to the iteration feedback module in which the data flow is controlled by the loss function with the Intersection over Union (IoU) and binary-weighted cross-entropy, as shown in [4,9,18,20].

2.4. Graph and Search Fusion in COD

A graph convolutional network (GCN) was mutually applied for edge- and region-based feature learning. The image features populated for the mutual graph learning [24] technique have a task-specific block of the extracted feature. A graph fusion [25] was implied between the inference stream of the extracted feature tensor F_1 followed by the basic block and the definite forward iteration Y^{in} of the network to obtain a resultant named Y' . These resultants were forwarded to the loss function and named iterative feedback loss to evaluate the network and update the training.

The instance-level method to separate the object in a single frame needs more attention and technical methodology. Camouflage fusion learning was performed to efficiently solve instance segmentation tasks. For this purpose, a search-based model was used with all the samples of the input images. The instance-level methodology was achieved using a vision transformer as well as multiple well-known methods, including Mask RCNN [26], Cascade Mask RCNN [27], MS RCNN [28], RetinaMask [29], CenterMask [30], YOLACT [31], SOLO [32], BlendMask [33] with ResNet50 [34], ResNet101-FPN [34] and ResNetXt101-FPN [35]. The framework [36] completes its task in two stages. Firstly, the instance segmentation model is trained using this loss function, and secondly, the search algorithm trained the predictor to choose the best instance in the segmentation model.

Hence, these techniques are for camouflage object and instance detection tasks. For this purpose, remarkable work also has been observed in the dataset collection and its preparation. However, there is still a need to fill the gap in the dataset collection and algorithm design to solve the task of COD. The search-based model combines instance search model implementations with the dual loss binary cross-entropy. The search for the best weightage of the image sample is imposed for updating the waiting list of the sample image, in this instance, with the use of the segmentation model.

3. Parallel Convolution Framework (PCF)

The PCF acquires the group convolution blocks input and surpasses it to the global refinement and optimizer. In PCF, the group convolved input g_{conv} is separated in two parallel paths for discreet structural feature extraction, and the channel outputs are concatenated in the $Conc_1$ states as shown in Equation (1).

$$Conc_1 = (P_1 \text{ C } P_2) \quad (1)$$

$$P_1 = \varphi (Bn(\text{Max}(4 \times Conv (g_{conv})))) \quad (2)$$

$$P_2 = \varphi ((8 \times Conv (g_{conv})) + Bn) \quad (3)$$

P_1 shows the pooled and batch normalization path that is concatenated with the unpooled path P_2 , as shown in Equations (2) and (3). The $Conc_1$ is responsible for the serial path ω_3 That is activated with the ELU activation function \mathcal{E} and initial features f_1 . ω_3 is followed by the batch normalization layer and its pooled and convolved $Conc_1$ input as shown in Equation (6).

$$Conc_2 = (f_1 \textcircled{C} \omega_3) \tag{4}$$

$$f_1 = P_2 - \varphi (Bn(\cdot)) \tag{5}$$

$$\omega_3 = \mathcal{E} (Bn(Max(Conc_1) + Conv)) \tag{6}$$

In Equation (4), $Conc_2$ is the concatenated output of the serial path ω_3 and initial features f_1 . Initial features f_1 are composed of $8 \times Conv$ of a group of convolved input g_{conv} , as shown in Equations (3) and (5). The final output of the ρ is retrieved after the series of convolution operations, as described in Equation (7).

$$\rho = Conc_2 + Max (Conv) + Conv_{k=1}^2 \tag{7}$$

Here, ρ is transferred to the global refinement and optimizer module independently. In the global refinement, the parallel convolved input is mixed with the average pooled input and passed to the optimizer.

4. Global Refinement Framework (GRF)

In this section, a global refinement framework is presented that employs the feature refinement of pooled input P_i and the convolved input C_i states from the average of g_{conv} and ρ , respectively, as shown in Equation (8). This P_i is composed of the average of the group convolution feature g_{conv} , as shown in Equation (9).

$$\mathfrak{R}^G (P_i \oplus C_i) \tag{8}$$

$$P_i = avg (g_{conv}) \tag{9}$$

\mathfrak{R}^G is the refined global function that handles two flows of stream, i.e., normal flow v_{in} and the vectorized flow Φ_{in} . The normal flow is activated with the ReLU and surpasses the batch normalization layer. Both flows are multiplied with the 1×1 index ratio, as shown in Equation (10).

$$\omega_1 = \sum_{i=1}^l v_{in}^i \odot \Phi_{in}^i \tag{10}$$

The first serial part ω_1 shows the partial vectorized multiplication, and the output is converted to the global average pooling layer, taking the v_{in} as convolved input stream and Φ_{in} as the pooled input stream is explored in Equations (11) and (12).

$$v_{in} = \varphi (Conv (C_i) + Bn) \tag{11}$$

$$\Phi_{in} = \aleph_{avg} (Conv (P_i)) \tag{12}$$

φ shows the activation function on the convolved input flow after convolution as $Conv$, and batch normalization as Bn make the output as v_{in} . \aleph_{avg} shows the global average pooling after the convolution on the pooled input. ω_1 is reshaped from the vector to two-dimensional vector space \mathbb{R}^2 to apply the convolution with the concatenation of activated convolved input as $v_{in} - Bn$ is shown in Equations (14) and (15).

$$\omega_2 = (v_p \textcircled{C} \Phi_p) \tag{13}$$

$$v_p = Conv ((v_{in} - Bn)) \tag{14}$$

$$\Phi_p = \aleph_{avg} (\mathbb{R}^2 (\omega_1)) \tag{15}$$

The second serial part ω_2 shows the output from the concatenation of partial convolved and partial pooled inputs as v_p and Φ_p , respectively, as shown in Equation (13). The final

output of this framework is attained after passing the batch normalization Bn and sigmoid activation Sig to ω_2 , as shown in Equation (16).

$$\mathfrak{R}^G = Sig (Bn (\omega_2)) \tag{16}$$

The objective of the global refinement module is to expand the global features by index-wise multiplication in a two-dimensional vector space and normalization of features with the average pooling and sigmoid activation function. The globally refined output is transferred to the optimizer module and the cost function. In the optimizer module, the best channels are selected for the best position of the feature matrices.

5. Optimizer Framework (OF)

In OF, channel-wise optimization is proposed for the best channel selection to avoid the overfitting and complexity of the training model. The optimizer framework \mathcal{O} is composed of three major tasks, i.e., binomial distribution, selective search, and optimal channel selections.

$$t = \mathbb{R} \left| \sum \rho + \mathfrak{R}^G + Conv \right| \tag{17}$$

It takes the input t from the parallel convolved ρ , globally refined \mathfrak{R}^G and simple convolution input as shown in Equation (17) and the input is transformed to the linear vector space \mathbb{R} .

The binomial distribution β having a lower boundary ℓ_d and the upper boundary ℓ_u , takes the vectorized input t and formats the values in 0 and 1, as shown in Equation (18). The purpose of these binary values transformation is to make the decision effectively for the selective search. This selective search works efficiently on the one-dimensional binary values β by ordering the distribution in ascending order $0 \leq i \leq n, \lambda_0^n \because \lambda_i \in \beta$.

$$\beta = \varphi (t, [\ell_d, \ell_u]) \tag{18}$$

The optimal feature selection takes the inner boundary values from β and calculates the fitness function Y for the searched instances λ_0^n as shown in Equation (19). The Y calculates the error e of λ_0^n and updates the cost using the objective function o as shown in Equations (21) and (22). For error calculation in λ_0^n , mean squared error (MSE) for the regression values and the objective function is expanded using the best instance value λ_0^n as shown in Equation (20).

$$Y = \sum_{k=1}^l (e + o), \quad \lambda_k^n = \lambda_0^n \tag{19}$$

$$\lambda_0^n = (\lambda_0^n = 1) \tag{20}$$

$$e = MSE (a + b\lambda_k^n) \tag{21}$$

$$o = a \times e + (1 - a) \times \left(\frac{1}{dim_0(\lambda_k^n)} \right) \tag{22}$$

l shows the number of iterations for the fitness function, composed of the error calculation and the objective function. The samples at a higher position, correspond to the fitness evaluation and are selected from β . The best features f_b from the λ_k^n are selected based on high boundary values as shown in Equation (23).

$$f_b = \begin{cases} \beta_i = 1 & \lambda_k^n = 1 \\ \beta_i = 0 & otherwise \end{cases} \tag{23}$$

The f_b and $Conv$ are transferred to the convolutional transpose layer and added to make a single flow τ_f as shown in Equation (24) to the cost function. The MAE is used as a cost function C_m of the model in which τ_f and ground truth G , as shown in Equation (25).

$$\tau_f = transConv(f_b) \oplus transConv(Conv) \quad (24)$$

$$C_m = MAE(\tau_f, G) \quad (25)$$

$transConv$ is transpose convolution that is applied to the best feature's output and convolved output to get the contracted feature map.

6. Experiments and Results

The proposed model is applied in the experimental setup by using PyTorch with the Adam Optimizer. The training batch size was experienced on the 8 and 12 and found the results were better on batch size 8 with the initial learning rate of 1×10^{-4} . The whole training time was almost 60 min for the 10 epochs. The experiment was performed on the platform of 12 CPUs and 1280-core graphic processors on all the datasets, and the image size was maintained at $352 \times 352 \times 3$. This input size of the model is further used in preprocessing, cleaning, and enhancement of the model. The preprocessing was performed for data normalization and defining the parameter of the input shape from the raw data. A pixel value normalization was also used on the model input gate from its minimum to maximum range. This input was forwarded to the model for feature extraction, and the results were compared with the predefined results to train the model. The evaluation time on this framework is 83.3 ms/sample.

The CAMO dataset [10] consists of generic camouflage objects that are dependent upon the specific situation or condition, as shown in Figure 6a. COD10K [4] contains 10,000 images, as shown in Figure 6b, in which 5066 are defined as camouflaged, 3000 background, and 1934 as non-camouflaged. It occupies the 78 sub-classes of biological camouflage. The collection of the 10,000 images includes 6000 for training and the remaining 4000 samples for testing purposes. The Kvasir-SEG dataset [11] is the medical camouflage dataset with a gastrointestinal binary segmented area designed for polyp segmentation. It has 1000 samples with the ground truth provided, a very small dataset of images. The provided annotations were generated for the images as shown in Figure 6c, and the images were collected through biomedical sources. This dataset was beyond consideration in the COD domain, but due to its application in the medical field, it is an important camouflage dataset, and researchers have been utilizing it for experiments.

Mean absolute error (MAE) [37] is the simplest cost function to evaluate the two values, i.e., predicted and actual. As shown in Equation (26), the formula is the qualitative measurement of the two values concerning the observed samples. The formula explains the difference between the actual and predicted value and the mean to the total number of observations.

$$MAE = \frac{\sum_{i=1}^n |y_i - x_i|}{n} \quad (26)$$

The precision [38] formula is modified with the average precised class on the predicted class with the True Positive TP and False Positive FP .

$$S - Measure(x, y) = \frac{(2 \mu_x \mu_y + c1)(2 \sigma_{xy} + c2)}{(\mu_x^2 + \mu_y^2 + c1)(\sigma_x^2 + \sigma_y^2 + c2)} \quad (27)$$



Figure 6. Camouflage Dataset Image Samples (a) COD10K (b) CAMO (c) Kvasir-SEG.

The structure similarity index measure [39] is described in Equation (27) as the *S-Measure* uses the average of x, y position as μ_x, μ_y , variance of x, y and covariance of x and y $\sigma_x^2, \sigma_y^2, \sigma_{xy}$. L as dynamic range, $c1$ and $c2$ constant as bits per pixel, 0.01 and 0.03 by default, respectively. This is a performance metric of two binary maps of predicted and true to find the similarity of the structure.

$$F_{\beta} = \left(1 + \beta^2 \right) \frac{Precision \cdot Recall}{\beta^2 \cdot Precision + Recall} \tag{28}$$

The F-measure beta [40] is denoted by F_{β} , the binary map using the *Precision (avg)* and *Recall* as described in Equation (28). It uses the beta constant β to control the complete and detection co-factors. The beta constant is 1 by default in order to have smoothness detection.

$$E - measure = \frac{1}{w \times h} \sum_{x=1}^w \sum_{y=1}^h \varnothing FM. (x, y) \tag{29}$$

It [41] is a binary map performance metric denoted by *E-Measure* dependent on the ground truth *GT* and foreground map *FM* bias metric $\varnothing FM$ as shown in Equation (29). With the weight w and height h of the foreground map, the alignment measurement is accomplished at the x and y position of the pixel.

The experimental procedure was initially adopted and performed using the benchmark publicly available dataset and the pre-trained model. Furthermore, all the datasets are trained and tested using the proposed framework and compared with the existing methods. The experiments and comparative results are presented in the following order:

1. Experiments on benchmark datasets using baseline and pretrained methods;
 2. Experiments on benchmark datasets using the proposed method;
- Comparison of the proposed method with baseline and pretrained results;
 - Comparison of the proposed method with the state-of-the-art method's results.

6.1. Experiment 1 on Pretrained Models

6.1.1. CAMO + COD10K Datasets—In the Light of Composite Experiments

These large-scale datasets need to explore predefined random methods. Firstly, the dataset is tested with binary accuracy and the Dice coefficient with the binary cross-entropy loss function. The model used for these performance measures was polyp segmentation based on the area color.

Experiments show that the results are about 10% in the ten epochs, even when transfer learning is performed. Furthermore, the cost function is very high. The efficient segmentation net has also been used to tackle these problems. The dataset is measured upon the binary accuracy performance metric as well. The loss or the cost function was still binary cross-entropy.

The graph in Figure 7a,b shows that the dice coefficient increases after 20+ epochs, while the loss is reduced, as shown in the figure, which is more costly than the model taking a significant amount of time to be trained, as one iteration takes 30 min to 60 min due to a large number of samples in the dataset. The same behavior has been identified in the other performance metric and the loss function applied to this model. The UNET model [16] and FCN [42] with the VGG16 pre-trained backbone [43] are also used with this dataset, but the same behavior of the model was observed. This limitation needs to be addressed to explore the dataset’s nature. The proposed model and its experiments are conducted to understand that refinement is needed for some spatial feature extraction processes and optimization. The experiment as mentioned above shows the behavior of the dataset before the benchmark pre-trained models.

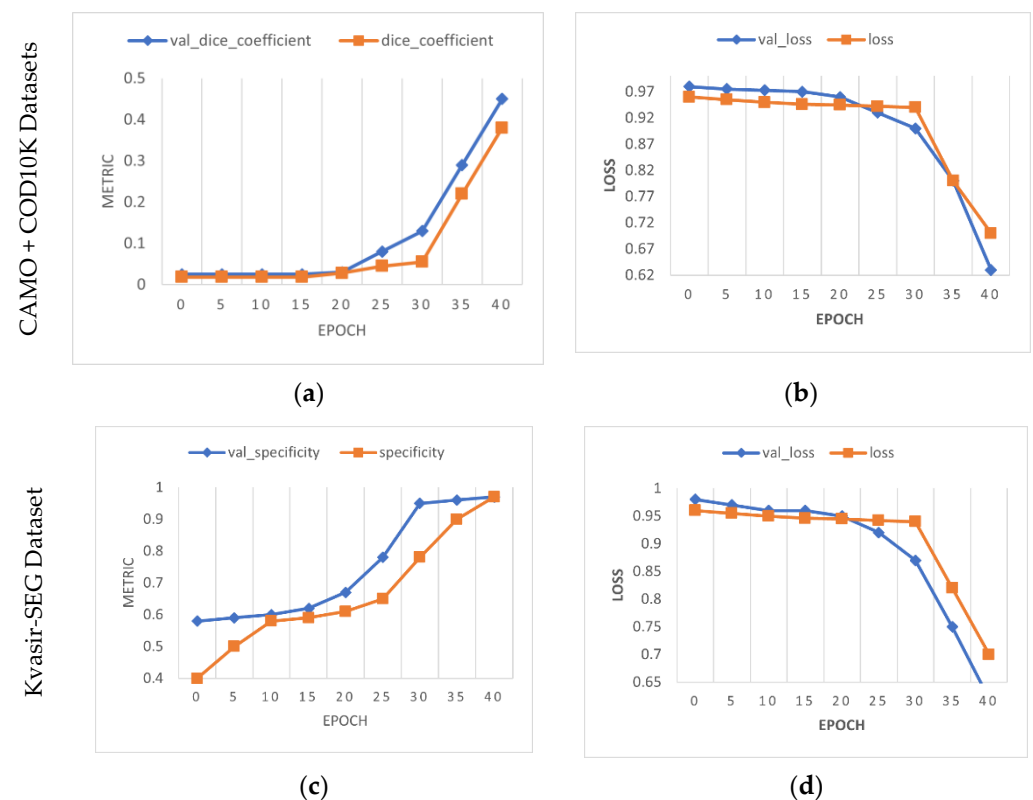


Figure 7. CAMO + COD10K, and Kvasir-SEG Datasets, Combined Experiments on training and validation dataset: (a) dice coefficient; (b) binary cross-entropy loss; (c) specificity; (d) binary cross-entropy loss.

6.1.2. Kvasir-SEG—In the Light of Composite Experiments

The experimental results on the ground truth and texture of the sample were achieved due to less complexity. The dataset was trained on the same set of hyperparameters and other dataset behavior such as sample size, batch size, and the transformation applied. The tabular data show the experiments based on the three datasets and their training with

the composite and proposed model architecture, as shown in Table 1. The specificity was used as the performance metric with built-in accuracy metric binary accuracy and the loss function of binary cross-entropy due to the binary segmentation of the dataset using the image samples and binary ground truth, as shown in Figure 7c,d. The analyses of all of the experiments using different deep-learning algorithms and the proposed model indicate the achievement of optimization.

Table 1. Comparison of the proposed method with pre-trained UNet and VGG-16.

Algorithm	Epochs				Dataset	Remarks
	6	10	20	20+		
UNET (Modified)	4%	2%	8%	12%+	CAMO	-
	3%	2%	7%	10%+	COD10K	-
	52%	60%	67%	70%+	Kvasir-SEG	-
VGG 16 pretrained backbone	1%	3%	6%	12%+	CAMO	-
	2%	4%	5%	10%+	COD10K	-
	55%	56%	65%	70%+	Kvasir-SEG	-
Optimize Global Refinement	70%	96%	-	-	CAMO	Optimized
	91%	94%	-	-	COD10K	
	87%	93%	-	-	Kvasir-SEG	

6.2. Experiment 2 Using OGR Framework

The training and validation graph based on the performance and the loss function is to be demonstrated to visualize the experimental achievements. The proposed model was applied to the three datasets, including CAMO, COD10K, and Kvasir-Seg, respectively. These datasets are from different branches of image segmentation.

6.2.1. Experiments on CAMO Dataset

The benefit of an average number of samples with the binary generated a mask of almost ~1250 and was trained with the proposed architecture and attained the desired results. In the preprocessing stage, the transformation was used on the image samples to exploit the flip, translate, and rotate parameters. The image samples were then passed to the proposed architecture of the semantic segmentation model to train the samples with masks. The MAE was a loss function for the model backpropagation and updating the weights.

In some experiments, the loss function was reduced by the iteration or epochs, but the accuracy was not attained. Additionally, in the iteration where the trainable parameters do not reduce the loss function, validation accuracy of >90% was achieved. Such behavior of the loss function on training and validation shows the inconsistency in the data samples. The CAMO dataset [10] has generalized camouflage, which is not only limited to natural camouflage but also contains situational or other camouflage. The Adam optimizer initialized with zero gradients showed the independent behavior of the model inference. The model's training sample showed some accuracy reduction, but on the test set, remarkable model behavior was observed. The strategy was observed in that the MAE was working independently, and the model parameter metrics were saying something else. The validation of the situation-based camouflage has the same impact on the model behavior in terms of inference and learning. The model was repeatedly trained on this dataset, wherein the Adam optimizer, variational accuracy and loss are better than the training accuracy and loss, which is very common.

The graph as shown in Figure 8 describes that the model validation accuracy has sudden behavioral learning, which is somehow a fair condition and the cause of dataset complexity, as well as learning rate (initially 0.001) for 10 epochs, and the 70 | 30% train-test ratio.

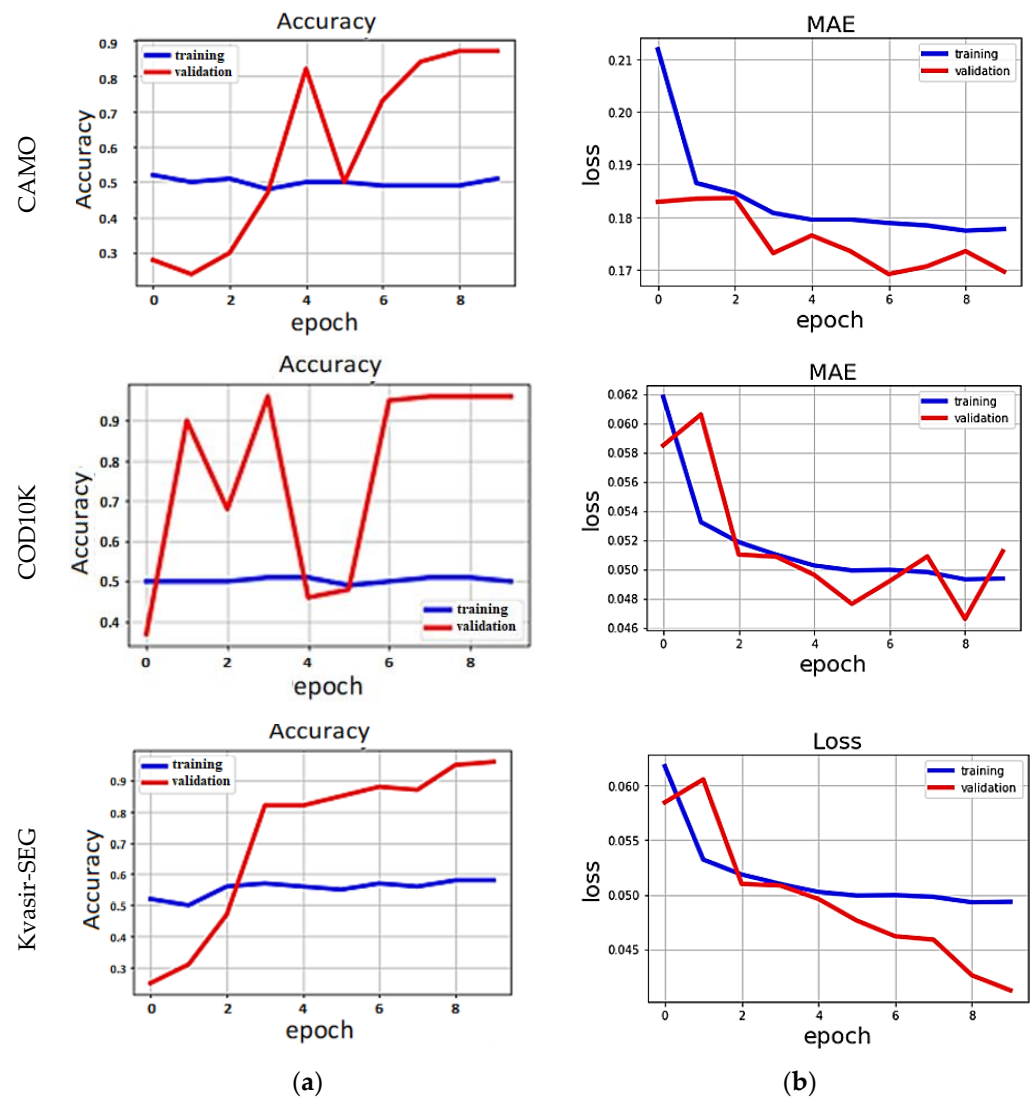


Figure 8. Training and validation graph on CAMO, COD10K, Kvasir-SEG on training and validation dataset: (a) binary accuracy; (b) mean absolute error loss.

6.2.2. Experiments on COD10K

This is a large dataset with high-resolution images for training (352×352), and the Adam optimizer was used for the learning rate control. With the loss function of MAE, the network was trained in 10 iterations. This dataset is complex in that the predefined model, such as UNeT, only brings an accuracy of 50%. The technique of parallel convolution and the optimizer function caused the dataset to have the highest accuracy. The learning parameters of the model remained at zero for each batch size until the model obtained the best results independent of the iterations. In another setting, the model optimizer became iterative of the previous epoch. The batch size of 2 and 3 was the only possible option for the machine to have an offline experiment of the proposed model.

The training results of the proposed model were explored in graph form, and the quantitative analysis of this dataset with the previous technique was in tabular form. The data was in the binary polyp segmentation type despite multiple instances. The concept of the existence and non-existence of the camouflage was used to keep the model smooth and less burdened. The data loader identified with the normalization and the preprocessing of the input sample image. The model is trained on the condition to reduce the loss function and the performance measurement. The training direction is independent of the metrics used in this model. The graph shows the accuracy, the losses, and the effect of the reducing

loss function, which was very low on this dataset. The constant lowering of the cost function can result in accuracy loss independent of the iterations and dependent on the data samples. As shown, the proposed model was trained on this dataset with the Adam optimizer, a variational learning rate initially at 0.001 with ten epochs, and 70 | 30% and 80 | 20% train–test ratios.

6.2.3. Experiments on Kvasir–Seg Dataset

The rarely used dataset used as camouflage is the polyp segmentation of gastrointestinal with the relevant segmentation mask. This dataset is based on binary polyp segmentation and was passed to the proposed model. The data is biological, so the behavior is different for the validation. The settings of the model hyperparameters were not changed to see the model’s behavior for each dataset under the same circumstances. The preferred input size was 256×256 instead of 352×352 . The optimizer was changed to RMSProp to show the dataset variations and the effect of the dataset sample on the model.

The comparison of multiple datasets is shown in Table 2. The binary segmentation was performed with batch sizes of 2 and 3, respectively. The learning iteration was 10, and the train–test ratios were 80 | 20% and 90 | 10% due to the 1000 samples.

Table 2. Comparison of the proposed OGR w.r.t Baselines Camouflage models and datasets.

Baseline	Dataset	↑ S _{measure}	↑ E _{measure}	↑ F _{measure}	↓ MAE
SiNet	CAMO	0.751	0.771	0.606	0.100
	COD10K	0.869	0.891	0.740	0.044
	Kvasir-SEG	-	-	-	-
PraNET	CAMO	-	-	-	-
	COD10K	-	-	-	-
	Kvasir-SEG	0.915	0.948	0.915	0.030
CubeNet	CAMO	0.788	0.850	0.788	0.085
	COD10K	0.795	0.864	0.644	0.041
	Kvasir-SEG	-	-	-	-
CANet	CAMO	0.799	0.770	0.865	0.075
	COD10K	0.809	0.885	0.703	0.035
	Kvasir-SEG	-	-	-	-
Pyvit+GCN	CAMO	0.544	0.902	0.801	0.057
	COD10K	0.868	0.932	0.798	0.024
	Kvasir-SEG	-	-	-	-
Proposed	CAMO	0.962	0.951	0.8952	0.0362
	COD10K	0.94	0.9353	0.8426	0.0402
	Kvasir-SEG	0.936	0.957	0.922	0.0258

Figure 9 shows the experiment results performed using three different datasets and validation and the training graphs that were generated. The loss function is also presented for comparative analysis of all the metrics. In this section, tables and charts are shown concerning the different datasets and the comparison with the previous benchmark techniques/algorithms based on the standard performance measures. The detailed information in the form of the graph is shown in Figure 9a–c, showing the performance of the proposed model and the dominance of all the previously proposed algorithms for accuracy gain and cost reduction, while Figure 10 shows the prediction result on the COD10K dataset.

For a comprehensive analysis, a discussion of the different performance metrics is shown in Table 2. In light of the following experiments and discussion, the loss function and MAE analysis found that accuracy was not attained, despite lowering the cost function. The lowering of MAE should increase the accuracy of the model. In the model’s controlling phase of the training, only the cost function can play a role in updating the gradient of the model’s parameters. The cost function is responsible for the model training control and

the parameters update. The observations indicate that lowering the cost function led to sudden changes in accuracy, which is very dangerous.

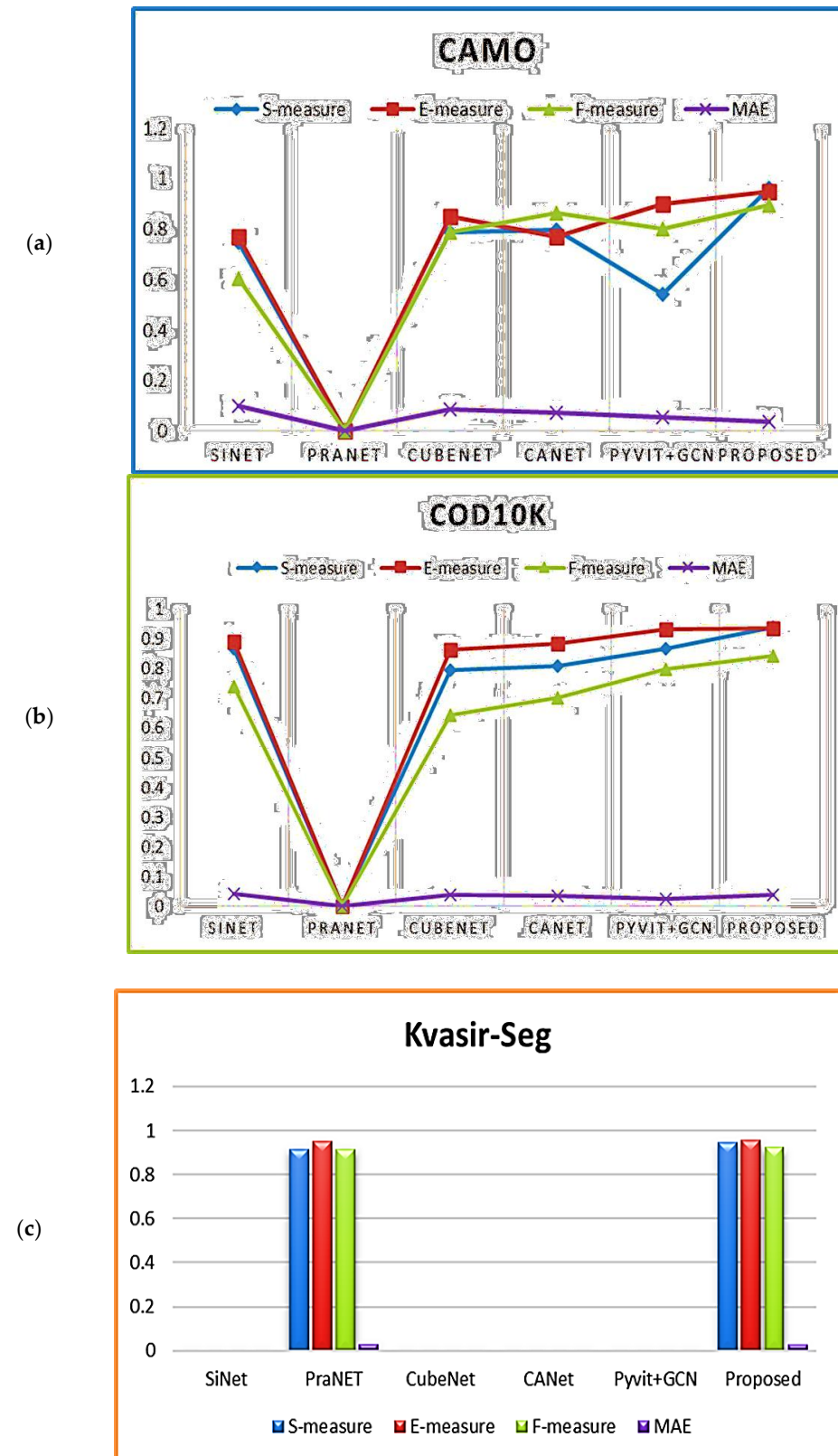


Figure 9. Comparative Analysis of multiple datasets: (a) CAMO; (b) COD10K; (c) Kvasir-SEG.

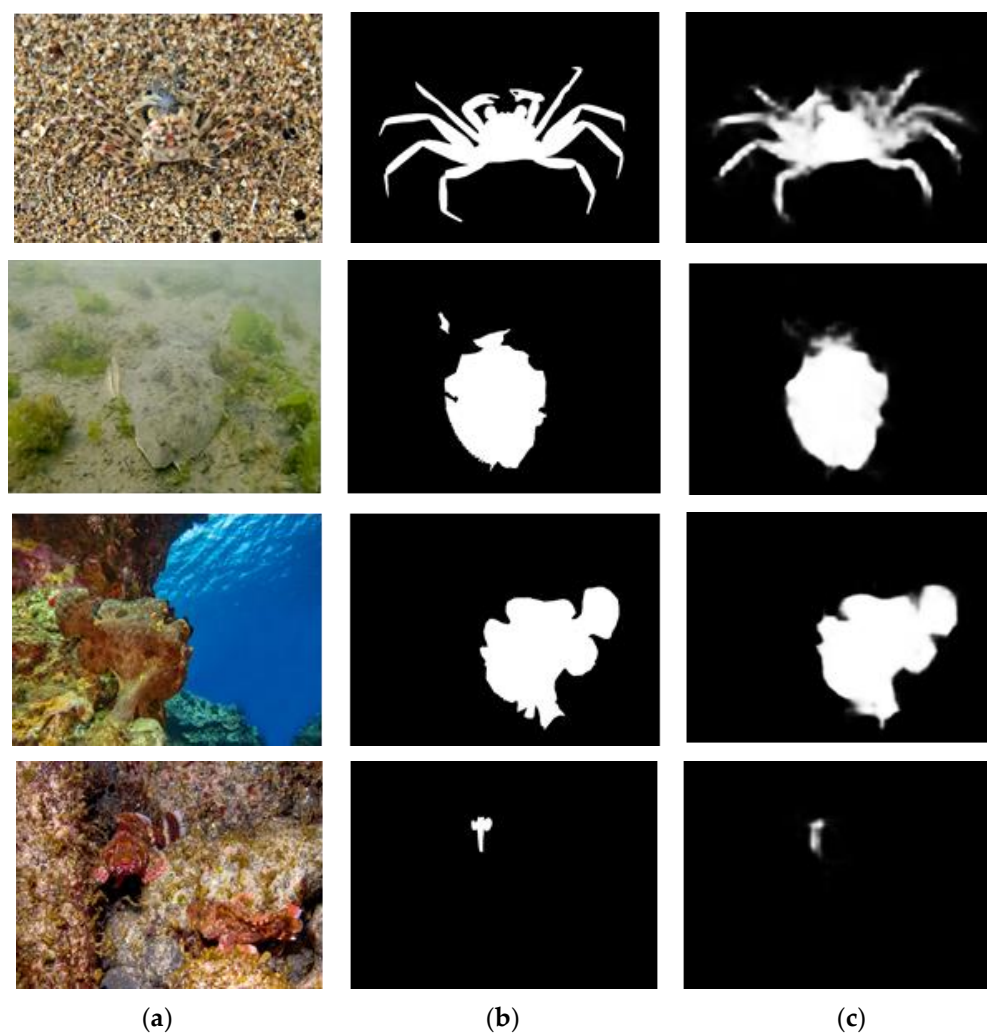


Figure 10. Prediction Results of OGR on COD10K: (a) Sample Images; (b) GT Label; (c) Predicted Label.

6.3. Comparison Results

Due to the very small number of samples in the past, the accuracy achievement and the deep learning strategy were hard to be accomplished. The large-scale working of the COD10K shows the way of interaction in the different parameters of exploration. A fully convolutional-based network has been applied so far in this type of dataset due to the image and the ground truth availability. The results indicated that the SINet achieved results of almost 86%, 86%, 75%, and 77% for the CHAMELEON, CAMO, and COD10K, respectively, with the different experiments of dataset generalization, meaning the different combinations of the multiple datasets, as shown in Table 3.

Another experiment was conducted but without the large-scale dataset (COD10K); it involved the internal body camouflage (Kvasir-SEG), and the results could reach 91%. Then, multiple models, e.g., parallel attention-based, Pyramid transformer-based, and pre-trained backbone with resnet50, were used with the multiple datasets, and average scores of 91.5%, [92%, 84%, 86%, 87%], [87%, 78%] were achieved. Comparing these results with this study to further analyze this method is necessary. The proposed framework attained accuracy results of 96.2%, 94.1%, and 93.4% on the CAMO, COD10K, and Kvasir-SEG datasets, respectively.

This section described the results of the previous study with another study in terms of quantitative analysis. The methods used in previous research were useful to this study by way of comparison.

Table 3. Comparison of different state-of-the-art methods, datasets, and their results.

References	Methods	Dataset	Results
[4]	A resnet50-CNN model with Search and Identification deep models	CHAMELEON, CAMO, COD10K	86.9%, 75.1%, 77.1%
[9]	PraNet	Kvasir-SEG	91.5%
[18]	A resnet50-CNN model with CANet	CHAMELEON CAMO, COD10K, NC4K	90.1%, 80.7%, 83.2%, 85.7%
[24]	Mutual GCN	CHAMELEON CAMO, COD10K	89.3%, 77.5%, 81.4%
[20]	A resnet50-CNN model with CubeNet	CHAMELEON CAMO, COD10K	87.3%, 78.8%, 79.5%
[25]	Pyramid vision Transformer + GCN fusion	CHAMELEON CAMO, COD10K, NC4K	92.2%, 84.4%, 86.8%, 87.0%
[36]	ResNet50,101-FPN, ResNetXt101-FPN with CFL	CAMO++, COD10K (Instance)	33.6% 31.4%
Proposed Model	Optimized Parallel Refinement	CAMO	96.2%
		COD10K	94.1%
		Kvasir-SEG	93.4%

7. Conclusions and Future Work

This study presents an optimization technique for COD tasks and a comparative analysis with the previous study. The dataset was used with the same parameters as previously used. The preprocessing techniques, e.g., transformation, image enhancement, noise removal, etc., are used in some experiments. The proposed model works efficiently and provides an optimal solution to the problem of camouflaged object detection. The performed experiments used to benchmark and publicly available datasets on pre-trained models, benchmark models, and the proposed model, indicating that the proposed model was the best in terms of the mentioned datasets concerning performance optimization and accuracy of the model. The training epochs, efficient learning, and optimization are best for the previously defined architecture. This study explored the comparative results of the experiment of the previous study with the performance of the proposed model.

The identification of the loss function is very important for model performance.

It can be further improved by selecting the best loss function for the specific model. The model can be generalized further, and the combined dictionary keys in the model can be modified for a better solution. This technique can also be optimized by combining supervised and unsupervised methods. The advancement in the run-time inference and the comparative analysis of the samples can also be enhanced.

Author Contributions: Conceptualization, M.K., S.U.R. and K.A.A.; Formal analysis, M.K.; Funding acquisition, K.A.A.; Methodology, M.K., S.U.R., T.M., K.A.A. and H.T.R.; Software, M.K., S.U.R., T.M. and H.T.R.; Supervision, S.U.R. and H.T.R.; Visualization, S.U.R., T.M. and K.A.A.; Writing—original draft, M.K., S.U.R., T.M., K.A.A. and H.T.R.; Writing—review and editing, M.K., S.U.R., T.M. and H.T.R. All authors have read and agreed to the published version of the manuscript.

Funding: The authors extend their appreciation to King Saud University, Saudi Arabia, for funding this work through the Researchers Supporting Project number (RSP-2021/305), King Saud University, Riyadh, Saudi Arabia.

Data Availability Statement: Not applicable.

Acknowledgments: The authors extend their appreciation to King Saud University, Saudi Arabia, for funding this work through Researchers Supporting Project number (RSP-2021/305), King Saud University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Horikawa, T.; Kamitani, Y. Generic decoding of seen and imagined objects using hierarchical visual features. *Nat. Commun.* **2017**, *8*, 15037. [[CrossRef](#)]
2. How, M.J.; Santon, M. Cuttlefish camouflage: Blending in by matching background features. *Curr. Biol.* **2022**, *32*, R523–R525. [[CrossRef](#)]
3. Soofi, M.; Sharma, S.; Safaei-Mahroo, B.; Sohrabi, M.; Organli, M.G.; Waltert, M. Lichens and animal camouflage: Some observations from central Asian ecoregions. *J. Threat. Taxa* **2022**, *14*, 20672–20676. [[CrossRef](#)]
4. Fan, D.-P.; Ji, G.-P.; Sun, G.; Cheng, M.-M.; Shen, J.; Shao, L. Camouflaged object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 2777–2787.
5. Borji, A.; Cheng, M.-M.; Hou, Q.; Jiang, H.; Li, J. Salient object detection: A survey. *Comput. Vis. Media* **2019**, *5*, 117–150. [[CrossRef](#)]
6. Shah, J.H.; Sharif, M.; Raza, M.; Murtaza, M.; Rehman, S.U. Robust Face Recognition Technique under Varying Illumination. *J. Appl. Res. Technol.* **2015**, *13*, 97–105. [[CrossRef](#)]
7. Yasmeen, U.; Shah, J.H.; Khan, M.A.; Ansari, G.J.; Rehman, S.U.; Sharif, M.; Kadry, S.; Nam, Y. Text Detection and Classification from Low Quality Natural Images. *Intell. Autom. Soft Comput.* **2020**, *26*, 1251–1266. [[CrossRef](#)]
8. Wang, H.; Jiang, X.; Ren, H.; Hu, Y.; Bai, S. Swiftnet: Real-time video object segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 1296–1305.
9. Fan, D.P.; Ji, G.P.; Zhou, T.; Chen, G.; Fu, H.; Shen, J.; Shao, L. Pranet: Parallel reverse attention network for polyp segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Lima, Peru, 4–8 October 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 263–273.
10. Le, T.-N.; Nguyen, T.V.; Nie, Z.; Tran, M.-T.; Sugimoto, A. Anabranched network for camouflaged object segmentation. *Comput. Vis. Image Underst.* **2019**, *184*, 45–56. [[CrossRef](#)]
11. Jha, D.; Smedsrud, P.H.; Riegler, M.A.; Halvorsen, P.; Lange, T.D.; Johansen, D.; Johansen, H.D. Kvasir-seg: A segmented polyp dataset. In Proceedings of the International Conference on Multimedia Modeling, Daejeon, Korea, 5–8 January 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 451–462.
12. Zou, Z.; Shi, Z.; Guo, Y.; Ye, J. Object detection in 20 years: A survey. *arXiv* **2019**, arXiv:1905.05055.
13. Dong, B.; Wang, W.; Fan, D.P.; Li, J.; Fu, H.; Shao, L. Polyp-pvt: Polyp segmentation with pyramid vision transformers. *arXiv* **2021**, arXiv:2108.06932.
14. Tian, C.; Fei, L.; Zheng, W.; Xu, Y.; Zuo, W.; Lin, C.-W. Deep learning on image denoising: An overview. *Neural Netw.* **2020**, *131*, 251–275. [[CrossRef](#)]
15. Zhu, H.; Li, P.; Xie, H.; Yan, X.; Liang, D.; Chen, D.; Wei, M.; Qin, J. I Can Find You! Boundary-Guided Separated Attention Network for Camouflaged Object Detection. *AAAI* **2022**, *36*, 3608–3616. [[CrossRef](#)]
16. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015; Springer: Berlin/Heidelberg, Germany, 2015; pp. 234–241.
17. Dubey, A.K.; Jain, V. Comparative Study of Convolution Neural Network’s ReLu and Leaky-ReLu Activation Functions. In *Applications of Computing, Automation and Wireless Systems in Electrical Engineering*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 873–880.
18. Liu, J.; Zhang, J.; Barnes, N. Confidence-Aware Learning for Camouflaged Object Detection. *arXiv* **2021**, arXiv:2106.11641.
19. Luthra, A.; Sulakhe, H.; Mittal, T.; Iyer, A.; Yadav, S. Eformer: Edge Enhancement based Transformer for Medical Image Denoising. *arXiv* **2021**, arXiv:2109.08044.
20. Zhuge, M.; Lu, X.; Guo, Y.; Cai, Z.; Chen, S. CubeNet: X-shape connection for camouflaged object detection. *Pattern Recognit.* **2022**, *127*, 108644. [[CrossRef](#)]
21. Han, K.; Wang, Y.; Chen, H.; Chen, X.; Guo, J.; Liu, Z.; Tang, Y.; Xiao, A.; Xu, C.; Xu, Y.; et al. A survey on visual transformer. *arXiv* **2020**, arXiv:2012.12556.
22. Le, T.-H.; Dai, L.; Jang, H.; Shin, S. Robust Process Parameter Design Methodology: A New Estimation Approach by Using Feed-Forward Neural Network Structures and Machine Learning Algorithms. *Appl. Sci.* **2022**, *12*, 2904. [[CrossRef](#)]
23. Wang, W.; Xie, E.; Li, X.; Fan, D.-P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; Shao, L. Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 548–558. [[CrossRef](#)]
24. Zhai, Q.; Li, X.; Yang, F.; Chen, C.; Cheng, H.; Fan, D.P. Mutual graph learning for camouflaged object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 12997–13007.
25. Hu, X.; Fan, D.P.; Qin, X.; Dai, H.; Ren, W.; Tai, Y.; Wang, C.; Shao, L. High-resolution Iterative Feedback Network for Camouflaged Object Detection. *arXiv* **2022**, arXiv:2203.11624.

26. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
27. Cai, Z.; Vasconcelos, N. Cascade r-cnn: Delving into high quality object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6154–6162.
28. Huang, Z.; Huang, L.; Gong, Y.; Huang, C.; Wang, X. Mask scoring r-cnn. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 6409–6418.
29. Fu, C.Y.; Shvets, M.; Berg, A.C. RetinaMask: Learning to predict masks improves state-of-the-art single-shot detection for free. *arXiv* **2019**, arXiv:1901.03353.
30. Lee, Y.; Park, J. Centermask: Real-time anchor-free instance segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 13906–13915.
31. Bolya, D.; Zhou, C.; Xiao, F.; Lee, Y.J. Yolact: Real-time instance segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 9157–9166.
32. Wang, X.; Kong, T.; Shen, C.; Jiang, Y.; Li, L. Solo: Segmenting objects by locations. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 649–665.
33. Chen, H.; Sun, K.; Tian, Z.; Shen, C.; Huang, Y.; Yan, Y. Blendmask: Top-down meets bottom-up for instance segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 8573–8581.
34. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
35. Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; He, K. Aggregated residual transformations for deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1492–1500.
36. Le, T.N.; Cao, Y.; Nguyen, T.C.; Le, M.Q.; Nguyen, K.D.; Do, T.T.; Tran, M.T.; Nguyen, T.V. Camouflaged Instance Segmentation In-the-Wild: Dataset, Method, and Benchmark Suite. *IEEE Trans. Image Process.* **2021**, *31*, 287–300. [[CrossRef](#)]
37. Chai, T.; Draxler, R.R. Root mean square error (RMSE) or mean absolute error (MAE)?—Arguments against avoiding RMSE in the literature. *Geosci. Model Dev.* **2014**, *7*, 1247–1250. [[CrossRef](#)]
38. Henderson, P.; Ferrari, V. End-to-End Training of Object Class Detectors for Mean Average Precision. In *Asian Conference on Computer Vision*; Springer: Cham, Switzerland, 2016; pp. 198–213. [[CrossRef](#)]
39. Wang, Z.; Bovik, A.; Sheikh, H.; Simoncelli, E. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [[CrossRef](#)] [[PubMed](#)]
40. Margolin, R.; Zelnik-Manor, L.; Tal, A. How to evaluate foreground maps? In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 248–255.
41. Fan, D.-P.; Gong, C.; Cao, Y.; Ren, B.; Cheng, M.-M.; Borji, A. Enhanced-alignment Measure for Binary Foreground Map Evaluation. *arXiv* **2018**, arXiv:1805.10421. [[CrossRef](#)]
42. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
43. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.