Camouflaged Object Detection Using Convolutional Neural Network*

Karolis Mickevičius[1], Arūnas Lipnickas[1]

[1]Kaunas University of Technology, Faculty of Electrical and Electronics Engineering, Department of Automation

Abstract

Camouflaged object detection assists in the detection of camouflaged animals, military personnel and equipment. The usage of such algorithms extends to finding specified cells in biology and detection of surface defects in industry. This study researches the task of detecting camouflaged objects using a convolutional neural network. A U-Net network is selected and adapted to receive a dataset of camouflaged objects and is tested to be able to process it. Different U-Net modifications, loss functions, photo sizes, colour channels and conversion spaces for camouflaged object detection are studied. The results of camouflaged object detection are shown as Dice, IoU and AUC metrics. The results are compared and it is found accurately did the U-Net model predict the location of camouflaged objects.

Keywords

Neural network, convolutional neural network, U-Net, Camouflaged Object Detection, image segmentation

1. Introduction

Camouflage - a defence mechanism to avoid detection and recognition from the environment using both natural and artificial methods, concealment by means of lighting, color, materials or camouflage, such as background matching (colors, appearance and patterns), disruptive painting, self-shadow concealment, and destructive shading. Animals , through evolution, have acquired the ability to adapt to their environment and to camouflage themselves in order to hide from predators and prey. Camouflage is also used in hide from animals when hunting or studying them and in the military to conceal personnel and equipment from enemy forces. Clothing with a specific texture and face paints are used to match the colors of the environment. Camouflaged Object Detection or COD is used to identify objects with colors and textures very similar to the background. Although COD is more difficult than other common image segmentation tasks, it has many possible applications. The Camouflaged Object Detection algorithm can be used for medical treatments, detecting new species in biology, and surface defects in industry.

The following Section 2 presents work related to ours. Section 3 explains the data used for this research and how it was prepared. Section 4 explains what U-Net models, loss functions and performance metrics were used. Section 5 presents the results of this study. Section 6 concludes this study.

2. Related Work

Camouflaged object detection is a task in computer vision. The goal of COD is to discern camouflaged objects from their environment. Deep learning is used to achieve this goal.

One of the neural networks used is the Residual Neural Network (ResNet), or its newer version, Res2Net. Using the basis of these models, they obtain preliminary properties for each layer. Since the output of the lowest layer features of the model basis contains more noise and less useful information, while the output of the top four model basis features contains more semantic information that can provide more effective guidance for further feature extraction, researchers choose only the top four model basis features [1].

*IVUS 2025: Information Society and University Studies 2025, May 15, Kaunas, Lithuania

arunas.lipnickas@ktu.lt (A. Lipnickas)

0000-0002-5686-0646 (A. Lipnickas)

In [2], researcher used TIM (Texture-aware Interaction Module) to extract fine textures that could be precisely localised. They first used a simple 1x1 convolutional layer to reduce the input feature channels to 64. They then split these features into two branches, one used to extract local features by adding a channel attention module ("CAM"), and the other used to capture spatial features guided by a position attention module ("PAM"). Researchers performed a feature addition operation to combine the above local and spatial features. Finally, they used the Atrous Spatial Pyramid Pooling or ASPP module to integrate the following discriminative features, which can more accurately model texture-aware interactions.

Similar neural networks can be built on top of ResNet. A Spatial Pyramid Attention (SPA) block was developed which allows the incorporation of structural information into the channel attention blocks to encode intermediate features. A modified Res2Net core network with the SPA block is designed to obtain better initial multi-level features [3].

ResNet uses all the layers you have created to extract features, or reuse them. Layers can be reused to create a new optimisation branch, the masking maps created by this branch. In addition, instead of directly selecting and providing an improved masking map for the optimised branch, taking full account of global and local information and use a similar aggregation approach with the Global Context-Aware Progressive Aggregation Network (GCPANet) to create the final masking maps [4].

BgNet reuses layers to create a second branch. The features extracted from the first branch of the main network are aggregated to infer the location of possible masked objects. The initial region and boundary prediction maps of the first branch are then used to refine the features of the second branch. The refined features are integrated by the second decoder to compute the final prediction maps [5].

In addition to ResNet, other neural network models such as VAN-small are used. The model performs a similar function and extracts a 5-layer convolutional neural network feature set from an RGB image. Comparing it with ResNet50 and Res2Net50 in the CAMO-Test and COD10K-Test shows that the model used achieves better results with fewer parameters [6].

The "VGG" model is used for extracting multi-depth features. Layers 1 and 2 of the VGG mainly extract texture, colour and boundary details, which are crucial for the detection of camouflaged objects. Therefore, researchers used features 1, 2, which were used earlier to represent the object. They used layer 3 as a bifurcation point. The routing flow shows that the communication extraction phase of a part of the object consists of layer 4 and layer 5 is used to integrate the possible communication of the part of the instance with the object. The route flow indicates that a concept reconstruction or feature key completion module consisting of a second layer 4 and 5 is used to modify the full description of the association relation [7].

In [8], researchers present Search and Identification Network System ("SINet"). A masked image is fed as input and the search module is first used to extract deep features to obtain a set of features. The extracted features are then sent to the identification module to obtain the masked maps provided by the network. Finally, the predicted masked object map is compared with the ground truth map in the training set to optimize the loss function and further obtain the optimal detection model.

The researchers developed the Minimum Boundary Contrast Detection (MBCD) model to use inputs as masked images that are first captured or sampled from the environment. This masked object uses a texture function to compute the exact resolution as input, observed from large-scale images or video. Then perform several operations, like thresholding and texture pixel placement, to require different texture feature extraction and pixel contrast identification for each template that is smaller than the original size [9].

This study focuses on the accuracy of using U-Net for detecting masked objects. Highest accuracy model parametres are found testing and comparing different U-Net modifications, loss functions, photo colors and sizes.

3. Research Data

The CAMO dataset is used to train the masked object detection algorithm and is chosen for its diversity of photos. CAMO is a dataset developed for the masked object segmentation problem [10, 11]. The masked images in the dataset are divided into two groups - naturally masked objects and artificially masked objects. The natural camouflaged photos consist of animals camouflaged in nature (forests, bushes, etc.). Artificially camouflaged photos are of people wearing camouflage clothing or painting their bodies to blend in with the background.

The dataset consists of 1,250 photos of camouflaged objects. The dataset is accompanied by a set of photographs where the masked object is defined in white on a black background. Such pictures allow to know where the answer is during the training of the algorithm.

The photos in the dataset are of different sizes and need to be standardised. In order to align the photos, a standart image size needs to be set. In image segmentation, the photos are converted into 32x32, 64x64, 96x96, 128x128, 256x256 wide and long pixel photos, for the speed and efficiency of the algorithm training.

Additionally, photos are modified by converting them from RGB to grayscale. RGB photos consist of 3 channels - red, green and blue. Converting them to grayscale changes the size of the channels from 3 to one. Such photos can give different results in the training process of the algorithm, because the complexity of the photo is reduced and the algorithm has to work with fewer features. However, removing channels can remove important features from the photo and the algorithm may lose accuracy.

Photos are also modified by converting their color spaces. Two color space conversions are used in this study – Haematoxylin-Eosin-Diaminobenzidine (HED) and Hue, saturation and value (HSV).

The dataset is split into test, training, and validation sets. In this study, the dataset is split into 70% training, 15% testing, and 15% validation. In addition, during the data set splitting, the images are randomly split using a random number sequence starting from the beginning.

4. Methodology

4.1. Convulutional neural network

Convolutional neural networks are a type of neural network designed to process network-like data. For this purpose, convolution is used - a function that processes data with the aim of giving the most accurate value. During convolution, the data is transformed according to the specified weights, thus we obtain the result. In order to obtain the most accurate result, it is necessary to determine what weights we will use in data processing. For this purpose, neural networks are used to calculate what weights need to be used to obtain the answer. First, the data is prepared for the neural network. Since the convolutional neural network is able to process data in the form of a network, it must be given data in the form of a matrix. In this case, the images are converted into matrices, the pixels in the image that have their own color receive a numerical value describing their color. After converting the data to a numerical value, we can start using neural networks to obtain the weight values.

4.2. U-Net model

U-Net is a convolutional neural network designed to handle a small number of training images and perform more accurate image segmentation [12]. The model has the ability to detect multiple different objects which is needed for camouflaged object that can be hidden between different types of objects. The U-Net consists of two pathways - contraction and expansion. The contraction path uses the 3x3 activation function of "ReLU" and the 2x2 reduction method of "Max-pool". The "ReLU" function converts the specified values to 0 if they are negative or equal to 0 and returns positive values unchanged.

"Max-pool" is a reduction method that divides a specified matrix into a specified number of parts and assigns the largest values in the old matrix to the parts in the new matrix. In this case, the given matrix

is a 2x2 matrix, so "Max-pool" divides the matrix into 4 parts, selects the largest values of the matrix and returns a 2x2 matrix.

Once the image is compressed, it is enlarged using the "Up-conv" enlargement method. "Up-conv" enlarges the matrix by repeating the same values according to a specified size. When a 2x2 size is specified, the rows and columns of the fed matrix are enlarged by a factor of two and close variables are assigned to the newly created locations. During expansion, to avoid data loss, copies, made during contraction, are used. Since the copies may not match in size, they are truncated.

Finally, after expanding the image, the number of filters needs to be reduced. For this purpose, a 1x1 convolution is used. Depending on the activation function, "ReLU" is used, taking the value of a given matrix, multiplying it by the number of filters and the number in the convolution matrix, and outputting an answer with fewer filters.

4.2.1. Base U-Net model

The U-Net model used for this study is inspired by [13]. The model consists of 4 deep layers. The model has 2 convolutional blocks in each deep layer in the contraction path. There are 3 convolutional blocks in the expansion path. The number of filters in the convolutional blocks differs from the standard model starting from 32 filters and reaching 256 filters. The "ReLU" 3x3 activation function is used during convolution. The "sigmoid" activation function is used at the output of the model. The "Max-pool" 2x2 downscaling method is used in the contraction path, and the "Upscale" 2x2 upscaling method is used in the expansion.

4.2.2. Residual U-Net model

Residual U-Net is a modification of U-Net using a residual block. The residual block creates transitions in the neural network. In a simple U-Net, the image must go through all the contraction and expansion stages. The residual block allows the algorithm to skip convolutional blocks, so the neural network can have a smaller number of parameters and protects the information propagation from degradation.

4.2.3. ASSP U-Net model

Atrous Spatial Pyramid Pooling or ASSP is a way to cover a larger number of information without increasing the number of parameters [14]. It uses dilated convolution. A dilation parameter is given by which the convolution expands from the starting point.

4.3. Loss functions

Loss functions are designed to evaluate the algorithm model and provide feedback on which parameters the model will change. This study uses different functions, monitors their results, and monitors the model's ability to detect a masked object.

4.3.1. Dice loss functions

Dice Loss function compares the similarity between two classes of an image - the object detected by the algorithm model and the real object. The loss function returns a numerical value between 0 and 1, and small values that result in better segmentation performance [15].

$$Dice = 1 - \frac{2|X \bigcap Y|}{|X| + |Y|}, \tag{1}$$

here X is the actual object, Y is the object detected by the algorithm model.

4.3.2. Binary Cross Entropy loss functions

Binary Cross Entropy loss function, also known as the logarithmic loss function, measures the difference between the labels of a dataset and the probability distribution predicted by the algorithm model. Binary Cross Entropy loss function outputs 0 or 1, true or false. The output values are then scaled according to their distance from the true value. Finally, the logarithmic function produces a negative mean and multiplies it by one for the negative value.

$$L_{binary} = -\frac{1}{N}\sum_{i=1}^{N}(y_i \log p_i + (1-y_i)\log(1-p_i)), \tag{2}$$

here N is the number of the object or class, y_i is the desired result, p_i is the probability of the algorithm model detecting the desired object or class. (1- p_i) is the probability of the algorithm detecting another class or object [16].

4.3.3. Weighted Binary Cross Entropy loss functions

Weighted Binary Cross Entropy loss function uses the formula for binary cross entropy, but adds weights to the variables. The weights are calculated from the number of true positive and negative classes divided by the total number of classes detected by the algorithm model [17]. The sum of the weights must sum to 1.

$$L_{wBCE} = -\frac{1}{N}\sum_{i=1}^{N}(w_1 y_i \log p_i + w_0(1-y_i)\log(1-p_i)), \tag{3}$$

here N is the number of the object or class, y_i is the desired result, p_i is the probability of the algorithm model to detect the desired object or class, w_1 is the first weight, w_0 is the zero weight.

4.4. Performance metrics

4.4.1. Dice similarity coefficient

Dice similarity coefficient, or "DSC", is a statistical indicator used to calculate the similarity between two objects.

$$DSC = \frac{2|X \bigcap Y|}{|X| + |Y|}, \tag{4}$$

here, X and Y are data sets. In the formula, the intersection of the X and Y sets is multiplied by 2 and divided by the sum of the X and Y data sets. A module is used so that the answer is not affected by negative values in the data sets. The formula is 2 because overlapping objects will result in twice as much as adding two objects.

4.4.2. IoU and Mean IoU

Intersection over Union (IoU) or “Jaccard’s Index” is a metric used to evaluate the performance of different detections in the testing phase [18]. IoU measures the ability of a model to distinguish an object from the background in an image. IoU calculates the overlap between two defined boundary windows – the one calculated by the algorithm model and the response. IoU returns a score between 0 and 1.

$$IoU = \frac{|X \bigcap Y|}{|X| \bigcup |Y|}, \tag{5}$$

here X and Y are the windows of the defined boundaries. The amount of overlap between the two boundary windows is found and the answer is divided by the combined X and Y boundary windows. This gives the overlap ratio.

4.4.3. AUC

Area Under the Curve (AUC) is the area under the Receiver operating characteristic (ROC) or Precision Recall (PR) curves, which is used as a quality indicator in binary classification. To calculate the AUC value, the Keras function package is used, which calculates the quality index using the Riemann sum [17]. The Riemann sum divides the area under the curve into small areas of different heights and sizes. These areas are then summed up, but this may leave unreachable areas under the curve that will not be included in the sum. For this reason, the function first takes into account the area under the curve. The height of the areas to be plotted against the ROC curve are determined by dividing the true positives by the result provided by the algorithm. The height of the areas plotted under the PR curve is determined by dividing the true positives by the true results. The resulting AUC quality indicator shall be in the numerical range 0 to 1.

5. Results

Research was conducted with the U-Net model. The optimizer used for training is Adam with a training rate of 0.001. The algorithm learns for 25 iterations, but is stopped after 5 iterations if the results did not change. The filters in the model are not changed, the blocks used for narrowing and expanding are 2x2 matrices. The model training was repeated if the training failed and empty model predictions were obtained. Dice, IoU and AUC performance metrics are used as results. The model is prepared and trained with the dataset. The resulting model is tested and finally several processed photos are shown. The photos presented in the study are randomly selected from tested images.

5.1. Study of different U-Net models

The impact of different U-Net models on the detection of camouflaged objects was studied. A binary cross function was used for the study. The images are in RGB format, 128x128 in width and length. The training runs for 25 iterations and was stopped if the result does not change within 5 iterations.

Table 1
Influence of U-Net types results

U-Net type	Dice performance, %	IoU performance, %	AUC performance, %
Base	42.22	43.13	84.71
Residual	42.25	43.13	83.85
ASSP	37.46	43.13	84.56
Residual + ASSP	33.55	43.13	82.97

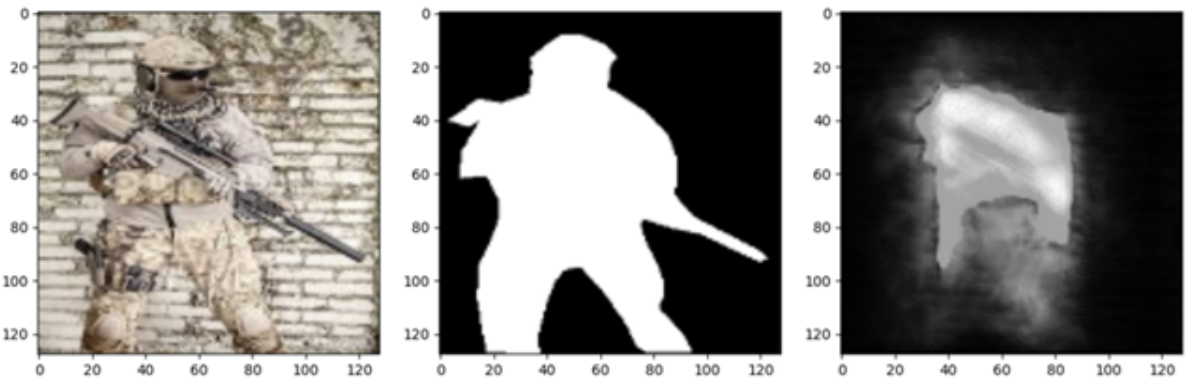

Figure 1: Photo prediction using Residual U-Net. From left - original photo, photo answer and model prediction.

According to the results (see Table 1), the base U-Net has the best detection performance, but the ASSP U-Net with a lower Dice rating than the base U-Net was able to detect branching objects. Figure 1 shows image prediction results using Residual U-Net and models capability to detect the object and seperate the lower body of the person.

5.2. Research of loss function impact

The impact of loss functions on the detection of a masked object was investigated. The base U-Net framework was used. The images used were 128x128 width and length, RGB format. Different loss functions were used during the study and the results compared.

Table 2
Impact of different loss function results

Loss functions	Dice performance, %	IoU performance, %	AUC performance, %
Dice	38.82	43.13	77.60
Binary Cross Entropy	42.22	43.13	84.71
Weighted Binary Cross Enropy	47.62	43.13	85.21

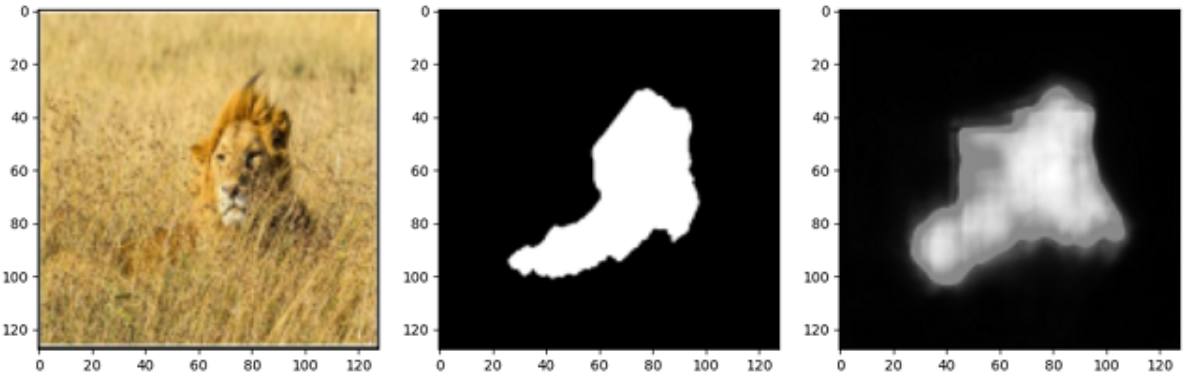

Figure 2: Photo prediction using Weighted Binary Cross Entropy loss function. From left - original photo, photo answer and model prediction.

The impact of loss functions on the detection of masked objects was investigated (see Table 2). The best results were achieved with the Weighted Binary Cross Entropy loss function. In Figure 2, the model predicted the lions body and head well. Furthermore, a square shape can be identified in the center of the prediction image which shows that the model learned to place it there for accuracy.

5.3. Research on image size, color channels and spatial conversion

The influence of image sizes, color channels and spatial conversion on masked object detection was investigated. The base U-Net architecture was used. The loss function used was binary cross entropy. The image size, color channels and spatial conversion, in the study were 128x128 pixels in width and length. RGB images were used in the study of image size.

RGB is the best choice compared to other image formats in terms of Dice and AUC scores (see Table 3). The HED photo format has the lowest scores. Figure 3 depicts HED format image prediction. The model predicted the location of the camouflaged object and a square which lowered its accuracy.

The study (see Table 4) shows that the Dice rating is highest with 96x96 photos, the IoU rating is highest with 32x32 photos and the AUC is highest with 128x128 photos. According to the results, the IoU estimation decreases as the size of the image increases. Furthermore, the AUC estimation increases with the size of the image, but decreases by 10.29% at the next image size of 256x256. Figure 4 shows that using 96x96 image size the model is capable to seperate two animals and their limbs.

Table 3
Influence of color channel and spacial conversion change results

Color channel and spacial conversion	Dice performance, %	IoU performance, %	AUC performance, %
RGB	42.22	43.13	84.71
Gray	35.26	43.13	82.73
HED	33.37	43.13	81.01
HSV	40.64	43.13	83.30

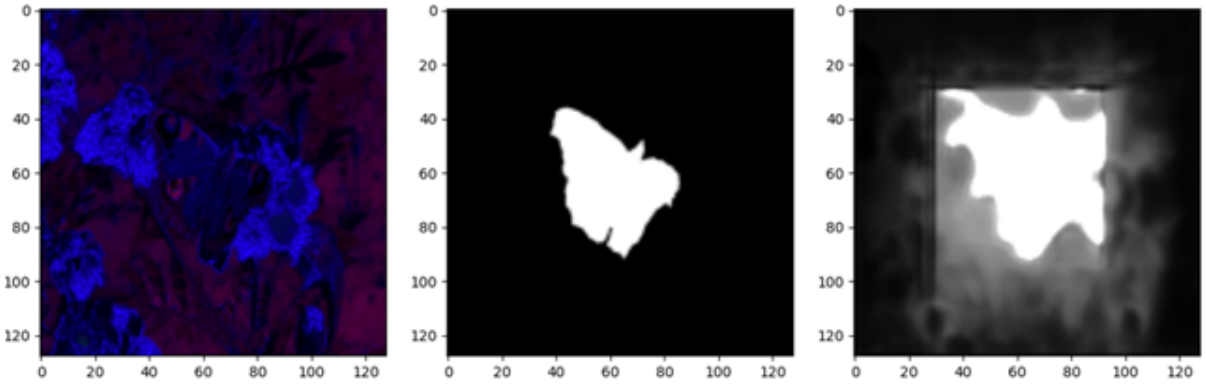

Figure 3: HED format photo prediction. From left - original photo, photo answer and model prediction.

Table 4
Influence of different image size results

Image size	Dice performance, %	IoU performance, %	AUC performance, %
32x32	42.13	47.09	80.26
64x64	37.70	44.81	81.52
96x96	46.61	43.75	82.22
128x128	42.22	43.13	84.71
256x256	31.07	42.13	74.42

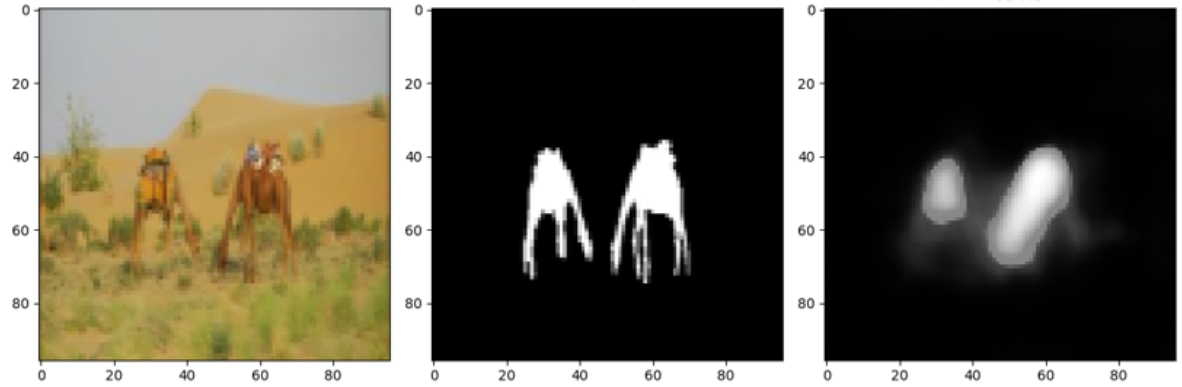

Figure 4: 96x96 size photo prediction. From left - original photo, photo answer and model prediction.

6. Conclusions

Different modifications of the U-Net model were used in the study, such as Residual and ASSP. The obtained estimates for all IoUs are 43.13% and the AUCs for all modifications are similar, as the highest (base U-Net - 84.71%) and the lowest AUC value (Residual + ASSP - 82.97%) differ by 1.74%. The Dice

estimation is the most different as the base U-Net (42.22%) and Residual (42.25%) are similar, ASSP (37.46%) is lower than them within 4.76-4.79% and Residual + ASSP (33.55%) is lower than them within 8.67-8.70%. However, during testing it was observed that the ASSP model was able to detect protrusions in objects when the Baseline and Residual U-Net were not able to do so. Additionally, IoU metric needs further testing to address the issue of showing the same result during testing except during image size test. The reason for this constant estimation can be caused by the performance of the function itself since the function used is taken from the Keras function set, the problem arises because without setting several IoU parameters the function itself tries to determine whether the given labels are variables of type integer or float.

The results obtained in the loss function test showed that the Weighted Binary Cross Entropy loss function obtained the best results (Dice - 47.62%, IoU - 43.13%, AUC - 85.21%) compared to Weighted Binary Cross Entropy and Dice. In the images detected by the model, it was observed that the Weighted Binary Cross Entropy loss function trained the model to create square shaped objects that do not define the object, but try to cover it. The Dice model tried to accurately define the masked object in the photo detection, but it was unaccurate at detecting branching objects.

The photo size study shows that a 96x96 photo has the highest Dice rating of 46.61%, a 32x32 photo has the highest IoU rating of 47.09% and a 128x128 photo has the highest AUC rating of 84.71%. In the color channel and spatial study, it was found that RGB format photos are better at detecting camouflaged objects as they got better Dice 42.22%, IoU 43.13% and AUC 84.71%.

Results from all of the tests show that model accuracy improves with correctly selected modifications of U-Net, loss functions, image size and color channel. The model accuracy can vary from 47.62% to 31.07% using Dice performance metric which reveals a 16.55% difference which is crucial in object detection. Future research can focus on the impact on accuracy of introducing images without a camouflaged object into the algorith. Moreover, real-time object detection can be investigated.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] J. Tong, Y. Bi, C. Zhang, H. Bi, Y. Yuan, Local to global purification strategy to realize collaborative camouflaged object detection, Computer Vision and Image Understanding 241 (2024) 103932. URL: https://www.sciencedirect.com/science/article/pii/S1077314224000134. doi:https://doi.org/10.1016/j.cviu.2024.103932.

[2] X. Li, H. Li, H. Zhou, M. Yu, D. Chen, S. Li, J. Zhang, Camouflaged object detection with counterfactual intervention, Neurocomputing 553 (2023) 126530. URL: https://www.sciencedirect.com/science/article/pii/S0925231223006537. doi:https://doi.org/10.1016/j.neucom.2023.126530.

[3] A. Wang, C. Ren, S. Zhao, S. Mu, Attention guided multi-level feature aggregation network for camouflaged object detection, Image and Vision Computing 144 (2024) 104953. URL: https://www.sciencedirect.com/science/article/pii/S0262885624000568. doi:https://doi.org/10.1016/j.imavis.2024.104953.

[4] C. Zhang, K. Wang, H. Bi, Z. Liu, L. Yang, Camouflaged object detection via Neighbor Connection and Hierarchical Information Transfer, Computer Vision and Image Understanding 221 (2022) 103450. URL: https://www.sciencedirect.com/science/article/pii/S1077314222000637. doi:https://doi.org/10.1016/j.cviu.2022.103450.

[5] T. Chen, J. Xiao, X. Hu, G. Zhang, S. Wang, Boundary-guided network for camouflaged object detection, Knowledge-Based Systems 248 (2022) 108901. URL: https://www.sciencedirect.com/science/article/pii/S0950705122004294. doi:https://doi.org/10.1016/j.knosys.2022.108901.

[6] Y. Liu, K. Zhang, Y. Zhao, H. Chen, Q. Liu, Bi-RRNet: Bi-level recurrent refinement network for camouflaged object detection, Pattern Recognition 139 (2023) 109514. URL: https://www.sciencedirect.com/science/article/pii/S0031320323002145. doi:https://doi.org/10.1016/j.patcog.2023.109514.

[7] M. Liu, X. Di, Extraordinary MHNet: Military high-level camouflage object detection network and dataset, Neurocomputing 549 (2023) 126466. URL: https://www.sciencedirect.com/science/article/pii/S0925231223005891. doi:https://doi.org/10.1016/j.neucom.2023.126466.

[8] Y. Liu, C.-q. Wang, Y.-j. Zhou, Camouflaged people detection based on a semi-supervised search identification network, Defence Technology 21 (2023) 176–183. URL: https://www.sciencedirect.com/science/article/pii/S2214914721001586. doi:https://doi.org/10.1016/j.dt.2021.09.004.

[9] Z. Xu, J. Wang, F. Hu, G. Abbas, E. Touti, M. Albekairi, O. I. El-Hamrawy, Improved camouflaged detection in the large-scale images and videos with minimum boundary contrast in detection technique, Expert Systems with Applications 249 (2024) 123558. URL: https://www.sciencedirect.com/science/article/pii/S0957417424004238. doi:https://doi.org/10.1016/j.eswa.2024.123558.

[10] T.-N. Le, T. V. Nguyen, Z. Nie, M.-T. Tran, A. Sugimoto, Anabranch Network for Camouflaged Object Segmentation, Journal of Computer Vision and Image Understanding 184 (2019) 45–56.

[11] J. Yan, T.-N. Le, K.-D. Nguyen, M.-T. Tran, T.-T. Do, T. V. Nguyen, MirrorNet: Bio-Inspired Camouflaged Object Segmentation, IEEE Access 9 (2021) 43290–43300.

[12] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional Networks for Biomedical Image Segmentation, in: N. Navab, J. Hornegger, W. M. Wells, A. F. Frangi (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Springer International Publishing, Cham, 2015, pp. 234–241.

[13] R. Augustauskas, A. Lipnickas, Improved Pixel-Level Pavement-Defect Segmentation Using a Deep Autoencoder, Sensors 20 (2020). URL: https://www.mdpi.com/1424-8220/20/9/2557. doi:10.3390/s20092557.

[14] F. Yu, V. Koltun, Multi-Scale Context Aggregation by Dilated Convolutions, 2016. URL: https://arxiv.org/abs/1511.07122, _eprint: 1511.07122.

[15] C. Mei, X. Yang, M. Zhou, S. Zhang, H. Chen, X. Yang, L. Wang, Semi-supervised image segmentation using a residual-driven mean teacher and an exponential Dice loss, Artificial Intelligence in Medicine 148 (2024) 102757. URL: https://www.sciencedirect.com/science/article/pii/S0933365723002713. doi:https://doi.org/10.1016/j.artmed.2023.102757.

[16] Q. Guo, C. Wang, D. Xiao, Q. Huang, A novel multi-label pest image classifier using the modified Swin Transformer and soft binary cross entropy loss, Engineering Applications of Artificial Intelligence 126 (2023) 107060. URL: https://www.sciencedirect.com/science/article/pii/S0952197623012447. doi:https://doi.org/10.1016/j.engappai.2023.107060.

[17] F. Chollet, others, Keras, 2015. URL: https://github.com/fchollet/keras, publisher: GitHub.

[18] D. Zhou, J. Fang, X. Song, C. Guan, J. Yin, Y. Dai, R. Yang, IoU Loss for 2D/3D Object Detection, in: 2019 International Conference on 3D Vision (3DV), 2019, pp. 85–94. doi:10.1109/3DV.2019.00019, iSSN: 2475-7888.