International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Small Object Detection in Remote Sensing Imagery Using Optimized Faster R-CNN

Princy Matlani[1*], Manish Shrivastava[2]

Submitted: 01/10/2024 **Revised:** 15/11/2024 **Accepted:** 25/11/2024

Abstract: Small object detection in remote sensing images is the procedure of recognising and integrating small, particular objects within an image taken by a remote sensing device, including satellite or aerial drone.An innovative hybrid deep learning based small object detectionmodel is introduced in this research work. The proposed model is divided into five main phases: (a) Pre-Processing (b) Segmentation (c) Feature Extraction (d) Feature Fusion (e) deep learning based small object detection. Initially, the collected raw image is pre-processed via gaussian filtering (for noise removal) and adaptive histogram equalization approach (for contrast enhancement). From the pre-processed image,the Region of Intertest (ROI) is identified viathe Fully Convolutional Networks (FCN). Then, from the identified ROI areas, the features like scale invariant feature transform, local binary patterns and Haar like features are extracted. The extracted features are fused using the score level fusion. From the fused features, the optimal features are selected using the new Dingo Optimization and Tunicate Swarm (DOTSA) algorithm, which is the conceptual amalgamation of the standard Tunicate Swarm optimizer (TSA) and standard Dingo Optimizer (DO), respectively. The small object detection is accomplished via the new optimized faster R-CNN, whose weight functions are optimized via the new Dingo Optimization and Tunicate Swarm (DOTSA) algorithm. The proposed model is implemented using the MATLAB platform.The findings are evaluated in terms of accuracy, sensitivity, precision, FPR, FNR, etc. using the present models. The proposed model has recorded the highest detection accuracyas 98%.

Keywords: *Small Object Detection;Remote Sensing Images; Fully Convolutional Networks (FCN); Dingo Optimization and Tunicate Swarm (DOTSA); Optimized Faster R-CNN.*

1. Introduction

Small object detection in remote sensing images is becoming increasingly important as the use of these images in real-world applications grows. The detection of planes and ships in optical images is a major application of small object detection in remote sensing. However, there are significant challenges in the remote sensing image processing process, such as small object size, complex backgrounds, and poor detection performance in noisy and low-resolution images. Object recognition in remote sensing images has many uses, such as environmental regulation, military, surveillance, and monitoring oil and gas activity. Researchers have explored various algorithms and deep convolutional neural network-based methods for object detection in remote sensing images, but these methods still face difficulties such as large input images, complex backgrounds, and poor performance in low-resolution images. Additionally, the cost of high-resolution imagery for large areas is a significant challenge for small object detection in remote sensing.Small object detection in remote sensing images is a significant contribution in the field of image analysis and interpretation.

It allows for the identification and analysis of small, often obscured objects in large, high-resolution images, which can aid in various applications such as urban planning, natural resource management, and disaster response.

This study's major contribution is exemplified below:

- To select the optimal features using the new Dingo Optimization and Tunicate Swarm (DOTSA) algorithm. This DOTSA model is the conceptual blend of standard Tunicate Swarm optimizer (TSA) and standard Dingo Optimizer (DO), respectively.
- To precisely identify the small objects using the new optimized faster R-CNN.
- To optimize the weight of faster R-CNN using the new Dingo Optimization and Tunicate Swarm (DOTSA) algorithm.

The remaining section of this research work is arranged as: Section II discusses about the literature work reviews regarding the subject and Section III presents the proposed mechanism and the DL algorithms employed in the work. Section IV presents and discusses the experimental results. Section V concludes this research.

*1*Department of Computer Science & Engineering, Guru Ghasidas University, Bilaspur, CG, India*

2Department of Computer Science & Engineering, Guru Ghasidas University, Bilaspur, CG, India

2. Literature Review

In 2019, Chen *et al.* [8] have proposed Multi-scale Spatial and Channel-wise Attention methodology that integrates spatial attention of different scales and contains spatial and channel-wise attention. MSCA improves the object region by paying explicit attention to it. The combination of MSCA greatly improves the models' anti-interference capability, allowing them to address the impact of the complex background and accomplish accurate detection.

In 2019, Pang *et al.* [9] have proposed are mote sensing region-based convolutional neural network, that was a unified and self-reinforcing network made up of Tiny-Net as backbone, a global attention intermediate block, and a final classification algorithm and detector. Tiny-Net was a lightweight residual framework that extracts features from inputs quickly and powerfully. Tiny-Net was used to build a global attention block that prevents false positives.

In 2019, Dong *et al.* [10] have presented an innovative HSRI object detection technique based on convolutional neural networks (CNNs) with appropriate object scale features. Initially, the appropriate ROI scale of object recognition was determined by gathering statistical data for the object scale range in HSRIs. CNN structure for object detection was developed and tested using a suitable ROI scale.

In 2021, Lu *et al.* [11] have proposed an end-end attention network and feature fusion SSD. Initially, a multilayer feature fusion framework was created to improve the semantic features of the shallow features. Following that, a dual-path attention module was presented to check the feature information. The network's feature representation capability is then improved further by a multi scale receptive field module.

3. Proposed Methodology

Small object detection is a computer vision task in which a model is trained to identify and locate small objects within an image or video. The goal of this research is to use a deep learning model to detect small objects. Small objects are harder to detect as they are smaller in size and can be easily occluded by other objects, which can make them difficult to locate. To overcome this issue, in this research work, a novel small object detection model is developed. Five main phases make up the projected model: (a) Pre-Processing (b) Segmentation (c) Feature Extraction (d) Feature Fusion (e) Deep learning based small object detection. Let the collected raw image be denoted as $D_j^{in}; j = 1,2, \ldots n$. Here, n denotes the count of image samples. Fig. 1 explains the overall architect's proposed model and includes five major phases.

Step 1: Pre-Processing: At the outset, the composed core image$D_j^{in}$is pre-processed through a gaussian filtering and adaptive histogram equalization approach. The pre-processed image acquired after pre-processing is denoted as D_j^{pre}.

Step 2: Segmentation: Then from the pre-processed imageD_j^{pre}, the region of interest (RoI) is identified by FCN.

Step 3: Feature Extraction: Subsequently, from the identified RoI segmented image,the features like scale invariant feature transform, local binary patterns and Haar like features are extracted. These extracted features are together denoted as F.

Step 4: Feature Fusion: Out of the extorted features F, the extracted features f are fused by score level fusion.

Step 5: Small Object Detection: The best features are chosen using the new Dingo Optimization and Tunicate Swarm (DOTSA) algorithm from the fused features. The new optimised faster R-CNN, whose weight functions are optimised by the new Dingo Optimization and Tunicate Swarm (DOTSA) algorithm, is used to detect small objects.

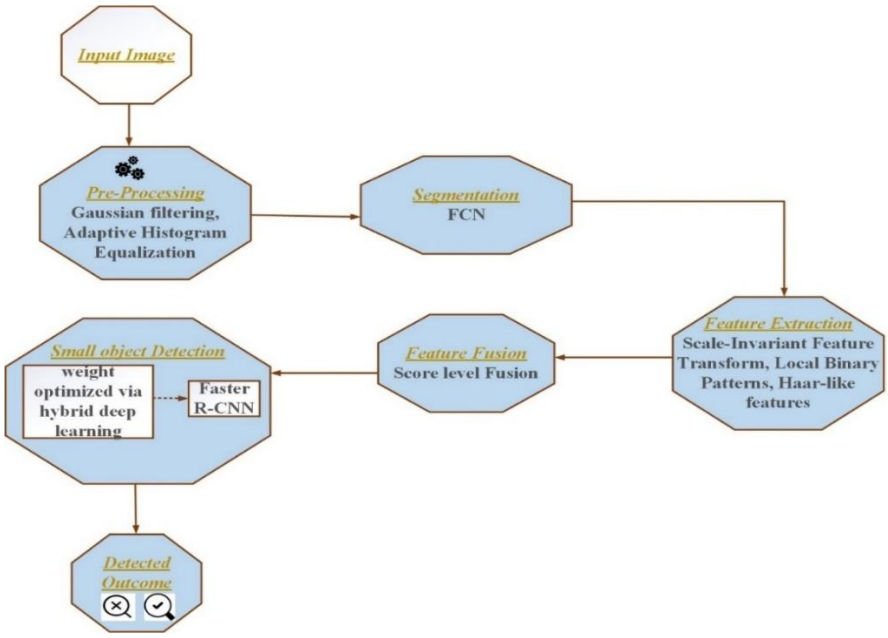

Figure 1: Overall architecture diagram

3.1 Preprocessing

In this research work, preprocessing is done using gaussian filtering and adaptive histogram equalization approach. refers to the techniques and methods used to prepare image for analysis or modeling. This can include tasks such as cleaning and formatting the image, reducing the dimensionality of the image, and normalizing or scaling the image. The goal of preprocessing is to improve the quality and suitability of the image or the intended analysis or model, and to make the image more amenable to the algorithms is used.

3.1.1 Gaussian Filtering

Initially, the collected raw image $D_j^{in}$is pre-processed via Gaussian filtering. Gaussian filtering is a smoothing technique used in image processing and computer vision to reduce noise and unwanted variations in an image. It is based on the Gaussian function, which is a bell-shaped curve that is used to weight the pixels in the image. The technique works by convolving the image with a kernel (a small matrix) that has the same shape as the Gaussian function. The kernel is then moved over the image, with the values at each position in the image being multiplied by the corresponding values in the kernel and the results being summed together. This process is repeated for every pixel in the image, resulting in a smoothed version of the original image.

The Gaussian Smoothing Operator performs a weighted average of surrounding pixels based on the Gaussian distribution. It is used to remove Gaussian noise and is a realistic model of defocused lens.

Gaussian filters are widely used in multi-resolution image processing. Multi-resolution techniques typically require convolution of the image with several Gaussian filters with increasing spread. For large values of the spread a finite impulse response implementation involves a large number of filter coefficients and hence a lot of calculations to compute the convolution. Pyramid techniques exploit the fact that a Gaussian function can be factorized into a convolution of Gaussian functions with smaller spread:

$$P(\sigma) = P(\sigma,) * P(\sigma_2) * \dots \dots * P(\sigma_M), \quad (1)$$

Where $\sigma_s \leq \sigma$ and $\sigma^2 = \sum_{s=1}^{M} \sigma_s^2$

The saving in calculations is achieved by using less filter coefficients for each $P(\sigma_s)$, since the spread σ_s is smaller than σ, and by subsampling the image after each intermediate convolution$P(\sigma_s)$, since each Gaussian filter$P(\sigma_s)$ reduces the bandwidth of the images. The noise free image is passed to adaptive histogram equalization.

3.1.2 Adaptive Histogram Equalization

The noise free image is pre-processed by adaptive histogram equalization. It is a digital image processing technique used to enhance the contrast of images. It differs from normal histogram equalization in the respect that the adaptive method enhances the contrast locally.By distributing the most frequent intensity values, it enhances the visual appeal of the image. AHE applies the transformation to various regions of the image adaptively, in contrast to standard histogram equalization, which applies a single global transformation to the entire image. This makes it possible to more precisely control the contrast, which is useful for images with uneven lighting or irregular backgrounds. The contrast enhanced image is fed as an input to segmentation phase.

3.2 Segmentation

In this research work, segmentation is done using FCN. Segmentation refers to the process of dividing an image or signal into multiple segments, or distinct regions. FCNs generate dense predictions for every pixel in an input image. Various uses for this include signal processing, object recognition, and image analysis. Edge detection, clustering, and thresholding are common segmentation methods. Image segmentation is a technique used in image processing to distinguish between objects and textures in images. In many image processing tasks, such as image editing, object recognition, and autonomous systems, it is a crucial step.

3.2.1 FCN

The contrast enhanced image is segmented by FCN. FCN for image segmentation cannot achieve the highest accuracy for the features of the FCN outputs are coarse features. After max-pooling layers, the main features can be retained while others discarded, so most of features cannot be used for image segmentation, then the edges of the object become blurred and some pixels are assimilated. Due to less max-pooling, the lower layers have finer features, then we should combine the high, coarse layers with law, fine layers to get the features of different levels and let the model make local predictions with respecting global structure that high coarse features are mainly used for what is the object which law, fine features are used for where is the object. The core component of a Fully Convolutional Network (FCN) is the convolutional layer, which is responsible for learning features from the input data. It consists of multiple feature maps processed by convolution kernels. Each kernel performs the convolution operation on its receptive field using shared weights, resulting in a reduction of the number of parameters and enabling the network to be deeper with fewer parameters. The formula for calculating the convolutional layers in an FCN is shown in Eq. (2),

$$conv(a,b) = r\left(\sum_{U=0}^{m-1} \sum_{V=0}^{m-1} W_{U,V} X_{a+U,b+V} + bi\right) \quad (2)$$

Where, $conv(a,b)$ defines convolution result also called feature map, m is the size of the kernel, $W_{U,V}$ is the weight of convolution kernel in line $U$and column V, $X_{a+U,b+V}$ defines the input, bi is bias and r is the activation function.One of the main advantages of using a Fully Convolutional Network (FCN) for image segmentation is its ability to preserve spatial information. The segmented image is fed as an input to the feature extraction phase.

3.3 Feature Extraction

In this research work, the features are extracted by scale invariant feature transform, local binary patterns and Haar like features. The process of locating and

removing pertinent data from raw image so it can be used in a machine learning model is known as feature extraction. In this preprocessing the feature extraction is Scale invariant feature transform, Local Binary Patterns and Haar like features.

3.3.1 Scale Invariant Feature Transform

A computer vision algorithm named Scale Invariant Feature Transform (SIFT) is used to recognize and describe local features in digital images. Scale-invariant key points are found in an image by SIFT, which then uses a feature descriptor to describe them. These key points and descriptors can be used to compare features in various images, enabling tasks like object recognition and image alignment are shown in Eq. (3).

$$h_s = TBD_s b + \eta = C_s b + \eta \quad (3)$$

Where h_s is the $M \times 1$ low resolution image, b is the $N \times 1$ original high-resolution image. T is the $M \times N$ sub sampling matrix, B is the $N \times N$ blur matrix reflecting the PSF, D_s is the image transformation matrix, η is noise. C_s $(M \times N)$ is the combination of matrix T, B, D_s.

3.3.2 Local Binary Pattern

A texture descriptor named Local Binary Patterns (LBP) is used in computer vision to describe an image's texture. It is predicated on the notion of contrasting a pixel's intensity with the intensity of the pixels around it. The LBP operator evaluates each surrounding pixel's intensity in relation to a central pixel's intensity. The corresponding bit in the LBP value is set to 1 if the surrounding pixel's intensity is greater than or equal to that of the central pixel, otherwise it is set to 0. As a result, a binary pattern is produced that can be used to describe an image's texture as shown in Eq. (4). LBP has a variety of uses, including image retrieval, object recognition, and face and face recognition.

$$LBP_{R,P} = \sum_{r=0}^{R-1} e(h_r - h_d)\, 2^R \qquad e(y) = \begin{cases} 1, if & y \geq 0; \\ 0, & otherwise \end{cases} \quad (4)$$

(R, P) defines the pixel neighbourhoods which means R sampling points on a circle of radius of P.

3.3.3 Haar like Features

The features, which represent the intensity of pixels in a specific area of an image, are based on the Haar wavelet. In order to identify edges, lines, and other features in an image, they compute the difference between the sum of the pixels in the white region and the sum of the pixels in the black region of a particular area. These features are frequently employed in applications for object and face detection. To compute sum of rectangular, arear in the image, at any position or scale, using only four lookups.

$$sum = J(E) + J(F) - J(G) - J(H) \quad (5)$$

Where the points E, F, G, H belong to the integral image J as per Eq. (5). The features are extracted and passed to feature fusion phase.

3.4 Feature Fusion

The extracted features are fed as an input to the fusion process. The process of combining various features extracted from an image to create a more detailed and reliable representation of the image is known as feature fusion in image processing. Edges and textures are examples of low-level features that can be used for this, as well as high-level features (e.g., objects, scene semantics). Enhancing the performance of image processing tasks like object recognition and image segmentation is the aim of feature fusion. For feature fusion, techniques like deep learning, PCA, LDA, etc. are used.

3.4.1 Score level Fusion

Score level fusion is a method used in the field of pattern recognition and computer vision to combine the extracted features. The combined extracted features are passed in to the small object detection phase.

3.5 Small object detection

The outcome of the combined extracted features is fed as an input to the detection phase. Finding and identifying small objects in an image or video is referred to as small object detection. Since small objects frequently have low resolution and might be hidden by other, larger objects in the scene, this can be difficult. Small object detection methods, object detection frameworks, and deep learning models like YOLO and ResNet are some of the techniques used.

3.5.1 Faster R-CNN

Faster R-CNN is a deep learning-based object detection algorithm. It is a widely used object detection algorithm that combines the power of Convolutional Neural Networks (CNN) with the speed of region-based proposals. The algorithm has two stages: The first stage is the region proposal network (RPN) which generates a set of region proposals, or potential bounding boxes for objects in the image. The second stage is the CNN, which takes the region proposals from the first stage and classifies them as object or background. It also refines the bounding boxes to be more precise. Faster R-CNN uses a shared convolutional layer for both the RPN and the CNN, which helps to speed up the process and reduce the number of computations required. This also makes the architecture more efficient and accurate. Faster R-CNN is widely used in various applications such as self-driving cars, drones, surveillance systems, and many other computers vision tasks. Its accuracy and efficiency make it a popular choice among researchers and practitioners alike.

By minimize an objective function following the multi-task loss in Fast R-CNN loss function for an image is defined as per Eq. (6)

$$LO(\{r_j\}, \{s_j\}) = \frac{1}{M_{cls}} \sum_j LO_{cls}(r_j, r_j^*) + \lambda \frac{1}{M_{reg}} \sum_j r_j^* LO_{reg}(s_j, s_j^*) \quad (6)$$

Where j is an anchor index in a mini batch and r_j is the anchor predicted probability and is the anchor is positive the ground truth label is r_j^* and is 0 when the anchor is negative.$LO_{cls}$is the log loss over two classes,

for positive anchors the regression loss is referred by means of the term $r_j^* LO_{reg}$.

3.5.3 Tuna Dingo Optimizer (TDO)

TDO model is the combination of the tuna swarm optimization and dingo optimizer. The encircling phase of the dingo phase is used within the tuna swarm optimizer. Tunicate has the ability to locate a marine food source. The location of the food source, however, is unknown in the search area. The tunicate uses two different behaviours to find the best source of food. Swarm intelligence and jet propulsion are two examples of these actions. A tunicate must satisfy three conditions in order to mathematically model the behaviour of jet propulsion: avoid conflicts between search agents; move toward the position of the best search agent; and maintain proximity to the best search agent. While the swarm behaviour will inform other search agents of the best optimal solution, updating their positions. The subsections before this one describes the mathematical modelling of these behaviours. In Tuna Dingo Optimizer, optimization is used to identify the ideal set of parameters for a machine learning model that reduces the loss function. This could enhance the model's performance and accuracy.

3.5.3.1 Avoiding the conflicts among search agents

To avoid the conflicts between search agents, vector $\vec{I}$is employed for the calculation of new search agent position.

$$\vec{I} = \frac{\vec{J}}{\vec{N}} \quad (7)$$

$$\vec{J} = i_2 + i_3 - \vec{M} \quad (8)$$

$$\vec{M} = 2.i_1 \quad (9)$$

However, $\vec{J}$is the gravity force and $\vec{M}$shows the water flow advection in deep ocean. Thevariables c_1,, i_2 and i_3 are random numbers lie in the range of $[0, 1]$. $\vec{N}$represents the social forces between search agents. The vector $\vec{N}$is calculated as per Eq. (10),

$$\vec{N} = \lfloor O_{min} + c_1.O_{max} - O_{max} \rfloor \quad (10)$$

where $O_{min}$and O_{max}represent the initial and subordinate speeds to make social interaction. In this work, the values of $O_{min}$and $O_{max}$are considered as 1 and 4, respectively.

3.5.3.2 Spiral Encircling Movement towards the direction of best neighbour (Proposed)

In TSA, the search agents move randomly reach the neighbours. Since the agents are moving randomly, they may not reach the optimal solution in a reasonable amount of time. Moreover, Random search methods can be computationally expensive because they may require a large number of iterations to find the optimal solution. Therefore, the solutions search agents are posed to move in a spiral path based on the dingo's optimizer. Encircling the optimal solution helps the algorithm avoid getting stuck in local optima, which is a common problem with many optimization algorithms. Encircling also assist to balance the trade-off between exploration and exploitation, by allowing the algorithm to explore a wide range of solutions while still focusing on the most promising areas. The agents (tunicates) move in the encircling phase in accordance with Eq. (11) to Eq. (15),

$$\overline{B_c} = |\bar{C}.\overline{D_e}(h) - \overline{D}(f)| \quad (11)$$

$$\overline{D}(f+1) = \overline{D_e}(f) - \bar{E}.\bar{B}(c) \quad (12)$$

$$\bar{C} = 2.\overline{g_1} \quad (13)$$

$$\bar{B} = 2\bar{d}.\overline{g_2} - \bar{d} \quad (14)$$

$$\bar{d} = 3 - \left(F * \left(\frac{3}{F_{maxi}}\right)\right) \quad (15)$$

The vectors g_1 and g_2 in this instance are random variable vectors in the range $[0, 1]$. These formulas use the vectors B and D to represent the distance and position, respectively. The prey is represented by the subscript e, while the tunicate s is represented by the subscript c. In this case, the best search agent is referred to as the prey, and all other search agents are referred to as the tunicates. The region of the solution space around the prey to which other tunicate converge is determined by vectors $\overline{CE}$. $\bar{E}$ Indicated by values less than -1 in the first case and greater than 1 in the second, it is used to determine whether the prey is fleeing from the search agent or is being pursued by the pack. For this class of biologically inspired algorithms, a common assumption during the hunting phase is that the pack members have a good sense of where the prey is located. The formulas' subscripts for the two top tunicate solutions in this phase are alpha and beta, with the other tunicate following them and updating their positions as necessary and shown in Eq. (16)-Eq.(21).

$$\bar{B}_\alpha = |\bar{C}_1.\overline{D}_\alpha - \bar{D}| \quad (16)$$

$$\bar{B}_\beta = |\bar{C}_2.\overline{D}_\alpha - \bar{D}| \quad (17)$$

$$\bar{B}_\circ = |\bar{C}_3.\overline{D}_\circ - \bar{D}| \quad (18)$$

$$\bar{D}_1 = |\overline{D}_\alpha.\bar{E} - \bar{B}_\alpha| \quad (19)$$

$$\bar{D}_2 = |\overline{D}_\alpha.\bar{E} - \bar{B}_\beta| \quad (20)$$

$$\bar{D}_3 = |\overline{D}_\circ.\bar{E} - \bar{B}_\circ| \quad (21)$$

Each tunicate 's intensities are determined using Eq. (22) to Eq. (24),

$$\bar{F}_\alpha = log\left(\frac{1}{G_\alpha - (1H-100)} + 1\right) \quad (22)$$

$$\bar{F}_\beta = log\left(\frac{1}{G_\beta - (1H-100)} + 1\right) \quad (23)$$

$$\bar{F}_\circ = log\left(\frac{1}{G_\circ - (1H-100)} + 1\right) \quad (24)$$

Tunicate es will start attacking the prey if the position is not updated, signalling the end of the hunt. A number of iterations cause a linear decrease in the value $\bar{d}$. The values of $\bar{B}_\alpha$ fall between $[-3d, 3d]$. As a result, as iterations go by, this range gets smaller and smaller until the tunicate es finally come to a stop.

3.5.3.3 Converge towards the best search agent

$$\overrightarrow{O_o}(k) = \begin{cases} \overrightarrow{SF} + \vec{I}.\overrightarrow{OP}, & if j_{and} \geq 0.5 \\ \overrightarrow{SF} - \vec{I}.\overrightarrow{OP}, & if j_{and} < 0.5 \end{cases} \quad (25)$$

As per Eq. (25), $\overrightarrow{O_o}(k')$ is a current position update for the tunicate in relation to the location of the food source $\overrightarrow{SF}$.

3.5.3.4 Swarm behaviour

The first two optimal best solutions are saved and update the positions of other search agents in accordance with the best search agents' positions in order to mathematically simulate the swarm behaviour of tunicate. The definition of tunicate swarm behaviour is shown in Eq. (26),

$$\overrightarrow{O_o}(k+1) = \frac{\overrightarrow{O_o}(k)+\overrightarrow{O_o}(k+1)}{2+i_1} \qquad (26)$$

4. Result and Discussion

4.1 Experimental Setup

To implement the proposed model, MATLAB has been used.Dataset1 [12] and Dataset2 [12] were used to gather the information for the evaluation. RCNN, ResNet, and InceptionNet are examples of existing algorithms that have been analyzed and compared to the performance of the suggested method. According to table I, varying the training percentage is used to process the performance analysis of the proposed model, and fig 2 shows the resulting graph representation. According to table I, all of the current models with varying training percentages are compared to the proposed model.

4.2 Overall Performance Analysis of the Proposed Model

4.2.1 NWPU VHR-10 Dataset

Table 1: Performance Analysis of the proposed model

Methods	TP	TN	FP	FN	Sen	Spec	Ac	Pre	Recall	F-M	NPV	FPR	FNR	MCC
Proposed ORCNN	107	108	2	3	0.97	0.98	0.97	0.98	0.97	0.97	0.97	0.01	0.02	0.95
RCNN	104	102	8	6	0.94	0.92	0.93	0.92	0.94	0.93	0.94	0.07	0.05	0.87
ResNet	101	99	11	9	0.91	0.9	0.90	0.90	0.91	0.90	0.91	0.1	0.08	0.81
InceptionNet	101	95	15	9	0.91	0.86	0.89	0.87	0.91	0.89	0.91	0.13	0.08	0.78

Table1 shows the NWPU VHR-10 dataset with the comparison of four different neural network architectures (RCNN, ResNet, and InceptionNet) for an image classification task yielded the following results. True positives (TP), true negatives (TN), false positives (FP), false negatives (FN), sensitivity or recall (Sen), specificity (Spec), accuracy (Acc), precision, recall, F-measure, negative predictive value (NPV), false positive rate (FPR), false negative rate (FNR), and Matthew's correlation coefficient are displayed in the columns (MCC). With the highest accuracy, precision, recall, F-measure, and MCC of the four models taken into consideration, the ORCNN may be the most effective one.

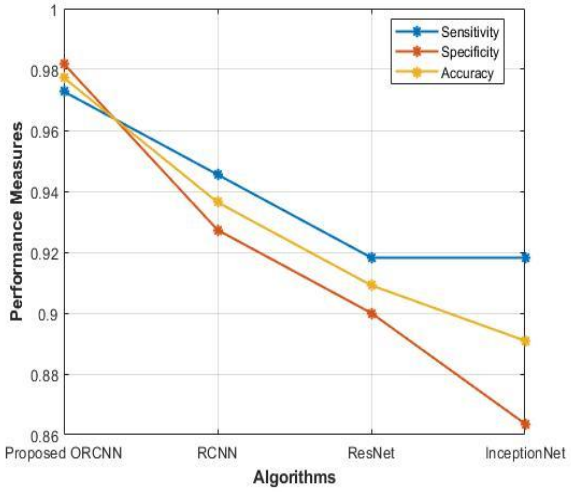

Figure 3: Overall performance analysis of graphical representation – Accuracy, Specificity, Sensitivity

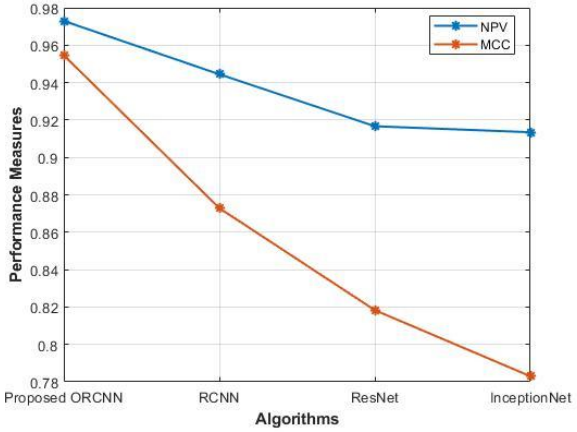

Figure 4: Overall performance analysis of graphical representation-NPV and MCC

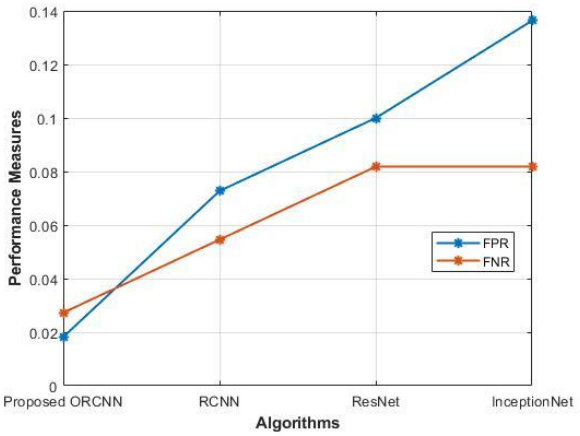

Figure 5: Overall performance analysis of graphical representation- FPR and FNR

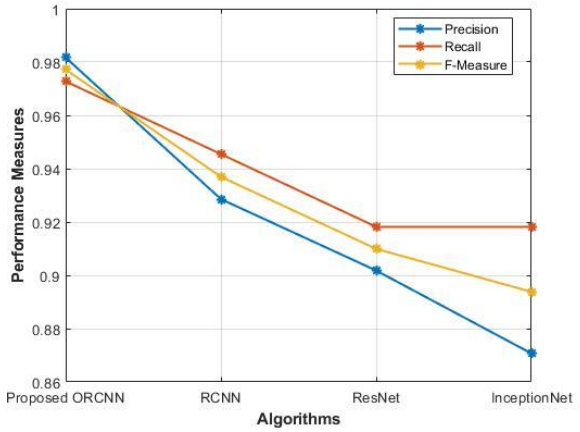

Figure 6: Overall performance analysis of Precision, Recall, F-Measure

4.2.2 SSDD Dataset

Table 2: Performance Analysis of the proposed model

Methods	TP	TN	FP	FN	Sen	Spec	Acc	Precision	Recall	Fmeasure	NPV	FPR	FNR	MCC
Proposed ORCNN	1277	1285	15	23	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.01	0.01	0.97
RCNN	1264	1272	28	36	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.02	0.02	0.95
ResNet	1251	1271	29	49	0.96	0.97	0.97	0.97	0.96	0.96	0.96	0.02	0.03	0.94
InceptionNet	1247	1236	64	53	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.04	0.04	0.91

The SSDD dataset has been used to assess the effectiveness of various image classification models. The performance of four different models—RCNN, ResNet, and Inception Net—is compared in the Table 2. The table displays various performance indicators for the models, including true positive (TP), true negative (TN), false positive (FP), false negative (FN), sensitivity (Sen), specificity (Spec), accuracy (Acc), precision, recall, F-measure, negative predictive value (NPV), false positive rate (FPR), false negative rate (FNR), and Matthew's correlation coefficient (MCC). According to the table, the Proposed ORCNN model has the best overall performance, scoring 0.98 for accuracy, 0.98 for precision, 0.98 for recall, and 0.98 for F-measure. The accuracy, precision, recall, and F-measure of the RCNN model are all slightly lower than those of the other models, at 0.97 0.97, 0.97, and 0.97 respectively. Even worse performance is displayed by the ResNet model, which has an accuracy of 0.97, precision of 0.97, recall of 0.96, and F-measure of 0.96. With accuracy, precision, recall, and F-measure all scoring at 0.95, the Inception Net model performs the worst.

Comparing the Proposed ORCNN model to the other three models, it appears that it performs the best on the SSDD dataset. However, it is crucial to note that these findings might not generalise to other datasets or real-world scenarios, and a model's performance should always be assessed on the particular task and dataset it was designed for.

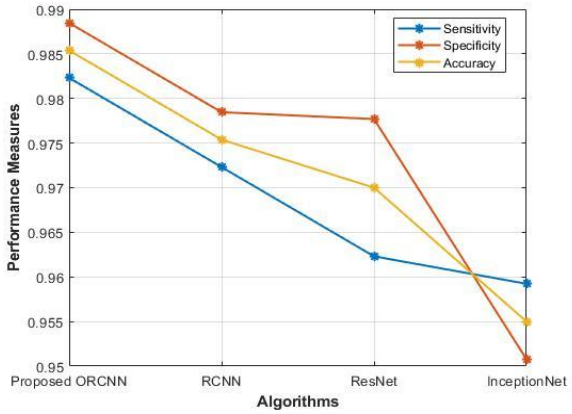

Figure 7: Overall performance analysis of Specificity,Sen and Accuracy

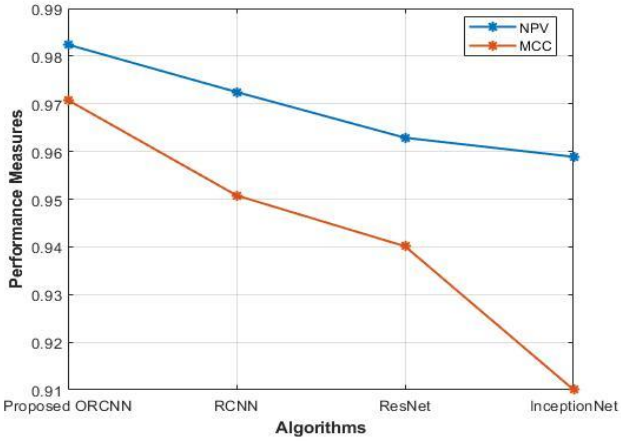

Figure 8: Overall performance analysis of NPV and MCC

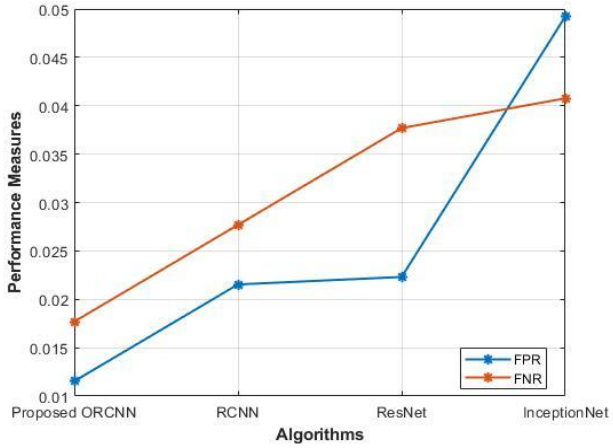

Figure 9: Overall performance analysis of FPR and FNR

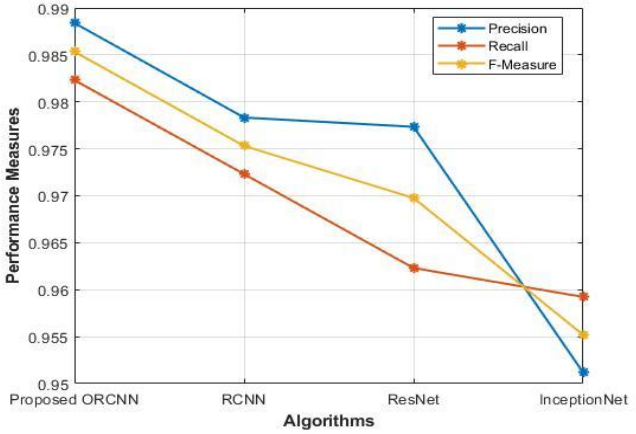

Figure 10: Overall performance analysis of Precision, Recall and F-Measure

(a) **(b)**

Fig 11 (a) and (b) sample detection of proposed model on NWPU VHR-10 and SSDD data set

Fig.11 (a) and Fig. 11 (b) show a small object detection in remote sensing imagery of the proposed model using NWPU VHR-10 dataset and SSDD dataset, respectively**.**

5. Conclusion

In this research work, an innovative hybrid deep learning small object detection model was developed for recognizing and locating small, specific objects within images taken by remote sensing devices such as satellites or aerial drones. The proposed model was divided into five main phases: (a) Pre-Processing (b) Segmentation (c) Feature Extraction (d) Feature Fusion (e) Small object detection. The collected raw image was pre-processed using a gaussian filtering and adaptive histogram equalization approach. From the pre-processed image, the images were segmented using Fully Convolutional Networks (FCN). Features were extracted using scale invariant feature transform, local binary patterns, and Haar-like features from the segmented image. The extracted features were fused using the approach called score-level fusion. The Dingo Optimization and Tunicate Swarm (DOTSA) algorithm was used for detecting small objects in remote sensing images. To enhance the prediction accuracy, the weight of faster R-CNN was fine-tuned by a hybrid optimization model. The proposed model was implemented using the MATLAB platform. The combination of the proposed techniques was successful in obtaining an accuracy rate of 98%. The findings were evaluated in terms of accuracy, sensitivity, precision, FPR, FNR, etc. using the present models.

Reference

[1] Wang, P., Sun, X., Diao, W. and Fu, K., 2019. FMSSD: Feature-merged single-shot detection for multi scale objects in large-scale remote sensing imagery. IEEE Transactions on Geo science and Remote Sensing, 58(5), pp.3377-3390.

[2] Lu, H., Li, H., Chen, L., Cheng, Y., Zhu, D., Li, Y., Lv, R., Chen, G., Su, X., Lang, L. and Li, Q., 2021. A ship detection and tracking algorithm for an airborne passive interferometric microwave sensor (PIMS). IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 14, pp.3519-3532.

[3] Sun, J., Zhang, Y., Wu, Z., Zhu, Y., Yin, X., Ding, Z., Wei, Z., Plaza, J. and Plaza, A., 2019. An efficient and scalable framework for processing remotely sensed big data in cloud computing environments. IEEE Transactions on Geoscience and Remote Sensing, 57(7), pp.4294-4308.

[4] Sun, X., Wang, B., Wang, Z., Li, H., Li, H. and Fu, K., 2021. Research progress on few-shot learning for remote sensing image interpretation. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 14, pp.2387-2402.

[5] Du, W., Addepalli, S. and Zhao, Y., 2019. The spatial resolution enhancement for a thermogram enabled by controlled subpixel movements. IEEE Transactions on Instrumentation and Measurement,69(6), pp.3566-3575.

[6] Huang, Z., Yang, S., Zhou, M., Li, Z., Gong, Z. and Chen, Y., 2022. Feature map distillation of thin nets for low-resolution object recognition. IEEE Transactions on Image Processing, 31, pp.1364-1379.

[7] Lei, J., Luo, X., Fang, L., Wang, M. and Gu, Y., 2020. Region-enhanced convolutional neural network for object detection in remote sensing images. IEEE Transactions on Geo science and Remote Sensing, 58(8), pp.5693-5702.

[8] Chen, J., Wan, L., Zhu, J., Xu, G. and Deng, M., 2019. Multi-scale spatial and channel-wise attention for improving object detection in remote sensing imagery. IEEE Geo science and Remote Sensing Letters, 17(4), pp.681-685.

[9] Pang, J., Li, C., Shi, J., Xu, Z. and Feng, H., 2019. $\mathcal {R}^ 2$-CNN: fast Tiny object detection in large-scale remote sensing images. IEEE Transactions on Geo science and Remote Sensing, 57(8), pp.5512-5524.

[10] Dong, Z., Wang, M., Wang, Y., Zhu, Y. and Zhang, Z., 2019. Object detection in high resolution remote sensing imagery based on convolutional neural networks with suitable object scale features. IEEE Transactions on Geo science and Remote Sensing, 58(3), pp.2104-2114.

[11] Lu, X., Ji, J., Xing, Z. and Miao, Q., 2021. Attention and feature fusion SSD for remote sensing object detection. IEEE Transactions on Instrumentation and Measurement, 70, pp.1-9.

[12] Dataset1 and Dataset2 collected from: "https://github.com/chaozhong2010/VHR-10_dataset_coco", 2023-01-20.