33RD CONGRESS
OF THE INTERNATIONAL COUNCIL
OF THE AERONAUTICAL SCIENCES
STOCKHOLM, SWEDEN, 4-9 SEPTEMBER, 2022

ICAS 2022 SWEDEN

A NOVEL METHOD FOR DETECTING UAVS USING PARALLEL NEURAL NETWORKS WITH RE-INFERENCE

Hubert Stepien[1], Martin Bilger[2], Håkan Forsberg[3], Billy Lindgren[4] & Johan Hjorth[3]

[1]Saab Aeronautics, Bröderna Ugglas gata, SE-581 88 Linköping, Sweden
[2]Saab Surveillance, Nettovägen 6, SE-175 88 Järfälla, Sweden
[3]Mälardalen University, Box 883, SE-721 23 Västerås, Sweden
[4] Saab Dynamics, Boforsvägen 1, SE-691 50 Karlskoga

Abstract

In this paper, we present a novel method for detecting UAVs using diverse parallel neural networks with re-inference. The parallel networks are of type Convolutional Neural Networks (CNNs). We first set up a low threshold (2 respectively 20%) for each of the individual networks to detect a flying object. If all networks detect a flying object in the same area of a video frame with some overlap, we zoom into that area and redo the object detection and classification (re-inference step). To ensure correctness and reliability of the results from several parallel CNNs, we introduce total confidence T_c as a measurement. We also introduce the intersection over union for multiple parallel networks, IoU^{All}, and use that as threshold for calculating a reliable T_c. The results show great improvements regarding accurate detection of flying drones, reduced mispredictions of other objects as drones, and fast response time when drones disappear from the scene.

Keywords: drones, detection, neural networks, re-inference, redundancy

1. Introduction

In recent years, several drones have been spotted close to government buildings, nuclear power plants and airports. Flying these drones nearby government properties or important infrastructures, whether it is malicious or not, constitute a high risk. It is therefore of importance to detect these air vehicles early, with high accuracy and integrity and then being able to track them successfully. Also, for vertical airports using drones as the main transportation vehicle, it would be of interest to detect incoming drones to warn eventual passengers or personnel being close to the landing spot.

In this paper, we suggest a novel method for detecting unmanned aerial vehicles (UAVs) using parallel convolutional neural networks (CNNs). CNNs are a specific type of Deep Neural Networks (DNNs). Our test system consists of three parallel and diverse CNNs, each with different properties, to detect UAVs to address said issue. The system utilizes re-inference by zooming into a region of interest to increase the detection capability.

The goal of using the method is twofold; 1) to reduce the mispredictions of detecting other flying objects as drones, and 2) to achieve higher accuracy in detecting UAVs compared to single CNN networks, under the assumption of real-time performance. Multiple methods for the detection of drones have been investigated by researchers. Investigated technologies include RADAR, acoustic, visual and radio frequency. Each of them has advantages as well as disadvantages depending on the selected scenario. Shi et al. [1] have explored the mentioned technologies and their strengths given the distance to the objects of interest. From this research, Shi et al. developed an anti-drone system consisting of an acoustic array, vision sensors, RF sensors, and an RF jamming unit. The idea was to use a single detector for a certain distance or environment, not to combine sensors at the same time. Taha and Shoufan [2] have explored several visual architectures for drone detection, again one by one, not combined. Other related research includes techniques for illegal drone detection [3] and comparisons of convolutional neural network models for anti-drone systems [4].

2. Background

When using CNNs for detection and classification of objects in the close airspace, it is important to detect as many flying objects as possible. If a flying object is not detected it is called a false negative (FN). If an object is detected but misclassified, it is called a false positive (FP). If the CNN correctly detects an object, it is called a true positive (TP). Figure 1 shows the relationship between the ground truth and predicted outputs. The numbers in the figure are for illustration purposes only.

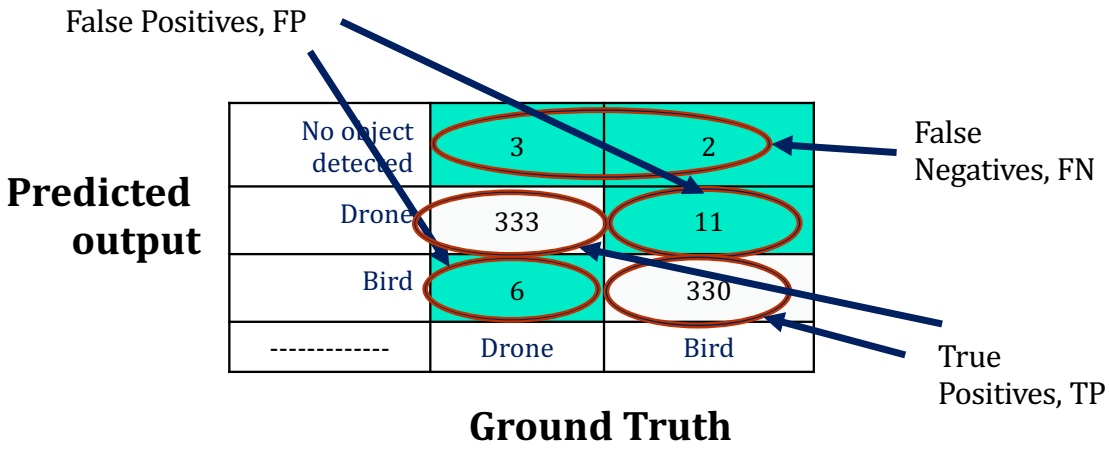

Figure 1 – The relationship between correct objects (ground truth) and predicted objects (predicted output). Correctly predicted outputs (333 drones and 330 birds in the figure above) are called true positives (TP). Incorrectly detected objects are called false positives (FP), and physical objects that are not detected are called false negatives (FN).

Most research today focuses on maximizing correctly classified objects. To correctly identify and classify objects is a necessity for a reliable system, but for highly dependable systems, it is of equal or higher importance to reduce FPs and FNs [5]. We believe diverse redundant systems are needed to cope with the above.

2.1 Network performance evaluation

Precision and recall are evaluation metrics typically used for performance evaluation of CNN networks. The *Precision*, see Equation 1, measures the percentage of correctly detected objects over all detected objects.

$$Precision = \frac{TP}{TP+FP} \tag{1}$$

Equation 2 shows the *Recall*, i.e., the percentage of correctly detected objects over the sum of correctly detected and undetected objects.

$$Recall = \frac{TP}{TP+FN} \tag{2}$$

Other measurements exist, e.g., Mean Average Precision [13] or Balanced F-score [14], but are not used in this article.

2.2 Accuracy of an object detector

When detecting and classifying objects in images, it is of interest to locate the object's position as well. Normally, a CNN detects an object near the ground truth object's location and size. An object's location and size is called a bounding box. The overlap region over a detected bounding box and the ground truth bounding box is called *Area of Intersection* (*AoI*), see Figure 2.

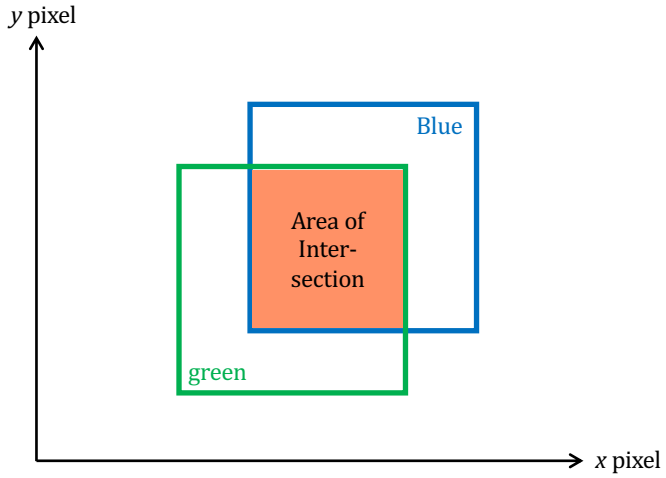

Figure 2 – Area of Intersection (AoI) is the overlap area between the detected object's bounding box (blue box) and the true (ground truth) bounding box (green box).

In the same way, it is possible to calculate the *Area of Union* (*AoU*), i.e., the union over the detected object's and the ground truth's bounding boxes, see Figure 3.

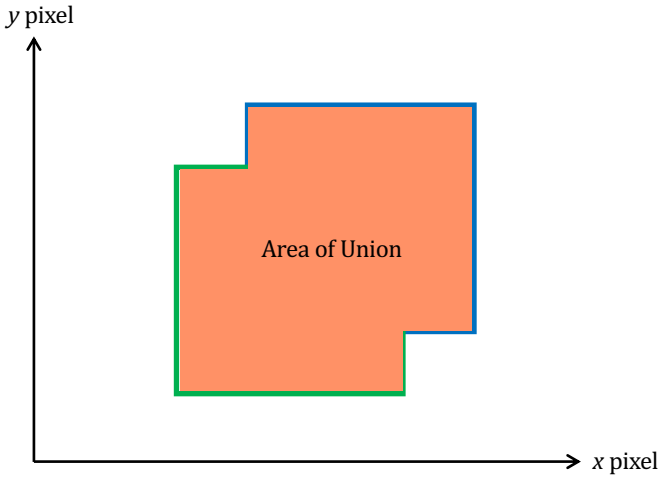

Figure 3 – Area of Union (AoU) is the union between the detected object's bounding box (the blue box to the upper right) and the true (ground truth) bounding box (green box to the lower left).

AoI and *AoU* are then used to calculate the *Intersection over Union* (*IoU*), see Equation 3.

$$IoU = \frac{AoI}{AoU} \tag{3}$$

The *IoU* is commonly used to compare the accuracy of different CNN object detectors on specific datasets. However, to perform the comparison, the ground truth's bounding boxes for every object in every image in the selected dataset must be present (typically added manually by humans).

2.3 Accuracy of parallel diverse object detectors

Since we use the strength of parallel and diverse CNNs to detect the same object, we need other measures. First, we introduce the intersection of the bonding boxes from multiple parallel diverse networks and call the intersection area for *Region of Interest (RoI)*. See Figure 4. We do not use the ground truth in the formula.

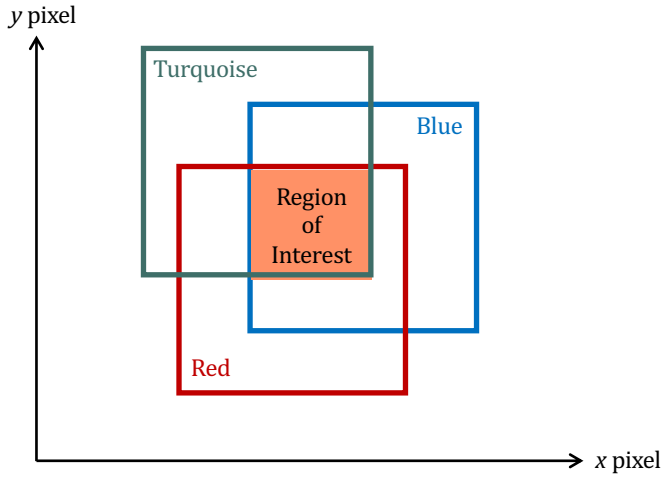

Figure 4 – The intersection of the parallel networks' bounding boxes (red, turquoise and blue in this example) form a Region of Interest (RoI).

In a similar way as *AoU* is calculated, we calculate the union of the output from all parallel networks and call the area for *Area of Union*^{All}, see Figure 5. We do not use the ground truth in the formula.

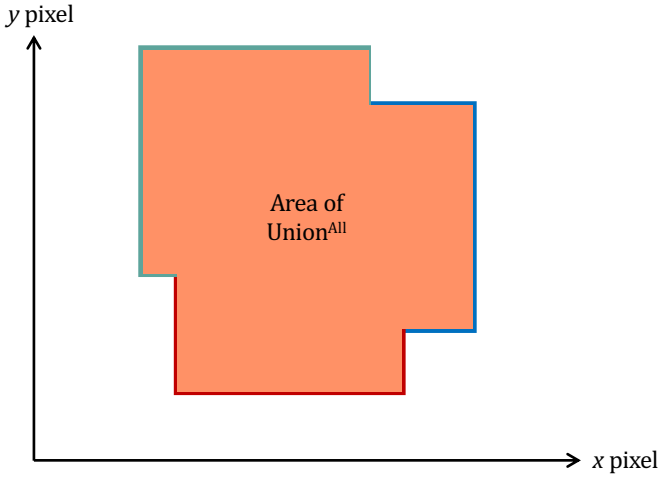

Figure 5 - The union of parallel networks' bounding boxes form the Area of Union^{All}.

We finally calculate the *IoU*^{All} as the Intersection over Union over multiple diverse networks' bounding boxes. See Equation 4. *IoU*^{All} may be used as an accuracy measurement for parallel networks but comparing different systems using different parallel networks may be difficult since the ground truth is removed from the equations.

$$IoU^{All} = \frac{RoI}{Area\ of\ Union^{All}} \tag{4}$$

2.4 Total Confidence over parallel networks

Another contribution in this paper concerns a new approach ensuring correctness and reliability of the results from several neural networks in parallel. We define total confidence (T_c) as a measurement, see Equation 5.

$$T_c = 1 - ((1 - C_{n1}) \times (1 - C_{n2}) \times (1 - C_{n3}) \times \ldots (1 - C_{nx})) \tag{5}$$

C_{n1} is the confidence of the first single network, i.e., the probability that the network believes it has detected an object at a certain location. $C_{n2}$is the confidence of the second network and so forth all the way up to the confidence of the *x*th network (C_{nx}.)

To be able to calculate a reliable T_c, the *IoU*^{All} for the identified object needs to be above 0.5. C_{n1-x} may be calculated in slightly different ways depending on the networks' architectures. The simplest

way is to use networks with entire predictive distributions, and not output of single point predictions.

3. Method

Our novel method consists of the following steps:

1. Let each parallel and diverse neural network (NN) detect flying objects from a frame. If a single network detects an object with a certain confidence C_{nx}, save its bounding box. (In our experiments we used an individual network confidence threshold value of 2% and 20% respectively.)

2. Use the intersection-over-union over all networks' bounding boxes (IoU^{All}) as an evaluation method to determine if the NNs detect an object at the same location. In this stage, it doesn't matter what kind of object the networks detect.

3. If the IoU^{All} exceeds a defined threshold, zoom into the region of interest (*RoI*, see Figure 4). (We use $IoU^{All} > 0.5$ as a threshold for our experiments.)

4. Use the zoomed in area to feed the parallel neural networks and let them detect the object again (re-inference step). Note, the zoomed in area should be slightly larger than a quadrant box containing the whole AoU^{All}. The size of the re-inferenced image needs to be large enough such that convolution can be performed on it. (We added 50 pixels on each of the four sides of the AoU^{All} in our experiments.)

5. By using each individual network's calculated confidence on the detected object in the zoomed in area, calculate a new T_c for the combined architecture's capability, and new data for
    a. correctly detect objects of interest (accuracy),
    b. minimize false detection of other objects as drones, and
    c. minimize undetected objects of interest

4. Test preparation

To test our method, careful preparations were made. The dataset used to train the three networks was initially created by Pawelczyk and WojTyra [6]. The original dataset includes other images than UAVs. To exclude unwanted objects, 8 800 images were carefully handpicked. The chosen images depict UAVs in the form of quadcopters in different scenarios with different backgrounds with shifting brightness, distortion and rotation. The images were selected to be as diverse as possible for improved detection capabilities. The images were split in two datasets, 8 000 for training, and 800 for validation purposes. Figure 6 shows some sample images used for training the network.

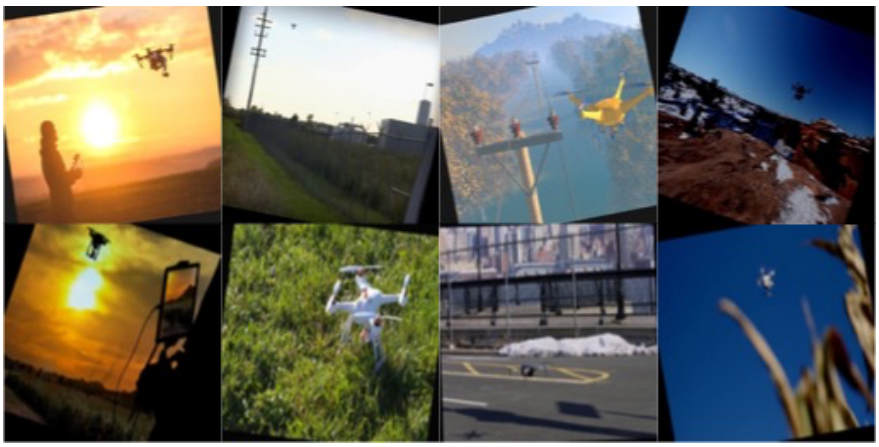

Figure 6 – Samples of images used for training the network.

4.1 Implemented neural networks

When we selected the three parallel and diverse networks, we did not do any deeper studies of network diversity such that they together perform as good as possible. The selected networks needed to be fast, efficient, and diverse, and be useful in real-time operations.

The selected neural networks are of type convolutional neural networks (CNNs). Their most outstanding features are described below. We used transfer learning on each of the networks to learn our handpicked dataset.

You Only Look Once V5s

YOLO V5s (and all previous versions of YOLO) is unique in its way the detector divides an image into MxM squares and performs detection on the individual squares. If an object of interest is detected in a square, a bounding box and a percentage reflecting confidence of the detected object is generated. The grid squares with high enough confidence are compiled into a heat map that uses labeled bounding boxes to perform a final detection [7]. See Figure 7.

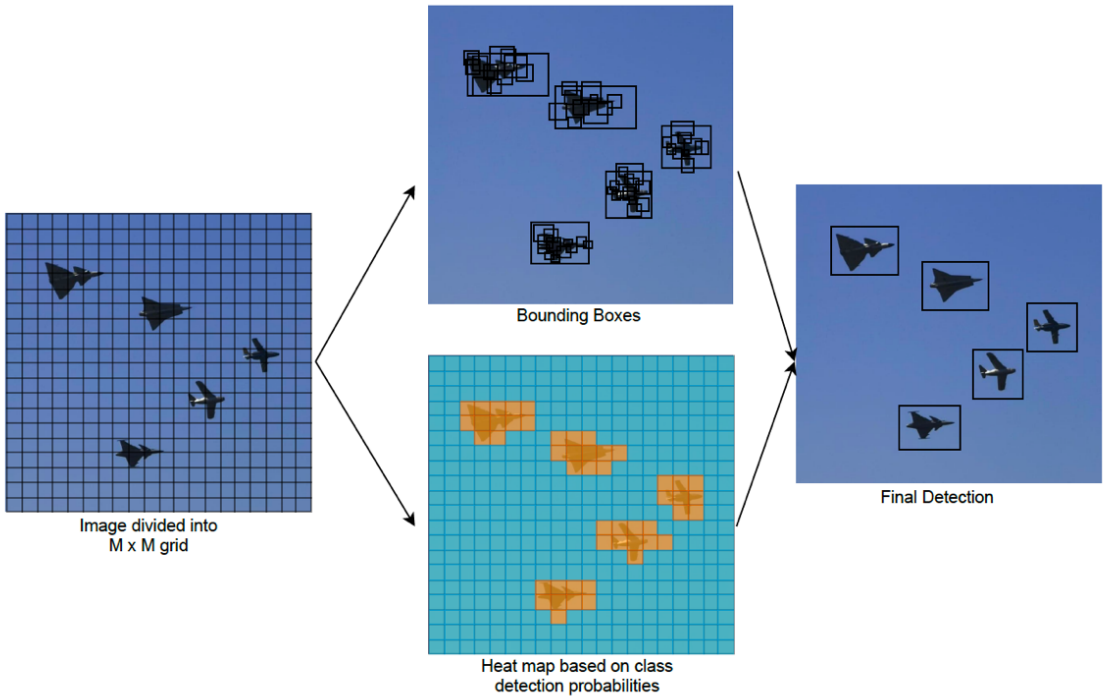

Figure 7 – Visualization of the different detection stages in YOLO.

MobileNet V2

MobileNet V1 and V2 are deep neural networks made for mobile platforms. MobileNets are based on depthwise separable convolutions which factorize a standard convolution into a depthwise and a pointwise convolution (*1x1* convolution). This factorization drastically reduces the computation and model size [8]. MobileNet V2 is based on the MobileNet V1 architecture but includes inverted residuals and inverted bottlenecks. Inverted residuals make the network truly unique in this implementation by allowing the network to connect the first convolutional layer block with the last fully connected layer. This allows the network to skip the layers in between thus saving computational resources by reducing the total amount of parameters that the network must learn and compute [9].

EfficientDet D1

EfficientDet D1's unique feature is a Bi-directional Feature Pyramid Network (BiFPN), which allows for simple and fast feature fusion, in combination with upscaling of the images fed into the network [10]. The BiFPN is based on the idea of PANets [11] which is based on the original Feature Pyramid Networks (FPNs) [12]. PANets take the idea of FPNs but add another bottom-up information path as

a compliment to the top-down information path. BiFPN removes nodes that only have a single input edge coming into them to save computational resources. Furthermore, an extra connection is made from the input node to the output node that allows for the preservation of already detected features in the backbone. To ensure better accuracy of the network, BiFPNs compute the same layer multiple times to allow for better detection of high-level features. Resolution for the D1 variant of the EfficientDet network is defined by the following:

$$Rinput = 640 + 1 * 128 = 768\ by\ 768\ pixels$$

By scaling the images in this way, the accuracy increases without increasing computational cost as compared to other scaling methods [10].

4.2 Test setup

The test environment was set up on Ubuntu Bionic Beaver operating system. The test software was created in the Python programming language version 3.8. Table 1 shows the test setup used in the experiments.

Table 1 – Test setup used in the experiments

Operating system	Processor	Graphics Unit	Memory
Ubuntu 18.04.5 LTS	Intel Xeon W2123 8 cores / 16 threads, 3.6 GHz	Nvidia Quadro P4000, 8 GB GDDR5 SDRAM, PCI-E 3.0 16x	32 GB DDR4, ECC, 3200 MHz

5. Results

After training, we tested our method by using a test video with quadcopters (QCs). The test video is completely independent from the training and validation dataset. When we selected the video sequence, we were particularly interested in frames where QCs enter from outside the borders as well as when they disappear out from the borders a shorter time and then enter the frames again. These situations are tricky to handle for certain neural networks.

5.1 Test results without re-inference

In the first test sequence, we explore the total confidence of the three parallel networks without using re-inference. We set the confidence threshold for each single network to 2%. That is, if each of the networks believe an object is a drone with only 2% probability, we then continue to check the IoU^{All} and if above 0.5, a positive has been found (includes both true and false positives).

In the test sequence, a drone flies from the right to the left in the video. The drone then disappears on the left side of the frames then enters the scene again (from left). In the end of the video the drone disappears on the right side of the frames and never enters again. The video sequence consists of 359 frames of which 184 contain a drone (true positives, TP) and then the rest of the frames (175) no flying object (true negatives, TN). Table 2 shows the results. Note that no false negatives (FNs) were identified in this test sequence, thus FN and Recall are not included in the table.

Table 2 – Test results without re-inference, C_{n1-3} set to 2% each.

Detection type	Yolo V5		MobileNet V2		EfficientDet		*Combined Network*	
TP	183	99.5%	182	98.9%	182	98.9%	184	100%
TN	112	64%	0	0%	0	0%	172	98.2%
FP	64	17.8%	177	49.3%	177	49.3%	3	0.84%
Precision	74.1%		50.7%		50.7%		98.4%	

The percentage of TPs is calculated by dividing the number of identified TP frames with the true total amount of TP frames. The percentage of TNs is calculated similarly, i.e., by dividing identified TNs with the total amount of negative frames.

When a network believes it has detected a drone, but it is something else, e.g. a building, it becomes a FP. In the case above, EfficientDet mispredicts a sign to be a drone in several frames when the drone is outside the scene, and MobileNet mispredicts several other objects to be a drone randomly in the frames when the drone is outside the scene. But even when a drone is present, a network sometimes detects other objects.

The percentage of FPs is calculated by dividing identified FPs with the total amount of frames in the video. Similarly, the percentage of FNs is the amount of identified FNs divided with the total amount of frames in the video. A FN is when a network doesn't detect a flying object but indeed there is one. The Precision and Recall are calculated according to Equations 1 and 2.

When calculating TP in the combined network, each network must first detect an object in the same area with a probability higher than 2% (C_{n1-3} are set to 2%). If the IoU^{All} is higher than 0.5, it is considered a TP. Since the combined network above detects all TPs, it must be a situation (for one or two frames) where one or two networks detect a FP in the same area as the other network/networks correctly detect a drone, such that IoU^{All} >0.5.

The only way the combined network above can detect a FP is when all three networks falsely detect a drone in the same region such that the IoU^{All} >0.5. In any other case, when the IoU^{All} < 0.5, no FP is detected. In the table above, three frames are classified as FPs. These frames are the ones not detected among the TNs.

Studying Table 2 there three two observations deserving attention; 1) MobileNet and EfficientDet seem to always detect flying objects even though there are no ones, 2) the networks altogether have detected false positives in the same location in three frames, and 3) the Precision is much higher for the combined network compared to each of the networks. Observation two is quite realistic since the threshold for detecting an object is set as low as 2% for each of the networks.

The IoU^{All} and the total confidence (T_c) are calculated for each frame and the results are visible in Figure 8. As can be seen, when there is a flying drone in a frame in the video, the T_c is higher than 0.8 and when the drone flies outside the scene (frames 94-225), the IoU^{All} becomes zero and the total confidence drastically changes (but is still high). The same thing happens in the end of the video when the drone flies outside the scene and never comes back again.

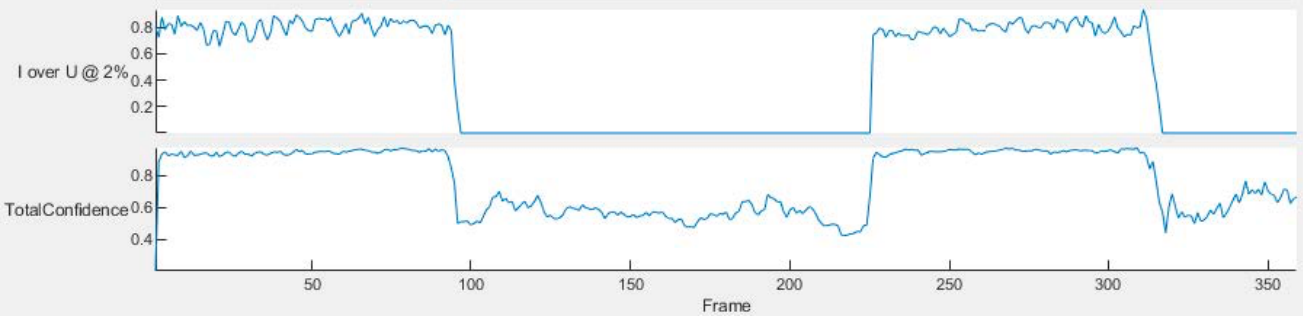

Figure 8 – The IoU^{All} and calculated total confidence (Tc) for each frame in the test sequence when confidence threshold for each single network is set to 2%.

Already in this first experiment (without re-inference), the strength of using multiple diverse networks is obvious. Figure 9 shows one example of this. EfficientDet and Yolo V5 both detect a drone with bounding boxes that are inaccurate (too high bounding boxes) but with high confidence (> 50%.) MobileNet, on the other hand, produces a much more accurate bounding box but to the expense of a lower confidence. Since the IoU^{All} is higher than 0.5, the total confidence for a positive can be trusted and in this frame (55 of 359 in the video sequence) T_c is 0.955.

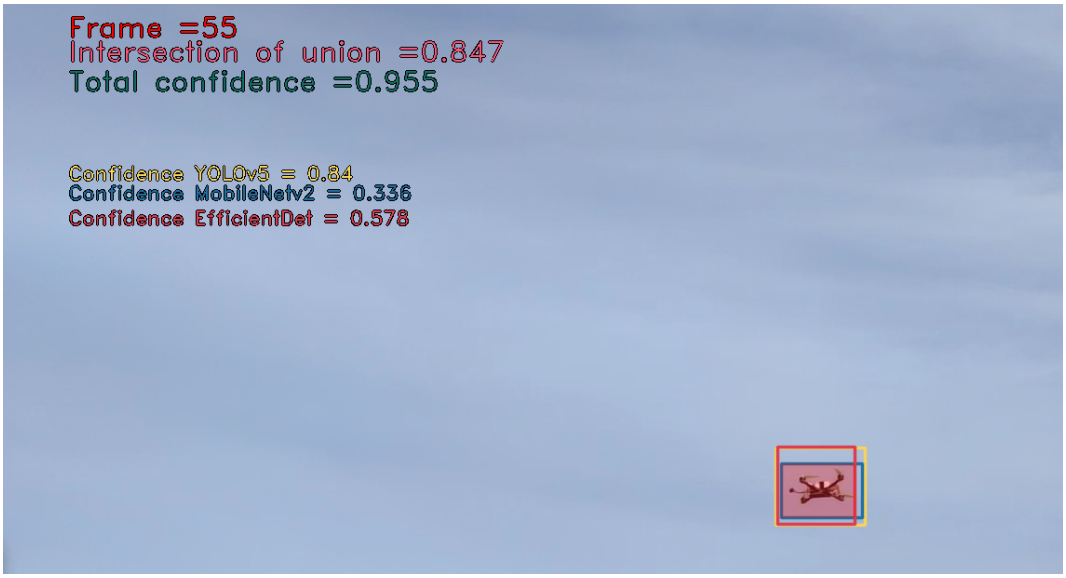

Figure 9 – Bounding boxes and confidences for the three parallel networks for frame 55 in the video sequence. IoU^{All} is 0.847 and T_c is 0.955.

The results from the first test show that the three neural networks together are much better in detecting non-threating objects, i.e., increased number of true negatives (TN), as well as in reducing the number of incorrectly detected objects (reduced number of false positives (FP)) compared to each of the single network. The accuracy, however, did not increase (0.5% increase is negligible.)

In the second test, we changed C_{nx} from 2% to 20% for each network (at the same time, not separately.) The results are shown in Table 3.

Table 3 - Test results without re-inference, C_{n1-3} set to 20% each.

Detection type	Yolo V5		MobileNet V2		EfficientDet		*Combined Network*	
TP	183	99.5%	170	92.4%	184	100%	184	100%
TN	167	95.4%	167	95.4%	0	0%	175	100%
FP	9	2.5%	4	1.1%	130	36.2%	0	0%
FN	0	0%	7	1.9%	0	0%	0	0%
Precision	95.3%		97.9%		58.6%		100%	
Recall	100%		96.0%		100%		100%	

Figure 10 shows the IoU^{All} and the T_c for each frame in the video.

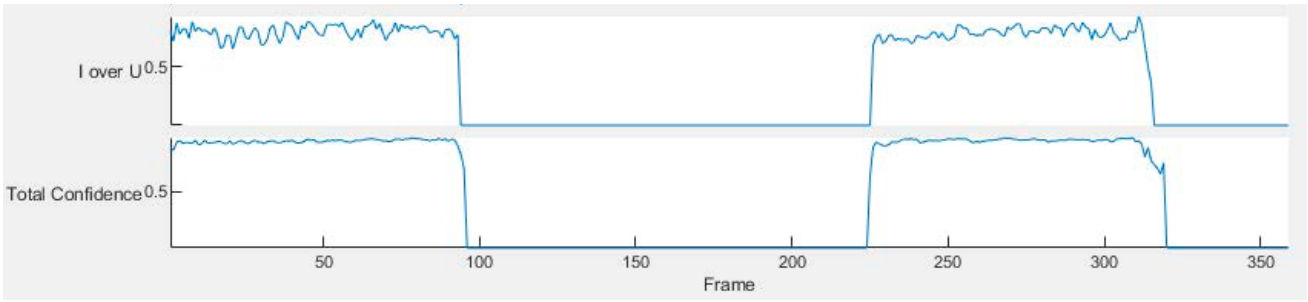

Figure 10 – The IoU^{All} and the total confidence (Tc) for each frame in the test sequence when confidence threshold for each single network was set to 20%.

The results show a marginal increase in accuracy with increased C_{nx} from 2 to 20% but the outcome indicates that by raising the confidence threshold of the networks, it is possible to faster detect a TN while reducing FP detections. In Figure 10 this can be seen when the T_c drops from a high level to

zero, which it does much steeper than it did in Figure 8, with $C_{n1\text{-}3}$ set to 2%. (It is a reduction from 4 to 1 frame when dropping from higher than 80% confidence to zero.) The *Precision* and *Recall* are both 100% for the combined network.

5.2 Test results with re-inference

In the following test sequence, we explore re-inference by first detecting a flying object in one frame. Then if the $IoU^{All} > 0.5$, we zoom into the area of interest (*AoI*) and let the CNNs detect if it is a drone in the zoomed in area (with individual network confidences as response).

Figure 11 shows the detected flying object from all three parallel networks in frame 75. The confidence for each network detecting the flying object is higher than 20% (which is our threshold value for the individual networks). The IoU^{All} is 0.837 which is higher than 0.5. Therefore, we zoom into the *AoI* and let the parallel neural networks detect drones in that zoomed in area (re-inference step). Without zooming into the *AoI*, the T_c is 96.4% for detecting a flying object. With re-inference using the *AoI* as the focus point, the total confidence for detecting a drone increases to 99.8% (see below).

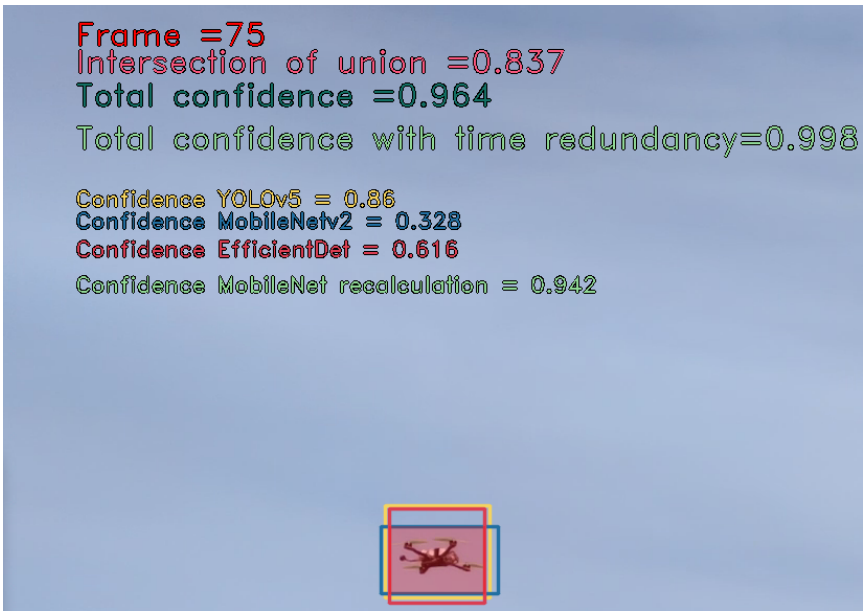

Figure 11 – The three parallel networks have detected a flying object in frame 75.

Figure 12 shows the bounding box for the MobileNet V2 network which has identified a drone within the zoomed in area. The confidence for MobileNet to detect it as a drone in the zoomed in area can be seen in Figure 11 (0.942).

Figure 12 – The MobileNet V2 network identifies a drone in the zoomed in area (AoU plus additional pixels). It identifies the drone with a confidence of 0.942.

By using the AoU plus some additional pixels on each side, a new image is created for the three neural networks to detect objects in. With this method, the confidence for detecting drones increases. Table 4 shows the re-inference confidence for each individual network, i.e., their probability that they have detected a drone in the zoomed in area in frame 75 in the video sequence.

Table 4 – re-inference confidence for the individual neural networks.

	Yolo V5	MobileNet V2	EfficientDet
Re-inference confidence for the zoomed in area in frame 75	74%	94.2%	88.8%

By using Equation 5, the total re-inference confidence for frame 75 is calculated to 99.8%.

$$T_{c,re-inference} = 1 - \big((1 - 0.74) \times (1 - 0.942) \times (1 - 0.888)\big) = 99.8\%$$

From the test results for frames 1 – 93, the $T_{c,re\text{-}inference}$ never falls below 99%.

Since the above re-inference method requires each of the three networks to detect objects twice per frame, we experimented with another approach using re-inference with reduced computational performance. We calculated a new T_c based on Equation (6).

$$T_{c,re-inference,new} = 1 - ((1 - C_{n1,orig}) \times (1 - C_{n2,orig}) \times (1 - C_{n3,orig}) \times (1 - C_{nx,re-inference})) \quad (6)$$

$C_{n1\text{-}3,orig}$ are the original confidences for each individual network detecting a drone in a frame. $C_{nx,re-inference}$ is the confidence for the best network of the three (x = 1, 2 or 3) of detecting a drone in the zoomed in area (re-inference step). The same rules apply as previous, i.e., each network's confidence must be higher than a certain threshold (20% in this test) and the IoU^{All} must be higher than 0.5. If that applies, only the network with best capability to detect drones in the zoomed in area is used in the re-inference step. The new confidence is then calculated for that network and this confidence is used in the calculations for the new total confidence according to Equation 6. Figure 13 shows the results for all frames in the video sequence using re-inference with only one network. In this case MobileNet V2. The figure shows each individual network's confidence in the original frame, the IoU^{All} and the T_c before re-inference, and the new total confidence after re-inference using Equation 6 for all frames in the video sequence.

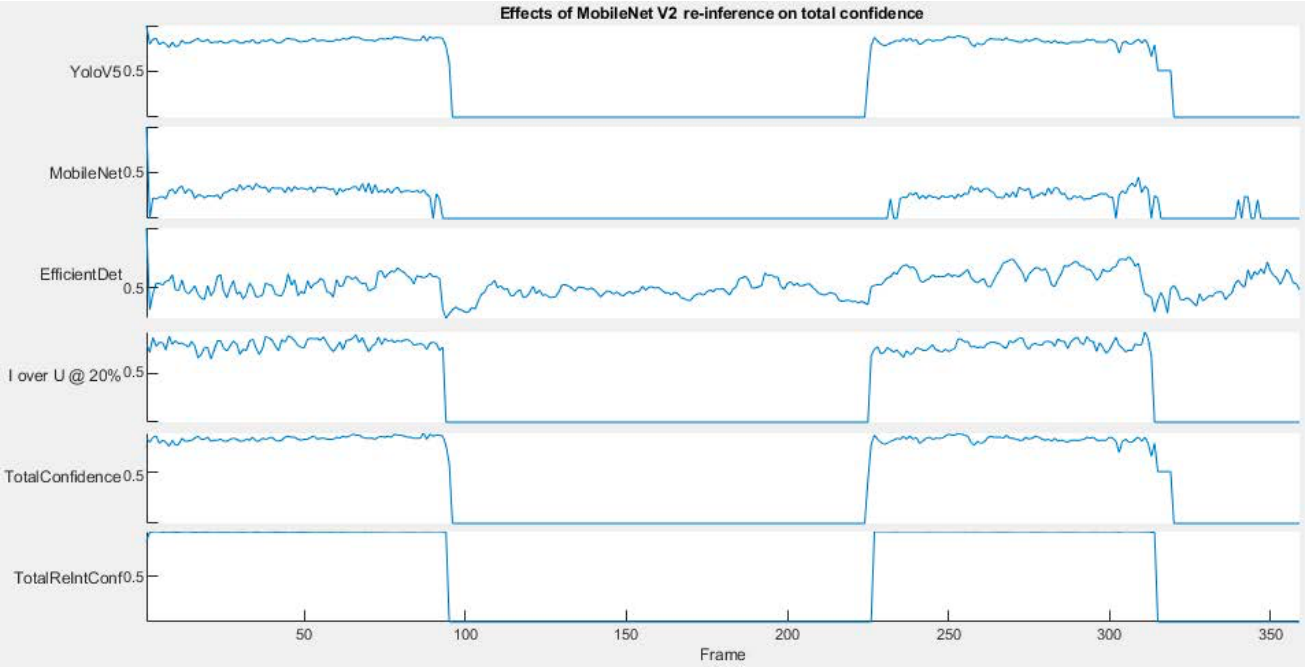

Figure 13 – Effects of using re-inference with one network only (MobileNet V2).

Table 5 shows the average total confidence using re-inference with only one network at a time. The average confidence is calculated on frames 1-93 in the video sequence.

Table 5 – Average re-inference confidence for detecting a drone using one network only in the re-inference step.

Network used in the re-inference step	Yolo V5s	MobileNet V2	EfficientDet
Average re-inference confidence using Equation 6 for frames 1-93	98.41%	99.50%	99.33%

The results from both methods of using re-inference (all three networks versus only one involved in the re-inference step) show improvements in total confidence and significant improvements compared to single network confidence for all frames.

6. Discussion

In this paper we introduced two new measurements to be used when multiple parallel neural networks work together to identify the same objects. The first one is the *IoU* over multiple diverse neural networks' bounding boxes, IoU^{All}, see Equation 4. The formula doesn't need the ground truth bounding boxes which is great but on the other hand it is harder to compare different constellations of diverse neural networks. The second measurement is the total confidence over multiple networks, T_c, see Equation 5. To be able to calculate a reliable T_c, the threshold for IoU^{All} needs to be 0.5 for three parallel networks. If more than three parallel networks are used, and if they are very dissimilar in the way they detect their objects and create their bounding boxes, the threshold for IoU^{All} may be set lower than 50% to be useful. We did not experiment with more than three parallel networks.

When we selected the three parallel and diverse convolutional neural networks (CNNs), we had to set single network accuracy aside in favor of performance. The selected networks needed to be fast, efficient, and diverse, and be useful in real-time operations. The last requirement forced us to select between few available CNNs. We did indeed investigate the different architectures of the networks but did not do any deeper study of which combination should be the best. For the purpose to validate our method, the three selected networks (MobileNet V2, EfficientDet, and YOLO V5s) did perform very well together. In some experiments, two of them produced less accurate bounding boxes but did detect the object of interest with high confidence, and the third network did the opposite, i.e., produced a more accurate bounding boxe but with lower confidence.

The test videos did not include simultaneous multiple objects of interest. If this would be the case, other neural networks and detection methods should be further investigated. The re-inference step would also be more complicated but still possible (frames may be buffered and reused for each object of interest). Although, the performance requirements would increase. Thus, the method of using only one network for the re-inference step would make more sense than using all three networks.

To further enhance the tracking of flying objects of interest, detection history can be used to calculate moving trajectories. We did not explore detection history in this paper.

7. Conclusion

In this paper we have introduced a novel method for detecting drones. The method relies on multiple diverse convolutional neural networks (CNNs) that together identifies objects of interest. We also introduced the Intersection over Union over multiple networks' bounding boxes, IoU^{All}, and the total confidence, T_c, over multiple networks as a measurement for the confidence in detecting the same object at the same location in a frame. By implementing diverse redundant CNNs, mispredictions during tests were heavily reduced compared to individual outputs from each single CNN. By using re-inference, average confidence over all frames in a video increased drastically. The detection time of objects of interest moving out from the scene and then back again was also improved. Finally, the

implemented re-inference was performed with two different algorithms. The first one used all parallel networks to recalculate the confidence of detecting an object of interest using a zoomed in area over the region of interest. The second algorithm used only one of the CNNs to detect an object of interest in a zoomed in area, and then combined the new confidence data with the previously detected T_c from all networks.

8. Acknowledgments

This work is partially supported by Vinnova within the project SafeDeep: Dependable Deep Learning for safety-Critical Airborne Embedded Systems and partially by the Swedish Knowledge Foundation within the project Dependable Platforms for Autonomous systems and Control.

9. Contact Author Email Address

mailto: hakan.forsberg@mdu.se

Mälardalen University, Box 883, SE-721 23 Västerås, Sweden, Telephone: +46 21101381

10. Copyright Statement

The authors confirm that they, and/or their company or organization, hold copyright on all of the original material included in this paper. The authors also confirm that they have obtained permission, from the copyright holder of any third party material included in this paper, to publish it as part of their paper. The authors confirm that they give permission, or have obtained permission from the copyright holder of this paper, for the publication and distribution of this paper as part of the ICAS proceedings or as individual off-prints from the proceedings.

References

[1] X. Shi, C. Yang, W. Xie, C. Liang, Z. Shi and J. Chen, “Anti-drone system with multiple surveillance technologies: Architecture, implementation, and challenges”, *IEEE Communications Magazine*, vol. 56, no. 4, pp. 68–74, April. 2018.

[2] B. Taha and A. Shoufan, “Machine learning-based drone detection and classification: State-of-the-art in research,” *IEEE access*, vol. 7, pp. 138669 - 138682, 2019.

[3] H. Chen, Z. Wang and L. Zhang, “Collaborative spectrum sensing for illegal drone detection: A deep learning-based image classification perspective,” in *China Communications*, vol. 17, no. 2, pp. 81-92, Feb. 2020.

[4] H. M. Oh, H. Lee and M. Y. Kim, “Comparing Convolutional Neural Network (CNN) models for machine learning-based drone and bird classification of anti-drone system,” *19th International Conference on Control, Automation and Systems (ICCAS)*, pp. 87-90, 2019.

[5] H. Forsberg, J. Lindén, J. Hjorth, T. Månefjord, and M. Daneshtalab, “Challenges in using neural networks in safety-critical applications,” *39th Digital Avionics Systems Conference*, DASC2020, Oct. 11-16, 2020.

[6] M. L. Pawelczyk and M. Wojtyra, “Real world object detection dataset for quadcopter unmanned aerial vehicle detection”, *IEEE Access*, vol. 8, pp. 174394–174409, Sep. 2020.

[7] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.

[8] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, 2017, *ArXiv, abs/1704.04861*.

[9] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. -C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510-4520, doi: 10.1109/CVPR.2018.00474.

[10] M. Tan, R. Pang and Q. V. Le, “EfficientDet: Scalable and Efficient Object Detection,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 10778-10787, doi: 10.1109/CVPR42600.2020.01079.

[11] S. Liu, L. Qi, H. Qin, J. Shi and J. Jia, “Path Aggregation Network for Instance Segmentation,” *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8759-8768, doi: 10.1109/CVPR.2018.00913.

[12] T. -Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie, “Feature Pyramid Networks for Object Detection,” *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 936-944, doi: 10.1109/CVPR.2017.106.

[13] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2009. doi: 10.1007/s11263-009-0275-4.

[14] Y. Sasaki, “The truth of the f-measure,” *Teach Tutor Mater*, pp. 5, Jan. 2007.