A Feasibility Study of Real-Time Image Processing Techniques for Small Flying Object Detection in Drones

Neil Loftus*, Cade Parlato*, Amelia McGinty*, Furkan Kizilay†, and Husnu S. Narman*
*Department of Computer Sciences and Electrical Engineering, Marshall University, USA
†Manisa Celal Bayar University, Turkey
Email: {loftus6, parlato2, mcginty2, narman}@marshall.edu, †190315027@ogr.cbu.edu.tr

***Abstract*—Drone usage is increasing significantly in our daily life, from military to delivery purposes. Although drones are also used to detect objects by using different techniques, they are limited to detecting flying small objects such as birds and responding quickly not to cause unintended collisions while flying at high speed. In this paper, we investigate the feasibility of using machine learning and image processing methods in drones while detecting birds mid-flight and responding to ensure their safety. This Real Time Bird Detection system (RTBD) is designed to detect birds so that proper response or evasive action can be taken by the drone. To avoid erroneous responses and observe the auto-behavior of drones while acting not to collide, we have developed an application with a graphical interface to easily control the drone's video feed and process that information using a machine-learning model. The application also has the capability to detect if a bird is close enough to interfere with the drone's flight path. Our test results show that the drone identified bird images within a 50-millisecond window of time, with Precision exceeding 96%, when Confidence exceeded 80%.**

***Index Terms*—Drones, Object Detection, Machine Learning.**

I. Introduction

The influx of drones and similar devices has drastically expanded the need for sophisticated small-craft aeronautics. Amazon and other logistics companies aspire and expect to use unmanned drones for more efficient and accessible deliveries. However, drone technology applications must address visual identification with subsequent drone response issues. Drones must be both compact and energy efficient, particularly for drones that maintain long-distance flight with considerable speed. Drone interaction with birds is an important scenario that accents these points. Unintended drone collisions with small clusters of birds and birds of prey mistaking drones for smaller birds represent two scenarios where both the drone could be damaged and the bird injured. These and other issues must be addressed before drones are used en masse, to avoid harm to wildlife, drones, and cargo.

There are other technological options, such as LiDAR and SONAR, employed for detecting aerial objects. However, these technologies require specialized sensor equipment. Image processing has the advantage of requiring only a video camera, which is often a standard feature on commercial drones. A trained video image processing system that implements machine learning techniques can precisely identify objects of concern and label them. There have been several studies that used machine learning and image processing to detect and differentiate birds from images, both on the ground and on drones [1] [2]. In these studies, image processing systems were fed sample images of birds and were trained to detect if an image or video contained any birds.

In many ways, object detection speed versus detection accuracy is the major challenge. This dichotomy is epitomized through the use of local or cloud computing with local calculation constrained by the onboard computer. More powerful hardware results in faster object detection speeds, however more powerful hardware may result in an increase in weight and power consumption that the drone can no longer support. This can be circumvented through the use of cloud computing [3]. Rather than designing the detection system to be completely on the drone, it may instead be operated remotely through a ground station. While this has the major advantage of allowing for the use of far more powerful computer hardware, it has several disadvantages. Primarily, cloud computing requires remote connectivity to the drone, and while in many cases the drone would already be connected to a ground station, this introduces an element of latency that causes decreases in object detection speed.

A balance can be obtained through the process of edge computing, which involves handling some information locally and some on the cloud, in order to minimize time delay due to latency and hardware strength [4], [5]. Therefore, a trade-off model that decides whether to make a decision on the cloud, edge, or drone should be implemented. The *aim* of this paper is to contribute to the field of aeronautic safety in small and micro-autonomous drones by examining real-time object detection.

The *objective* of this paper is to investigate the feasibility of using real-time Image processing to detect small flying

objects in small to micro drones. The key *contributions* of this paper are as follows: (i) The suitable model for bird detection in small and micro drones out of main object detection algorithms is determined, (ii) the dataset to train and test the determined bird detection model is created, and the model is analyzed under two test settings based on with and without Graphics Processing Unit (GPU), (iii) we further analyzed the explainability of the machine learning model while detecting birds by using Shapley Additive Explanations (SHAP) which is a well-known strategy in Explainable Artificial Intelligence (XAI), and (iv) finally, we developed the application with GUI to observe real-time prediction and behaviors of drones in real-time.

The rest of the paper is organized as follows: Section II summarizes the related work. In Section III, the machine learning model with analysis are explained. In Section IV, we discuss RTBD's graphical user interface (GUI) to control and display information. Finally, Section V has the conclusion and our plan for future work.

II. Related Works

There is a significant amount of research works on object detection with drones. In [6], the authors propose a novel method to improve object detection from drones, which face challenges such as tiny-scale objects, uneven distribution of objects, and environmental variations. The method consists of a global-local detection network, a self-adaptive region-selecting algorithm, and a local super-resolution network. In [7], the authors present a framework for evaluating the performance of drone-based object detection algorithms, considering various factors such as flight duration, camera resolution, and altitude. They show that the flying altitude and camera resolution have a significant impact on the accuracy of the object detection algorithm. In [8], the authors describe an object detection system using deep learning for drone delivery of medical aids. They use MobileNet and Single Shot Detector framework to detect and locate objects in video streams from drone and stereo cameras. In [9], the authors show an approach to detect and track objects for a drone using Convolutional Neural Network (CNN) in a Parrot AR Drone 2. In [10], the authors introduce Pareto Refocusing Detection (PRDet) to detect objects by focusing on the challenging regions that contain difficult objects, such as small or occluded ones. It uses a reverse-attention mechanism to find these regions and a region-specific context-learning module to enhance their features.

Due to You Only Look Once (YOLO)'s features such as compatibility and faster training time, many versions of YOLO were proposed [11], and their versions were also improved based on the usage scenarios such as drones [12]. In this paper, we use YOLOv7 and its versions due to its prediction speeds. Interested readers can learn more details about the new methods and their performances on object detection from [13], [14]

III. Machine Learning Model with Its Analysis

This Real Time Bird Detection system (RTBD) was trained using computer vision principles and practices sometimes identified as Objection Detection. During the bird detection process an image, usually a frame of a video, is analysed to label and find the position of each bird in the image. Object Detection is facilitated through the use of machine learning, specifically neural networks. The machine learning training focused models were executed via a process known as Supervised Learning. In supervised learning, the model is provided with a very large amount of labeled data during training, which is used to adjust internal prediction weights. After several iterations of training, the model is able to apply labels to unlabeled data, which is key for object detection [15]. In addition, existing published models provide checkpoints that our used as the basis for the RTBD, through a process known as transfer learning [16].

The model(s) employed for this RTBD system training included versions of Yolov7 and Yolov7-Tiny. This selection was primarily due to its speed of prediction compared to other models available at the time [17]. However, before training could occur, the training data sets needed were acquired using publicly available datasets from the machine learning website, Kaggle [18]. Two different datasets were used, one of bird images and a second data set of empty skies, which is used to decrease false positive predictions. As these datasets are unlabeled, each image used for training had to be labeled by hand. This was accomplished with a Python GUI application, labelimg [19]. After training the models, test data was used to determine the average time needed to identify each bird in the image. Additionally, two different models were trained with Yolov7-Tiny; one had negative images, one did not. In this case, negative images refer to sample data that is unlabeled and contains no objects to label. In addition, analysis was performed both with and without a graphics processing unit (GPU).

The model trained with Yolov7 took an average of 15.5 milliseconds (ms) per image with the GPU, and 226.8 ms without the GPU (only CPU prediction). In comparison, the model trained with Yolov7-Tiny (without negatives) as a base took an average of 5.0 ms per image with the GPU and 45.3 ms without the GPU. Similarly, the model trained with Yolov7-Tiny with negative images took an average of 5.1 ms with the GPU and 46.2 ms without. Although there was minimal difference between the two Yolov7-Tiny models, the model with Yolov7 took, on average, about 3 times as long with GPU and around 5 times as long without GPU. These results showed that Yolov7-Tiny performs faster for the benchmark dataset in both cases, but especially without GPU. These results are heavily dependent on the specific

hardware being used to run the application. For these tests, a computer with a GTX 1060 dedicated graphics card was used. Without dedicated graphics, the average time would be much longer. Fig. 1 illustrates average speed per image for the models, with and without GPU assistance.

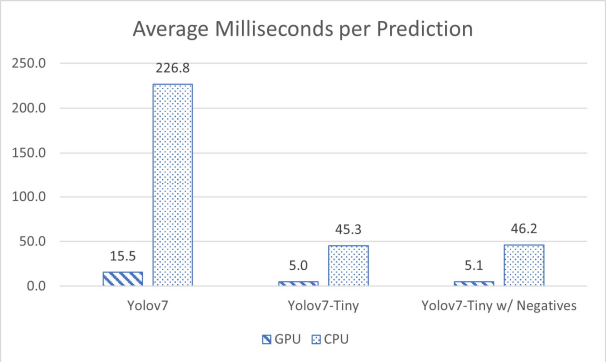

Fig. 1. Average speed per image for the models, with and without GPU assistance.

A. *Precision and Recall*

Several metrics were collected during the training and testing process; Precision, Recall, F1 score and Confidence. Precision (P), the percentage of positive predictions that were identified correctly, is calculated as the number of True Positives divided by the sum of the True and False Positives (TP) / (FP + TP). This ratio of values represents what percentage of predictions made by the model are actually what the model identified them as. Recall (R) is the percentage of labeled samples in the data that were properly identified. It is the percentage of objects identified that should have been identified and is the ratio of True Positives divided by True Positives plus False Negatives (TP) / (TP + FN) [20]. F1 score is a composite of these two attributes and serves as a more general evaluation of the model. F1 is calculated as twice the ratio of the product of the Precision and Recall values divided by the sum of those individual values. (2 * ((P*R)/(P+R)) [21]. Confidence refers to the percentage the model outputs to describe how certain it is in its prediction.

Respectively, Figs. 2 and 3 show Precision and F1 over prediction Confidence. These graphs indicate the accuracy of the models as they make more confident predictions. For the RTBD, low Confidence predictions are discarded, and these graphs only display predictions with a minimum Confidence of 80 %. Above 80% Confidence, the models resulted in 1.0 or 100% Recall for all values. This demonstrates there was no FN among the test data, meaning the model never missed birds that should have been detected. However, the same cannot be said about Precision, meaning FP are still present.

The RTBD test data represents limited conditions, and in an actual drone flight, False Negatives are more likely;

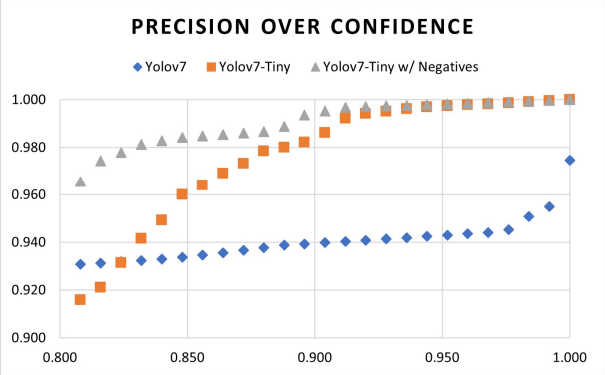

Fig. 2. Precision over Confidence, per model

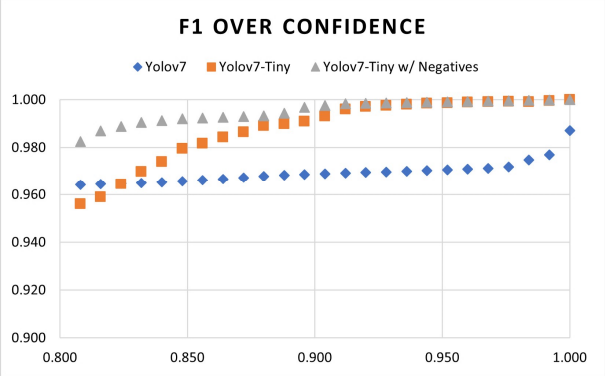

Fig. 3. F1 over Confidence, per model

however these ideal results still bode well, especially the Yolov7-Tiny employed model with negative images. Since Recall is constant, Precision serves as a better quantifying metric for expected RTBD behavior. Yolov7-Tiny with negative images had noticeably higher Precision values for all Confidence values. Among the test data, a Confidence of over 90% correlates with a Precision of 99% or higher. While Yolov7 had a precipitous increase in Confidence as its value approaches 90%, Yolov7-Tiny with negative images has higher Precision at 80% Confidence and quickly exceeds 98% Precision by the 85% Confidence level. This better Precision at lower Confidences is likely due to the inclusion of negative images, decreasing false positives.

The Yolov7 model produced the worst Precision and F1 values, this was likely due to less experimentation with training settings. Through a combination of fast detection time and high Precision, Yolov7-Tiny was selected for RTBD implementation. The base model, Yolov7, is capable of identifying numerous classes [17]. However, for this pilot demonstration, this model is trained to identify and respond to only 2 classes, “bird” and “flock”. Flock refers to a large collection of birds. This class was added to manage large numbers of birds. This option rarely came up during testing

and Figs. 2 and 3 show results specifically for the bird class.

B. External Analysis

Looking at the model more deeply, as it is trained with Yolov7, a large amount of information can be gleaned from existing publications. The Yolo family of machine learning models are primarily convolutional neural networks (CNNs), which have been proven as immensely successful in image based deep learning [11]. While the exact function has a number of additional complexities to it, the model is designed to detect specific features, which assist the model in making its prediction. However, in these cases, the features in a model are often abstracted. This is where black box model analysis tools can be useful. These tools are designed to analyze the function of the model by repeatedly testing it, as opposed to internal analysis. The chosen analysis tool was the SHAP (SHapley Additive exPlanations) explainer [22]. This tool allows us to load our model and test images to generate several explanatory graphs, with the primary graph we are interested in being the super pixel graph [23]. The super pixel graph shows which super pixels (or square sections of the image) have increased or decreased effect on the output of the model. The red / pink corresponds with relative greater effect and blue corresponds with lesser effect [24]. We ran several predictions through this analyser to get a better understanding of how the model functions.

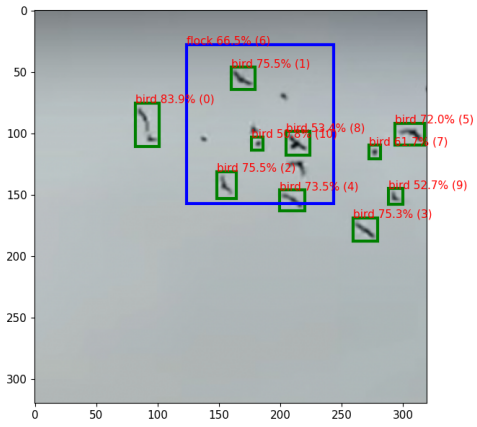

Fig. 4. Detection test with captured bird image

Fig. 4 shows the full frame analysis, with all detections, classes, and confidences. The sample we provided was one that we collected from a field test, and had considerable distance from the camera. This image serves as a difficult test image due to the distance, and this is represented in the overall low confidence from the model.

Fig. 5 shows a long distance classification of a single bird. The detection is almost entirely contained in 4 super pixels. The super pixel the model finds the most relevant is the red top left one, which contains the largest wing segment. Similarly, the bottom left contains a large portion of the bird

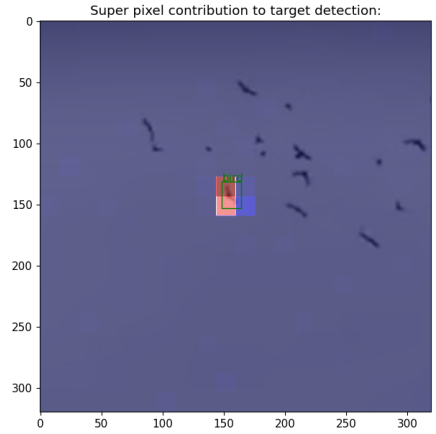

Fig. 5. Super pixel graph of first selected bird detection

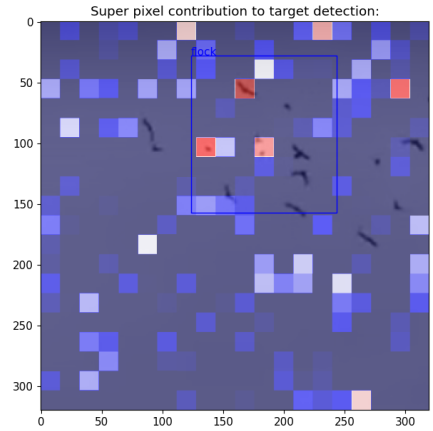

Fig. 6. Super pixel graph of flock detection

and is marked red. To the right of these two red pixels, we see two blue pixels, meaning the contents of those two pixels decreased the confidence in the pixels. We hypothesize that the model expects the right half of the bird to be located here, as opposed to the vertical position the bird is actually taking.

Looking at Fig. 6, which displays a flock detection, the segments that are highlighted red primarily contain smaller birds, as opposed to the larger birds, which the model associates less strongly with flocks. Around the image, we can see a number of blue pixels. The model only classifies this flock at 66.5%, which is below our stated standard of 80%. Most likely, this flock is sparse compared to the flocks in the training set, with the blue pixels representing spots that the model would expect birds to be present in for a flock. Many of the flock images in the training set occupied most of the frame, so it is possible that is a learnt feature that we may wish to correct.

We hypothesized that the model was primarily looking for “V” shaped objects that are relatively darker than the surrounding pixels in the frame. This is an apt description of

how a bird appears for a reasonable distance, so we believed this hypothesis to be likely. To test this, we created an abstract illustration to input into the model.

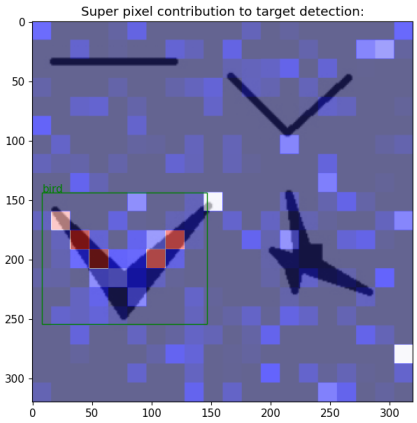

Fig. 7. Super pixel graph of the larger V shape

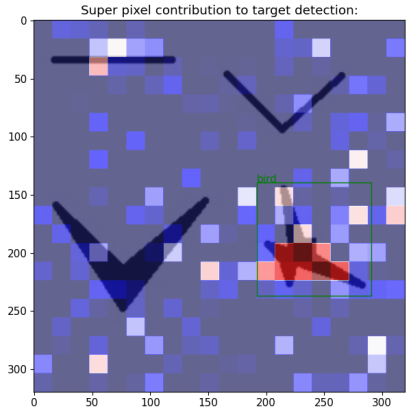

Fig. 8. Super pixel graph of the star shape

Figs. 7 and 8 show the model's detection for our illustration. This illustration contains a straight line, a V shape, a fuller V shape, and a star shape. While the straight line serves as a control, the other 3 shapes superficially resemble birds. As such, we expected them to be detected with some confidence. The straight line in the top left corner was not detected. This is good, as otherwise the straight lines present in other objects would cause false positives. The simple V shape in the top right has the second highest confidence at 96.6%. The fuller V shape in the bottom left has the highest confidence at 98.4%. Finally, the more complex star shape in the bottom right corner had the lowest, but still relatively high confidence of 94.2%.

In the bottom left corner of Fig. 7, is the fuller V shape. The model values the middle of the wings and the tips more than the center. This may be due to the V shape being thicker in the center than expected.

In the bottom right corner of Fig. 8 is the detection for the star shape. This prediction has a very large amount of red pixels located in the center and right wing, but far less in the left wing. It is possible the tail and the right wing make a V shape that is valued more. Fortunately, the bounding box still reflects the extent of the bird, so it did not cause the bird to only be partially detected. This figure has a number of red pixels interspersed among the frame. While it may be possible this is due to the model treating the absence as the lack of a flock, this is not seen in the other samples. As such, these pixels are likely erroneous and should be corrected.

IV. User Interface

The RTBD requires a mission focused graphical user interface (GUI). Based on user selection, this user-interface displays streamed flight information or the Artificial Intelligence (AI) model results and allows for selection of the source URL, AI model, and other options for input detection and stream output. A screenshot of the pilot RTBD user-interface, written in Python, with code for AI detection and stream options is presented in Fig. 9.

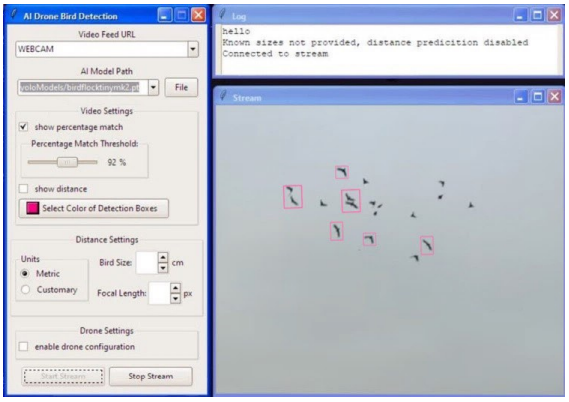

Fig. 9. A screenshot of our application in use

The GUI for the RTBD uses Python's standard User Interface (UI) framework, Tkinter. Although Tkiner generated GUIs could be considered outdated, it has many advantages. These include access to other Python libraries, support for multiple operating systems, and access to extensive support resources available on the internet. More development time is required to make the GUI visually appealing. Ways to improve its visual design include thoughtful spacing and placement of UI elements and creative widgets styling. The module "tkinter.ttk" is used to improve style the UI visual design. Although the theme built-in to the application is "clearlooks", it is easily changed to any other "ttk" theme without interfering with the application's function or rewriting the UI code.

The RTBD has the stream and the console log separate from the options and input window. The stop/start stream function and options/input elements are combined into one window display positioned at the top left of the screen.

When the stream is started, the window for the stream opens separately and scales in response to changes the window size. The console log, also resizable and scalable, is positioned below the stream. The stream options window, which is the main window and the one that allows for closing the program, is not resizable. The stream options include selecting a video URL for the stream to load, getting a live video feed from a Tello Drone, or from the computer's built in video camera. This can either be manually entered or chosen as a preset from a dropdown menu. Similarly, the machine learning model that the application uses can be selected from a number of presets. Additionally, any machine learning model trained with Yolov7 can be selected using a file system menu [17]. Since any model can be loaded into the RTBD, there are potential uses beyond bird detection. The GUI allows the user to configure settings for the video. It has a checkbox for whether percentage match values are shown on the boxes, and a scale to select the percentage match threshold (the minimum detected percentage to be displayed). Finally, the video settings include a button to select the color of detection boxes.

V. Conclusion

In this paper, we investigate the feasibility of using real-time Image processing to detect small flying objects in small to micro drones. Through the use of YOLOv7, we were able to develop an object detection system for drone cameras that allows to control it with a graphical user interface. Our tests showed that as our selected model for the RTBD exceeded 80% confidence, the precision exceeded 96% and the F1 exceeded 98%. Moreover, we investigate and try to explain why models detect or does not detect certain shape types by using Shapley Additive Explanations.

For future research, we wish to focus on Object distance estimation and on-board detection with edge detection to find a balance between accuracy of predictions and response time.

Acknowledgements

The authors thank everyone who provided their feedback during the writing process. The authors also thank NSF: Project Works Studio and the NASA West Virginia Space Grant Consortium to support this research.

References

[1] S. Fujii, K. Akita, and N. Ukita, "Distant bird detection for safe drone flight and its dataset," in *2021 17th International Conference on Machine Vision and Applications (MVA)*, 2021, pp. 1–5.

[2] A. Coluccia, A. Fascista, A. Schumann, L. Sommer, A. Dimou, D. Zarpalas, M. Méndez, D. de la Iglesia, I. González, J.-P. Mercier, G. Gagné, A. Mitra, and S. Rajashekar, "Drone vs. bird detection: Deep learning algorithms and results from a grand challenge," *Sensors*, vol. 21, no. 8, 2021.

[3] R. Buyya, S. N. Srirama, G. Casale, R. Calheiros, Y. Simmhan, B. Varghese, E. Gelenbe, B. Javadi, L. M. Vaquero, M. A. Netto *et al.*, "A manifesto for future generation cloud computing: Research directions for the next decade," *ACM computing surveys (CSUR)*, vol. 51, no. 5, pp. 1–38, 2018.

[4] C. Long, Y. Cao, T. Jiang, and Q. Zhang, "Edge computing framework for cooperative video processing in multimedia iot systems," *IEEE Transactions on Multimedia*, vol. 20, no. 5, pp. 1126–1139, 2017.

[5] R. Zhu, L. Liu, H. Song, and M. Ma, "Multi-access edge computing enabled internet of things: Advances and novel applications," *Neural Computing and Applications*, vol. 32, no. 19, p. 15313–15316, 2020.

[6] S. Deng, S. Li, K. Xie, W. Song, X. Liao, A. Hao, and H. Qin, "A global-local self-adaptive network for drone-view object detection," *IEEE Transactions on Image Processing*, vol. 30, pp. 1556–1569, 2021.

[7] P. Petrides, C. Kyrkou, P. Kolios, T. Theocharides, and C. Panayiotou, "Towards a holistic performance evaluation framework for drone-based object detection," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, 2017, pp. 1785–1793.

[8] W. Budiharto, A. A. S. Gunawan, J. S. Suroso, A. Chowanda, A. Patrik, and G. Utama, "Fast object detection for quadcopter drone using deep learning," in *International Conference on Computer and Communication Systems (ICCCS)*, 2018, pp. 192–195.

[9] A. Rohan, M. Rabah, and S.-H. Kim, "Convolutional neural network-based real-time object detection and tracking for parrot ar drone 2," *IEEE Access*, vol. 7, pp. 69 575–69 584, 2019.

[10] J. Leng, M. Mo, Y. Zhou, C. Gao, W. Li, and X. Gao, "Pareto refocusing for drone-view object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 3, pp. 1320–1334, 2023.

[11] J. Terven and D. Cordova-Esparza, "A comprehensive review of yolo: From yolov1 and beyond," 2023.

[12] O. Sahin and S. Ozer, "Yolodrone: Improved yolo architecture for object detection in drone images," in *2021 44th International Conference on Telecommunications and Signal Processing (TSP)*, 2021, pp. 361–365.

[13] "Object detection on coco test-dev," accessed:May 31, 2023. [Online]. Available: https://paperswithcode.com/sota/object-detection-on-coco

[14] P. Mittal, R. Singh, and A. Sharma, "Deep learning-based object detection in low-altitude uav datasets: A survey," *Image and Vision Computing*, vol. 104, p. 104046, 2020.

[15] S. V. Mahadevkar, B. Khemani, S. Patil, K. Kotecha, D. R. Vora, A. Abraham, and L. A. Gabralla, "A review on machine learning styles in computer vision—techniques and future directions," *IEEE Access*, vol. 10, pp. 107 293–107 329, 2022.

[16] Y. Quan, Z. Li, C. Zhang, and H. Ma, "Object detection model based on deep dilated convolutional networks by fusing transfer learning," *IEEE Access*, vol. 7, pp. 178 699–178 709, 2019.

[17] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," 2022. [Online]. Available: https://arxiv.org/abs/2207.02696

[18] "Your machine learning and data science community." [Online]. Available: https://www.kaggle.com/

[19] "Labelimg," Dec 2018. [Online]. Available: https://github.com/heartexlabs/labelImg

[20] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 233–240.

[21] J. Gao, Z. Wang, Y. Yang, W. Zhang, C. Tao, J. Guan, and N. Rao, "A novel approach for lie detection based on f-score and extreme learning machine," *PloS one*, vol. 8, no. 6, p. e64704, 2013.

[22] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.

[23] P. H. Avelar, A. R. Tavares, T. L. da Silveira, C. R. Jung, and L. C. Lamb, "Superpixel image classification with graph attention networks," in *2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, 2020, pp. 203–209.

[24] T. Nowak, M. R. Nowicki, K. Ćwian, and P. Skrzypczyński, "How to improve object detection in a driver assistance system applying explainable deep learning," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 226–231.