Realtime Drone Detection Using YOLOv8 and TensorFlow.JS

B. Srinivasa S. P. Kumar [1] , Irshad Ahmad Wani [2]

[1]*Assistant Professor, Department of MCA, Chaitanya Bharathi Institute of Technology (A), Gandipet, Hyderabad, Telangana, India*
[2]*MCA Student, Chaitanya Bharathi Institute of Technology (A), Gandipet, Hyderabad, Telangana, India*

***Abstract*— The escalating prevalence of drones in airspace has amplified concerns over potential misuse, encompassing privacy infringements and illicit activities. Autonomous drone detection systems emerge as a promising solution to tackle these mounting challenges. This study introduces an advanced real-time drone detection system, harnessing the cutting-edge YOLOv8 (You Only Look Once version 8) algorithm, renowned for its prowess in object detection. Real-time detection serves as an imperative prerequisite for timely identification and response. To achieve this, the YOLOv8 model is integrated with TensorFlow.JS, enabling seamless deployment and execution of the system on web-based applications without the need for server-side processing. Our proposed real-time drone detection system consistently attains exceptional accuracy rates, consistently exceeding 95% in drone detection, setting a robust benchmark for reliability. Through diligent model fine-tuning, we have substantially mitigated false-positive detections. By extending the model training to encompass birds and airplanes, we have minimized unnecessary alerts, enhancing the overall user experience.**

Keywords—Machine Learning, YOLOv8, CNN, TensorFlow.JS, Drone Detection

I. INTRODUCTION

Drones are becoming increasingly popular. These small, unmanned aircraft have a wide range of applications, from delivering packages to inspecting infrastructure [1]. However, their increasing use also poses some risks to public safety.

Drones are relatively inexpensive and easy to operate, making them accessible to a wide range of people. This means that there is a growing number of hobbyist drone pilots, some of whom may not be aware of the potential risks associated with their hobby. For example, drones can interfere with aircraft operations, crash into people or property, or be used to carry illegal substances or used for terrorism activities [2]. As of July 2023, there are 2,588,621 drones registered in the US [3].

In addition, drones are becoming increasingly sophisticated. Some drones are now equipped with cameras and sensors that can be used to gather intelligence or collect data. This raises the possibility that drones could be used for malicious purposes, such as spying or terrorism [4].

For instance, In August 2017, a major security breach occurred at the Indira Gandhi International Airport in New Delhi when unidentified drones were spotted flying near the runway. This incident led to the temporary suspension of flight operations, causing inconvenience and highlighting the vulnerability of airports to drone-related threats [5]. In December 2018, Gatwick Airport in London experienced a drone disruption that lasted for several days. Multiple unauthorized drones were sighted near the airport, leading to the cancellation and diversion of numerous flights, affecting thousands of passengers [6]. In September 2019, a drone attack targeted an oil refinery in Abqaiq, resulting in a fire. The attack was claimed by Houthi rebels in Yemen, who have been known to employ weaponized drones in their conflict with Saudi Arabia [7].

Identifying drones can pose a challenge due to the existence of similar objects in the sky, like birds and other aircraft. In this paper, our objectives are to develop an automated detection system that can distinguish between drones, birds and planes using a machine learning model and to also test the effectiveness of this system on a diverse dataset of images, including those of different types of drones currently available in the market. In [8], the author used a dataset of 2,395 images

of birds and drones, which they then augmented. However, they did not test their method on airplane images, as most methods fail when tested on airplanes after being trained on bird and drone images. This is because drones have a significant resemblance to aircraft.

To Achieve these objectives, we have curated a dataset consisting of images of drones, birds, and airplanes. The images that make up this dataset were gathered and then labelled using an online tool offered by Roboflow. After collecting more than 3500 images of birds, drones, and airplanes from public sources, such as Google, Kaggle, and others. The author labelled the images and divided them into three categories: drones, birds, and airplanes. The YOLOv8 model was then trained on google collab with NVIDIA-SMI 525.85.12 GPU. Using YOLOv8 Authors have presented a method for detecting drones flying in prohibited or where drones are restricted. YOLOv8 is the state-of-the-art object detection model that outperforms other models in terms of speed and accuracy.

Our key contributions to this study were the addition of a diverse dataset of different types of drone images which can detect most of the drones in the market and using latest state of the art YOLOv8 object detection model by Ultralytics. Authors have utilized a random train:test split of 80:20, the fine-tuning of the original YOLOv8 based on our collected customized dataset, then tested the model on a wide variety of backgrounds (dark, sunny), and the testing of different views of images.

II. LITERATURE REVIEW

Historically, a range of techniques including radar were employed for drone detection [9]. However, due to the low levels of electromagnetic signals transmitted by drones, radar-based detection proved challenging [10]. Other techniques including acoustic and radio frequency-based drone detection were also explored, but these approaches were often expensive and lacked the desired accuracy [11].

In recent times, machine learning-based drone detectors have emerged as promising alternatives. Classifiers such as Support Vector Machines (SVM) and Artificial Neural Networks have been successfully applied to drone detection, outperforming traditional radar and acoustic detection systems [12]. A notable advancement in this field has been the application of the YOLO (You Only Look Once) algorithm. The author has shown superior performance compared to competitor algorithms such as the R-CNN and SSD due to its complex feature-learning capability coupled with fast detection [13]. The YOLO algorithm has become a cornerstone in object detection tasks, with its rapid detection and high accuracy making it suitable for real-time implementation. YOLOv8, the latest iteration of this algorithm, has made significant strides in improving the performance of the YOLO family of algorithms, offering a marked improvement over its predecessors, like YOLOv5, YOLOv6, YOLOv7. In the present research, we leverage the capabilities of YOLOv8 to develop an automated drone detection system. The algorithm's speed, accuracy, and adaptability to real-time applications make it a strong candidate for such a task [14].

Deep learning-based object detection techniques are broadly categorized into one-stage and two-stage detection algorithms. R-CNN is representative of the two-stage object detection technique, while YOLO and SSD are examples of one-stage object detection techniques [15], [16]. One-stage detectors, utilizing the sliding window technique, operate swiftly and in real-time, making them well-suited for real-time applications [17]. YOLO, in particular, is a preferred choice due to its easy training, speed, accuracy, and ability to train an entire image immediately. It starts by dividing an image into SXS grids and assigns class probabilities with bounding boxes around the object. Subsequently, a single convolutional network is leveraged to perform the entire prediction. Conversely, R-CNNs initiate the process by generating numerous region proposals using a selective search method, then leverage a CNN to extract features from each region proposal, and finally classify and define bounding boxes for different classes [18]. Several studies have successfully applied these algorithms for UAV detection.

III. METHODOLOGY

The study leverages YOLOv8, a recent advancement in the YOLO algorithm series, renowned for its exceptional performance and rapid object detection [19]. This is crucial in applications like drone detection where the objects of interest (drones) can move at high speeds, necessitating a fast detection mechanism. YOLOv8 is built on the PyTorch framework, an open-source deep learning platform that simplifies the process of training and testing customized datasets, and it delivers superior detection performance. The YOLOv8 algorithm is structured into three principal segments: the backbone, neck, and head [20].

Backbone: The backbone segment of YOLOv8 is constructed using a Cross Stage Partial Network (CSPNet). The CSPNet is designed to decrease the model's complexity, resulting in fewer hyperparameters and reduced computational load (FLOPS). It also addresses the issues of gradient vanishing and explosion commonly encountered in deep neural networks. The CSPNet includes several convolutional layers, four CSP bottlenecks with three convolutions each, and spatial pyramid pooling. This component extracts features from the input image and combines them into a comprehensive feature map.

Neck: The middle segment of YOLOv8, referred to as the neck or the Path Aggregation Network (PANet), serves as a bridge between the backbone and the head. Its role is to collect the features extracted by the backbone, store them, and forward them to the deeper layers for feature fusion. This fusion of features ensures that the output layer has access to high-level features for the final object detection.

Head: The head segment of YOLOv8 performs the actual object detection. It consists of *1x1* convolutions that predict the object's class, generate bounding boxes around the detected object, and assign a class probability score.

In this study, we demonstrate real-time drone detection using a combination of YOLOv8 for training and TensorFlow.js for detection. Our methodology involves using YOLOv8 to train our model on a dataset consisting of drone, airplane, and bird images. The YOLOv8 model provides us with an efficient and powerful API built around it, abstracting away unnecessary details while allowing customizability. It also supports all usable export formats and employs practices that make the project both efficient and as optimal as it can be. The project provides pre-trained weights on MS COCO, a staple dataset on objects in context, which we use to transfer general knowledge of objects to our custom dataset.

After the training phase, we employ TensorFlow.js, a JavaScript library for training and deploying ML models in the browser and on Node.js, for the detection phase. This allows the trained model to be used in real-time applications, such as detecting drones in video streams.

A key advantage of using TensorFlow.js is its ability to perform computations in the browser, making it possible to create interactive applications without the need for server-side computation. This is particularly useful for real-time drone detection, as it reduces latency and allows for immediate feedback. The trained YOLOv8 model is converted to a format compatible with TensorFlow.js and then loaded into the application for real-time detection.

YOLOv8 is an anchor-free model, which means it directly predicts the center of an object instead of calculating the offset from a known anchor box [21], [22]. In earlier versions of YOLO, anchor boxes needed to be manually identified to assist in object detection. These predefined bounding boxes captured the scale and aspect ratio of specific object classes in the dataset. However, YOLOv8 automatically predicts anchor boxes at the center of an object, removing the need for manual identification.

Anchor boxes were a challenging aspect of earlier YOLO models because they may represent the distribution of the target benchmark's boxes but not the distribution of the custom dataset. In YOLO, anchor boxes are used to capture the scale and aspect ratio of specific object classes in the dataset. However, manually selecting anchor box dimensions can be difficult and may not accurately represent the objects in the custom dataset. This is because the distribution of object sizes and shapes can vary between different datasets.

Anchor free detection reduces the number of box predictions, which speeds up Non-Maximum

Suppression (NMS), a complicated post processing step that sifts through candidate detections after inference.

The first *6x6* convolution in the stem was replaced with a *3x3* convolution. The main building block was also changed, and *C2f* replaced *C3*. The module is summarized in the figure below, where "*f*" is the number of features, "*e*" is the expansion rate, and *CBS* is a block composed of a convolution, a batch normalization, and a *SiLU* activation function.

In *C2f*, all the outputs from the bottlenecks (two *3x3* convolutions with residual connections) are concatenated. In *C3*, only the output of the last bottleneck was used.

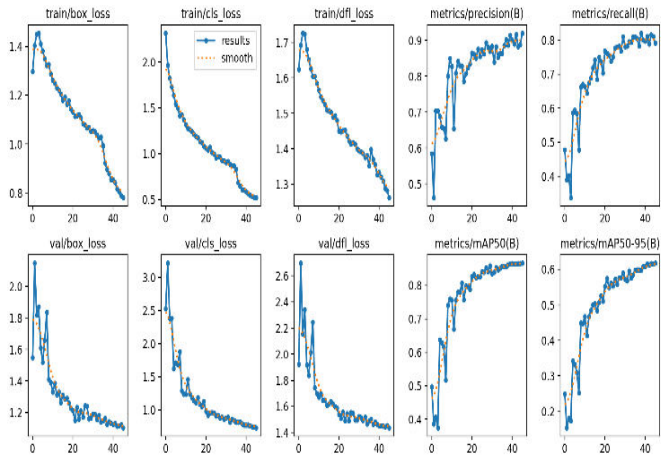
The Bottleneck in YOLOv8 is the same as in YOLOv5, but the first convolution's kernel size was changed from *1x1* to *3x3*. This change indicates that YOLOv8 is starting to revert to the ResNet block, which was defined in 2015.

In the neck, features are concatenated directly without forcing the same channel dimensions. This reduces the number of parameters and the overall size of the tensors.

Experimentation and data gathering.

We assembled a diverse dataset of more than 3500 images, which included 400 bird pictures, more than 600 airplane images, and 2500 drone images, sourced from various public platforms such as Google, and Kaggle. The images are from various angles, altitudes, and backgrounds to ensure a varied dataset. The bird images represented different species. The dataset was then split in an 80:20 ratio for training and testing the YOLOv8 model, resulting in more than 2500 training images and more than 1000 testing images. The images were then annotated using a freely available tool, categorizing them into three classes; drones were labelled as the "zero class", airplanes as ‘second class’, and birds as the "third class". For YOLO implementation, all images were saved in .txt format, which includes four coordinates for the object and its class (0, 1, or 2).

Our experiment was conducted on Google Collab, a free cloud-based notebook for coding, where we implemented YOLOv8. We fine-tuned the existing YOLOv8 model using our custom dataset and transfer learning for enhanced detection accuracy. We used the pre-trained weights of the original YOLOv8 model, specifically the YOLOv8s.pt weight that was saved during training on the COCO dataset. PyTorch was our chosen framework for this task. At the time of our model training, Google Collab provided a Tesla T4 with a 15110MiB memory NVIDIA GPU.



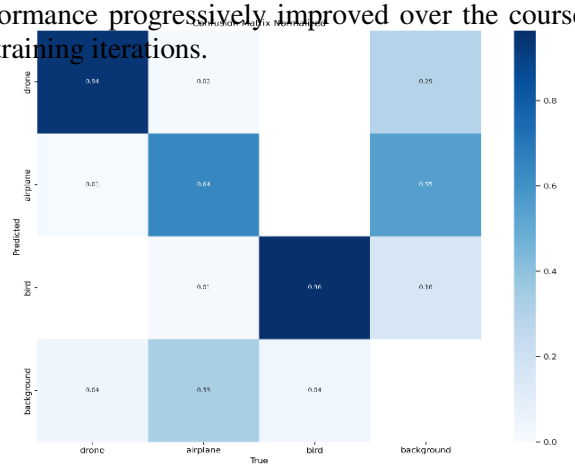
To fine-tune the YOLOv8 model, we used the hyperparameters recommended in the original model. We modified the original YOLOv8 model, reducing the number of classes from 80 to 3 to accommodate our three classes: drone, airplane, and bird. We used Roboflow for data augmentation and preprocessing. We then randomly split the dataset into an 80:20 train:test split. We trained the model for only 46 iterations, saving the best weight for testing with the testing images. A

flowchart of the overall experiment is provided in Figure 3.

IV. RESULTS

The performance of our trained model was gauged using several evaluation metrics such as the mean Average Precision (mAP), precision, recall, and F1-scores. To determine the speed of detection in the videos, we utilized Frames Per Second (FPS) as an evaluation metric.

We carried out the evaluation on a testing dataset, derived from a random split of the original dataset into training and testing sets. The testing images exhibited a wide range of variability, with different backgrounds (like bright, dark, blurred, etc.) and varying weather conditions (such as cloudy, sunny, foggy, etc.). Furthermore, some images contained multiple classes. To keep track of the evaluation metrics over time, we graphed these values over the course of the training iterations. An overall summary of the model's training can be seen in Figure 4. During the training process, the loss curves displayed a decreasing trend, which indicates that the losses for both training and validation were successfully minimized. In contrast, the metrics curves show an upward trend, signifying that the model's performance progressively improved over the course of the training iterations.



To further assess the model's ability to accurately predict, we plotted a precision-recall curve, which can be viewed in Figure 5. This curve veered towards the top right corner, suggesting that the values were predominantly close to one. This indicates a low rate of misclassification, thereby demonstrating the model's high level of accuracy in making predictions.

V. DISCUSSION

Drones have become a prominent topic in recent times, with their increasing usage across various domains and for diverse purposes. As drones continue to evolve with advanced technologies and tools, they pose new challenges. Certain areas or security zones prohibit the flight of unmanned aerial vehicles (UAVs). To ensure the security of such areas, an automatic drone detection system is necessary to detect drones without manual intervention. In this research, we employed the state-of-the-art object detection model YOLOv8, which outperforms other models in terms of speed, accuracy, and simplicity. Building an object detection system with YOLOv8 is easier compared to other models.

Previous studies have also focused on building drone detection systems and achieved a mean average precision (mAP) of approximately 0.7 to 0.8. However, most of these studies primarily included drones and birds in their datasets. Given that the skies are not limited to drones and birds, airplanes are increasingly common. We aimed to develop a system that can differentiate between drones, airplanes, and birds. When training object detection models solely with drone and bird images, they perform well on those categories. However, when provided with airplane images, these models often fail to deliver accurate results. Therefore, we prepared a dataset consisting of approximately 3500 distinct environmental images (sunny, cloudy, and dark) containing drones, airplanes, and birds. After training YOLOv8 with this dataset, we obtained a mAP of 0.86, F1-Curve of 0.85, P-Curve of 0.91, and R-Curve of 0.94. Our implemented approach is more resilient and more robust because of having airplane and bird images as we know airplanes have more resemblance to drones than birds have. So, it is essential to have airplane images than having bird images.

After training and evaluating our model, we aimed to develop a detection system that is easy to implement and deploy in any environment. We discovered a JavaScript ML library called TensorFlow.js that enables us to run and deploy machine learning

models within web browsers. YOLOv8 provides user-friendly APIs that facilitate the conversion of PyTorch weights to TensorFlow.js. By deploying the converted weights in the browser, we achieved surprising results in terms of accuracy and speed. The results were consistent with inference on PyTorch weights using YOLOv8. Leveraging web browsers for implementation and monitoring purposes offers convenience and accessibility.

However, this approach has a disadvantage. Computer vision tasks demand significant computational power, often relying on GPUs for efficient mathematical computations. For a real-time detection system, a GPU capable of handling the computational tasks is essential. Otherwise, there may be slight delays in detection due to the resource-intensive nature of real-time detection.

In terms of future scope, we aim to develop models that are smaller and can make inferences on hardware with minimal specifications. We came across a tool called SparseML and the concept of sparcification provided by Neural Magic. These techniques, such as pruning and quantization, aim to reduce the size of the model without significantly affecting its performance. Smaller models offer several benefits for real-time applications, including faster download times, reduced memory requirements, and less reliance on computational resources. This makes them suitable for deployment on devices with limited resources, such as browsers, mobile devices, or embedded systems.

As Neural Magic doesn't support converting weights to TensorFlow.js and may in future provide support for it.

VI. CONCLUSION

Conclusion: In this study, we conducted a comparative analysis of the performance using YOLOv8 and TensorFlow.JS for drone detection. We adapted the original YOLOv8 model to our specific dataset, which consisted of three classes: drones, airplanes, and birds. By fine-tuning the hyperparameters and utilizing transfer learning with pre-trained weights from MS COCO, we were able to enhance the detection precision.

To address data scarcity and overfitting issues, we implemented data augmentation using Roboflow for preprocessing techniques. This helped us generate a diverse and robust training dataset, leading to improved model performance. We evaluated the model using precision, recall, F-1 score, and mAP metrics, achieving values of 0.910, 0.94, 0.85, and 0.86, respectively.

For future research, we aim to focus on developing smaller models that can make inferences on hardware with minimal specifications. We will explore techniques such as pruning and quantization, which are facilitated by tools like SparseML and concepts like sparcification provided by Neural Magic. These techniques offer the potential to reduce the model size without significantly impacting its performance.

Smaller models have several advantages for real-time applications, including faster download times, reduced memory requirements, and improved efficiency on devices with limited computational resources. By pursuing these avenues, we can further enhance the practicality and accessibility of drone detection systems.

VII. REFERENCES

[1] Mohsan SAH, Khan MA, Noor F, Ullah I, Alsharif MH. Towards the Unmanned Aerial Vehicles (UAVs): A Comprehensive Review. *Drones*. 2022; 6(6):147. https://doi.org/10.3390/drones6060147

[2] Yaacoub JP, Noura H, Salman O, Chehab A. Security analysis of drone's systems: Attacks, limitations, and recommendations. Internet of Things. 2020 Sep;11:100218. doi: 10.1016/j.iot.2020.100218. Epub 2020 May 8. PMCID: PMC7206421.

[3] Drones by the Numbers. Available online: https://www.faa.gov/node/54496 (accessed 29 07 July 2023).

[4] Mithra Sivakumar1, Naga Malleswari TYJ. A Literature Survey of Unmanned Aerial Vehicle Usage for Civil Applications. https://doi.org/10.1590/jatm.v13.1233

[5] Drone spotted at IGI Airport led to temporary halt of flight operations. https://indianexpress.com/article/india/delhi-airport-live-igflight-operation-at-delhi-airport-halted-as-pilot-spots-drone-4805435/ (accessed 29 07 July 2023)

[6] "Gatwick Airport Drone Incident." Wikipedia, Wikimedia Foundation, 9 Feb. 2023, en.wikipedia.org/wiki/Gatwick_Airport_drone_incident. Accessed 30 Jul. 2023.

[7] "Abqaiq." Wikipedia, Wikimedia Foundation, 19 Apr. 2023, en.wikipedia.org/wiki/Abqaiq. Accessed 30 Jul 2023.

[8] Aydin, Burchan, and Subroto Singha. 2023. "Drone Detection Using YOLOv5" *Eng* 4, no. 1: 416-433. https://doi.org/10.3390/eng4010025

[9] Gong J, Yan J, Li D, Kong D. Detection of Micro-Doppler Signals of Drones Using Radar Systems with Different Radar Dwell Times. *Drones*. 2022; 6(9):262. https://doi.org/10.3390/drones6090262

[10] Elsayed, M.; Reda, M.; Mashaly, A.S.; Amein, A.S. Review on Real-Time Drone Detection Based on Visual Band Electro-Optical (EO) Sensor. In Proceedings of the 2021 Tenth International Conference on Intelligent Computing and Information Systems (ICICIS), Cairo, Egypt, 5–7 December 2021; pp. 57–65.

[11] S. Basak, S. Rajendran, S. Pollin and B. Scheers, "Combined RF-Based Drone Detection and Classification," in IEEE Transactions on Cognitive Communications and Networking, vol. 8, no. 1, pp. 111-120, March 2022, doi: 10.1109/TCCN.2021.3099114.

[12] B. Taha and A. Shoufan, "Machine Learning-Based Drone Detection and Classification: State-of-the-Art in Research," in *IEEE Access*, vol. 7, pp. 138669-138682, 2019, doi: 10.1109/ACCESS.2019.2942944.

[13] Y. Yang, "Drone-View Object Detection Based on the Improved YOLOv5," 2022 IEEE International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA), Changchun, China, 2022, pp. 612-617, doi: 10.1109/EEBDA53927.2022.9744741.

[14] I. V. S. L. Haritha, M. Harshini, S. Patil and J. Philip, "Real Time Object Detection using YOLO Algorithm," 2022 6th International Conference on Electronics, Communication and Aerospace Technology, Coimbatore, India, 2022, pp. 1465-1468, doi: 10.1109/ICECA55336.2022.10009184.

[15] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 2014, pp. 580-587, doi: 10.1109/CVPR.2014.81.

[16] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 6517-6525, doi: 10.1109/CVPR.2017.690.

[17] M. Nalamati, A. Kapoor, M. Saqib, N. Sharma and M. Blumenstein, "Drone Detection in Long-Range Surveillance Videos," 2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Taipei, Taiwan, 2019, pp. 1-6, doi: 10.1109/AVSS.2019.8909830.

[18] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, no. 1, pp. 142-158, 1 Jan. 2016, doi: 10.1109/TPAMI.2015.2437384.

[19] N. M. Krishna, R. Y. Reddy, M. S. C. Reddy, K. P. Madhav and G. Sudham, "Object Detection and Tracking Using Yolo," 2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2021, pp. 1-7, doi: 10.1109/ICIRCA51532.2021.9544598.

[20] Al-Qubaydhi N, Alenezi A, Alanazi T, Senyor A, Alanezi N, Alotaibi B, Alotaibi M, Razaque A, Abdelhamid AA, Alotaibi A. Detection of Un vauthorized Unmanned Aerial Vehicles Using YOLOv5 and Transfer Learning. *Electronics*. 2022; 11(17):2669. https://doi.org/10.3390/electronics11172669

[21] YOLOv8 Architecture and how it works: https://blog.roboflow.com/whats-new-in-yolov8/ (Accessed on 31-07-2023)

[22] YOLOv8 Working Architecture: https://github.com/ultralytics/ultralytics/issues/189 (Accessed on 31-07-2023)