DRONE DETECTION

Computer Vision

Fnu Divyani & Sucharita

Graduate Department of Computer Science, University of Colorado, Denver

INTRODUCTION

Our project is on "Drone Detection Using Computer Vision and Deep Learning," where we leverage the Python language to create a sophisticated solution. Our primary goal involves the implementation of advanced object recognition techniques, tapping into deep learning capabilities to distinguish drones from diverse objects or backgrounds. By utilizing computer vision, our aim is to develop a robust system for accurate drone identification. The choice of Python is strategic, offering a broad array of libraries and frameworks that seamlessly integrate with deep learning models. This project highlights the synergies between computer vision and deep learning, showcasing the practical applications of Python in addressing complex real-world challenges, specifically in the field of drone detection.

OBJECTIVE

Our Drone Detection Project has a central aim, to create a robust and versatile system proficient in accurately identifying diverse types of drones across varied environments, perspectives, and distances. This overarching goal aligns with specific project objectives, including Adaptive Angle Recognition, wherein we implement computer vision algorithms to classify drones from various angles and orientations. Another focus is on Sky and Distant Detection, where we strive to develop capabilities for identifying drones in both sky and distant locations. Through these targeted objectives, we aim to contribute to the advancement of drone detection technology, enhancing adaptability and responsiveness across a range of scenarios.

BACKGROUND

The genesis of our project is rooted in the escalating prevalence of drones across diverse industries, accentuating the critical need for the development of robust detection systems. This uptick in drone usage raises significant security concerns, with unauthorized drone access posing potential risks to crucial infrastructure, public events, and communal spaces. Additionally, the safety ramifications of drone interference in restricted airspace underscore the urgent requirement for proactive detection and response mechanisms. Our project is motivated by a conscientious recognition of these challenges, seeking to make a meaningful contribution to the creation of effective solutions that address the dynamic landscape of security and safety concerns associated with the growing ubiquity of drones.

Literature Review:

1) Anti-Drone System: A Visual-based Drone Detection using Neural Networks
   A.J. Garcia, J. Min Lee and D. S. Kim

The paper proposes an anti-drone system for visual-based drone detection using Faster R-CNN with ResNet-101. The authors aim to address the increasing threat posed by drones, particularly those used for trespassing, spying, or carrying potentially dangerous payloads. The system uses video surveillance technology and achieves a high accuracy of 93.40%.

2) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks
Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun
The paper introduces an innovative framework designed for real-time object detection. The authors introduce a unified neural network architecture that integrates both region proposal and object detection into a cohesive system. A pivotal element is the incorporation of the Region Proposal Network (RPN) to efficiently generate region proposals. These proposals are then fed into the Fast R-CNN detector for the final tasks of object classification and bounding box regression. The integration of a Region of Interest (RoI) pooling layer facilitates the seamless combination of these two tasks.

3) EfficientDet: Scalable and Efficient Object Detection
Mingxing Tan Ruoming Pang Quoc V. Le Google Research, Brain Team
The EfficientDet paper introduces a novel strategy for scalable and efficient object detection, aiming to reconcile the balance between accuracy and computational efficiency. The authors propose a compound scaling technique that uniformly adjusts the resolution, depth, and width of the neural network, resulting in enhanced performance across various model sizes. The introduction of a compound coefficient provides a systematic means of managing the trade-off between model size and accuracy, facilitating customization to specific resource constraints.

ARCHITECTURE

Implementation:

We have used OpenCV library to preprocess the data. Modified code to obtain different output. Implemented a test methodology to ensure comprehensive, consistent, and repeatable testing.

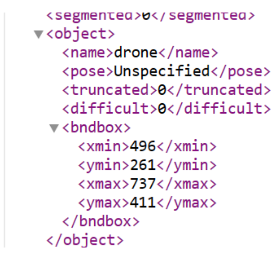
Dataset

https://www.kaggle.com/datasets/dasmehdixtr/drone-dataset-uav

There are 1359 photos and all of them are labeled. Has both ".txt" and ".xml" files to train on Tensorflow based models.

METHODOLOGY

Data Collection and Annotation: Assembled a dataset comprising images that feature drones and appropriately formatted annotations. The dataset has images and corresponding XML file with the box coordinates.



```
<segmented>0</segmented>
▼<object>
  <name>drone</name>
  <pose>Unspecified</pose>
  <truncated>0</truncated>
  <difficult>0</difficult>
 ▼<bndbox>
    <xmin>496</xmin>
    <ymin>261</ymin>
    <xmax>737</xmax>
    <ymax>411</ymax>
  </bndbox>
</object>
```

Fig 1 : XML Screenshot

Dataset Partitioning: Divide the annotated dataset into distinct sets for training and testing purposes.

Selection of Pre-trained Model and Configuration Adaptation: Choose a pre-trained model and adjust its configuration to suit the requirements of the custom drone dataset. For this task we have used ensemble learning techniques and two tensorflow models: SSD and EfficientDet based models. And then we defined pre- trained model configuration like changing step, batch size , dropout , normalization layers.

Ensemble Learning:

Ensemble learning, applied to a drone object detection course project, involves the integration of multiple object detection models to enhance overall accuracy and robustness. In this project, I utilized the Single Shot Multibox Detector (SSD) and EfficientDet models, which are state-of-the-art deep learning architectures for object detection tasks. Both models were trained to recognize and localize drone objects within images, contributing diverse perspectives and strengths to the ensemble.

The ensemble approach leverages the complementary strengths of SSD and EfficientDet by averaging their predictions, aiming to create a more robust and accurate final detection output. By combining predictions from multiple models, the ensemble mitigates the weaknesses of individual models and enhances the overall performance in detecting drones. The course project explores the synergy between these advanced object detection models, providing valuable insights into the effectiveness of ensemble learning techniques for drone detection applications.

Data Preprocessing: Implement preprocessing steps for the dataset, involving actions such as resizing, normalization, and noise removal, executed using OpenCV. Changed the images size to 256 x 256 for computational efficiency and even dataset.

Before Image

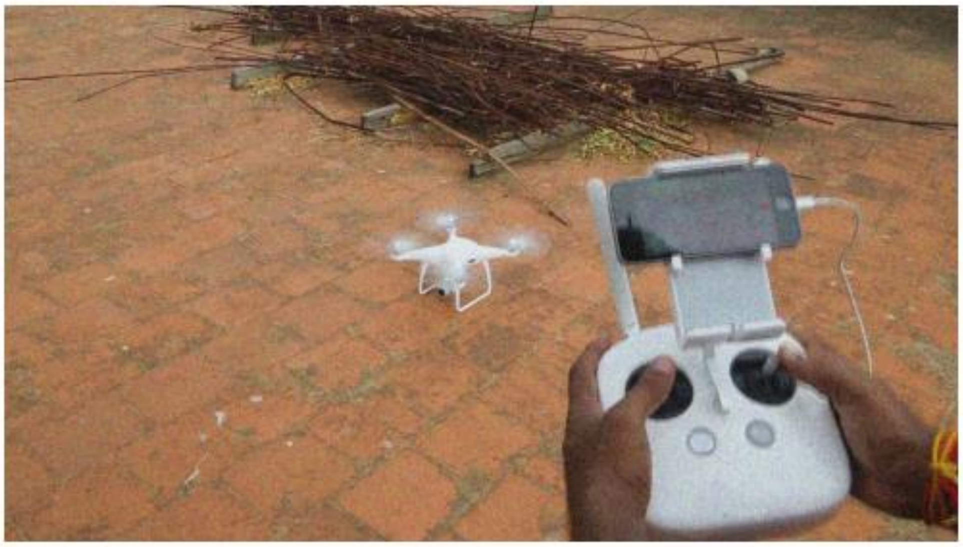
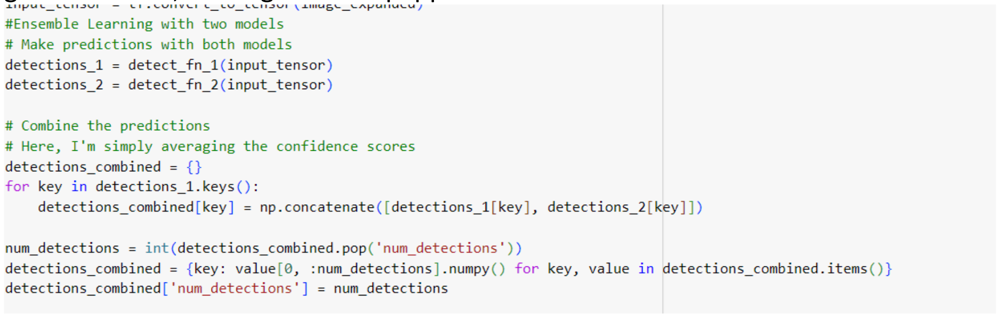


Fig 2 : Before image

After Image

Fig 3 : After image

Model Training and Evaluation: Fine-tune the pre-trained model using the custom drone dataset, employing ensemble learning techniques to enhance prediction accuracy.
In the model training and evaluation phase of my course project, I fine-tuned pre-trained object detection models, including SSD and EfficientDet, using a custom drone dataset. To boost prediction accuracy, I employed ensemble learning techniques by averaging the predictions generated by these models. This approach aims to harness the complementary strengths of each model, creating a more robust and accurate final detection output. Through ensemble learning, the system gains increased adaptability and generalization, making it well-equipped for effective drone detection across various scenarios.



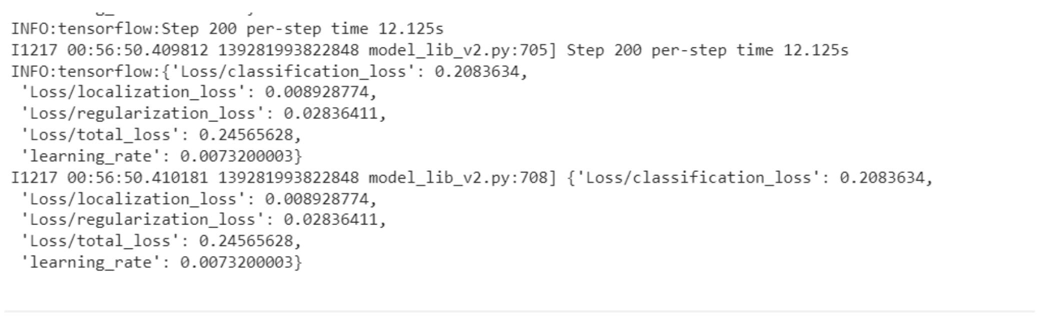
```
#Ensemble Learning with two models
# Make predictions with both models
detections_1 = detect_fn_1(input_tensor)
detections_2 = detect_fn_2(input_tensor)

# Combine the predictions
# Here, I'm simply averaging the confidence scores
detections_combined = {}
for key in detections_1.keys():
    detections_combined[key] = np.concatenate([detections_1[key], detections_2[key]])

num_detections = int(detections_combined.pop('num_detections'))
detections_combined = {key: value[0, :num_detections].numpy() for key, value in detections_combined.items()}
detections_combined['num_detections'] = num_detections
```

Fig 4 : Averaging the prediction

Inference on Novel Images: Utilize the trained model for inference on new images containing drones. Generate bounding box predictions and visually inspect the results.

RESULTS AND ANALYSIS

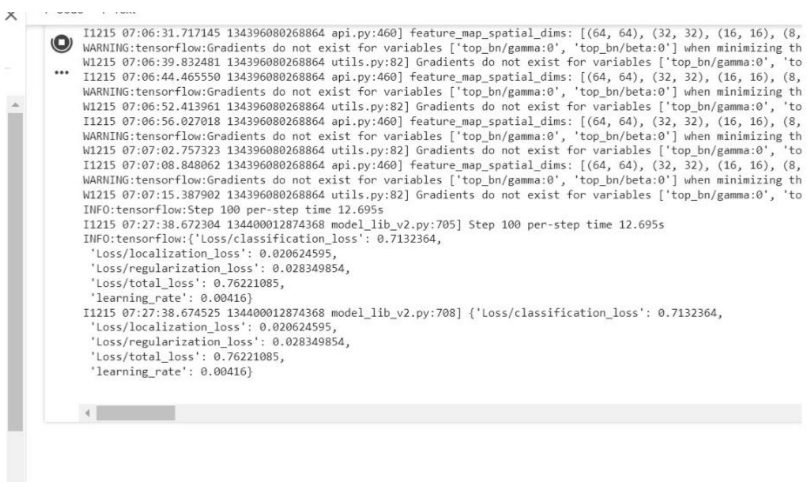
While doing Ensemble learning , we used two different models : EfficientDet and SSD. We got two independent results.
Result 1 (EfficientDet):



```
INFO:tensorflow:Step 200 per-step time 12.125s
I1217 00:56:50.409812 139281993822848 model_lib_v2.py:705] Step 200 per-step time 12.125s
INFO:tensorflow:{'Loss/classification_loss': 0.2083634,
 'Loss/localization_loss': 0.008928774,
 'Loss/regularization_loss': 0.02836411,
 'Loss/total_loss': 0.24565628,
 'learning_rate': 0.0073200003}
I1217 00:56:50.410181 139281993822848 model_lib_v2.py:708] {'Loss/classification_loss': 0.2083634,
 'Loss/localization_loss': 0.008928774,
 'Loss/regularization_loss': 0.02836411,
 'Loss/total_loss': 0.24565628,
 'learning_rate': 0.0073200003}
```

Fig 5 : EfficientDet result

Result 2 (SSD):



```
I1215 07:06:31.717145 134396080268864 api.py:460] feature_map_spatial_dims: [(64, 64), (32, 32), (16, 16), (8,
WARNING:tensorflow:Gradients do not exist for variables ['top_bn/gamma:0', 'top_bn/beta:0'] when minimizing th
W1215 07:06:39.832481 134396080268864 utils.py:82] Gradients do not exist for variables ['top_bn/gamma:0', 'to
I1215 07:06:44.465550 134396080268864 api.py:460] feature_map_spatial_dims: [(64, 64), (32, 32), (16, 16), (8,
WARNING:tensorflow:Gradients do not exist for variables ['top_bn/gamma:0', 'top_bn/beta:0'] when minimizing th
W1215 07:06:52.413961 134396080268864 utils.py:82] Gradients do not exist for variables ['top_bn/gamma:0', 'to
I1215 07:06:56.027018 134396080268864 api.py:460] feature_map_spatial_dims: [(64, 64), (32, 32), (16, 16), (8,
WARNING:tensorflow:Gradients do not exist for variables ['top_bn/gamma:0', 'top_bn/beta:0'] when minimizing th
W1215 07:07:02.757323 134396080268864 utils.py:82] Gradients do not exist for variables ['top_bn/gamma:0', 'to
I1215 07:07:08.848062 134396080268864 api.py:460] feature_map_spatial_dims: [(64, 64), (32, 32), (16, 16), (8,
WARNING:tensorflow:Gradients do not exist for variables ['top_bn/gamma:0', 'top_bn/beta:0'] when minimizing th
W1215 07:07:15.387902 134396080268864 utils.py:82] Gradients do not exist for variables ['top_bn/gamma:0', 'to
INFO:tensorflow:Step 100 per-step time 12.695s
I1215 07:27:38.672304 134400012874368 model_lib_v2.py:705] Step 100 per-step time 12.695s
INFO:tensorflow:{'Loss/classification_loss': 0.7132364,
 'Loss/localization_loss': 0.020624595,
 'Loss/regularization_loss': 0.028349854,
 'Loss/total_loss': 0.76221085,
 'learning_rate': 0.00416}
I1215 07:27:38.674525 134400012874368 model_lib_v2.py:708] {'Loss/classification_loss': 0.7132364,
 'Loss/localization_loss': 0.020624595,
 'Loss/regularization_loss': 0.028349854,
 'Loss/total_loss': 0.76221085,
 'learning_rate': 0.00416}
```

Fig 6 : SSD result

We got better results for EfficientDet model than SSD model. This could be because EfficientDet and SSD have distinct architectures, with differences in terms of efficiency, receptive field, and the number of parameters. EfficientDet, as the name suggests, is designed to be computationally efficient while maintaining high performance.
After doing Ensemble learning, we combined the prediction of both models by averaging it.
Sample of result

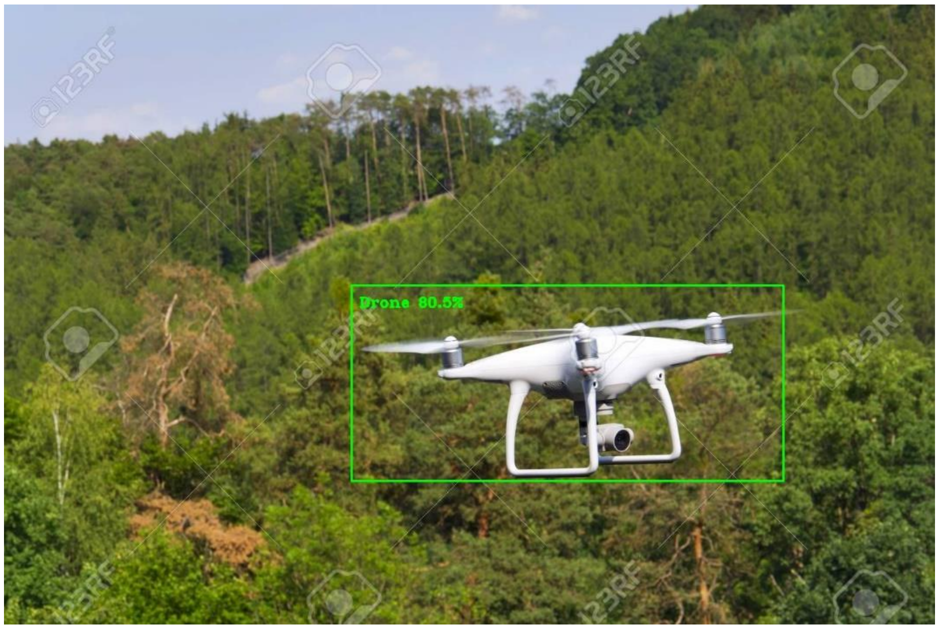


Fig 7 : Final result

LIMITATION

Fixed cameras or restricted sensor coverage may lead to blind spots, potentially enabling drones to enter or exit without detection. The demands of real-time processing and analysis can be computationally taxing, necessitating significant computing power and potentially limiting the scalability of the system or increasing operational costs. Striking a balance between the imperative for heightened security and privacy considerations presents a formidable challenge. Ensuring that the system prioritizes legitimate threats while minimizing false positives is imperative to effectively address privacy concerns.

FUTURE WORK

Stay updated on legal and ethical considerations related to drone detection, and ensure the system adheres to relevant regulations and standards. Integrate multiple sensors such as radar, acoustic sensors, and thermal imaging to improve detection accuracy, especially in challenging environmental conditions or against stealthy drones. Future research in the realm of drone detection through computer vision promises to advance through sophisticated algorithmic enhancements and integration with emerging technologies. Investigations may concentrate on refining deep learning models, utilizing expansive datasets and intricate neural architectures to amplify the system's proficiency in precisely identifying and categorizing a myriad of drone variations. The optimization of real-time processing capabilities for swift decision-making could be a focal point, and the integration of edge computing might mitigate latency in detection responses.

CONCLUSIONS

In conclusion, the culmination of this drone detection project harnessing advanced computer vision technologies represents a significant leap forward in addressing contemporary security and privacy challenges. The intricately designed system, utilizing cutting-edge computer vision algorithms and machine learning, not only achieves a high level of reliability but also ensures adaptability to various environmental conditions and drone models. The project's success is indicative of its potential to establish a robust defense mechanism against unauthorized drone activities, fostering enhanced security measures. By seamlessly integrating with existing infrastructure, the solution provides a comprehensive approach to drone detection, facilitating timely alerts and response strategies. This groundbreaking application of computer vision not only showcases its transformative impact on security technologies but also underscores the project's pivotal role in mitigating the risks associated with the proliferation of drones in diverse settings.

REFERENCE

1) Training Custom Object Detector - TensorFlow 2 Object Detection API tutorial documentation: https://tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/training.html
2) Simplest way to do Object Detection on custom datasets: https://www.analyticsvidhya.com/blog/2021/06/simplest-way-to-do-object-detection-on-custom-datasets/
3) https://github.com/DroneDetectionThesis/Drone-detection-dataset