Proceedings of the 2021 AIAA/IEEE 40th Digital Avionics Systems Conference (DASC), 3-7 October, San Antonio, USA
DOI:10.1109/DASC52595.2021.9594392

Drone Model Identification by Convolutional Neural Network from Video Stream

Mariusz Wisniewski, Zeeshan A. Rana, Ivan Petrunin
Digital Aviation Research and Technology Centre (DARTeC)
Cranfield University
Cranfield, Bedfordshire, MK43 0AL, United Kingdom
{m.wisniewski, zeeshan.rana, i.petrunin}@cranfield.ac.uk

***Abstract*—We present a convolutional neural network model that correctly identifies drone models in real-life video streams of flying drones. To achieve this, we show a method of generating synthetic drone images. To create a diverse dataset, the simulation parameters (such as drone textures, lighting, and orientation) are randomized. This synthetic dataset is used to train a convolutional neural network to identify the drone model: DJI Phantom, DJI Mavic, or DJI Inspire. The model is then tested on a real-life Anti-UAV dataset of flying drones. The benchmark results show that the DenseNet201 architecture performed the best. Adding Gaussian noise to the training dataset and performing full training (as opposed to freezing layers) shows the best results. The model shows an average accuracy of 92.4%, and an average precision of 88.6% on the test dataset.**

***Keywords*—Unmanned Aerial Vehicles, drones, airport security, convolutional neural network, anti-uav, synthetic images, domain randomization, synthetic drones**

I. Introduction

Amateur drones pose a risk to the security of airports. In 2018, a rogue drone at Gatwick airport affected approximately 1,000 flights, causing large financial losses. To prevent similar incidents from happening in the future, detection systems are required to monitor all air traffic in the proximity of airports.

Identifying the features of the detected drone, such as model, size, and payload, is of interest to airport security teams, as this data could help decide the mitigation actions. For example, an off-the-shelf drone might be operated by an amateur not aware of flying restrictions in the area, whereas a non-identifiable drone may be operated by a person with malicious intent.

This paper investigates the visual classification of 3 popular DJI drone models using a convolutional neural network. This is achieved by creating a synthetic dataset to train the neural network and testing its performance on a real-life dataset of flying drones.

The novel contribution presented in this paper is the convolutional neural network that is trained on a purely synthetic dataset and can accurately (90%+) predict the model of the drone in a real-life video feed. Although there exist attempts at predicting the drone model, we believe our approach is better because it relies on only using 3D model of the drone. This reduces the time-cost associated with training any new drone models. To the best of authors knowledge, such a convolutional neural network has not been presented in literature.

Counter drone technologies can be split into prevention, detection, and mitigation. Prevention aims to stop users from mistakenly causing disruptions. Geo-fencing, for example, stops drone operators from flying into restricted airspaces based on the GPS position of the drone. However, it does not stop malicious users from circumventing the restrictions. Geo-fencing can be turned off and cannot be enforced on homemade/hobbyist drones. Hence, detection and mitigation systems are required alongside.

Commercial detection systems most commonly use the following systems [1]:

- Acoustic (6%)
- RF (26%)
- Radar (28%)
- Visual (40%)

Acoustic sensors are the least used type of sensor for drone detection. They suffer from external noise, making them unreliable at areas such as airports, where aircraft noise may affect their performance. They also have short range compared with other sensors. However, they do not required line of sight with the drone, and microphones are relatively cheap. An acoustic system presented in [2] correctly detects and identifies drone models using a neural network trained on the sound of each drone. Another system [3] uses a hybrid system of visual cameras, and microphones to detect drones. They find that using the audio improves detection accuracy.

Radio frequency (RF) methods look at the RF signature of the drone. They are generally cheaper than other sensors and can perform at long range. However, they fail to detect autonomous drones that have no RF transmissions. A hierarchical classifier that detects the presence of a drone is presented in [4]. It can identify 3 models (Parrot Bebop, Parrot AR, DJI Phantom), and for the Parrot models it is able to identify the flight mode of the drone.

Holographic radars [5] can detect and classify $1m^2$ drones at a range of 20nmi. A convolutional neural network [6] can correctly classify drones and non-drones at an accuracy of 98.9%.

Visual methods should detect, track, and identify drones in order to successfully recognise drones in video feeds. A system that uses a matrix of static cameras and a background

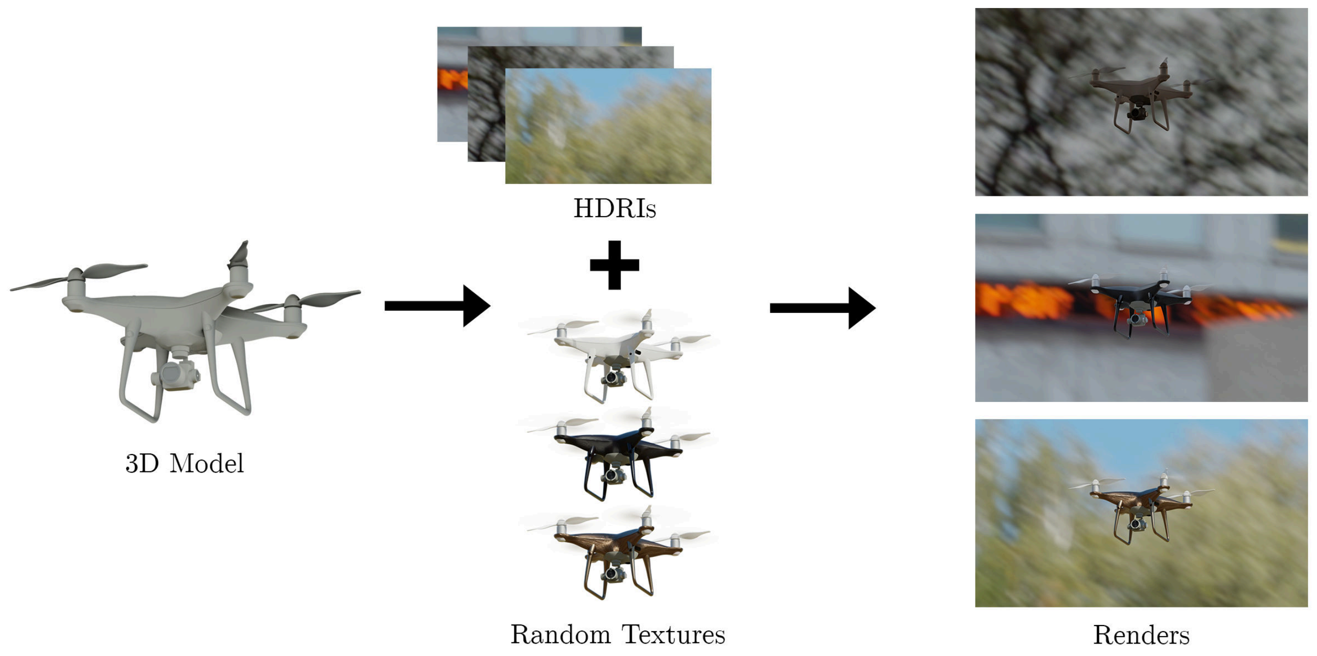


Fig. 1: Process of rendering a synthetic drone. Random textures can be applied to the 3D model. A High-Dynamic-Range-Image (HDRI) containing a realistic lighting setup can be set as the background scene. The drone can then be rendered using the Cycles engine.

subtraction method [7] is used to detect drones as far as 700 metres away in real-time. Another system [8] also uses background subtraction to detect the drones, but then uses a convolutional neural network to classify the object into drone, bird, and background categories.

The Anti-UAV challenge [9] compares tracking methods on video streams of flying drones, recorded by a pan-tilt-zoom camera. Both visual and thermal video feeds are provided, with the position of the drone labeled in each of the frames. Siam R-CNN [10] reports the best precision score of 95.70% on the infrared validation dataset.

Classification of drones with other objects is commonly achieved by using a convolutional neural network. Bird vs Drone Challenge [11] challenges participants to identify birds from drones in provided videos. The best submission [12] uses a trained ResNet110v2 convolutional neural network to classify the flying objects into drone, bird, and none categories.

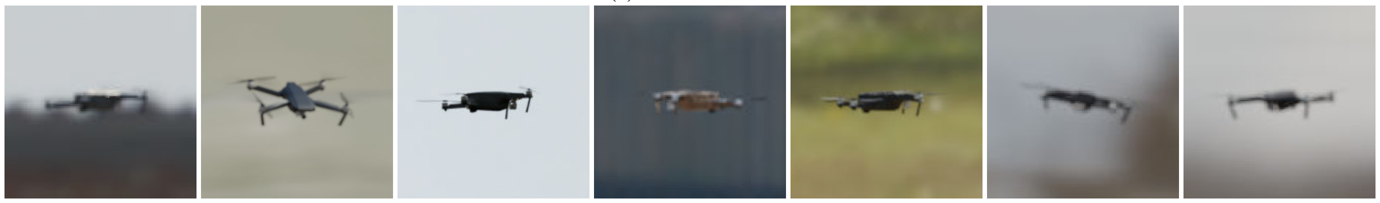
There exists limited literature on the identification of drone models. A drone identification system [13] uses 2,000 images scraped from internet to train a convolutional neural network to identify drone models. Another method [14] uses the YOLOv3 regional proposal network to predict whether the drone is a tricopter, quadcopter, or a hexacopter. Lastly, a method shown in [15] attempts to classify whether the drone is carrying a payload using YOLOv2.

Attempts at using a synthetic dataset for drone classification have been previously seen in literature. The advantage of using a synthetic dataset is that the ground truth is known. This is not the case with real-life images that have a time-cost associated with labelling the ground truth. A method of generating synthetic drones is shown in [16]. It is done by overlaying 3d drone models on top of random backgrounds. Another method [17] improves on this by using a Physically Based Rendering Toolkit to render photorealistic images of drones.

The use of synthetic images is not limited to drone detection systems. Synthetic images combined with domain randomization [18] are used to detect cars on the KITTI dataset. Domain randomization is the process of changing parameters of the simulation in a non-realistic way in order the train the neural network model to better learn the features of the object. A deep neural network trained used to control a robot arm based on visual input [19] appears to be one of the first successful applications of transferring a model trained on purely synthetic data to a real-world application. A ship classification convolutional neural network [20] is used to classify different types of ships from overhead imagery using a mix of real world and synthetic data.

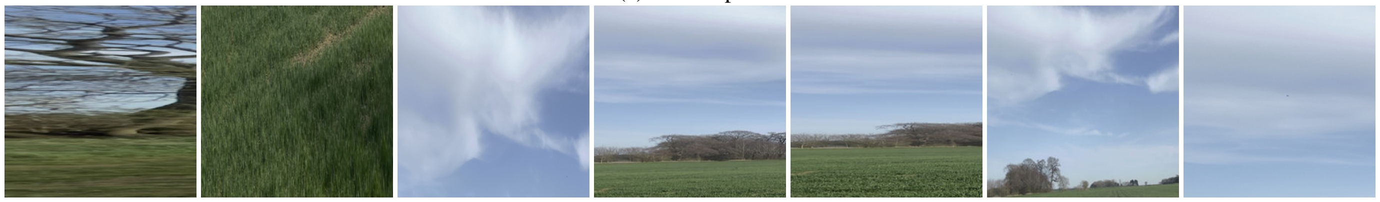
In the following sections of this paper, the methodology of generating the synthetic images is explained. Then, the process of training the convolutional neural network using the synthetic images is shown. This includes the choice of hyper-parameters, data augmentation, and choice of architecture. The results of the model when tested on a real-life dataset are then presented. These results are discussed with regard to existing literature. Finally, we conclude the paper and suggest further work.

(a) DJI Phantom



(b) DJI Mavic

(c) DJI Inspire



(d) No drone

Fig. 2: Synthetic drones. Used for training and validation.

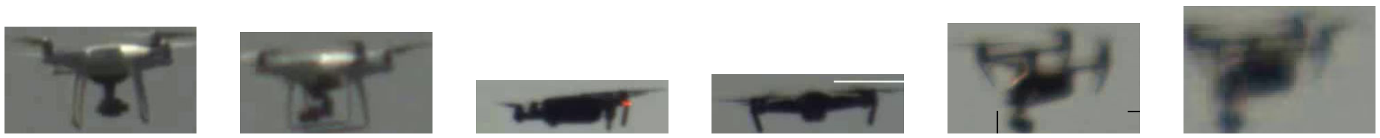


Fig. 3: Real drones (from the Anti-UAV dataset [9]). Used for testing.

II. Methodology

In this section we explain the process of creating the dataset using synthetic images generated in Blender[1], an open-source 3D modelling program. We then describe the process of training the neural network.

A. Synthetic Images

A challenge associated with training a neural network is the creation of the dataset. Gathering real-life images has a time-cost associated with drawing bounding boxes around the target object. Although we have found the Anti-UAV dataset, we wanted to use it to test our results. We then decided that synthetic image generation with the use of domain randomization made it possible to transfer our model to a real-life dataset. This posed the advantage of not having to label real-life drone images, and it allowed us to create datasets of drone models that we did not physically own.

To do this, we modelled DJI-Phantom and DJI-Mavic, and we found a free 3D model of DJI-Inspire[2] online. We then created a script to generate random images of the drone. The process of image generation is shown in figure 1.

1) Rendering Script: The script to generate the synthetic images keeps the drone animated to fly around the origin (0,0,0) for 200 frames. In this animation the drone moves in xyz space, rotates, and spins propellers. The drone models are scaled to a diameter of 1 m. The camera is programmed to always follow the drone. The xyz position of the camera is randomized and kept within a distance of 200 m from the drone. The background and lighting are also randomized

[1]https://www.blender.org/

[2]https://sketchfab.com/3d-models/dji-inspire-2-with-zenmuse-x5s-3979efe28b3a4221bdd462638582d0a6/

every 200 frames. To do this, the High Dynamic Range Images (HDRI) is changed. The HDRIs contains a background image and a realistic lighting setup and were acquired from HDRIHaven[3].

2) Domain Randomization: After performing some preliminary training and testing on a real-world dataset, the model translated poorly to the real-world dataset during testing. We hypothesised that this could be because the real-world dataset differs considerably from the synthetic images. Reference [18] applies random, unrealistic textures to their 3D objects with the aim of teaching the neural network about the features of the object, as opposed to the colour or texture. Further, we looked at our testing dataset, as shown in figure 3, and found that it contains images that are not perfectly focused on the drones.

Hence, we randomized the texture of the main body of the drone every 200 frames. We acquired the textures from ambientCG[4]. The focus point of the camera is also randomized, creating blur in some images.

The aim of this is to make the neural network invariant to the colour of the drone. Instead, we want the model to learn the features of the drone's shape. The aim of randomizing the focus point is to make the synthetic images more similar to the real-world scenarios. It is hard to focus and keep focus on a flying drone at a far distance by manually operating a camera to follow the object. A no drone class is also added, containing random images of background. It is shown in figure 2d.

B. Training the Neural Network

The models in the following sections are trained on a dataset of randomly generated 1,000 synthetic images, as shown in figure 2. Each of the models is pretrained on ImageNet [21]. A learning rate of 0.01, a momentum of 0.9 [22], and a batch size of 64 is used. As [23] notes, it is possible to start with a high learning rate that decays overtime. Hence, the learning rate is decayed by a factor of 0.1 every 7 epochs. Each of the models is trained for 100 epochs. A dropout of 0.25 is applied, to prevent from overfitting [23]. The validation is performed on a separate dataset of randomly generated 1,000 synthetic images.

To measure the performance of the models, average accuracy

$$\frac{\sum_{i=1}^{l} \frac{tp_i+tn_i}{tp_i+fn_i+fp_i+tn_i}}{l} \tag{1}$$

and average precision

$$\frac{\sum_{i=1}^{l} \frac{tp_i}{tp_i+fp_i}}{l} \tag{2}$$

are calculated [24]. tp_i is the true positive value for class i, tn_i is the true negative value for class i, fp_i is the false positive value for class i, fn_i is the false negative value for class i, and l is the number of classes.

[3]https://hdrihaven.com

[4]https://ambientcg.com/

1) Testing the Neural Network: The trained model is tested and validated on synthetic images of drones. Since the aim of this work is to create a neural network that can correctly identify the drone models in real life, the model is tested on the real life Anti-UAV dataset [9], as shown in figure 3.

The dataset provides videos and ground truth labels for each frame of: DJI Inspire, DIJ Mavic-Pro, DJI Phantom, DJI Mavic-Air, DJI Spark, and Parrot drones. Two video feeds are provided, one from visual camera, and another from a thermal camera. Some of the scenarios provided are filmed at night. For the purposes of testing our model, we are only interested in daytime videos from the visual camera. So, 4 daytime videos of each of DJI Phantom, DJI Mavic, and DJI Inspire were selected for the testing of our model. We use the provided ground truth label to extract an image of the drone from the video feed. This image is then inputted into the neural network to predict the drone model.

2) Synthetic Noise and Freezing Layers: Both [25] and [18] use Gaussian noise as a data augmentation on their training dataset. However, they do not explicitly test the effectiveness of adding the noise. Further, [19] found the effect of noise to be negligible, and [25] found that freezing the layers during the training of the neural network improved the precision. Contrary to this, [18] found that full learning, without freezing the layers, improved performance.

Architecture	Average Accuracy (%)	Average Precision (%)
Frozen Layers, No Noise	62.3	43.3
Frozen Layers, Added Noise	73.0	59.3
Full Learning, No Noise	79.4	69.3
Full Learning, Added Noise	**92.4**	**88.6**

TABLE I: Effects of freezing layers and adding Gaussian noise

Table I contains two variations of the neural network. Freezing layers compared with full learning, and adding Gaussian noise compared with not adding noise. A DenseNet201 [26] architecture is used for the comparison. It shows that both, full learning and added Gaussian noise, improve the performance of the neural network significantly. Full learning with added noise shows an average accuracy of 92.4%, compared with 62.3% for a model trained with frozen layers and no noise added. The average precision also increases to 88.6%, from 43.3%. This is in line with the findings of [18].

To explain the effects of adding noise to the dataset, [27] tries to visualise the effect of noise on neural networks by using sensitivity maps. The aim of this method is to find pixels that strongly influence the final decision. It shows that adding noise to the training and dataset provides a de-noising effect to the sensitivity map.

3) Data Augmentations: The images are resized to 256 pixels and cropped to a 224x224 pixels. A random horizonal flip is applied. The image is then transformed into a tensor and normalized. Lastly, the Gaussian noise of mean 0.75 and standard deviation 0.75 is applied to 75% of the images in the dataset. During validation and testing, the Gaussian noise is not added.

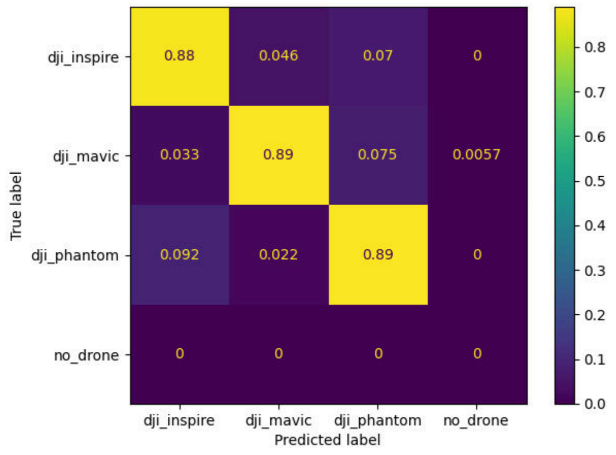


Fig. 4: Densenet201

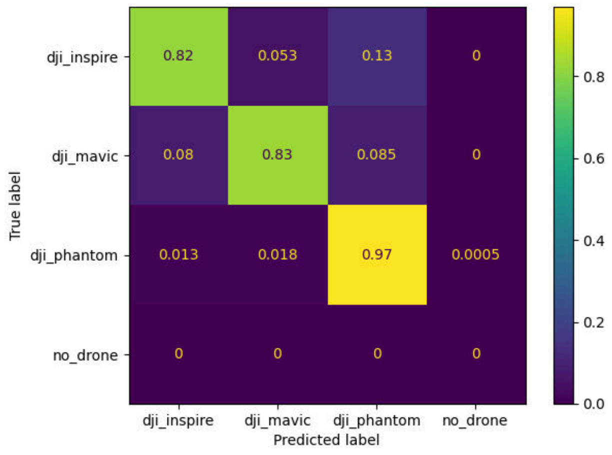


Fig. 5: ResNet50

4) Benchmarking Neural Architectures: To find the best performing architecture for this problem, a benchmark is performed by using open-source implementations in PyTorch [28]. Based on the results from a previous section, the layers are not frozen and Gaussian noise is added to the training dataset.

Architecture	Average Accuracy (%)	Average Precision (%)
AlexNet [29]	73.2	59.6
DenseNet121 [26]	84.0	75.9
DenseNet161 [26]	78.3	67.6
DenseNet169 [26]	87.2	80.8
DenseNet201 [26]	**92.4**	**88.6**
ResNet18 [30]	86.9	80.3
ResNet50 [30]	**91.6**	**87.3**
Wide ResNet50 [31]	75.8	63.7
ResNet101 [30]	83.1	74.5

TABLE II: Benchmark of convolutional neural architectures

Table II shows the results of the benchmark. Densenet201 and ReseNet50 architectures perform particularly well, achieving and average accuracy of 92.4% and 91.6% respectively.

III. Results

Figure 4 and 5 show the Densenet201 and ReseNet50 confusion matrices respectively. Although their overall accuracy is similar, the DenseNet201 has an even distribution of accuracy. ResNet50 is biased towards detecting one class and performs worse when detecting the other two classes. For this reason, we believe that DenseNet201 is the best architecture for this problem. Figures 6 and 7 show the accuracy and the loss for both the training and validation datasets of training the DenseNet201 model across 100 epochs.

A convolutional neural network that classifies drone models [13], reports an accuracy of 91.6%. The neural network is trained on images of drones scraped from the internet. However, a validation dataset is not used. When training neural network models, it is generally recommended to use separate validation and testing datasets to prevent overfitting [32]. The presented accuracy figure is alike to our validation accuracy shown in figure 6. The authors have designed their own convolutional neural network. They did not compare it with open-source state of the art methods such as AlexNet, Resnet, or DenseNet because they lacked computing power.

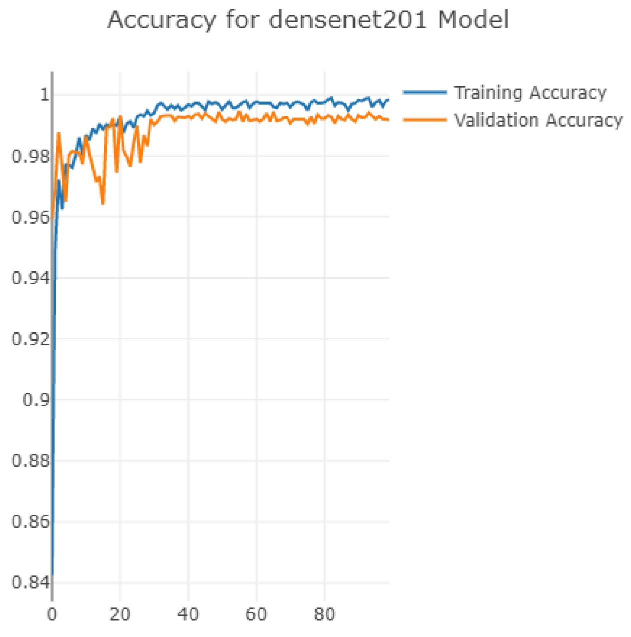


Fig. 6: Training accuracy across 100 epochs

Although the reported accuracy by [13] is similar to the accuracy of our model, we have presented reasons why we believe the two values are not comparable. We believe that if we tested the models side-by-side on the Anti-UAV dataset, the model from [13] would struggle to reach the reported

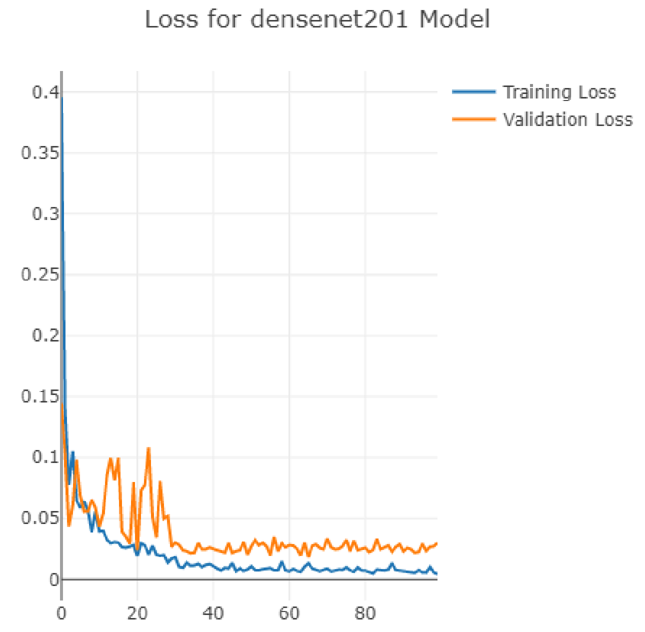


Fig. 7: Training loss across 100 epochs

accuracy of 91.6% because of the possibility of their model being overfitted. In contrast, we have taken steps, such as using domain randomization, to make sure that our model translates to a real-life dataset.

A model that attempts to classify loaded and unloaded drones [15], reports a mean average precision of 75.0%. The model uses the YOLOv2 object detection model. The authors were unable to find a public dataset of images of loaded and unloaded drones, so they created their own. The effects of size of the drone in the image are not examined. The mean average precision for detection methods is calculated differently to the average precision presented in our paper, because it relies on the overlap with the ground truth bounding box. So, it is hard to make a direct numerical comparison to our classification results.

The automotive detection method [18] is similar to our results in the sense that they train their model using synthetic images only. The model is tested on the real-life KITTI dataset. It uses the Faster-RCNN object detection model and reports 83.7% mean average precision when using domain randomization. Similarly with the previous method, mean average precision presented here is calculated differently to our average precision.

The convolutional neural network used to classify ships [20] is trained using both real and synthetic images, to predict the class of a ship (barge, cargo, container, or tanker) from overhead satellite images. This is similar to our approach, although for a different domain of ship identification. It uses a mix of real and synthetic ships for training. An accuracy of 96.9% is reported by using the ResNet34 architecture. When using a purely synthetic dataset, an accuracy of 59.2% is reported.

The accuracy of radio frequency detection of drones presented in [4] of 99.2% is higher compared with the visual detection approach presented in this paper.

IV. Conclusion

We presented a convolutional neural network trained on a purely synthetic dataset that correctly classified drone models in real-life video feeds of the Anti-UAV dataset. This method generated a more accurate convolutional neural network model than what is currently available in literature. To achieve this, we created a synthetic dataset by applying domain randomization (random positions, orientations, lighting conditions, and textures) to the 3D models of the drones. A benchmark identified that the DenseNet201 architecture showed the highest precision. Prior to training, we found that adding Gaussian noise to the training dataset increases the performance of the classifier. We have also found that freezing layers during the training has a negative effect on the performance of the classifier. We believe that this is an initial step in accurately identifying threats posed by different kinds of drones.

V. Further Work

There exist numerous avenues that could improve on this research. In this paper, we have only considered image classifiers, but we did not attempt to detect the drones in the images. This work could be expanded to object detectors such as Faster R-CNN, SSD, or YOLO. This would improve the practicality of the method as it would allow the drones to be detected by inputting the whole image rather than the bounding box around the drone.

This approach could be expanded to other challenges such as classifying drones and birds or classifying other drone models not mentioned here. Further information could be extracted from the synthetic images, such as the location of propellers, cameras, or payload. This could be used to produce more information about the drone. Different flying objects, such as planes or helicopters, could also be generated using synthetic images.

A study into increasing the size of the dataset should be conducted. We used 1,000 images, but it would be useful to find if producing more images improves the performance of the classifier. Adding real-life images to the training dataset could further improve the classification accuracy, which is something that was not investigated in this paper but has been successfully implemented in literature. To better understand the learnt features of the convolutional neural network, they could be visualized using sensitivity maps. Additionally, an investigation on the size of the drone in the input image should be conducted. To examine this effect more closely, the size of the input drone (in terms of pixels occupied) should be correlated with the accuracy of correct classification.

In this paper, we attempted to generate photorealistic images of drones, while applying domain randomization. However, we did not investigate the effects of adding or subtracting each of the methods. An ablation study, similar to the one done in [18], could be conducted to quantify the effect of each of the methods used for domain randomization.

Lastly, a dataset designed to test identification and detection methods would improve the state of research in this area. The Anti-UAV dataset was particularly useful in this problem, but it was not designed with the purpose of classifying the drone models.

Acknowledgement

This project is funded by the Department for Transport, Connected People Catapult, in collaboration with Aveillant Ltd and Autonomous Devices Ltd. All simulations have been carried out using the HPC facility (HILDA) at Digital Aviation Research and Technology Centre (DARTeC), Cranfield.

References

[1] G. Lykou, D. Moustakas, and D. Gritzalis, "Defending Airports from UAS: A Survey on Cyber-Attacks and Counter-Drone Sensing Technologies," en, *Sensors*, vol. 20, no. 12, p. 3537, Jun. 2020.

[2] H. Kolamunna, T. Dahanayaka, J. Li, S. Seneviratne, K. Thilakaratne, A. Y. Zomaya, and A. Seneviratne, "DronePrint: Acoustic Signatures for Open-set Drone Detection and Identification with Online Data," en, *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 5, no. 1, pp. 1–31, Mar. 2021.

[3] H. Liu, Z. Wei, Y. Chen, J. Pan, L. Lin, and Y. Ren, "Drone Detection Based on an Audio-Assisted Camera Array," en, in *2017 IEEE Third International Conference on Multimedia Big Data (BigMM)*, Laguna Hills, CA, USA: IEEE, Apr. 2017, pp. 402–406.

[4] I. Nemer, T. Sheltami, I. Ahmad, A. U.-H. Yasar, and M. A. R. Abdeen, "RF-Based UAV Detection and Identification Using Hierarchical Learning Approach," en, *Sensors*, vol. 21, no. 6, p. 1947, Mar. 2021.

[5] M. Jahangir and C. Baker, "Robust Detection of Micro-UAS Drones with L-Band 3-D Holographic Radar," en, in *2016 Sensor Signal Processing for Defence (SSPD)*, Edinburgh, United Kingdom: IEEE, Sep. 2016, pp. 1–5.

[6] H. Dale, C. Baker, M. Antoniou, and M. Jahangir, "An Initial Investigation into Using Convolutional Neural Networks for Classification of Drones," en, in *2020 IEEE International Radar Conference (RADAR)*, Washington, DC, USA: IEEE, Apr. 2020, pp. 618–623.

[7] B. Demir, S. Ergunay, G. Nurlu, V. Popovic, B. Ott, P. Wellig, J.-P. Thiran, and Y. Leblebici, "Real-time high-resolution omnidirectional imaging platform for drone detection and tracking," en, *Journal of Real-Time Image Processing*, vol. 17, no. 5, pp. 1625–1635, Oct. 2020.

[8] U. Seidaliyeva, D. Akhmetov, L. Ilipbayeva, and E. T. Matson, "Real-Time and Accurate Drone Detection in a Video with a Static Background," en, *Sensors*, vol. 20, no. 14, p. 3856, Jul. 2020.

[9] N. Jiang, K. Wang, X. Peng, X. Yu, Q. Wang, J. Xing, G. Li, J. Zhao, G. Guo, and Z. Han, "Anti-UAV: A Large Multi-Modal Benchmark for UAV Tracking," en, *arXiv:2101.08466 [cs]*, Feb. 2021. arXiv: 2101.08466 [cs].

[10] P. Voigtlaender, J. Luiten, P. H. S. Torr, and B. Leibe, "Siam R-CNN: Visual Tracking by Re-Detection," en, *arXiv:1911.12836 [cs]*, Apr. 2020. arXiv: 1911.12836 [cs].

[11] A. Coluccia, A. Fascista, A. Schumann, L. Sommer, M. Ghenescu, A. O. Avenue, and T. Piatrik, "Drone-vs-Bird Detection Challenge at IEEE AVSS2019," en, p. 7,

[12] C. Craye and S. Ardjoune, "Spatio-Temporal Semantic Segmentation for Drone Detection," en, in *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Taipei, Taiwan: IEEE, Sep. 2019, pp. 1–5.

[13] D. Lee, W. Gyu La, and H. Kim, "Drone Detection and Identification System using Artificial Intelligence," en, in *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, Jeju: IEEE, Oct. 2018, pp. 1131–1133.

[14] D. K. Behera and A. Bazil Raj, "Drone Detection and Classification using Deep Learning," en, in *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, Madurai, India: IEEE, May 2020, pp. 1012–1016.

[15] U. Seidaliyeva, M. Alduraibi, L. Ilipbayeva, and A. Almagambetov, "Detection of loaded and unloaded UAV using deep neural network," en, in *2020 Fourth IEEE International Conference on Robotic Computing (IRC)*, Taichung, Taiwan: IEEE, Nov. 2020, pp. 490–494.

[16] Y. Chen, P. Aggarwal, J. Choi, and C.-C. J. Kuo, "A deep learning approach to drone monitoring," en, in *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, Kuala Lumpur: IEEE, Dec. 2017, pp. 686–691.

[17] J. Peng, C. Zheng, T. Cui, Y. Cheng, and L. Si, "Using Images Rendered by PBRT to Train Faster R-CNN for UAV Detection," en, in *26. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2017*, 2018.

[18] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, "Training Deep Networks with Synthetic Data: Bridging the Reality Gap by Domain Randomization," en, *arXiv:1804.06516 [cs]*, Apr. 2018. arXiv: 1804.06516 [cs].

[19] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," en, in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC: IEEE, Sep. 2017, pp. 23–30.

[20] C. M. Ward, J. Harguess, and C. Hilton, “Ship Classification from Overhead Imagery using Synthetic Data and Domain Adaptation,” en, in *OCEANS 2018 MTS/IEEE Charleston*, Charleston, SC: IEEE, Oct. 2018, pp. 1–5.

[21] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database,” en, in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL: IEEE, Jun. 2009, pp. 248–255.

[22] N. Qian, “On the Momentum TeArmlgoinritGhrmads ient Descent Learning,” en, p. 14,

[23] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” en, *arXiv:1207.0580 [cs]*, Jul. 2012. arXiv: 1207 . 0580 [cs].

[24] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” en, *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, Jul. 2009.

[25] S. Hinterstoisser, V. Lepetit, P. Wohlhart, and K. Konolige, “On Pre-Trained Image Features and Synthetic Images for Deep Learning,” en, *arXiv:1710.10710 [cs]*, Nov. 2017. arXiv: 1710.10710 [cs].

[26] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” en, *arXiv:1608.06993 [cs]*, Jan. 2018. arXiv: 1608.069 93 [cs].

[27] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, “SmoothGrad: Removing noise by adding noise,” en, *arXiv:1706.03825 [cs, stat]*, Jun. 2017. arXiv: 1706.03825 [cs, stat].

[28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “PyTorch: An Imperative Style, High-Performance Deep Learning Library,” en, p. 12,

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” en, *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, May 2017.

[30] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” en, *arXiv:1512.03385 [cs]*, Dec. 2015. arXiv: 1512.03385 [cs].

[31] S. Zagoruyko and N. Komodakis, “Wide Residual Networks,” en, *arXiv:1605.07146 [cs]*, Jun. 2017. arXiv: 1605.07146 [cs].

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.