Binary Drone Classification

Authors

[redacted]

Matteo [redacted]

[redacted]

Students - Copenhagen Business School
MSc. Business Administration and Data Science

Abstract

The rapid rise of battlefield drones in the Ukraine conflict has created a critical need for labeled imagery for effective drone-type classification. In this paper we are exploring to what extent synthetic drone imagery can effectively supplement limited real-world data when training simple CNN models to accurately perform binary classification of drone types, and how this approach compares to traditional machine learning methods such as SVMs. The project involved key concepts like synthetic data generation, supervised learning methods and transfer learning in image classification. The datasets used included synthetic drone images created in Blender and real drone images. The models were built using TensorFlow for CNNs and scikit-learn for SVMs. The project demonstrated that training models solely on the synthetic data is insufficient, and that models trained directly on a mixture of data perform better than those trained using transfer learning. As a CNN trained on mixed data outperformed all other models with an F1-score of 0.93, we conclude that combining synthetic and real-world data when training CNNs is an effective approach when real-world data is limited.

Keywords: drone classification; synthetic data; supervised learning; convolutional neural networks; support vector machines; transfer learning

Introduction

In early March of 2025, the New York Times titled “A Thousand Snipers in the Sky: The New War in Ukraine” (Schwirtz & Browne, 2025). The Ukraine–Russia conflict has evolved into a war fought mostly with drones (Michel, 2023, pp. 34–35). This rapid shift towards drone warfare makes it clear that systems must be developed that can accurately and efficiently identify and classify drones.

Drones are taking on increasingly diverse roles. They are used for defensive and offensive strategic operations, reconnaissance, and direct combat. Therefore, both offensive and defensive capabilities depend on drone recognition technology (Tucker, 2022). The rise of these drones has accelerated military innovation in this field. However, it is not only European start-ups like Helsing (The War Zone, 2025) that are pushing for innovation in this sector; Ukraine’s opponents are too. Considering that military innovations usually evolve in secret, there is only limited access to real world image data of newly developed unmanned aerial vehicles (UAVs), making it challenging to classify them correctly on the battlefield.

This poses the question: To what extent can synthetic drone imagery effectively supplement limited real-world data in training simple Convolutional Neural Network (CNN) models to accurately perform binary classification of drone types, and how does this approach compare to traditional machine learning methods such as an Support Vector Machine (SVM)?

Related Work

Multiple contemporary research publications focus on drone identification. The shared goal among these studies is to improve upon drone detection and classification capabilities with the aim to improve situational awareness and response efficiency.

Wisniewski et al. (2022) developed such an approach. They classified drones by training a Convolutional Neural Network (CNN) entirely on synthetic images. Their dataset comprised realistic synthetic images generated using domain randomization techniques to improve the model´s ability to generalize to real-world conditions. They employed a customized CNN architecture made specifically for drone image classification. They achieved an accuracy over 90% when evaluated on real drone video footage, showcasing that synthetic data can act as a good substitute for real-world data, if it cannot be come by.

Furthermore, Mahdavi and Rajabi (2021) compared CNNs with traditional machine learning approaches such as Support Vector Machines (SVMs) with drones as a subject. They found that CNNs consistently outperformed SVMs on drone image datasets. This research showed the advantages of deep learning architectures in complex visual tasks.

Nalamati et al. (2022) investigated combining real-world drone images with synthetic augmentations to train robust classifiers. They introduce specialized image augmentations such as random rotations, scaling, and lighting variations to simulate real-world variability. They used a modified MobileNet architecture and found, that augmenting datasets with synthetic variations enhanced classification performance, outperforming classifiers that only were trained on real-world images.

All three studies showcase the viability of using synthetic and augmented data to not only train CNNs but ultimately enhance their ability to perform drone classification.

Conceptual Framework

Data Preparation

The data preparation process is comprised of five distinct stages: download/create, label, filter, augment and normalize. See Figure 1. Of our three datasets, one is synthetic and two are real-world images. They consist of images of different fixed-wing and multi rotor drones. To make the data usable for our case, we added bounding boxes and class labels, where they were missing. To ensure high quality we filtered out images that did not meet our requirements. To compensate for the lack of real-world data, we performed data augmentation on the two real-world datasets. To ensure the smooth running of all our models we normalized the data, so that the model inputs were consistent. To ensure comparability, 30% of the real-world drone images were split into a test set with an emphasis on a 50/50 class balance.

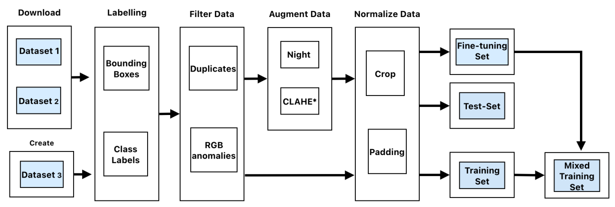

Figure 1: *Data preparation process*

Training Strategy

During this project, six models were trained. Four of these were based on CNNs, and three of them utilized SVMs.With one approach combining both methods. See Table 1. The evaluation of all models was conducted using the same test set. See Figure 2.

Table 1: *Data used to train each model.*

Model	Data
CNN1	Trained on synthetic data (Training-Set).
CNN2	Transfer model, fine-tuned on real-world data (Fine-tuning-Set), using CNN1 as a base model.
CNN3	Trained on both synthetic and real-world data (Training- and Fine-tuning-Set).
SVM1	Trained on synthetic data (Training-Set).
SVM2	Transfer model, fine-tuned on real-world data (Fine-tuning-Set), using CNN1 as a base model.
SVM3	Trained on both synthetic and real-world data (Training- and Fine-tuning-Set).

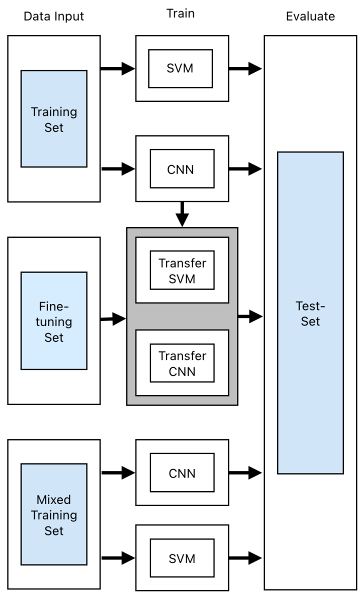

Figure 2: *Training Strategy*

Methodology

Dataset Description

This project makes use of three datasets. One consists of synthetic images of different types of aerial drones in various environments. These images are generated using Blender's Python API. The second dataset comprises real-world images of multi-rotor drones and the third consists of fixed-wing drones.

Table 2 shows the instances in each Dataset loaded before pre-processing.

To improve understanding of the data on which the models were trained and tested, we conducted the following exploratory data analysis on the filtered and augmented images. For simplicity, we are no longer differentiating between the three distinct datasets, but rather between synthetic or generated

Figure 3: *Synthetic multi-rotor drone*

Figure 4: *Synthetic fixed-wing drone*

Figure 5: *Real multi-rotor drone*

Figure 6: *Real fixed-wing drone*

Table 2: *Summary of datasets before preprocessing.*

Dataset	Source	Drone Type	Images
Dataset 1	Real-World	Fixed-Wing	554
Dataset 2	Real-World	Multi-Rotor	2 456
Dataset 3	Generated	Both Classes	2 398

images (Figure 3 and 4) and real-world images (Figure 5 and 6). The real pictures were manually filtered to confirm that they were all taken from the ground while the drone was in flight. Table 3 shows the distribution between multi-rotor and fixed-wing drones. Both the real-world dataset and the generated dataset contain more multi-rotor drones than fixed-wing drones. Multi-rotor drones have an overall share of 54.50%, while fixed-wing drones have an overall share of 46.50%.

Table 3: *Image counts by type*

Drone Type	Real	Generated	Overall
Fixed-wing drone	678	1090	1768
Multi-rotor drone	711	1323	2034

Due to the fact that real-world images stem from various data sources, their sizes vary. Figure 7 illustrates the size distribution of the real-world pictures. With the smallest image being 209x241 pixels and the largest 5760x3840 pixels. This highlights the importance of robust resizing methods.

The bounding boxes stored in the metadata surround the drones, thereby indicating their positions in the images. These boxes vary in size and shape.

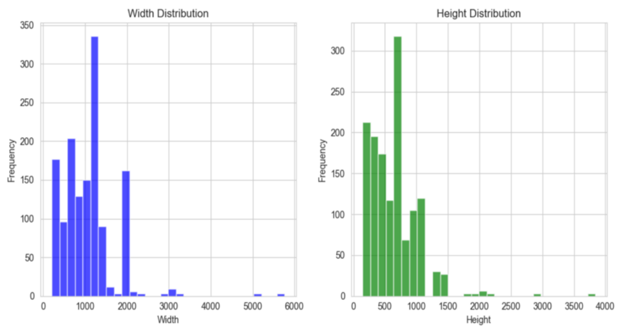

Figure 7: *Image Size distribution*

Figure 8 and 9 illustrate their position by displaying heatmaps for the normalised positions of the bounding boxes. In both cases, the centre of the bounding box is most often located around the centre of the image; however, for the generated images, the boxes tend to spread more towards the top.

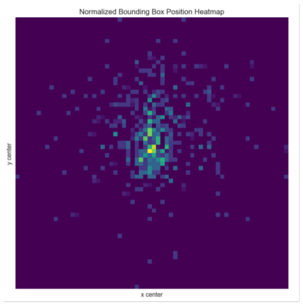

Figure 8: *Heatmap for real images*

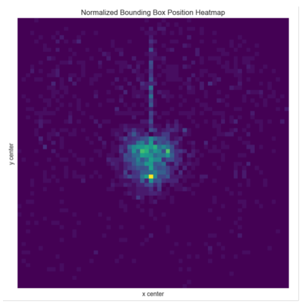

Figure 9: *Heatmap for synthetic images*

Data Preprocessing

Use of Generative Artificial Intelligence The code described and used in the following sections was partly generated or debugged using GenAI.

Image Generation The synthetic images were generated within a three-dimensional space in Blender using a series of methods to set up a scene, position the drones, store relevant annotations, and render an image. In the following, we will discuss the framework and process of generating the images.

The initial phase of the process entailed the generation of a unique scene, informed by a series of specifications. To accomplish this, we initially chose one of four drone models that would be used in the image. The drone in question was situated within the bounds of a dome shape, with a realistic rotation. The camera angle was determined through a calculation of the drone's position and the camera's orientation, with the camera itself being directed towards the drone. The configuration is illustrated in Figure 10, with the cone denoting the camera positioned at the origin.

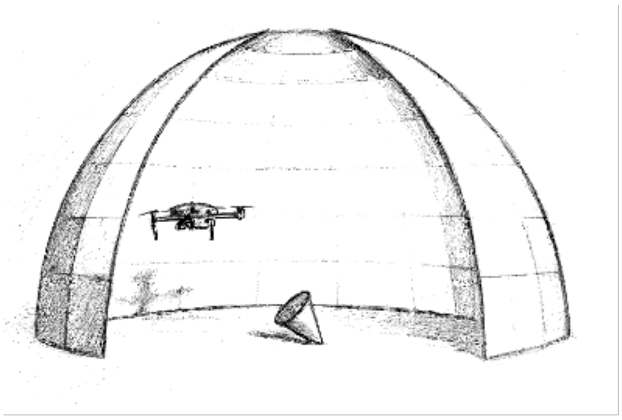

Figure 10: *Illustration of Blender setup*

Subsequently, the scene was transferred to a separate class. This class calculated and stored annotations in a CSV file, loaded an HDRI file to generate a realistic background and lighting, and rendered the image. The class stored the relevant labels for the drones and computed bounding boxes based on the camera rotation and drone location. The parameters of the camera's rotation, the position of the drone, and the designated HDRI file were documented for future reference.

Data Filtering To ensure consistency and that each image contributes equally to the model's learning, as well as to reduce noise, we filtered the data in three steps before further processing. First, the real-world image datasets (Dataset 1 and Dataset 2) were filtered for drone pictures taken from the ground, rather than from other aerial vehicles. This process involved reviewing both datasets and removing all images that do not fulfill the requirements. Secondly, all duplicates were removed. An image was classified as a duplicate if its perceptual hash matches that of another image. This method, proposed by Li et al. (2016), was used due to its lightweight nature, which made it easy to run through thousands of high-resolution pictures. Using this method, 25 duplicates were removed across all datasets. The final step of the filtering process involved detecting RGB anomalies. This was achieved by calculating the average red, green and blue values across an image. Images with an average value of less than five or greater than 250 were filtered out. The aim of this method was to detect pictures that were either too bright or too dark.

Table 4: *Summary of detected duplicates and RGB outliers*

Dataset	Duplicates	RGB Outliers
Dataset 1	6	0
Dataset 2	2	0
Dataset 3	17	0

Data Augmentation To have enough real-world data for a solid test set, as well as for fine-tuning CNN2 and SVM2, the number of instances in Datasets 1 and 2 had to be increased. Rather than searching for more data that fulfilled the requirements, we opted for data augmentation. Using the Albumentations library, which can handle bounding boxes in the metadata, we created night and Contrast Limited Adaptive Histogram Equalization (CLAHE) versions of each image. This method enabled us to increase the total number of real-world pictures from 463 to 1,389.

Data Normalization To provide consistent training and testing data for the models, we normalised the inputs. We used the bounding boxes from the metadata to crop the drone out of the image. To maintain the aspect ratio, we added padding to each drone, making it rectangular, before downscaling it to 128x128 pixels for the CNNs and 64x64 pixels for the SVMs.

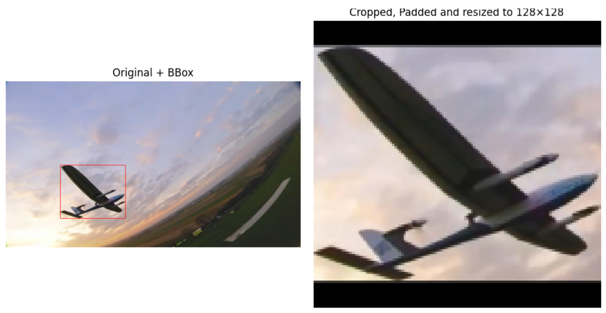

Figure 11: *Picture before normalization (including bounding-box) (left) and after (right)*

Modelling Framework

Convolutional Neural Networks Initially, three base architectures were tested for CNN1 and CNN3, splitting the training set into training and validation sets. To find the optimal model configuration we used the built-in Keras hyperparameter tuning function. The models tested comprised two, three or four convolutional layers with filter sizes of 32, 64, 128 and 256, respectively, and incorporated regularisation methods.

For each convolutional block, parameters like kernel size, activation function and the L2 regularization parameter were tuned. Optional batch normalization was deployed and each block used max pooling with kernel size 2x2. The activation functions to be considered was ReLu, ELU, and leaky ReLu. Max pooling layers were used in all blocks. Every model further contained a flatten layer that converted two-dimensional matrix data to a one-dimensional vector, and a fully connected dense layer with 128 units, where the activation function and dropout rate were tuned. Finally, the output layer where a single unit with sigmoid activation as the model concerns a binary classification task. The relevant kernel size was 5x5 and we consequently used filter sizes of 32, 64, 128, and 256. We found that the above provides a balance between capturing fine details and broader contextual information. The activation functions were chosen for their computational efficiency, and their ability to handle complex patterns and improve learning dynamics. Max pooling was used to reduce computational load, memory usage, the number of parameters, and add some level of variance to small translations. (Géron, 2019, p. 456) The dense layer was used for feature extraction, flexibility, and nonlinearity. It uses an Adam optimizer, where the learning rate is tuned using Keras tuner, and a binary cross-entropy loss function.

As we found the models to be prone to overfitting in the early stages of the project; a series of measures were deployed to avoid this. This includes early stopping, L2 regularization, batch normalization and dropout.(Géron, 2019, pp. 364-368)

The best performing model, trained on the generated data, was used as a basis for training a transfer-CNN on the real-world training set. This was motivated by the practical applicability of transferring models trained on synthetic data to real-world applications with a limited amount of data. The transfer model was first expanded by adding an extra layer and then fine-tuned. In the next step, the last two layers of the previous model were unfrozen and fine-tuned. This model was then trained on the fine-tuning dataset.

Support Vector Machines To compare the more complex neural network approach with a non-neural one, we opted to train a Support Vector Machine (SVM) classifier using the same dataset as that used by the neural networks. This supervised learning method aims to find a decision boundary — a straight line in two dimensions or a flat hyperplane in higher dimensions — that best separates the data points by class. The SVM selects the boundary that maximizes the margin between the classes. If we impose the strict condition that all instances lie on the correct side of the margin, this is known as hard margin classification. However, hard margin SVMs have two key limitations: they only work with linearly separable data and are highly sensitive to outliers (Géron, 2019, p. 156). To address these limitations, we adopted a more flexible soft margin approach and used grid search to determine the optimal hyper parameters (Géron, 2019, pp. 79-81). Prior to training, we reduced the dimensionality of the data to using Principal Component Analysis (PCA) with the aim to preserve approximately 95% of the explained variance on both the training and the test set (Géron, 2019, pp. 222-226). For SVM2, we applied feature-based transfer learning. High-level feature vectors were extracted from CNN1 by removing its final classification layer. These vectors were then used as input to train an SVM on real-world drone data. As with SVM1 and SVM3, we optimized the SVM's hyper parameters using cross-validated grid search.

Evaluation Metrics

All models were evaluated using a test set, which only contained real-world images. To measure model performance, we compared the respective F1 scores. The F1 score is the harmonic mean of precision and recall (Géron, 2019, p. 94). Precision is defined as the proportion of true positive predictions among all positive predictions made by the model. It quantifies how accurate the model is when it predicts the positive class and is calculated with formula (1). A high precision score means that there are few false positives.

$$\text{Precision} = \frac{TruePositives}{TruePositives + FalsePositives} \tag{1}$$

Recall, or Sensitivity or True Positive Rate measures the proportion of actual positive cases that are correctly identified by the model. It captures the model's ability to detect all relevant instances of the positive class and is calculated with the formula (2). A high recall means few false negatives

$$\text{Precision} = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{2}$$

Both precision and recall are needed to calculate the F1-score as can be seen in formula (3).

$$F_1 = \frac{2}{\text{Recall}^{-1} \ + \ \text{Precision}^{-1}} \tag{3}$$

Precision and recall represent a trade-off. Increasing recall typically results in more false positives, which lowers precision, and vice versa. The F1-score captures this balance by combining both metrics into a single harmonic mean. As a result, maximizing the F1-score requires improving both precision and recall simultaneously, rather than optimizing one at the expense of the other. To evaluate overall model performance, we report the standard F1-score, which balances precision and recall for the positive class. This metric is particularly useful when both false positives and false negatives carry consequences. Although our dataset is relatively balanced, using the F1-score ensures that both rotor and fixed-wing drone classifications are assessed with equal emphasis (Géron, 2019, pp. 93-95).

Results

We evaluated all our models using the same test set. The best-performing model was CNN3, which was trained on mixed data with no transfer learning, achieving an F1-score of 0.92. Second is SVM3 which was also trained on mixed data, with an F1-score of 0.83. The worst-performing model was CNN1, which was trained on synthetic data and did not employ transfer learning, with an F1-score of 0.54. Further details and data for all models can be found in Table 4.

A clear trend is that models trained solely on synthetic data perform significantly worse than those trained using other approaches. The performance metrics show that a CNN trained on a combination of real and synthetic data substantially outperforms a CNN that was initially trained on synthetic images only and subsequently fine-tuned using real-world data. This remains true even though both approaches ultimately used the same set of data points, indicating that exposure to both domains simultaneously yielded more robust feature representations than sequential synthetic-to-real training. The same effect could be observed with SVMs. Training an SVM directly on mixed data produced a better result (F1 = 0.83) than extracting features from a CNN that was initially trained on synthetic data (F1 = 0.81). However, the difference is less significant compared to CNNs.

Table 5: *Precision, recall, and F_1-score by drone type for each model*

CNN1 ($F_{1,\text{macro}} = 0.536$)			
Drone Type	Precision	Recall	F_1-score
Fixed-wing drone	0.5474	0.8647	0.6704
Multi-rotor drone	0.6782	0.2850	0.4014
CNN2 ($F_{1,\text{macro}} = 0.6908$)			
Drone Type	Precision	Recall	F_1-score
Fixed-wing drone	0.6926	0.686	0.6893
Multi-rotor drone	0.689	0.6956	0.6923
CNN3 ($F_{1,\text{macro}} = 0.9275$)			
Drone Type	Precision	Recall	F_1-score
Fixed-wing drone	0.9311	0.9227	0.9272
Multi-rotor drone	0.9234	0.9324	0.9279
SVM1 ($F_{1,\text{macro}} = 0.5842$)			
Drone Type	Precision	Recall	F_1-score
Fixed-wing drone	0.5799	0.6135	0.5962
Multi-rotor drone	0.5897	0.5556	0.5721
SVM2 ($F_{1,\text{macro}} = 0.8163$)			
Drone Type	Precision	Recall	F_1-score
Fixed-wing drone	0.8325	0.7927	0.8112
Multi-rotor drone	0.8018	0.8406	0.8208
SVM3 ($F_{1,\text{macro}} = 0.8380$)			
Drone Type	Precision	Recall	F_1-score
Fixed-wing drone	0.8211	0.8647	0.8426
Multi-rotor drone	0.8571	0.8116	0.8337

Complexity and Running Time

In addition to assessing model performance, it is important to consider the complexity and running time of the models. A less well-performing model may be preferable if it is simpler or has a lower running time. Table 6 shows the training and inference times of our models on an Apple M3 Chip with 16GB of RAM. The training time of the transfer models is determined by the cumulative training time of all their components. Although CNN3 has the longest training time, it is the most efficient model. By way of comparison, SVM3 takes approximately one-sixth of the time to train, but lacks the performance of the stronger CNN.

Table 6: *Training and inference times for each model*

Model	Training Time (s)	Inference Time (s)
CNN1	3315.71	1.5208
CNN2	3375.16	1.1751
CNN3	3628.07	1.1719
SVM1	266.58	0.0392
SVM2	3328.12	1.6205
SVM3	595.09	0.0522

Inference time is also an important factor, particularly for real-world use cases where classification must happen in near real time. We use the time taken by the different models to classify the test set as an indicator of their respective inference times. It is important to note that our model pipelines may not be fully efficient, which could affect the accuracy of this measurement. Nevertheless, this method has been found to be adequate. Pure SVM models significantly outperform CNN-based models in terms of inference time. However, there is a trade-off between performance and speed. Furthermore, CNN3, the model with the highest F1-score, has a shorter inference time than all the other CNN-based models. This is because it is less complex: CNN3 has one fewer layer than both CNN1 and CNN2.

When choosing a model, it is important to consider running time and complexity. If the use case requires fast inference and low resource usage, a support vector machine (SVM) would be the better option. However, if performance is the top priority, a convolutional neural network (CNN) might be a better option.

Discussion

Technical Considerations

Model Comparison In general it can be said that classical machine learning approaches such as SVMs are easier to train and classify images faster than the more complex deep learning approaches using neural networks such as CNNs.

SVM Analysis When building the SVMs, we used grid search to find the optimal hyper parameters. The objective is to strike a balance between maximizing the separation between the two classes and minimizing margin violations. This is known as 'soft margin classification' (Géron, 2019, p. 157). This balance can be controlled using the hyper parameter C: a smaller C value leads to a wider margin between the two classes, and vice versa. We tried hyper parameters of 0.1, 1 and 10 for all three SVMs and achieved the best results with a hyperparameter of 10 across all the models. Further improvements to the models could be achieved by fine-tuning this hyperparameter further and trying even larger C values. To verify that increasing C would not lead to infinite improvement, we also ran the models with a hyperparameter of 100 and achieved worse results. This indicates that the sweet spot lies between 10 and 100.

CNN Analysis For the more complex CNNs, we conducted a random search to find the optimal model configuration. 5x5 kernels were used for all models. CNN1 and CNN2 used four and five layers respectively (with an additional layer added to CNN2 for fine tuning), while the best-performing CNN3 only required three layers. This showed that more layers and greater complexity do not necessarily produce better results. Regarding overfitting measures, early stopping, dropout and L2 regularization were applied similarly across all models.

To further compare the decisions made by CNN models we use a Grad-CAM analysis. Grad-CAM was chosen due to its ease of interpretation and visualisation (Selvaraju et al., 2017). More specifically, we use the final layer of the CNNs for visualisation because it strikes a balance between capturing high-level semantic information and retaining spatial details. (Selvaraju et al., 2017). The final layers of our CNN models are fully connected layers with a single neuron. This layer produces a logit value between 0 and 1, with values above 0.5 indicating multi-rotor drones and values below 0.5 indicating fixed-wing drones.

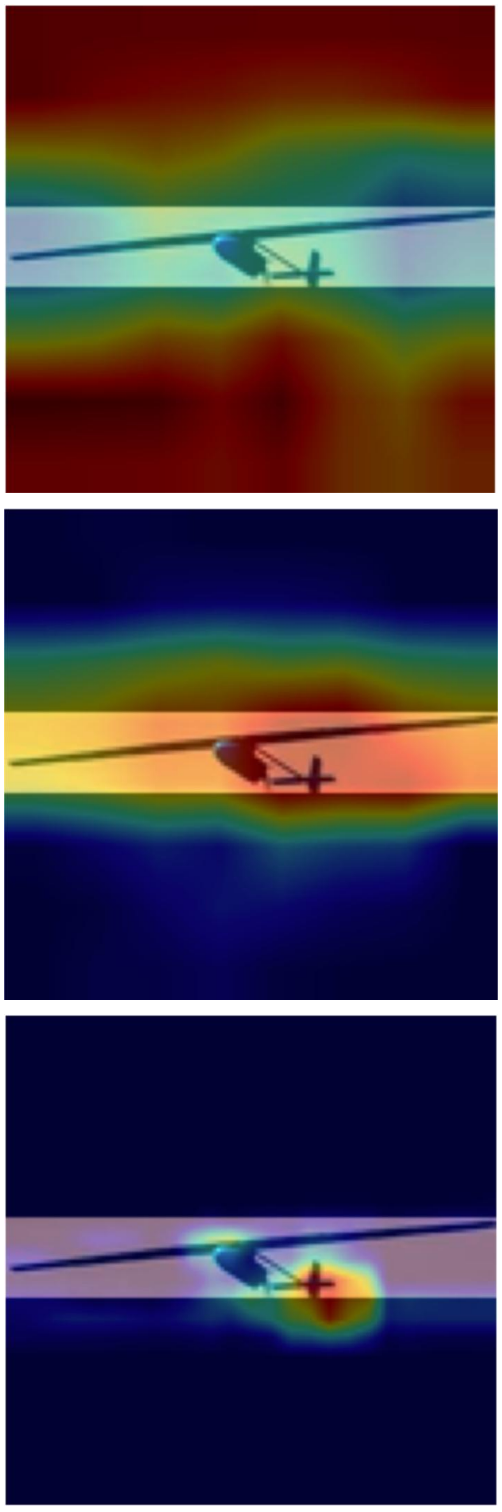
Figure 12: *Grad-CAM heatmaps for CNN1, CNN2, and CNN3.*

Figure 12 illustrates Grad-CAM images that compare the focus of spatial details for each model, revealing some interesting insights. The poorer-performing models appear to have a notable bias towards the padding. They appear to have learned to associate padding with a certain class. This is particularly evident in CNN1, where a significant amount of padding at the top and bottom of an image may suggest a fixed-wing drone, as these tend to be wider. CNN2 performs somewhat better, valuing the padding less. The best-performing model, CNN3, assigns almost no value to the padding. Rather than valuing it and identifying multi-rotor drones, CNN3 seems to recognise patterns unique to fixed-wing drones. However, the F1 score of 0.93 still indicates room for improvement.

Limitations

Despite the decent results, particularly from CNN3, several limitations emerged, primarily relating to the quality of the dataset, the evaluation choices made and the data processing methods employed.

One major issue is the domain gap between synthetic and real-world images. Despite the synthetic data being generated with randomized drone positions, lighting, and HDRI backgrounds in Blender, it still lacked the nuanced complexity of real-world images. The textures, lighting artifacts, and noise patterns found in field conditions were underrepresented, which likely hindered models like CNN1 and SVM1, which were both trained on synthetic data only. To close this gap, more realistic synthetic generation, such as style transfer, could help reducing this gap.

Furthermore, CNN3´s performance advantage may stem not just from its exposure to both domains but also from a simple increase in data volume. Models trained on more data tend to perform better in general, so further testing is needed to distinguish whether diversity or quantity played the larger role. Training models on matched dataset sizes but with varying domain composition would help clarify this.

For evaluation we used the F1 score, which balances precision and recall. However, for high-risk use cases such as detecting kamikaze drones, recall is far more critical. In such contexts, adopting the F2-score, which weighs recall more heavily, would be more appropriate for assessing real-world utility.

Another important artefact stems from bounding box shape differences. Fixed-wing drones tended to have more oblong bounding boxes, while multi-rotors were often more compact. When these images were cropped and padded to square input shapes, the amount of background padding varied systematically between the two classes. As a result, some models may have learned to associate padding amount or shape with the drone type, despite padding being a preprocessing artifact unrelated to the drone´s actual features.

Finally, there was a high amount of pre-processing and augmentation. This was necessary in our case, but it may have introduced artificial patterns: techniques like CLAHE or night filters improve variation but risk creating biases if not grounded in real data characteristics. Over-conditioning the model through aggressive augmentation can reduce its ability to generalize.

Ethical Considerations

The use of AI for drone classification in military contexts raises ethical challenges. Responsible AI (RAI) emphasises fairness, transparency, robustness, and accountability (Stryker, 2024). Although CNNs outperform traditional methods such as SVMs, they are less transparent, making their decisions difficult for humans to understand and predict. If used in a combat situation in conjunction with an automated system, misclassification by such a CNN can have fatal consequences. Therefore, when using such systems in combat applications, they should never be integrated into fully automated systems to minimise the risk of misclassification as much as possible by following an augmented approach.

Conclusion and Future Work

For this project, we trained six models to classify whether a drone is fixed-wing or multi-rotor. We built two convolutional neural networks (CNNs) from scratch: one was trained on synthetic data, and the other on a combination of synthetic and real-world data. For comparison, we also built two SVMs using the same approach. Finally, we created two transfer models based on the CNN that was trained using synthetic data. One of these was a CNN and the other was an SVM using feature-based transfer learning. All the models were evaluated using the same real-world image test set. Overall, the models trained solely on synthetic data performed significantly worse than all the other models (SVM: F1 = 0.58; CNN: F1 = 0.54). Training on mixed data yielded better results for both CNNs and SVMs (SVM: F1 = 0.83; CNN: F1 = 0.93) than transfer learning did (SVM: F1 = 0.82; CNN: F1 = 0.69). The difference between the two CNNs is more significant than that between the two SVMs here.

This project demonstrates how a combination of synthetic and real-world data can be used to train classifiers for drone identification, opening up a variety of possibilities for future research in this area. Firstly, we can refine and optimize the generation of synthetic data (for example, by using more realistic and diverse backgrounds according to the application area) to further improve the performance of CNNs. Secondly, extending our framework to multi-class problems, such as distinguishing kamikaze drones from reconnaissance UAVs, could reveal where alternative metrics, such as an F2-score that gives more weight to recall, add value beyond the standard F1. Finally, integrating state-of-the-art object detectors such as YOLOv8 would enable real-time, in-flight classification of drones from video streams, paving the way for deployable, low-latency surveillance systems.

References

Géron, A. (2019). *Hands-on machine learning with scikit-learn, keras, and tensorflow: Concepts, tools, and techniques to build intelligent systems* (2nd ed.). O'Reilly Media.

Li, X., Li, J., & Huang, F. (2016). A secure cloud storage system supporting privacy preserving fuzzy deduplication. *Soft Computing*, *20*(4), 1437–1448.

Mahdavi, F., & Rajabi, R. (2021). Drone detection using convolutional neural networks. https://arxiv.org/abs/2107.01435

Michel, A. H. (2023). *Drone wars: Pioneers, killing machines, artificial intelligence, and the battle for the future*. Princeton University Press.

Nalamati, M., Kapoor, A., & Srivastava, A. (2022). Robust drone classification using synthetic data augmentation. *IEEE Access*, *10*, 65423–65433. https://doi.org/10.1109/ACCESS.2022.3184229

Schwirtz, M., & Browne, M. (2025, March). *The second drone age: Defining war in the 2020s* [The New York Times]. https://www.nytimes.com/interactive/2025/03/03/world/europe/ukraine-russia-war-drones-deaths.html

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-CAM: Visual explanations from deep networks via gradient-based localization. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 618–626. https://openaccess.thecvf.com/content_ICCV_2017/papers/Selvaraju_Grad-CAM_Visual_Explanations_ICCV_2017_paper.pdf

Stryker, C. (2024, February). *What is responsible ai?* [IBM Think]. https://www.ibm.com/think/topics/responsible-ai

The War Zone. (2025, January). *Ai-enabled hx-2 kamikaze drones now in production for ukraine*. https://www.twz.com/air/ai-enabled-hx-2-kamikaze-drones-now-in-production-for-ukraine

Tucker, P. (2022, October). *Ukraine's drone war has become a lab for military innovation* [Defense One]. https://www.defenseone.com/technology/2022/10/ukraines-drone-war-has-become-lab-military-innovation/378770/

Wisniewski, M., Rana, Z. A., & Petrunin, I. (2022). Drone model classification using convolutional neural network trained on synthetic data. *Journal of Imaging*, *8*(8), 218. https://doi.org/10.3390/jimaging8080218