Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 02, June 2023.

FORENSIC ANALYSIS OF DRONE COLLISION WITH TRANSFER LEARNING

Arda Surya Editya, Tohari Ahmad and Hudan Studiawan

(Received: 13-Jan.-2023, Revised: 19-Mar.-2023, Accepted: 4-May-2023)

ABSTRACT

Drones are among the devices that are used in many different activities. There is a time when drones have accidents and authorities need to find the cause. Drone forensics is used to determine the cause of an accident. The analysis phase of drone forensics is one of the most important steps in determining accident causes. In this paper, we applied a deep-learning technique to classify drone collisions. We investigate the use of Inception V.9 as the deep-learning framework. Additionally, this study compares the performance of the proposed method with other techniques, such as MobileNet, VGG and ResNet, in classifying drone collisions. In this experiment, we also implement transfer learning as well as its fine tuning to speed up the training process and to improve the accuracy value. Additionally, our investigation shows that Inception V.9 outperforms other frameworks in terms of accuracy, precision and F1 score.

KEYWORDS

Inception, MobileNet, VGG, RestNet, Drone forensics, Transfer learning, Network infrastructure.

1. INTRODUCTION

Drones are devices that have been developed and used in both academia and industry over the past few years. This type of device can assist humans working in many sectors to complete a particular task [1]. When an accident occurs, for example when a drone breaks or crashes, users need to find the cause of the incident. Drone forensics is a sub-discipline of digital forensics dedicated to drone investigations. Drone forensics can be used to discover evidence from various types of media collected by a drone to determine the cause of an accident [2].

As an example, when a drone has an accident while taking a picture in a hilly area, the owner can find out what caused the accident. To find out the cause of the accident, one can look at the media installed on the drone. This includes the camera on the drone and the sensor data log on the drone. Therefore, the causes of drone accidents can be known. In drone grand prix, drone-collision classification is used as evidence. To determine whether there is a foul in a race where drones collide, the referee needs to know the cause of the collision. Furthermore, drone-collision classification can be used in military applications. It is possible for a drone to collide with an enemy or to be attacked by an enemy if it flies into enemy territory. In order to conclude that a drone collision occurred, the pilot or the authority needs evidence. Alternatively, when drones are used in the entertainment industry, the director or the industry may need to know how the drone was involved to determine whether the worker WBB is negligent.

In drone forensics, there are three types of evidence that can be examined: antemortem, live and post-mortem data [3]. Several previous studies used post-mortem data to analyze the cause of a drone crash, such as a collision. Pedro et al. [4] used post-mortem data to build a model that could detect a collision with a moving object. In their study, MobileNetV2 was applied to extract features from acquired video and an agglomerative algorithm was then applied to determine object positions. However, this approach has certain weaknesses. For example, it took a long time to train the model and its accuracy could still be improved.

Furthermore, Jain et al. [5] examined basic drone architecture to gain evidence that can be applied to forensic analysis. In their study, they also proposed a generic drone forensic model to improve the digital investigation process. The approach has helped another study begin investigating drone forensics and further improvements can be made with an analysis method based on deep learning.

A study on improving drone forensics was also conducted by Ashraf et al. [6]. In their study, Ashraf et al. [6] applied ResNet50 to detect drones from drone camera data. In their study, they used six datasets

A. Editya, T. Ahmad and H. Studiawan are with Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia. Emails: ardasurya.tif@unusida.ac.id, tohari@if.its.ac.id and hudan@its.ac.id

and achieved an average Fl score of 0.92. However, using a different deep-learning architecture may give a better result.

Contribution. In this paper, we investigate drone collision using InceptionV3 [7] to classify collision events from frames extracted from drone videos. To reduce the computational process and speed up the training process, we implement a transfer-learning technique. The transfer-learning weights are taken from ImageNet [4]. We perform fine-tuning based on the weights used in InceptionV3 to obtain the most appropriate weights for the case of a drone collision.

This paper is organized as follows. Section 2 discusses related work in drone-collision detection and drone forensic framework. We describe how we analyze the collected data, validated our approach and implemented our scheme in Section 3. This paper presents proposed solutions and describes the experimental results from our transfer-learning method in Section 4. Finally, we present the conclusion and possible avenues for future research in Section 5.

2. RELATED WORK

2.1 Drone Collision Detection

Several drone research projects are focusing on collision detection. Pedro et al. [4] used post-mortem data to analyze drone collisions. They used MobileNetV2 and a convolutional neural network (CNN) to extract features from post-mortem data and detect collisions. An agglomerative algorithm was then used to detect the positions of the objects. The accuracy of the developed model increased the mean Average Precision (mAP) from 54.4% to 91.4%.

Furthermore, Anwar et al. [8] used training data from a drone computer simulation and transferred the results to the real operation of a drone. Their method was able to reduce both energy consumption and the detection error for collisions. As the research used computer simulation at the training stage, the weights of the model could not represent actual conditions. In a real situation, the model is strongly influenced by the drone camera.

In addition, Ouahouah et al. [9] used deep reinforcement learning to make a drone fly without colliding with any obstacles. This takes a lot of time to produce the model, because the method that they used is different from deep learning. Deep reinforcement learning involves training an agent to make a series of decisions in an environment with the goal of maximizing accuracy.

A study that implemented deep learning into the drone was also conducted by Obaid et al. [10]. The researchers used artificial neural networks (ANNs) to train a drone using simulation software. The accuracy obtained was 85.3% and this value can be optimized using a different architecture with a good dataset.

Rehmatullah and Kelly [11] used parallel tracking and mapping (PTAM) and dense tracking and mapping (DTAM) to detect an object that may have been hit by a drone in real time. This technique had an accuracy of 34.53% in detecting all objects from images captured by the drone. The drawbacks of this approach are that accuracy is relatively low and many GPU resources are needed to train the model.

Furthermore, Lima et al. [12] conducted research about collision detection and obstacle classification based on the physical interaction between a drone and the environment. The experiment used three classes of classification, including no collisions, collisions with soft objects and collisions with hard objects. The experiment used three different methods: FNN (Fully Convolutional Network), ResNet and Inception. The most accurate result of the experiment was from Inception with an accuracy of 98.7%.

Furthermore, Park and Kim [13] conducted research about an algorithm to detect fast drone movement. The algorithm needs the location and time of the drone. It makes the collision of the drone categorized as a hard collision or a soft collision.

Drone-collision research includes not only digital imaging data, but also drone sensor value. Weinert et al. [14] initiated research on drone-collision calculation. In their work, they performed simulations of drone collision impact based on dimension and maximum gross take of weight (MGTOW). Based on the results of this experiment, collisions with dimensions and MGTOWs have an effect on damage.

Moreover, Xiaohua et al. [15] concluded in their research that aircraft-nose shapes reduce drone-collision damage. The degree of aircraft nose damage is determined by the shape of the nose, especially when the nose is collided with a drone. The research used ten simulations with a variety of parameters. The same problem was studied by Zhang et al. [16]. To minimize drone-collision risk, they calculated the aircraft's position when starting flying. Additionally, this research determined the equation that is used to optimize the calculation of the risk of aircraft collisions with drones.

In addition, the drone may perform a show when deployed at several events, which results in a swarm formation of drones. The drones need to calculate to avoid colliding with each other when they form a swarm. This problem was also studied by Naveed et al. [17]. Their research proposed an algorithm to optimize drone positioning in a swarm formation. In the research, they used the distance between drones as well as the value of standard deviation to make drones not collide with each other.

A similar research was conducted by Sabetghadam et al. [18] aiming to ensure that drones did not collide when flying together. The drones may assemble in a formation while flying in the air. In this case, drones may collide if the algorithm is incorrect. The research adopted Voronoi-based space partitioning to derive local constraints that guarantee collision avoidance with neighboring vehicles.

2.2 Drone Forensic Framework

Drone forensics is a method that collects all artifacts stored in the internal or external memory of the drone system. Such data includes flight records, flight control, sensor data and internal monitoring data. Al-Dhaqm et al. [3] proposed a forensic framework that could be used for most types of drones and comprised of three phases, specifically the preparatory, data-acquisition and data-analysis phases. The phases of this framework are described below.

1) Preparatory phase: This phase determines what components are installed on the drone. In addition, forensic evidence is identified, a pilot interview is conducted and the tools used for drone forensics are prepared.
2) Data-acquisition phase: All data artifacts are collected from media using live and post-incident acquisition and a report on the information collected from the drone is created.
3) Data-analysis phase: In this phase, the data is analyzed and conclusions are drawn regarding the reasons for damage to the drone.

We take this framework as the starting point for this paper and focus on the data-analysis phase. The goal of this work is to detect drone-collision cases accurately.

3. PROPOSED METHOD

As shown in Figure 1, the proposed method uses a video dataset from ColaNet [4]. This method begins with observations of datasets. In this phase, we observe the dataset and record the timestamp value at which the drones collide. Following that, we use Algorithm 1 to convert the video dataset into frames. By inserting the timestamp value into Algorithm 1, the system automatically converts the video into labeled frames. There are two labels, one for "normal" and one for "collision". Following that, all labeled frames are divided into three folders: training, validation and testing. Moreover, we train the whole dataset in the training folder with InceptionV3. We used the transfer-learning technique to speed up the training process. After we have trained the model using pre-trained data, we obtain initial results. In the subsequent step, the accuracy value is improved by fine-tuning the model.

3.1 Data Pre-processing

In this step, we convert 100 videos of drone collisions into labeled frames. It starts by inputting a timestamp value that we have gathered from drone-collision datasets. Algorithm 1 labels a frame as "normal" or "collision" after we enter a timestamp. In this step, we gathered 18,872 frames. Specifically, the collision class contain 1,380 frames, while the normal class contains 17,492 frames.

Labeled frames are placed into three folders; namely, training, validation and testing. Figure 2 shows sample frames that are labeled with normal and collision. The composition of folders is as follows. The training folder contains 11,380 frames, where 9,712 frames are labeled normal and 1,668 are labeled as collision. The validation folder contains 3,292 frames, where 2,856 frames are labeled as normal and 436 are labeled as collision. The test folder contains 4,200 frames, where 3,012 frames are labeled as normal and 1,188 frames are labeled as collision. Overall, the training data represents 60.3% of the dataset, the validation data 17.44% and the testing data 22.26%. There are two classes of normal and collision in each dataset. Due to the unbalanced dataset, we divided the dataset using these percentages so that the model can run well.

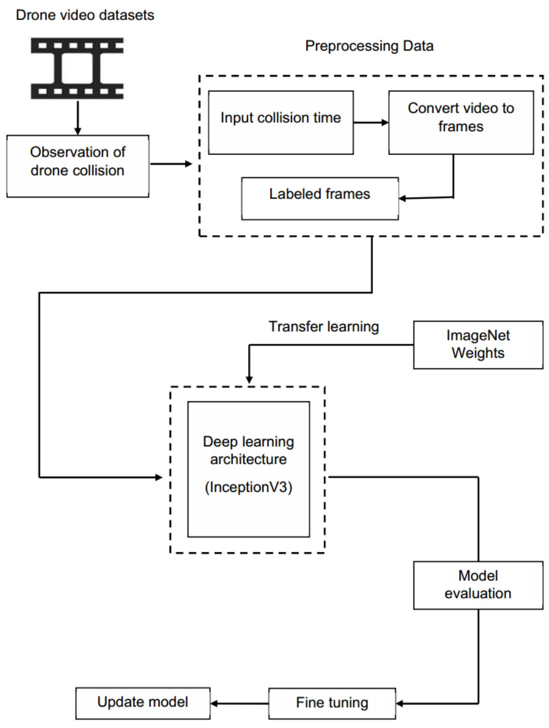

Figure 1. Research flow diagram.

Algorithm 1. Algorithm to convert video into labeled frames

```
1: for $iteration$ =1, 2, ... do
2:     if $time>time\_collision$ then
3:         $frame = collision$
4:     else
5:         $frame = normal$
6:     end if
7: end for
```

3.2 Transfer Learning

Transfer learning is a technique to speed up the training process in deep learning. The idea behind transfer learning is to apply an existing model that has trained on massive datasets to a new task in order to classify another case. Figure 3 shows the diagram of transfer learning, where it uses the domain, task and marginal probabilities framework to explain transfer learning [19]. The domain of transfer learning can be formalized as in Equation (1) below:

$$D=\{x, P(X)\} \tag{1}$$

where D is defined as a domain built by two elements of a tuple containing a feature space that is represented by x, while $P(X)$ is a marginal probability of sample data points. In each specific domain, there is a task to determine the probabilistic view point that is shown in Equation (2) below:

$$T=\{\gamma, \eta\} \tag{2}$$

where T is a task running on each transfer learning domain, γ is defined as a label of a frame and η is a predictive function obtained from the learning process. It can also be demonstrated that using $P(Y|X)$ represents the prediction of the corresponding label.

In this paper, we employ the models from lmageNet and apply them to the drone-collision case. Transfer learning involves several components, specifically a deep-learning architecture and pre- trained model templates from ImageNet. We use InceptionV3 for deep-learning architecture in this work.

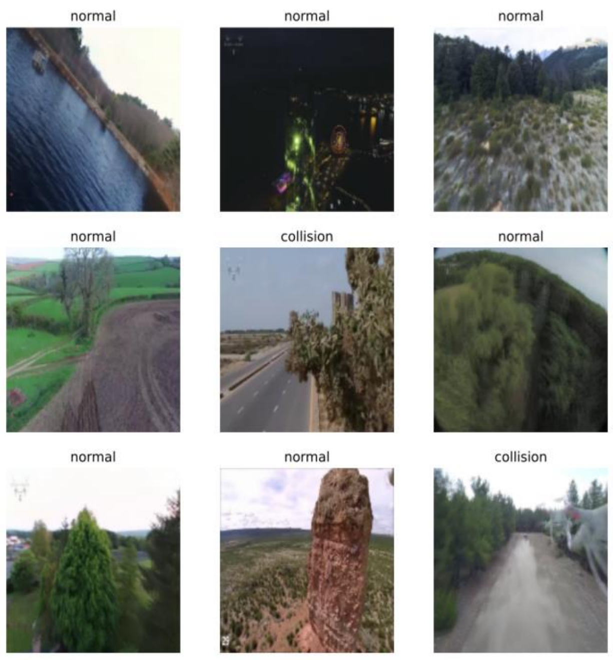

Figure 2. Frame label from the ColaNet dataset.

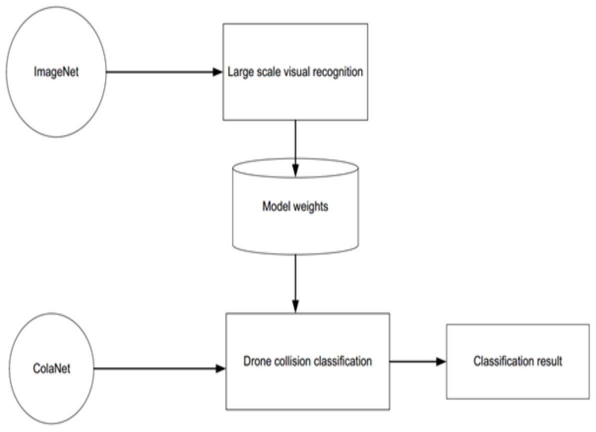
ImageNet
Large scale visual recognition
Model weights
ColaNet
Drone collision classification
Classification result

Figure 3. Block diagram of transfer learning.

3.3 Transfer Learning with InceptionV3

InceptionV3 is a deep-learning method that uses the CNN architecture. It combines several convolutions and its results are combined into one to improve the performance of the model. InceptionV3 is commonly used in a classifying task [7]. It performs the procedure using convolution on the input with different filter sizes, such as 1×1, 3×3 or 5×5. In the last part of the architecture, there is a max pooling layer and each result is sent to the subsequent inception module [7].

The previous inception methods, InceptionV1 and InceptionV2, use a lot of convolutional processes, which makes the CPU work harder. To tackle this issue, InceptionV3 limits the total number of input channels by adding an extra 1×1 convolution before 3×3 and 5×5 convolutions. By adding 1×1 convolution, the CPU does not have a heavy computation, because 1×1 convolution is lighter than other ones. This process can reduce the number of input channels. However, adding 1×1 convolution after the maximal pooling layer has been introduced in another work [20]. In addition, lnceptionV3 has the label smoothing regularization (LSR) which makes InceptionV3 more adaptable and performs well when handling low-resolution images. LSR can be described using the mathematical formula shown in Equation (3) below:

$$H(q',p) = -\sum_{k=1}^{K} \log p(k) q'(k) = (1-\varepsilon)H(q,p) + \epsilon H(u,p) \tag{3}$$

LSR is equivalent to replacing single cross-entropy loss defined by $H(q,p)$. q is the ground truth distribution, u represents the uniform distribution and p is the predicted distribution. In the equation, epsilon is the value for label smoothing. Label smoothing can help improve the generalizability of a model by reducing the confidence of the model in its predictions. This can be especially useful when training on a. dataset with a large number of classes or when the model is overfitting. It can provide regularization and a more adaptable model. The main procedure of this operation is depicted in Figure 4.

3.4 Model Fine-tuning

In deep learning, fine-tuning is the process of updating the weights to increase accuracy. In this study, fine-tuning is applied in transfer learning. This is because we apply a pre-trained model to learn novel cases in transfer learning [21]-[22]. As a result of using the accuracy value in a new case, we may get a bad result. To be applied to new cases, the pre-trained model needs to be fine-tuned.

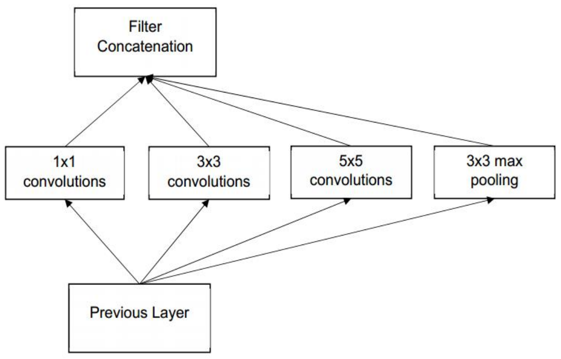

Figure 4. The workflow of InceptionV3.

Table 1. Layers of deep-learning architectures used for experiments.

No.	Deep learning architecture	Number of layers
1	InceptionV3	311
2	MobileNetV2	154
3	ResNetV2	190
4	VGG16	19
5	VGG19	22

A fine-tuning process takes less time than a training process, since we unfreeze a few nodes and update their weights based on the training process. The benefit of this method is that we do not have to spend a lot of time creating a customized model [23]. Figure 5 shows how the fine-tuning progress begins to convert data from datasets into convolutional layers. The process continues until all of the data enters the fully connected (FC) layer. The FC layer is used to detect specific global configurations of the features detected by the lower layers of the network. They usually sit at the top of the network hierarchy, at a point when the input has been reduced (by the previous, usually convolutional layers) to a compact representation of features. Each node in the FC layer learns its own set of weights from all of the nodes in the convolution layer.

The FC layer may have the same level as the softmax layer. The softmax is a function that turns a vector of K real values into a vector of K real values that sum to 1. The input values can be positive, negative, zero or greater than one, but the softmax transforms them into values between 0 and 1, so that they can be interpreted as probabilities. The fine-tuned process concentrates on the weight in the FC layer, which can make the output from the softmax layer appear high. It increases accuracy and minimizes loss.

4. Experimental Results and Analysis

4.1 Environment Setup

In this sub-section, we describe the environment used in this experiment. First, we implement the InceptionV3 model using TensorFlow version 2.10.0. In this experiment, we also compared InceptionV3 [7] with other deep-learning architectures, such as MobileNetV2 [24], ResNetV2 [25], VGG16 [26] and VGG19 [26]. The specifications of ea.ch deep learning architecture are shown in Table 1.

Based on Table 1, we can see that InceptionV3 has the most number of layers. In this experiment, we used the transfer-learning technique to make the training process faster. The weights were obtained from lmageNet [4], [27][28][29] and then transferred to the discussed deep-learning architecture.

4.2 Training Results with Pre-trained Model

We run a pre-trained model based on the weights from the ImageNet dataset and evaluate our model based on the accuracy value. The experiment uses 32 batches and 10 epochs for each model. The results are presented in Figure 6.

Five deep-learning architectures have been tested, as shown in Figure 6.Each deep-learning architecture produces different reeults.lnoeptionV3 has a high accuracy value, with values ranging from 87% to 98%. This is because lnceptionV3 architecture has several layers that execute in parallel. In addition, there is a pool that accommodates each layer process that works in parallel, which increases the accuracy value. This is shown in Figure 6, where it can be seen that the results of the first test are similar to the training accuracy at the tenth epoch. The most significant advantage of Inception V3 is that this method can be applied to complex images. This is because the architecture allows several blocks to run in parallel, making it possible to obtain many features. The main weakness of this method is that there are many blocks that run in parallel. The computational needs are also more intense, meaning that this approach requires a very large number of resources [30].

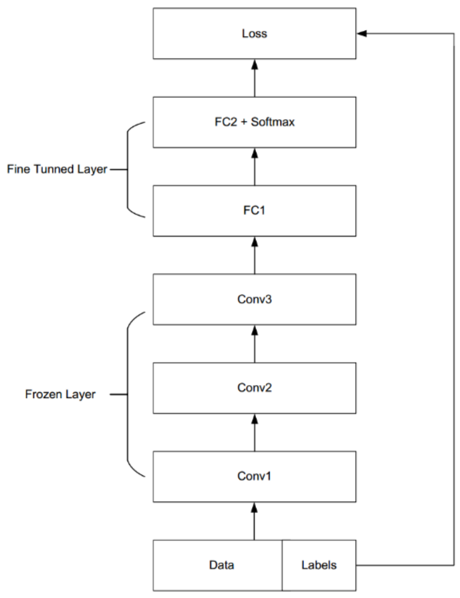

Figure 5. Block diagram of fine-tuning.

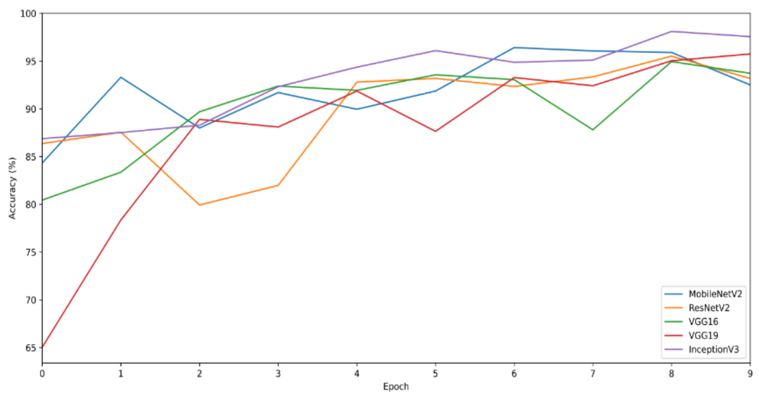

Figure 6. Accuracy of deep-learning architectures using pre-trained model.

For MobileNetV2, the resulting graph shows an unstable behavior, as depicted in Figure 6. The accuracy is between 85% and 95%. This is because the MobileNet V2 architecture uses a residual connection, which aims to keep the image constantly having the same color channel [11], [31]. This approach has certain advantages and disadvantages. The main advantage is that the frames have lower information loss, since MobileNetV2 stores the information contained in each frame using the residual connection. If we have two frames with a high degree of similarity, this can increase accuracy [4]. The primary disadvantage is that the classification results will be inaccurate if the original data contains significant noise or inappropriate features [5].

ResNetV2 provides lower accuracy values. Its results are between 81% and 92%. This is because ResNetV2 uses the value of each node obtained from the results of calculations based on the previous value. This architecture also uses a skip connection to skip nodes with the poorest accuracy value [6], [32].

For VGG16, the resulting accuracy is unstable, with values ranging between 81% and 91%, as shown in Figure 6. This is because in the VGG16 architecture, the frame is convoluted. Each process block in the architecture has a filter, which is classified at the end of the process using a fully connected neural network or a softmax layer [32]. The accuracy value obtained in the first test is unstable due to the loss of many features during the convolution process. In detecting a drone collision from image frames, it will be difficult to find features [33]. The VGG16 method is very effective in terms of recognizing objects. This is because the architecture finds the key features of the image data and when these are found, the classification accuracy is very high. The most significant drawback of this method is that if the image data does not have many features, the result will be out of focus or blurry [34]. The image frames of a drone in a collision scene have a lot of noise, for instance.

For the VGG19 method, the accuracy values range from 65% to 94%. This is because, according to the VGG19 architecture, each process block has a convolution process. Subsequently, it is classified at the end of the process using a fully connected neural network [27]. The first layer of VGG 19 applies a convolution process that is not too focused so that it can find numerous features in the image data. The data is then forwarded to the classification process in the fully connected block. VGG19 performs well if the image has many features or if the intended feature has a somewhat vague value. This method can find those features that will later be used in the classification process, meaning that accuracy is much more stable and relatively high. This method has a primary weakness in that if the processed image data is similar, the accuracy value will stack at a high level of accuracy.

Based on the graphs in Figure 6, it can be concluded that InceptionV3 has the highest accuracy. This is because the weight values in each node are drawn from ImageNet, a model that was trained on large amounts of image data and this allows the architecture to reach an accuracy value of above 60% on the training process [35].

4.3 Fine-tuning Results

After testing the pre-trained model using weights from ImageNet, we modified and updated some node weights using the process described as follows. In this phase, we unfroze some nodes to update their weights. The updated numbers of nodes are shown in Table 2. After updating the node weights, we re-tested the model on the dataset and the results are shown as a graph of the accuracy values in Figure 7.

The accuracy of InceptionV3 improves significantly after model fine-tuning, as shown in Figure 7. This is because InceptionV3 uses a parallel process in which each process calculates a node weight value. In the final step, it combines all the node weight values, which has an impact on the current accuracy value, as shown in Figure 7. The accuracy reaches a peak value of 98%, because in the fine-tuning process, we modified all of the nodes in the softmax layer and InceptionV3 has more convolutional blocks. This is the reason why InceptionV3 achieves such high accuracy values. In addition, InceptionV3 combines residual blocks with numerous convolutional blocks and this procedure can increase the accuracy.

Table 2. Table of updated weight nodes.

No.	Deep-learning architecture	Number of updated weight nodes
1	MobileNetV2	56
2	ResNetV2	84
3	VGG16	2
4	VGG19	2
5	InceptionV3	130

For MobileNetV2, we can see that the accuracy value is more stable than before. This is because the value of each node in the fully connected layer or softmax layer has been modified. This makes it more accurate in terms of detecting whether the frame contains a collision or not. The weakness of this approach is that the accuracy plateaus at 97%, since in MobileNetV2, less computer memory is used to process image frames [36].

For ResNetV2, we can see that the accuracy value reaches a steady value after fine-tuning. This is because ResNetV2 uses a skip connection in each epoch to skip blocks that give poor performance, which affects accuracy. The accuracy reaches 96%, since in the convolutional block process, the frame may have lost much information, since ResNetv2 skips blocks with poor performance and frames containing collision may have even greater noise or blurriness [37].

For VGG16, we can see the accuracy value becoming more stable than before. This is because VGG16 uses 16 block convolution layers to extract features from the image frame. In addition, it uses one fully connected neural network or a softmax layer to classify image frames. Figure 7 shows that the maximum accuracy value is around 94%. This is because in VGG16, the image quality is affected by the collision frame, which has more noise and is blurry. If noise and blurry frame images are processed by a convolutional block, many of the information in those frames will be lost. In addition, VGG16 does not use a residual block that causes a vanishing gradient. This means that this method failed to find a specific feature m several frames [38].

For VGG19, the accuracy value is lees accurate when we fine-tune the VGG19 model. The reason is that VGG19 uses more filters before it processes a fully connected CNN. The accuracy value is around 91%. VGG19 has more filters that give the frame a feature. This causes an explosive gradient in the output. This means that in several frames, VGG19 fails to find features, especially in collision frames, since these contain a great deal of noise and ate blurred.

4.4 Discussion

We compare the performance of InceptionV3 with other deep-learning architectures, specifically MobileNetV2, ResNet, VGG16 and VGG19. To begin with, we use the same configuration, including batch, epoch, optimizer and hardware. To compare deep-learning architecture performance, we calculate recall, precision and F1 score values. These three parameters are used to assess the performance of the architecture in the image data-classification process [39]. Precision is obtained using Equation (4), recall is calculated with Equation (5) and F1 score is obtained by Equation (6).

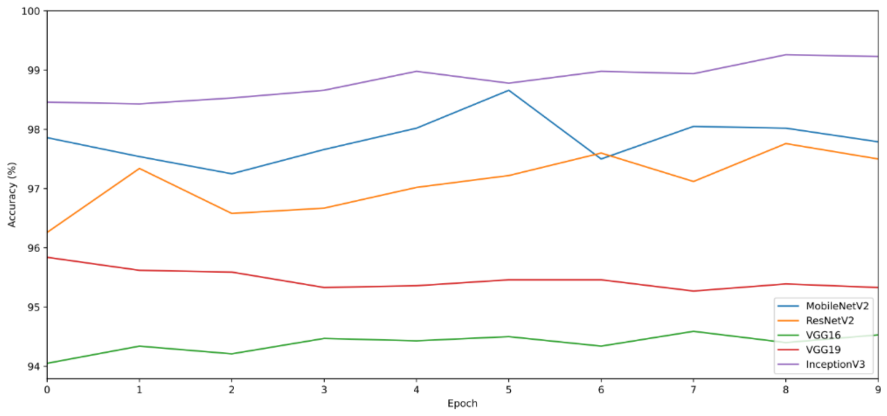

Figure 7. Accuracy values of deep-learning architectures after fine-tuning.

Table 3. Summary of model evaluation.

Deep-learning architecture	Accuracy (%)	Loss (%)	Recall (%)	Precision (%)	F1 score (%)
InceptionV3	99.35	3.95	98.23	99.54	99.56
MobileNetV2	98.18	7.48	68.56	64.56	66.56
ResNetV2	97.79	9.74	99.53	97.21	98.35
VGG16	94.27	14.09	99.48	87.34	93.45
VGG19	93.49	19.41	99.65	93.32	96.45

$$Precision = \frac{TP}{TP+FP} \tag{4}$$

$$Recall = \frac{TP}{TP+FN} \tag{5}$$

$$F1\ score = \frac{2\times precision\times recall}{precision+recall} \tag{6}$$

A summary of the results obtained using Equation (4) to Equation (6) is shown in Table 3. From Table 3, we can see that InceptionV3 has the best evaluation values with an accuracy of 99.35%, a precision of 99.54% and F1 score of 99.56%. This means that InceptionV3 has a stable model to classify drone collisions. When we use MobileNetV2, the opposite occurs. The method has a high accuracy of 98.18%, but a low precision of 66.56%. MobileNetV2 employs depth-wise block convolution, with each block consisting of a 3×3 convolution layer filtering the input, followed by 1×1 pointwise convolution combining the filtered values to create a new feature.

MobileNetV2 uses the depth-wise block convolution, where each block consists of a 3×3 convolution layer to filter the input, followed by a 1×1 pointwise convolution which combines the filtered values to create a new feature. In MobileNetV2, there are inputs and outputs between the models, while the inner layer encapsulates the model's ability to change the input from the pixel level concept to the image classification [40]. The total memory is constructed from all inputs and outputs in whole operations. Meanwhile, when we use the bottleneck residual block and make it a single operation, the memory will be filled by the bottleneck tensor instead of the internal tensors that can be larger. This means that this method cannot be used to classify drone collision although it has a good accuracy value. Also, MobileNetV2 has a lack of value in recall, precision and F1 score parameters. The lack of values of MobileNetV2 in those parameters is due to unbalanced datasets. We have a high recall value of 99.65% in VGG16, which could be because we rely on image quality in VGG16 and the collision frame has more noise and blur. If noisy and blurry frame images are processed by a convolutional block, they will lose much of the information that is contained in a frame. In addition, VGG16 does not use a residual block that causes a vanishing gradient. This means that this method failed to find a specific feature in several frames [38].

The VGG 19 result has the highest value in recall, because in VGG 19, there are more filters to reduce noise in collision frames. In a collision frame, a blurred frame can affect the accuracy value. According to Equation (5), recall is proportional to the number of true positives (TP) and false negatives (FN), implying that VGG19 can accurately detect whether the result is positive or negative. Furthermore, ResNetV2 has the second highest recall after VGG19, because ResNetV2 used the residual weight to improve the ability of the model to classify the drone-collision frame. Since the collision frame is the blurred frame, it has an advantage for ResNetV2. Also, ResNetV2 uses a skip connection in each epoch to skip blocks that give poor performance, which affects accuracy.

5. Conclusion and Future Work

In this research, we present a forensic analysis of drone collisions using a transfer-learning technique; namely, InceptionV3. After model fine-tuning, it was discovered that InceptionV3 was the best model, with a peak accuracy value of 99.35%. We also compared InceptionV3 with other architectures, such as MobileNetV2, ResNetV2, VGG16 and VGG19.

The model generated from experiments can be applied to drone forensic investigation based on videos found in the drone's storage. Although only collision videos from the ColaNet dataset were used in this paper, we believe that the results of this experiment can be applied to achieve faster classification of drone collisions for drone forensics.

The contribution of this paper is a methodology to classify drone-collision events based on frames extracted from drone videos using InceptionV3. By implementing a transfer-learning technique, we reduce the computational load and speed up the training process. Weights for transfer learning are taken from lmageNet. By fine-tuning the weights used in lnceptionV3, we are able to determine the most appropriate weights for drone collisions.

Future work will involve further updating the modules presented in this article. Data pre-processing will be improved through the use of neural network pipelines to classify frames containing collisions. We can improve the proposed method by minimizing the amount of human labor needed to classify the frames from the video datasets. In addition, this work can be extended by modifying our deep-learning framework to maximize accuracy and minimize loss. Finally, we intend to develop a detection system for attackers and collisions with objects, which may be useful in terms of gaining information to assist in forensic investigations.

References

[1] L. P. Osco et al., "A Review on Deep Learning in UAV Remote Sensing," Int. J. of Applied Earth Observation and Geoinformation, vol. 102, p. 102456, 2021.

[2] A. L. Renduchintala, A. Albehadili and A. Y. Javaid, "Drone Forensics: Digital Flight Log Examination Framework for Micro Drones," Proc. of the 2017 Int. Conf. on Computational Science and Computational Intelligence (CSCI 2017), pp. 91-96, DOI: 10. 1109/CSCI. 2017. 15, 2018.

[3] G. Horsman, "Unmanned Aerial Vehicles: A Preliminary Analysis of Forensic Challenges," Digital Investigation, vol. 16, pp. 1-11, DOI: 10.1016/j .diin.2015.11.002, 2016.

[4] D. Pedro et al., "Collision Avoidance on Unmanned Aerial Vehicles Using Neural Network Pipelines and Flow Clustering Techniques," Remote Sensing, vol. 13, DOI: 10.3390/rs13132643, 2021.

[5] U. Jain, M. Rogers and E. T. Matson, "Drone Forensic Framework: Sensor and Data Identification and Verification," Proc. of the 2017 IEEE Sensors Applications Symposium (SAS), pp. l-6, DOI: 10. 1109/SAS. 2017.7894059, 2017.

[6] M. W. Ashraf, W. Sultani and M. Shah, "Dogfight: Detecting Drones from Drones Videos," CVPR, The Computer Vision Foundation, pp. 7063- 7072, DOI: 10.1109/CVPR46437.2021.00699, 2021.

[7] C. Szegedy, S. Ioffe, V. Vanhoucke and A. Alemi, "Inception-v4, Inception-resnet and the Impact of Residual Connections on Learning," Proc. of the 31st AAAI Conf. on Artificial Intelligence (AAAI'17), pp. 4278-4284, DOI: 10.48550.arXiv.1602.07261, 2016.

[8] A. Anwar and A. Raychowdhury, "Autonomous Navigation *via* Deep Reinforcement Learning for Resource Constraint Edge Nodes Using Transfer Learning," IEEE Access, vol. 8, pp. 26549-26560, DOI: 10 .1109/ACCESS.2020.2971 172, 2020.

[9] S. Ouahouah et al., "Deep-reinforcement-learning-based Collision Avoidance in UAV Environment," IEEE Internet of Things Journal, vol. 9, no. 6, pp. 4015-4030, DOI: 10.1109/JIOT.2021.3118949, 2022.

[10] A. A. Obaid and H. Koyuncu, "Obstacle Avoidance in Unmanned Aerial Vehicles Using Image Segmentation and Deep Learning," Proc. of the 2022 Int. Symposium on Multi-disciplinary Studies and Innovative Technologies (ISMSIT), pp. 912-915, DOI: 10.1109/ISMSIT56059. 2022.9932865, 2022.

[11] F. Rehmatullah and J. Kelly, "Vision-based Collision Avoidance for Personal Aerial Vehicles Using Dynamic Potential Fields," Proc. of the 12th Conf. on Computer and Robot Vision (CRY 2015), pp. 297-304, DOI: 10.1109/CRV.2015.46, 2015.

[12] R. R. Lima and G. A. S. Pereira, "Drone Collision Detection and Classification Using Proprioceptive Data," Proc. of the Int. Conf. on Unmanned Aircraft Systems (ICUAS), pp. 562-569, DOI: 10.1109/ICUAS54217.2022.9836207, 2022.

[13] J. K. Park et al., "UAV Collision Detection Algorithm Based on Formula for Fast Calculation," Turkish J. of Computer and Mathematics Education (TURCOMAT), vol. 12, no. 6, pp. 452-456, 2021.

[14] A. Weinert, L. Alvarez, M. Owen and B. Zintak, "Near Midair Collision Analog for Drones Based on Unmitigated Collision Risk," J. of Air Transportation, vol. 30, no. 2, pp. 37-48, 2022.

[15] X. Lu, X. Liu, Y. Li, Y. Zhang and H. Zuo, "Simulations of Airborne Collisions between Drones and an Aircraft Windshield," Aerospace Science and Technology, vol. 98, p. 105713, DOI: 10.1016/j.ast.2020 .105713, 2020.

[16] N. Zhang, H. Liu, B. F. Ng and K. H. Low, "Collision Probability between Intruding Drone and Commercial Aircraft in Airport Restricted Area Based on Collision-course Trajectory Planning," Transportation Research Part C: Emerging Technologies, vol. 120, p. 102736, DOI: 10.1016/j.trc.2020.102736, 2020.

[17] J. N. Yasin et al., "Formation Maintenance and Collision Avoidance in a Swarm of Drones," Proc. of the 3rd Int. Symp. on Computer Science and Intelligent Control, ACM, DOI: 10.1145/3386164. 3386176, 2019.

[18] B. Sabetghadam, R. Cunha and A. Pascoal, "A Distributed Algorithm for Real-time Multi-drone Collision-free Trajectory Replanning," Sensors, vol. 22, no. 5, DOI: 10.3390/s22051855, 2022.

[19] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 1345—1359, 2010.

[20] S. Arakelyan, S. Arasteh, C. Hauser, E. Kline and A. Galstyan, "Bin2vec: Learning Representations of Binary Executable Programs for Security Tasks," Cybersecurity, vol. 4, no. 1, p. 26, DOI: 10.1186/s42400-021-00088-4, 2021.

[21] A. Ramdan et al., "Transfer Learning and Fine-tuning for Deep Learning-based Tea Diseases Detection on Small Datasets," Proc. of the 2020 Int. Conf. on Radar, Antenna, Microwave, Electronics and Telecommunications (ICRAMET), pp. 206-211, DOI:10.1109/ICRAMET51080.2020 .9298575, 2020.

[22] J. Luttrell et al., "A deep Transfer Learning Approach to Fine-tuning Facial Recognition Models," Proc. of the 13th IEEE Conf. on Industrial Electronics and Applications (ICIEA), pp. 2671-2676, DOI: 10.1109/ICIEA. 2018.8398162, 2018.

[23] G. Vrbancic and V. Podgorelec, "Transfer Learning with Adaptive Fine-tuning," IEEE Access, vol. 8, pp. 196197-196211, DOI:10.1109/ACCESS.2020.3034343, 2020.

[24] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L.-C. Chen, "Mobilenetv2: Inverted Residnals and Linear Bottlenecks, arXiv: 1801.04381, pp. 4510-4520, DOI: 10.1109/CVPR.2018.00474, 2018.

[25] C. Szegedy et al., "Rethinking the Inception Architecture for Computer Vision," Proc. of the 2016 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 2818-2826, DOI: 10.1109/CVPR.2016.308, 2016.

[26] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778, DOI: 10.1109/CVPR.2016.90, 2016.

[27] J. Jaafari, S. Douzi, K. Douzi and B. Hssina, "Towards More Efficient CNN-based Surgical Tools Classification Using Transfer Learning," Journal of Big Data, vol. 8, DOI:10.1186/s40537-021-00509-8, 2021.

[28] G. Li, Z. Ji, Y. Chang, S. Li, X. Qu and D. Cao, "ML-ANet: A Transfer Learning Approach Using Adaptation Network for Multi-label Image Classification in Autonomous Driving," Chinese J. of Mechanical Engineering (English Edition), vol. 34, DOI: 10.1186/s10033-021-00598-9, 2021.

[29] H.-T. Theng, C.-C. Hsieh, W.-T. Lin and J.-T. Lin, "Deep Reinforcement Learning for Collision Avoidance of Autonomous Vehicle," Proc. of the 2020 IEEE Int. Conf. on Consumer Electronics-Taiwan (ICCE-Taiwan), DOI: 10. 1109/ICCE-Taiwan49838.2020.9258199, 2020.

[30] A. Sarkar and Kalyani, "Survey of Smart Grid Network Using Drone and PTZ Camera," Proc. of the 3rd Int. Conf. on 2019 Devices for Integrated Circuit (DevIC), Kalyani, India, 2019.

[31] C. Shorten and T. M. Khoshgoftaar, "A Survey on Image Data Augmentation for Deep Learning," Journal of Big Data, vol. 6, no. 1, p. 60, DOI: 10.1186/s40537-019-0197-0, 2019.

[32] N. Best, J. Ott and E. J. Linstead, "Exploring the Efficacy of Transfer Learning in Mining Image-based Software Artifacts," Journal of Big Data, vol. 7, DOI: 10.1186/s40537-020-00335-4, 2020.

[33] N. Attari, F. Ofli, M. Awad, J. Lucas and S. Chawla, "Nazr-CNN: Fine-grained Classification of UAV Imagery for Damage Assessment," Proc. of the 2017 Int. Conf. on Data Science and Advanced Analytics (DSAA 2017), pp. 50-59, DOI: 10.1109/DSAA.2017.72, 2017.

[34] W. Qiu, S. Bi, C. Zhong, Y. Luo, J. Li and B. Sun, "Obstacle Avoidance of Aerial Vehicle Based on Monocular Vision," Proc. of the 2017 IEEE 7th Annual Int. Conf. on CYBER Technology in Automation, Control and Intelligent Systems (CYBER), (2017) 657-662, DOI: 10.1109/CYBER.2017.8446065, 2017.

[35] L. Huang, X. Feng, C. Zhang, L. Qian and Y. Wu, "Deep Reinforcement Learning-based Joint Task Offloading and Bandwidth Allocation for Multi-user Mobile Edge Computing," Digital Communications and Networks, vol. 5, pp. 10-17, DOI:10.1016/j.dcan.2018.10.003, 2019.

[36] Y. Chen et al., "Classification Methods of a Small Sample Target Object in the Sky Based on the Higher Layer Visualizing Feature and Transfer Learning Deep Networks," EURASIP Journal on Wireless Communications and Networking, vol. 2018, no. 1, p. 127, DOI:10.1186/s13638-018-1133-2, 2018.

[37] B. Wei, L. Zhang, K. Wang, Q. Kong and Z. Wang, "Dynamic Scene Deblurring and Image De-raining Based on Generative Adversarial Networks and Transfer Learning for Internet of Vehicle," EURASIP J. on Advances in Signal Processing, vol. 2021, no. 1, p. 121, DOI: 10.1186/s13634-021-00829-0, 2021.

[38] Y.-H. Choi et al., "Using Deep Learning to Solve Computer Security Challenges: A Survey," Cybersecurity, vol. 3, no. 1, p. 15, DOI: 10.1186/s42400-020-00055-5, 2020.

[39] X. Chen, W. Qi and L. Xi, "Deep-learning-based Motion-correction Algorithm in Optical Resolution Photoacoustic Microscopy," Visual Computing for Industry, Biomedicine and Art, vol. 2, no. 1, p. 12, DOI:10.1186/s42492-019-0022-9, 2019.

[40] P. Drews-Jr, I. d. Souza, I. P. Maurell, E. V. Protas and S. S. C. Botelho, "Underwater Image Segmentation in the Wild Using Deep Learning," Journal of the Brazilian Computer Society, vol. 27, no. 1, p. 12, DOI: 10.1186/s13173-021-00117-7, 2021.

ملخص البحث:

تُعدّ الطّائرات المسيّرة من بين الأجهزة التي تُستخدم في أنشطة منوّعة. وأحياناً تتعرّض تلك الطّائرات لحوادث، ممّا يجعل السّلطات في حاجةٍ الى الوقوف على أسبابها. وقد أصبح تحليل حوادث الطّائرات المسيّرة ومعرفة أسبابها مجالاً خصْباً للبحث.

في هذه الورقة، تمّ توظيف إحدى تقنيات التّعلُّم العميق من أجل تصنيف الاصطدامات التي تتعرض لها الطّائرات المسيّرة، وقد تمّ استخدام (Inception V9) كإطار عملٍ للبحث. من جانب آخر، تقارن هذه الورقة بين الطّريقة المقترحة وتقنياتٍ أخرى جرى استخدامها في أدبيات الموضوع من أجل تصنيف اصطدامات الطّائرات المسيّرة. وفي التّجارب المتعلّقة بهذا البحث، تم استخدام تعلُّم النّقْل والضّبْط الدّقيق لتسريع عملية تدريب النّظام وتحسين قيمة الدّقّة التي يتمّ الحصول عليها من الطّريقة المقترحة. يُضاف الى ذلك أنّ التّجارب أثبتت أنّ الطّريقة المقترحة تفوّقت على الطّرق الاخرى التي تمّت مقارنتها بها من حيث مؤشّرات الأداء.