A NOVEL APPROACH FOR DETECTION FAULT IN THE AIRCRAFT EXTERIOR BODY USING IMAGE PROCESSING

Noura Nayef Almansoori

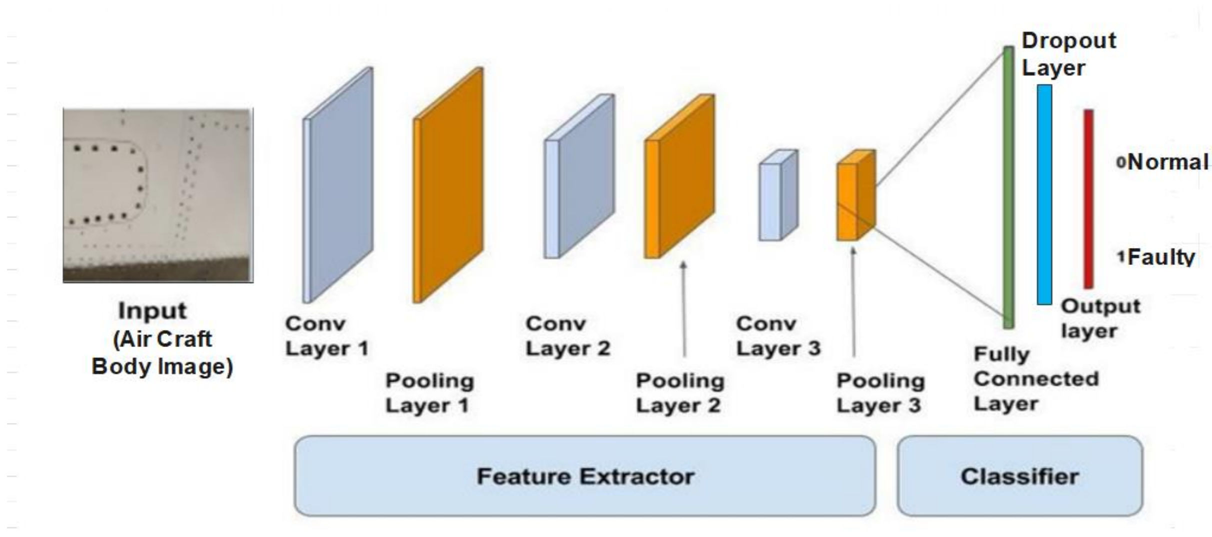
MASTER THESIS NO. 2023: 30

College of Engineering

Department of Electrical and Communication Engineering

A NOVEL APPROACH FOR DETECTION FAULT IN THE AIRCRAFT EXTERIOR BODY USING IMAGE PROCESSING

Noura Nayef Saeed Saleh Almansoori



April 2023

United Arab Emirates University

College of Engineering

Department of Electrical and Communication Engineering

A NOVEL APPROACH FOR DETECTION FAULT IN THE AIRCRAFT EXTERIOR BODY USING IMAGE PROCESSING

Noura Nayef Saeed Saleh Almansoori

This thesis is submitted in partial fulfilment of the requirements for the degree of Master of Science in Electrical Engineering

May 2023

United Arab Emirates University Master Thesis
2023: 30

Cover: The Architecture of Proposed CNN Model

(Photo: By Noura Nayef Saeed Saleh Almansoori)

Declaration of Original Work

I, Noura Nayef Saeed Saleh Almansoori, the undersigned, a graduate student at the United Arab Emirates University (UAEU), and the author of this thesis entitled "*A Novel Approach for Detection Fault in the Aircraft Exterior Body Using Image Processing*", hereby, solemnly declare that this is the original research work done by me under the supervision of Prof. Falah Awwad, in the College of Engineering at UAEU. This work has not previously formed the basis for the award of any academic degree, diploma or a similar title at this or any other university. Any materials borrowed from other sources (whether published or unpublished) and relied upon or included in my thesis have been properly cited and acknowledged in accordance with appropriate academic conventions. I further declare that there is no potential conflict of interest with respect to the research, data collection, authorship, presentation and/or publication of this thesis.

Student's Signature: [illegible]

Date: 9/1/2023

Approval of the Master Thesis

This Master Thesis is approved by the following Examining Committee Members:

1) Advisor (Committee Chair): Falah Awwad
   Title: Professor
   Department of Electrical and Communication Engineering
   College of Engineering

   Signature [illegible] Date 15-5-2023

2) Member (Co-Advisor): Sheharyar Malik
   Title: Assistant Professor
   Department: Aircraft Engineering Technology
   Abu Dhabi Polytechnic

   Signature [illegible] Date 12-5-2023

3) Member: Mamoun Awad
   Title: Associate Professor
   Department of Computer Science and Software Engineering
   College of Information Technology

   Signature [illegible] Mamoun A. Awad Date 30-5-2023

4) Member (External Examiner): Hani Saleh
   Title: Associate Professor
   Department of Electrical Engineering and Computer Science
   Institution: Khalifa University, Abu Dhabi, UAE

   Signature [illegible] Date 31-5-2023

This Master Thesis is accepted by:

Dean of the College of Engineering: Professor Mohamed H. Al-Marzouqi

Signature Mohamed AlMarzouqi

Date June 15, 2023

Dean of the College of Graduate Studies: Professor Ali Al-Marzouqi

Signature Ali Hassan [illegible]

Date June 16, 2023

Abstract

The primary objective of this thesis is to develop innovative techniques for the inspection and maintenance of aircraft structures. We aim to streamline the entire process by utilizing images to detect potential defects in the aircraft body and comparing them to properly functioning images of the aircraft. This enables us to determine whether a specific section of the aircraft is faulty or not. We achieve this by employing image processing to train a model capable of identifying faulty images. The image processing methodology we use involves the use of images of both defective and operational parts of the aircraft's exterior. These images undergo a preprocessing phase that preserves valuable details. During the training period, a new image of the same section of the aircraft is used to validate the model. After processing, the algorithm grades the image as faulty or normal.

To facilitate our study, we rely on the Convolutional Neural Network (CNN) approach. This technique collects distinguishing features from a single patch created by the frame segmentation of a CNN kernel. Furthermore, we use various filters to process the images using the image processing toolbox available in Python. In our initial trials, we observed that the CNN model struggled with the overfitting of the faulty class. To address this, we applied image augmentation by converting a small dataset of 87 images to an augmented dataset of 4000 images. After passing the data through multiple convolutional layers and executing multiple epochs, our proposed model achieved an impressive training accuracy of 98.28%.

In addition, we designed a GUI-based interface that allows users to input an image and view the results in terms of faulty or normal. Finally, we propose that the application of this research in the field of robotics would be an ideal area for future work.

Keywords: Aircraft Maintenance, Robotics, Image Processing, Data Augmentation.

Title and Abstract (in Arabic)

طريقة جديدة لاكتشاف الأخطاء في الهيكل الخارجي للطائرة باستخدام معالجة الصور

الملخص

الهدف الأساسي من هذه الأطروحة هو تطوير تقنيات مبتكرة لفحص وصيانة هياكل الطائرات، نهدف إلى تبسيط العملية بأكملها من خلال استخدام الصور لاكتشاف العيوب المحتملة في هيكل الطائرة ومقارنتها بصور الطائرة التي تعمل بشكل صحيح. يتيح لنا ذلك تحديد ما إذا كان جزء معين من الطائرة معيبًا أم لا، يتم ذلك من خلال استخدام معالجة الصور لتدريب نموذج قادر على تحديد الصور المعيبة. تتضمن منهجية معالجة الصور التي نستخدمها استخدام الصور للأجزاء المعيبة والتشغيلية للجزء الخارجي للطائرة، تخضع هذه الصور لمرحلة ما قبل المعالجة التي تحافظ على التفاصيل القيمة، يتم استخدام صورة جديدة لنفس الجزء من الطائرة للتحقق من صحة النموذج وذلك خلال فترة التجربة. بعد عملية المعالجة، تقوم الخوارزمية بتصنيف الصورة على أنها معيبة أو عادية.

نعتمد في دراستنا هذه على استراتيجية الشبكة العصبية الالتفافية. (CNN) تقوم هذه التقنية بجمع السمات المميزة من جزء واحد تم إنشاؤه بواسطة تجزئة إطار CNN وبناءً على ذلك، نستخدم العديد من المرشحات لمعالجة الصور باستخدام أدوات معالجة الصور المتاح في Python وذلك في تجاربنا الأولية، كما أنه لاحظنا بأن نموذج CNN كافح الإفراط الذي قد يحصل في تجهيز الطبقة المعيبة و لمعالجة هذا الأمر، قمنا بتطبيق تكبير الصورة عن طريق تحويل مجموعة صغيرة تتكون من 87 صورة إلى مجموعة معززة تتكون من 4000 صورة وذلك بعد تمرير البيانات عبر طبقات التفافية متعددة وتنفيذ عهود متعددة، حقق نموذجنا المقترح دقة تدريب فريدة بلغت 98.28%.

كما أنه، قمنا بتصميم واجهة قائمة على واجهة المستخدم الرسومية (GUI) تسمح للمستخدمين بإدخال صورة وعرض النتائج التي توضح الصورة المعيبة أو الصحيحة. وفي الختام، نقترح أن يكون تطبيق هذا البحث في مجال الروبوتات مجالًا مثاليًا للعمل في المستقبل.

مفاهيم البحث الرئيسية: صيانة الطائرات، الروبوتات، معالجة الصور، زيادة البيانات.

Acknowledgements

This work would not have been feasible without the assistance of many people. Many thanks to Prof. Falah Awwad and Dr. Sheharyar Malik for providing persistent guidance, support, and assistance, including their continuous proofreading and editing of several drafts of this thesis.

Thank you so much for your unwavering support, my parents. You've always been there for me, and this time was no different. Thank you for your thoughtfulness and for always telling me I'm wonderful even when I didn't believe it. Maha, my soulmate, thank you for answering so many phone calls, calming me down, and always listening to me. I also want to thank my husband for his continuous support throughout the years. I'm thankful to those who predicted I wouldn't be able to finish my master's degree while having five children and working.

I also want to thank myself and my children, who have been through so much, for their unwavering support throughout my journey to complete my master's degree while raising a family and working.

Dedication

First and foremost, I shall dedicate this dissertation to our almighty God, who provides me with the power and wisdom I need in everyday life.
My adoring spouse and children, as well as my lifelong thanks to my soulmate, Maha, for her unending love.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AI	Artificial Intelligence
AMT	Aircraft Maintenance Technician
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
CV	Computer Vision
GUI	Graphical User Interface
ML	Machine Learning
RF	Random Forest
SI	Swarm Intelligence
SVM	Support Vector Machine

Chapter 1: Introduction

1.1 Overview

The purpose of this work is to automate the whole process of detection of faults in aircraft bodies using images and then to compare it with functioning images of that section of the normal aircraft and conclude whether that section is damaged or normal. For image acquisition, the concept of robotics can be used to acquire different parts of aircraft. An automated robot machine is designed to move freely on the designated path around the aircraft and capture images. The captured images are then passed to the model to detect defectiveness in the part of the body.

The framework of this study includes processing images of an aircraft's exterior body for a specific model, a Machine Learning (ML) approach is used to extract features from these images, wherein a kernel is used to capture discriminating features from a single patch generated through image segmentation. The feasibility of using different filters for processing the images using the Python image processing toolbox is also explored. These extracted features, enough to describe the corresponding image, are then stored with a class label of two types, damaged and normal. The class labels are termed as output against the extracted features, and the deep neural image processing technique of Convolutional Neural Networks (CNNs) is tested for various inputs. We trained our image processing model with sets of features extracted from images from the training sets and tested with our captured images of the testing set.

1.2 Statement of the Problem

Aviation is a huge market and there is an involvement of massive financial aspects to it. In the given scenario the aircraft maintenance and test report prior to the flight is of much concern. To date, an aircraft maintenance check is widely dependent on human activities. Thus, causing a lot of time and cost expended on aircraft checks before every flight. The aviation industry needs rapid development of techniques to develop an automated system for an aircraft maintenance check. There exists a satisfactory amount of work in this domain, however, a reliable and efficient model is still missing in the practical aspect. In this work, we address the issue of aircraft maintenance checks through an automated robot using an image processing model.

1.3 Objectives/Milestones

To successfully implement the project, we have divided the study into several steps as follows:

1. Firstly, we collected a dataset of aircraft images and performed an analysis of these images.
   a. We labeled the images as *faulty* or *normal.*
   b. After formulating a dataset of sufficient images, we applied data augmentation techniques to the original set of images.
   c. The next step involved the use of analysis software such as Python for image acquisition, image rectification, preprocessing, image transformation, and identifying discriminating features that can better classify the *damaged* image from the *normal* image.

2. In the second phase, we explored Convolutional Neural Networks (CNNs) and checked their feasibility.
   a. We tested different implementations of CNN as feature selectors and classifiers.
   b. We designed an algorithm to decide whether a new aircraft image is damaged or normal.

3. Finally, we tuned the algorithm to produce efficient and more accurate results.

1.4 Potential Contributions and Limitations of the Study

The potential contribution of this project is providing a moveable pre-trained image processing model for the aircraft body scan prior to the flight for checking the aircraft body parts damages. The scope of the project is limited to the body parts scan and self-made dataset of aircraft body images.

Chapter 2: Background and Literature Study

2.1 Introduction

Aviation is a massive industry, generating total revenue of $2.7 trillion and contributing approximately 3.6% of the world's gross domestic product (GDP). However, delays in flights due to the maintenance of aircraft can have a massive impact on the overall customer experience, resulting in economic losses of approximately $26.6 billion in the United States alone (Apo, 2016). Moreover, significant accidents occurred during aircraft traveling, primarily attributed to the maintenance of airplanes (Apo, 2020). Maintenance of aircraft is an indispensable part of the overall booming aviation industry, and the whole process includes a complex organization centered around the aircraft maintenance technician (AMT).

2.1.1 The Role of Aircraft Maintenance Technicians

Maintenance is crucial for keeping airplanes functional and ensuring their safety of the aircraft. The primary role of AMTs is the detection and addressing of any potential problems. However, the dependency of the aircraft's health on the AMT's detection induces a risk factor and human error.

2.1.2 Challenges in Aircraft Maintenance

With almost 100,000 commercial flights per day and thousands of aircraft inspections per hour, the aviation industry presents a massive gap for researchers to reduce aircraft downtime and maintenance costs. Approximately 70% of these inspections are performed visually by AMTs, representing a significant challenge for the industry (Yasuda *et al.*, 2021).

2.1.3 Mobile Robots for Aircraft Maintenance

Mobile robots are mainly employed to aid operators in the maintenance of aircraft. Earlier work on aircraft maintenance was based on automating skin inspections of the exterior, but these robots only perform visual inspections, and the decision to objectify a part as faulty rests with the maintenance staff (Karaoğlu *et al.*, 2023). Recent research has focused on assembling robots for the pre-check of the aircraft before flight based on visual inspections, relying solely on processing captured images. To move around the aircraft freely, the robot uses laser and vision methods to center itself around the aircraft. Recognition and avoidance of obstacles are used in the navigation mode of the robot (Zhang *et al.*, 2021).

2.2 Background

2.2.1 Application of Artificial Intelligence in Aviation

The advent of Artificial Intelligence (AI) has revolutionized several commercial industries, optimizing costs, reducing the duration of design cycle time, prototyping, optimization, maintenance, manufacturing, and updating products. AI has found a significant audience in industries ranging from aviation to autonomous driving cars and the health industry. However, in the aviation industry, machine learning and AI techniques are still not widely applied for the detection and maintenance of aircraft due to the absence of standard and benchmark work and dependability on simple models compared to much-intrigued models (Zhao *et al.*, 2022).

2.2.2 Advancements in Computer Vision

Machine vision has progressed from an embryonic phase to a mature field, encompassing a wide range of applications such as surveillance, inspection, robot manufacturing, obstacle avoidance, vehicle guidance, directing traffic, biometric measurement, and analysis of images from space shuttles. However, the use of computer vision and image processing in the aviation industry is mainly limited to inspection and surveillance. Its application while combining machine learning and AI is yet to be explored (Jovancevic *et al.*, 2015; Miranda *et al.*, 2019; Ding *et al.*, 2022).

In summary, the identification and utilization of a machine learning model for a specific problem is an essential factor for the accurate prediction of overall results. It generally depends on the choice of algorithms and models. The aviation industry is primed for the application of AI and machine learning in maintenance, and further research in this area can help optimize detection processes and reduce aircraft downtime and maintenance costs.

2.2.3 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a type of Artificial Neural Network (ANN) that are commonly used in machine learning and computer vision applications, particularly for image processing.

AI (Artificial Intelligence) is a broad field that encompasses various techniques and technologies for creating intelligent machines capable of performing tasks that typically require human intelligence, such as natural language processing, decision-making, and problem-solving.

Image processing refers to the use of computer algorithms to manipulate digital images to improve their quality or extract useful information. Image processing techniques can be used in a wide range of applications, including medical imaging, satellite imagery analysis, and security surveillance.

CNNs are particularly effective in image classification tasks, such as object detection and recognition, due to their ability to automatically learn features directly from the raw image data. This is achieved through the use of convolutional layers, which apply a set of learned filters to the input image to extract relevant features. These features are then fed through fully connected layers to produce a classification output.

In summary, CNNs are a type of AI technology that is commonly used for image processing tasks and have demonstrated impressive results in areas such as object detection and recognition.

2.2.4 The Architecture of Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a type of deep neural network that have been designed specifically to handle image processing tasks. A CNN is made up of several layers, each with a specific role in feature extraction and classification. The layers include convolutional layers, pooling layers, activation functions, and fully connected layers. The main objective of a CNN is to learn and extract relevant features from images, which can then be used to classify objects.

The convolutional layers are responsible for extracting features from the input image. Each convolutional layer has multiple filters, which are small matrices that are convolved with the input image to produce a feature map. The filters learn to detect local patterns in the image, such as edges, corners, and textures. This allows CNN to learn more complex features by stacking multiple convolutional layers.

Pooling layers reduce the spatial resolution of the feature maps produced by the convolutional layers by taking the maximum or average value in a small neighborhood. This helps to reduce overfitting and improve computational efficiency.

The activation functions introduce non-linearity into the network, allowing CNN to model complex relationships between the input and output. The most commonly used activation function in CNNs is the Rectified Linear Unit (ReLU), which sets all negative values to zero. This helps to speed up the training process by making the optimization problem easier to solve.

Finally, the fully connected layers perform the final classification of the input image by mapping the learned features to the output classes. Each neuron in the fully connected layer is connected to every neuron in the previous layer, allowing the CNN to learn complex relationships between features.

In conclusion, the architecture of a CNN plays a crucial role in the success of image processing tasks. Each layer of the network has a specific function, from feature extraction to final classification. CNNs offer several advantages over other machine learning and deep learning models, making them a popular choice for image processing tasks.

Figure 2.1 shows a typical CNN architecture with multiple convolutional and fully connected layers (Image Classification, 2009; Gupta, 2017).

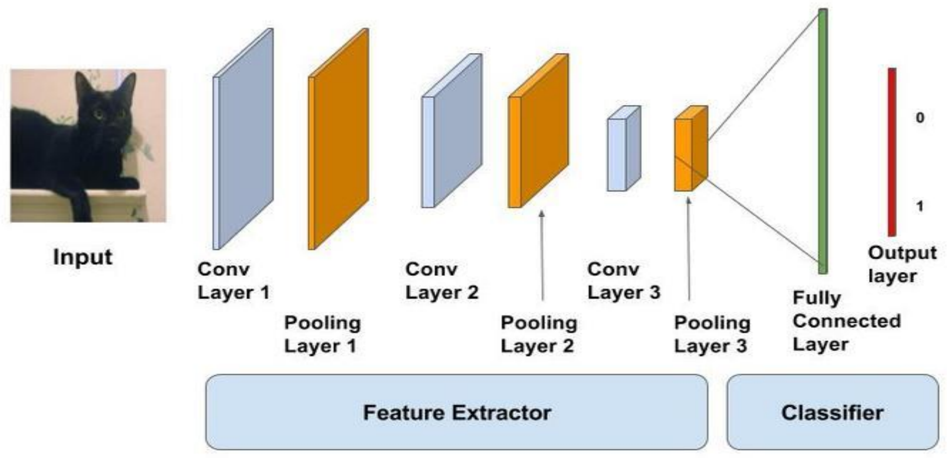


Figure 2.1: CNN Architecture.

2.2.5 Benefits of CNN over ML Algorithms for Image Processing

Convolutional Neural Networks (CNNs) have gained a lot of attention and popularity in recent years due to their exceptional performance in image-processing tasks. CNNs have numerous advantages over traditional machine learning and deep learning models. One of the key advantages of CNNs is their ability to handle images of varying sizes. Unlike traditional machine learning models, which require images to be resized to a fixed size, CNNs can handle images of varying sizes. This is due to the convolutional layers, which are designed to learn features at different spatial scales.

CNNs are also capable of learning spatial relationships between pixels. Traditional machine learning models treat each pixel in an image as an independent feature, which can lead to poor performance on image classification tasks. On the other hand, CNNs can learn spatial relationships between pixels by sharing weights across the filters. This allows CNNs to detect patterns and features in an image, which makes them suitable for image classification and recognition tasks.

Another advantage of CNNs is their ability to learn hierarchical features. CNNs can learn hierarchical features by stacking multiple convolutional layers. The lower layers learn simple features such as edges and corners, while the higher layers learn more complex features such as object parts and textures. This hierarchical learning enables CNNs to capture the essence of the images and extract the most informative features. Figure 2.2 shows the hierarchical feature learning of CNN. CNNs are also robust to translation and rotation. This means that they can recognize objects in images regardless of their position and orientation. The convolutional layers in CNNs learn to detect features in local regions of an image. This makes CNNs capable of handling images with different orientations and positions. The feature maps produced by the convolutional layers are similar for all variations of an image, indicating that the CNN is invariant to translation and rotation.

Finally, CNNs can be trained end-to-end using backpropagation. This makes CNNs more efficient in learning and optimizing their parameters. End-to-end training means that the entire network is trained as a single unit, which allows CNNs to learn complex features and patterns in an image. This also means that CNNs can be trained on large datasets, which is essential for achieving high accuracy on image classification tasks.

In conclusion, CNNs have several advantages over traditional machine learning and deep learning models. They can handle images of varying sizes, learn spatial relationships between pixels, learn hierarchical features, are robust to translation and rotation, and can be trained end-to-end. These advantages make CNNs suitable for various image-processing tasks, such as image classification, object recognition, and segmentation.

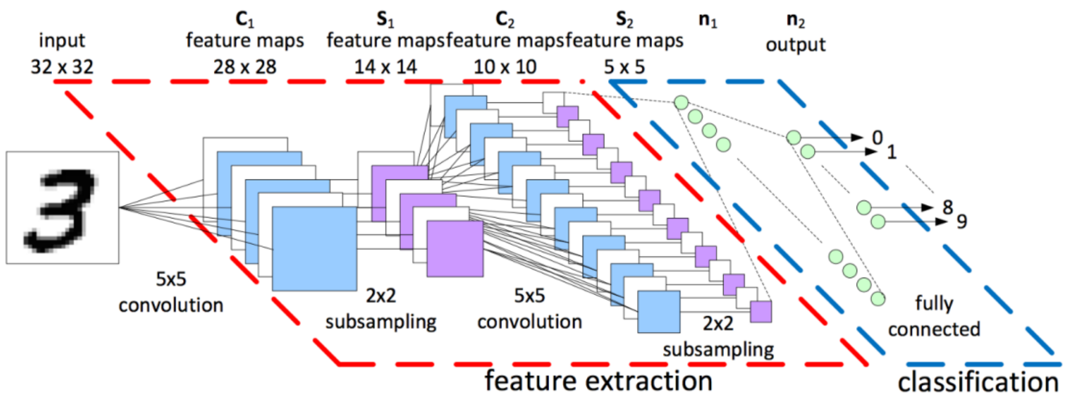


Figure 2.2: Example of Hierarchical Features in a CNN.

2.3 Related Work

2.3.1 The Impact of Aircraft Maintenance on the Aviation Industry

The aviation industry generates a staggering revenue of $2.7 trillion, accounting for roughly 3.6% of the world's GDP. However, flight delays caused by aircraft maintenance can severely affect customer satisfaction and lead to economic losses. In the United States alone, these losses amount to approximately $26.6 billion (Sims, 2019). It is also a well-known fact that significant accidents occurred during aircraft traveling, attributed primarily to the maintenance of airplanes. Maintenance of aircraft is an indispensable part of the overall booming aviation industry. The whole process of maintenance includes a complex organization that is centered around the aircraft maintenance technician (AMT) (Latorella and Prabhu, 2000). Maintenance is keeping airplanes functional and ensuring the safety of the aircraft. The primary role of AMTs is the detection and address a similar problem (Pettersen and Aase, 2008). The dependency of the aircraft's health on the AMT's detection induces a risk factor and human error (CAA, 2002). Also, there are almost 100,000 commercial flights per day and thousands of aircraft inspections per hour; these huge numbers represent a massive gap for researchers to reduce the aircraft's downtime and maintenance costs, 70% of these inspections are performed visually by AMT (Futterlieb, 2017).

Mobile robots are mainly employed for the aid of operators for the maintenance of aircraft (Parker and Draper, 1998; Kroll, 2008). Earlier work on the maintenance of the aircraft was based on automating skin inspections of the exterior (Siegel, Gunatilake and Podnar, 1998); however, these robots just perform the visual inspection, and the decision to objectify a part as faulty rests with the maintenance staff. In recent work (Donadio, Frejaville, Larnier, and Vetault, 2016; Jovančević *et al.*, 2016), the researcher assembled a robot for the precheck of the aircraft before flight based upon the visual inspection in which they solely relied on processing the captured image. To steer in the airport, Air-Cobot can reach the prescribed location of the aircraft (Donadio, Frejaville, Larnier, and Vetault, 2016) or can be made to follow a specific path such as a line following robot, in which the ground is marked with a specific strip (Bauda, Bazot and Larnier, 2017). The robot can also follow a human (Donadio, Frejaville, Larnier, and Vetault, 2016). To move around the aircraft freely, the robot uses laser and vision methods to center itself around the aircraft (Donadio, Frejaville, Larnier, and Vetault, 2016; Jovančević *et al.*, 2016). Recognition and avoidance of obstacles are used in the navigation mode of the robot (Futterlieb, Cadenat and Sentenac, 2014).

2.3.2 Potential Applications of AI and ML in the Aviation Industry

With the advent of Artificial Intelligence (AI) in several commercial industries, cutting costs, depleting the duration of design cycle time, prototyping, optimization, maintenance, manufacturing, and updating products is made much more optimized. Machine learning and AI have found a significant audience in industries ranging from aviation to autonomous driving cars and the health industry. AI has outperformed humans in high computationally expensive tasks. In aviation, (Ian, Elena and Michael, 2019) used AI for the reduction of risk involved in the manufacturing of the complex and intrigued parts of an aircraft. In (Gonzalez *et al.*, 2016), the researchers proposed a methodology for operating an un crewed combat aerial vehicle being controlled and driven by AI. In earlier work, a model is proposed for deciding to land an aircraft on aircraft carriers using a neuro-fuzzy algorithm (Richards, 2002). AI is used in the decision-making process of designing parts of the aircraft and thereby reducing the time taken for designing (Azizi Oroumieh *et al.*, 2013). In earlier work, AI was used to find the health of aging aircraft and predict those that need maintenance or need to be disposed of (‘Journal of intelligent material systems and structures’, 1990).

Advances in AI could help aerospace companies optimize their detection processes. However, machine learning and AI techniques are still not applied in the aviation industry for the detection and maintenance of aircraft, and the main reason for this is the absence of standard and benchmark work and dependability on simple models as compared to much-intrigued models (AXISCADES Engineering Technologies Limited, 2020). Machine vision has progressed from an embryonic phase to a mature field encompassing a more extensive range of applications such as surveillance, inspection, robot manufacturing, obstacle avoidance vehicle guidance, directing traffic, biometric measurement, and analysis of images from space shuttles (Davies, 2004). Computer vision in general and image processing, in particular, is the extraction of relevant information from an optical snap (Ballard and Brown, 1992). Autonomous mobile robots have been studied and researched in the previous four decades (Waxman *et al.*, 1987; Thorpe *et al.*, 1988). However, the use of computer vision and image processing in the aviation industry is utilized only for inspection and surveillance, and its application while combining machine learning and AI is yet to be explored (Sims, 2019).

Identification and utilization of a machine learning model for a specific problem is an essential factor for the accurate prediction of overall results. It generally depends on the type and nature of the input data and forms of prediction classes (Chan, Oktavianti and Puspita, 2019). As many artificial intelligence techniques can help to identify damaged parts in mechanical parts (Miranda *et al.*, 2019; Zou, 2020). Among the recent AI techniques, Convolutional Neural Networks (CNN) have proved their efficiency in most computer vision problems. The use of CNN for the identification of damage in aircraft bodies can give promising results (Zou, 2020). Among many AI-based types of research, CNN has outperformed other algorithms specifically when image processing is involved in aviation aircraft (Miranda *et al.*, 2019; Schlegl *et al.*, 2019; Bouarfa *et al.*, 2020).

Furthermore, image processing is divided into several phases (Guo, Yu and wu, 2020) and for each phase of image processing, there are many different algorithms used (Ma *et al.*, 2020). For example, images undergo pre-processing to eliminate distortion and enhance segmentation accuracy. Several elements contribute to the presence of disturbances in images. These strategies, though, should only be used whenever the dataset is limited and stable (Verma, Kaur and Arora, 2016; Konstantinov and Baryshnikov, 2017; Dey *et al.*, 2020). Once the noise has been removed, the next step is the segmentation to extract the region of interest (ROI). However, all these features cannot be extracted and utilized due to various factors related to the quality and depth of the image (Setlur Nagesh *et al.*, 2020). Similarly, to produce the best results, different machine learning classifiers have been implemented for the retrieved features throughout the classification process (Amit, 2019; Bouarfa *et al.*, 2020). In order to generate an efficient classification technique, the CNN method must first solve two key problems: First, choose the most appropriate kernel settings, and then choose the most essential collection of characteristics (Guo, Yu and wu, 2020; Stoean, 2020). To obtain the optimum subset of attributes, it is obligatory to have the best feature selection mechanism implemented. Feature grading provides certain benefits over feature extraction approaches, such as needing less preprocessing cost and presenting appropriate settings with fewer data instances. Essentially, multiple evolution metaheuristic algorithms for CNN and SVM parameter optimization have been established. Metaheuristic approaches include a randomized aspect that allows them to investigate diverse parts of the solution space, increasing solution variety and avoiding becoming trapped in local optimum solutions (Doerr *et al.*, 2013; Konstantinov and Baryshnikov, 2017; Dey *et al.*, 2020). Furthermore, heuristic strategies have already been frequently used in the research to resolve feature extraction problems; nevertheless, few methods have been created to optimize SVM features concurrently with feature extraction. Various evolutionary or swarm-based techniques, in particular, have been used for features and parameter refinement (Ma *et al.*, 2020). However, in case of time constraint and accuracy, SVM is preferred over SI in many image processing models (Ma *et al.*, 2020).

2.3.3 Automated Detection of Faulty Parts Using ML and CV Techniques

Automated and semi-automated methods for detecting faulty aircraft body parts have become increasingly important in recent years. Several studies have been conducted to explore various approaches for automating the detection process, including the use of machine learning algorithms, computer vision techniques, and signal processing.

One popular approach for the automated detection of faulty aircraft parts is through the use of machine learning algorithms. In a study by Ghodrati et al. (2019), a deep learning-based approach was proposed for the detection of cracks in aircraft fuselage using Convolutional Neural Networks (CNNs). The authors demonstrated that their proposed method achieved high accuracy and robustness in detecting cracks in real-world images of the aircraft fuselage.

Shafi *et al.* (2023) used a dataset consisting of 4500 images of the aircraft skin surface, captured using a high-resolution camera, and included different types of damage such as corrosion, cracks, and dents. The dataset was divided into training and testing sets, and the CNN model was trained using transfer learning with the VGG16 architecture. The proposed method achieved better results compared to traditional machine learning algorithms such as support vector machines and decision trees.

The authors also evaluated the performance of the CNN model by comparing it with the visual inspection results conducted by experts. The results showed that the proposed method can provide accurate and reliable results in detecting aircraft damage. The proposed method can potentially reduce the time and cost required for manual inspection and improve the safety and reliability of aircraft operations.

Another study by Ewald *et al.*, (2018) proposed a framework for the automated detection of dents on aircraft skin using a combination of machine learning and image processing techniques. The authors used a Support Vector Machine (SVM) algorithm to classify the severity of the dents based on the extracted features from the images. The results showed that their proposed framework achieved high accuracy in detecting and classifying dents on aircraft skin.

Cha, Choi and Büyüköztürk (2017), Ewald *et al.*, (2018), Bouarfa *et al.*, (2020), and Laxman *et al.* (2023) also proposed a method for automated detection and recognition of cracks in aircraft structures using deep learning techniques. The study involved collecting images of different types of cracks in aircraft structures. These images were labeled and then used to train and test the deep learning model. The authors developed a custom-built deep learning model with a CNN for feature extraction and an

LSTM for sequence modeling. The proposed model achieved an accuracy of 93.2% in detecting and recognizing cracks in the aircraft structure.

The authors further evaluated the model's performance using real-world data from aircraft structures. The proposed model was able to accurately detect and recognize cracks in aircraft structures, demonstrating its potential for real-world applications. The proposed method can potentially reduce the time and cost required for manual inspection and improve the safety and reliability of aircraft operations.

In addition to machine learning-based approaches, computer vision techniques have also been explored for the automated detection of faulty aircraft parts. In a study by Karaoğlu *et al.* (2023), a computer vision-based approach was proposed for the detection of corrosion on aircraft parts. The authors used image segmentation techniques to separate the corrosion regions from the background, followed by feature extraction and classification using a Random Forest algorithm. The results demonstrated high accuracy in detecting corrosion on aircraft parts.

Zhang *et al.* (2021) proposed a computer vision-based approach for the detection of corrosion on aircraft parts using a deep convolutional neural network (CNN) algorithm. The researchers used a dataset of images of corroded aircraft parts and developed an algorithm to automatically detect and classify the level of corrosion in the images. The study found that their approach was effective in detecting and classifying corrosion on aircraft parts with high accuracy. The authors suggest that this approach could be used to improve the efficiency and accuracy of corrosion inspection in the aviation industry.

Chapter 3: Methodology

The basis of this work is to automate the process of detection using Image processing where the computer (Artificial Intelligence) decides if the part in the image is damaged or not. The process starts with capturing images of both the *damaged* and *normal* body of the aircraft with a specific model. Once these images are captured, it starts with using Python for processing these images. Here extraction serves two tasks, firstly processing the image at the pixel level and finding the fine difference between an image in which the body is damaged and a normal body image; and secondly, as we analyzed images at pixel levels, so, these extracting features reduce the size. These vital pieces of information, i.e., features, have been extracted for all the images, both with damaged and normal bodies and were stored alongside their corresponding image.

The data in our case were the features extracted from each image. Then we applied our algorithm to learn these features, training consists of making the algorithm learn patterns in the images against their class label. Once the training phase is complete, the testing phase starts in which the features from new images were fed to the CNN algorithm but with the class label hidden and we checked how efficiently our method can detect the damaged location in a body of aircraft. Figure 3.1 shows the basic steps of image processing and classification. A detailed description of the overall process of each phase is depicted in Figure 3.1 (Gupta, 2017):

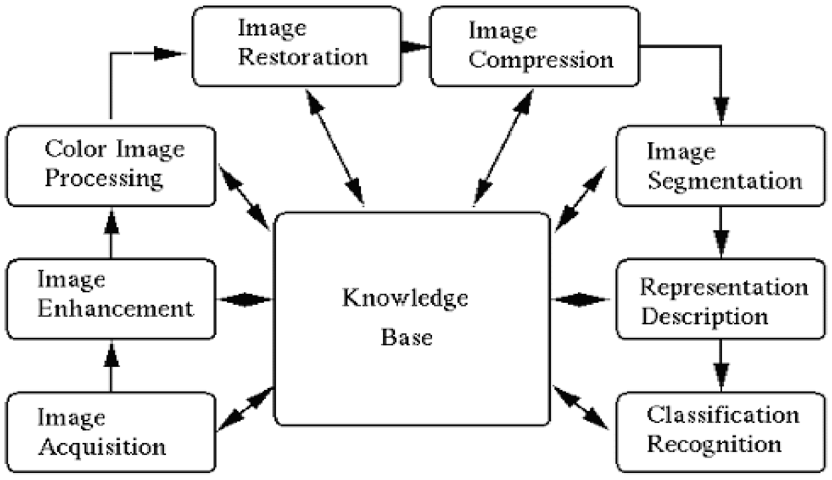


Figure 3.1: Phases of Image Processing.

3.1 Collecting Data Set

The most important factor of this research is the accurate dataset. Therefore, we collected refined images of aircraft bodies for specific models. Each image has been stored in a repository with the class label.

3.2 Pre-Processing

A better preprocessing technique results in refined data for the modeling. The image data should be pre-processed before creating the CNN model. Reduction of speckles helps in improving the identification accuracy. And the data should be converted to tensors acceptable by the Neural Networks.

3.2.1 Image Acquisition to Image Compression

The first step in image processing is to acquire an image. In python, there is a number of libraries to acquire an image or read a jpg or bitmap file. After acquiring, the transformation of the image has been done using erode, dilute, and histogram equalization concepts. Finally, to prepare an image for segmentation, the dimensions of the image has been set to a predefined value using TensorFlow built-in libraries.

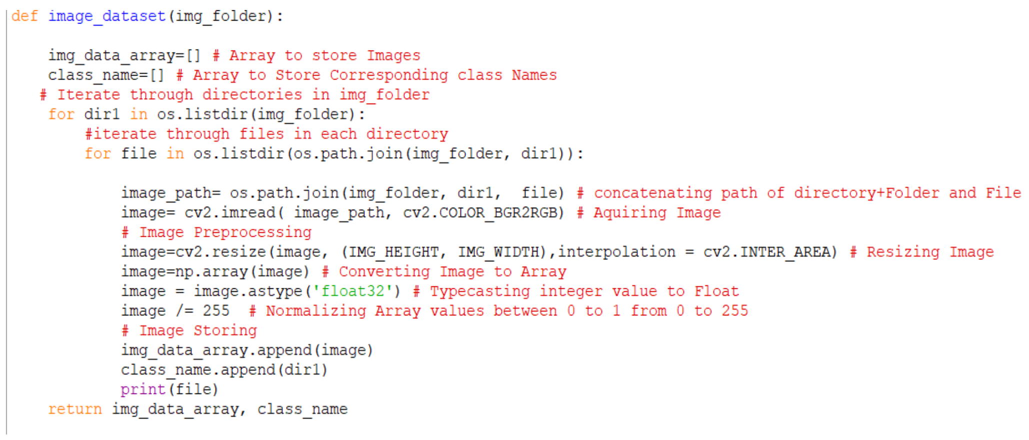
3.2.2 Image Segmentation

Image segmentation is the prior task in image classification. Correct Segmentation leads to accurate results. The two forms of segmentation techniques used in image processing that is selective and extractive. The selective approach is more robust than the extractive costing extra space. However, in aviation, there are certain similar parts integrated to form a single working machine. Therefore, a selective approach is a feasible choice for this sort of image processing.

3.2.3 Data Cleaning

The segmentation process may generate some missing, noisy, and redundant values at the corners of the segments. Missing values in the data have been evaluated, and depending on the nature of the features and the records, these values may be replaced with calculated values, the files dropped, or the features removed entirely. During this phase, missing, noisy, and inconsistent data has been addressed using a regression model. The most common model used in these situations is Naïve Bayes.

Figure 3.2 shows the step performed from image acquisition to image compression using a Numpy array in Python.



```python
def image_dataset(img_folder):

    img_data_array=[] # Array to store Images
    class_name=[] # Array to Store Corresponding class Names
   # Iterate through directories in img_folder
    for dir1 in os.listdir(img_folder):
        #iterate through files in each directory
        for file in os.listdir(os.path.join(img_folder, dir1)):

            image_path= os.path.join(img_folder, dir1,  file) # concatenating path of directory+Folder and File
            image= cv2.imread( image_path, cv2.COLOR_BGR2RGB) # Aquiring Image
            # Image Preprocessing
            image=cv2.resize(image, (IMG_HEIGHT, IMG_WIDTH),interpolation = cv2.INTER_AREA) # Resizing Image
            image=np.array(image) # Converting Image to Array
            image = image.astype('float32') # Typecasting integer value to Float
            image /= 255  # Normalizing Array values between 0 to 1 from 0 to 255
            # Image Storing
            img_data_array.append(image)
            class_name.append(dir1)
            print(file)
    return img_data_array, class_name
```

Figure 3.2: Image Preprocessing Steps.

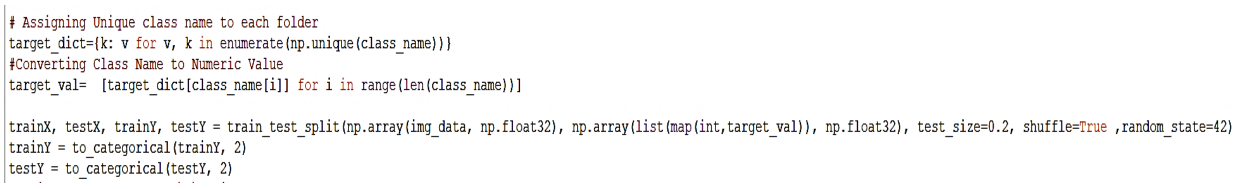
3.3 Feature Selection

In this phase of the process, a dataset is passed through ML algorithms again to select the key features for the dataset. The overfitting quality of the ML algorithm makes sure the chosen data must remain inbound, thus avoiding overlapping of features and data.

3.4 Data Splitting

As per standards of machine learning model implementation, the data is divided into three categories of training, testing, and validation. The training data have been used to train the model; testing verifies the outcome, and the correctness of the trained model, and the validation portion applies specific techniques to validate the data. For this work, we have initially divided the data into two splits of 80-20. 80% data is used for training and validation, while 20% is used for testing after the model is selected. The initial 80% is further divided using three different splits of 70-30, 80-20, and 90-10. The most appropriate among these is an 80-20 split for our dataset.

Figure 3.3 shows the train-test split applied for the experiments. Also, the last two lines show the transformation of textual data of class labels to encoded form using the categorical function of the Numpy array. It assigns 0 to the Faulty class and 1 to the Normal class of images.



```
# Assigning Unique class name to each folder
target_dict={k: v for v, k in enumerate(np.unique(class_name))}
#Converting Class Name to Numeric Value
target_val=  [target_dict[class_name[i]] for i in range(len(class_name))]

trainX, testX, trainY, testY = train_test_split(np.array(img_data, np.float32), np.array(list(map(int,target_val)), np.float32), test_size=0.2, shuffle=True ,random_state=42)
trainY = to_categorical(trainY, 2)
testY = to_categorical(testY, 2)
```

Figure 3.3: Train Test Split of the Dataset.

3.5 Model Definition

The goal of this research is to define a model for damage detection class in aircraft body maintenance checks. The study of the literature suggests that there are many techniques used for this purpose, both in the domain of traditional and machine learning. However, it is noticed that traditional statistical and robotics models show better results only with manual decision capabilities. Therefore, in this work, the focus is on machine learning AI models (Fatima and Pasha, 2017; Konstantinov and Baryshnikov, 2017; Ma *et al.*, 2020) to fully automate the system to improve efficiency and time. Image data is mostly dealt with by neural networks because of their random approach toward data and the division of data into subsequent layers (Chan, Oktavianti and Puspita, 2019). The key features are extracted through pruning and set into a new matrix. However, the neural networks alone fall to the local optimal following the divergence of the data scheme. To avoid this, mostly, some other techniques are combined with neural networks (Stoean, 2020). In the domain of image processing, the two most promising hybrid techniques are CNN with SVM and Swarm Intelligence (SI) with CNN. However, SI costs more time than SVM. There exists an alternative to the hybrid approach which is by increasing the number of images using augmentation. In this study, the augmentation technique has been applied. Moreover, a CNN should be built by stacking Convolution, max pooling, activation, and fully connected layers. The arrangement of these layers depends on the problem to be solved. This arrangement is referred to as CNN architecture. We can build the CNN architecture referring to the available pre-trained architectures, (e.g.: LeeNet, VGGNet, UNet, etc).

Moreover, the hypothesis equation of CNN is given in Equation (3.1), while Equation (3.2), shows the mathematical formula for computing cost:

$$Z = W^T . X + b \qquad (3.1)$$

Where, W represents the weight matrix, X shows the input and b describes the bias term.

$$J(w_0, w_1) = \frac{1}{2m}\sum_{i=1}^{m} Loss(x^i, y^i, w) \qquad (3.2)$$

Similarly, to compute Gradient Descent for weight optimization, the mathematical form given in Equation (3.3) can be used:

$$w_j := w_j - \alpha \frac{\partial}{\partial w_j} J(w) \qquad (3.3)$$

3.6 Training

The model was trained with the set of data selected for training purposes. Before training data, some random inputs were generated to test the correctness of the model.

3.7 Model Optimization and Validation

After training and testing the dataset, the model has been validated with certain mathematical libraries. The Mean Absolute Percentage Error (MAPE) is the most common method of validation from the literature. To be more precise a second metric assumed for testing is accuracy, precision, and other performance metrics using the confusion matrix can also be used.

Following equations represent the mathematical form of the evaluation metrics used in this study which are recall, precision, and accuracy respectively.

$$Recall = \frac{TP}{TP+FN} \qquad (3.4)$$

$$Precision = \frac{TP}{TP+FP} \qquad (3.5)$$

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \qquad (3.6)$$

$$F1\ Score = \frac{2*Precision*Recall}{Precision+Recall} \qquad (3.7)$$

Where, TP: True Positive, FP: False Positive, TN: True Negative, FN: False Negative. For further explanation in our case, Positive is Normal Class and Negative is Faulty class. Similarly, True represents correct identification, and False shows wrong prediction.

3.8 Experimental Setup

In this thesis, two different neural network architectures are used. Firstly, a simple Artificial Neural Network is applied using five-layer architecture having 1 input, 3 hidden, and 1 output layer. Secondly, a thirteen-layer architecture of CNN is used that contains 1 input, 1 output, and 11 hidden layers. The hidden layers also contain Batch Normalization, Dropout, and Max Pooling layers to maintain the data in between the layers. Table 3.1 represents the hyperparameter used for the experiments.

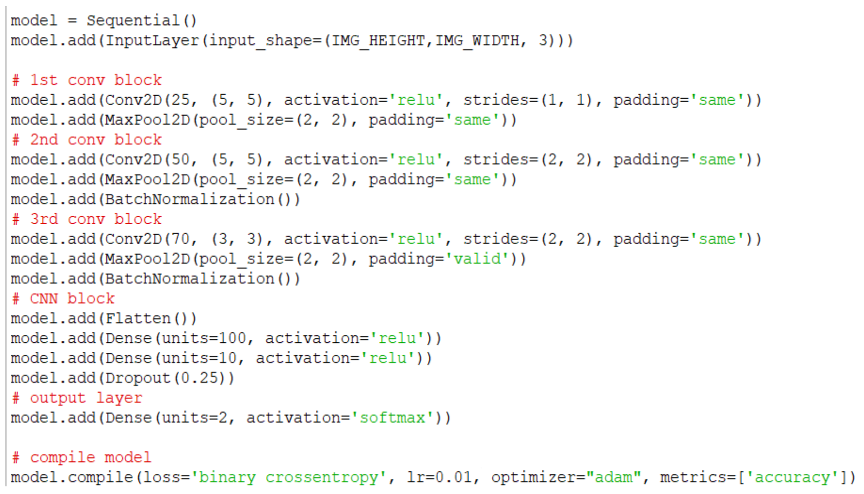
Table 3.1: Hyperparameter Descriptions.

Hyperparameter	**Value**	**Description**
l1_ratio	None	Regularization of the output values
filters	32,64	The number of filters (also known as channels) used in the convolutional layers.
kernel_size	3,5	The size of the convolutional kernel used to extract features from input data.
	ReLU	Ignores the negative values
	Softmax	Converts values based on the Entropy function
Rate (drop)	0.25	Fraction of the input units to drop in the dropout layer.
LR	0.01	The step size shrinkage is used in updating the weights of the functions at each layer.
Optimizer	Adam	Algorithms used to change the attributes of neurons, for instance, weighs and learning rate to optimize the loss
epoch	5, 10	Number of iterations of the model
Loss	Categorical_crossentropy	Method for computing and returning loss value
Metrics	Acuracy	Model Evaluation metric

Table 3.1: Hyperparameter Descriptions (Contd.).

Hyperparameter	**Value**	**Description**
verbose	2	The nature of output shown on the console screen
Batch_size	8,32	The minimum number of samples at each epoch
Strides	2	Number of pixel shifts on the input matrix
Padding	0,2	Number of pixels added to the boundary of an input matrix
Batch Normalization	-	Batch Normalization is a technique used in neural networks to improve the training process and performance. It involves normalizing the input to a layer to have zero mean and unit variance, which helps to stabilize the distribution of inputs to the layer.
Dropout	0.25	Dropout is a regularization technique used in Convolutional Neural Networks (CNNs) to prevent overfitting of the model. Overfitting occurs when the model learns to fit the training data too closely, resulting in poor generalization of new data.

Figure 3.4 shows the CNN architecture applied to represent a number of layers and a hyperparameter of each layer in the model.



```
model = Sequential()
model.add(InputLayer(input_shape=(IMG_HEIGHT,IMG_WIDTH, 3)))

# 1st conv block
model.add(Conv2D(25, (5, 5), activation='relu', strides=(1, 1), padding='same'))
model.add(MaxPool2D(pool_size=(2, 2), padding='same'))
# 2nd conv block
model.add(Conv2D(50, (5, 5), activation='relu', strides=(2, 2), padding='same'))
model.add(MaxPool2D(pool_size=(2, 2), padding='same'))
model.add(BatchNormalization())
# 3rd conv block
model.add(Conv2D(70, (3, 3), activation='relu', strides=(2, 2), padding='same'))
model.add(MaxPool2D(pool_size=(2, 2), padding='valid'))
model.add(BatchNormalization())
# CNN block
model.add(Flatten())
model.add(Dense(units=100, activation='relu'))
model.add(Dense(units=10, activation='relu'))
model.add(Dropout(0.25))
# output layer
model.add(Dense(units=2, activation='softmax'))

# compile model
model.compile(loss='binary crossentropy', lr=0.01, optimizer="adam", metrics=['accuracy'])
```

Figure 3.4: Model Definition.

3.9 Python Libraries

Table 3.2 presents the python libraries used in the development of this project briefly describing the task they are used for. A few of these libraries such as Numpy, Sklearn, and Matplotlib are default libraries for any image processing or computer vision tasks. While TensorFlow is used for the implementation of CNN models. The alternative approach can be the use of a Pytorch. TensorFlow is preferred over Pytorch due to its ease of implementation and integration with the GUI module.

Table 3.2: Use of Python Libraries.

Python Library	**Task**
Numpy	• For Image arrays manipulation and filtering
Tensor Flow	• Formation of tensors for CNN.
Tensor Flow. keras	• Modelling (.models/sequential) • Layering functions such as Dense, Conv2D (.layers) • Preprocessing of images such as image (.preprocessing)
Matplotlib	• Reading images (imread) • Erosion and dilution • Histogram equalization
Theano	• Alternative for TensorFlow
Sklearn	• For SVM • For data splitting

Chapter 4: Results

4.1 Overview

This section presents the results of various experiments conducted in this work for image classification using the CNN classifier. The results are evaluated for two different datasets of aircraft images that are actual images collected and augmented images created. The actual images are small in number that face certain fitting and learning issues so therefore a new dataset is formed using image augmentation techniques. It is concluded from the selected five executions of the model that the CNN classifier achieves a maximum of 98.28% training accuracy and 81.78% of validation accuracy on the augmented dataset.

4.2 Image Data Augmentation

The data augmentation technique is used to increase the size of the dataset by flipping, rotating, and skewing the provided images. In this study, the images from each of the faulty and normal folders are selected randomly and applied with a random augmentation feature. For instance, images ‘5b’ and ‘43b’ are the same images of a faulty body part that are augmented to form ‘97’ more images in random selection as shown in Figure 4.1. Beside flip and rotation, blur and sharpen features are used to transform the images. Furthermore, in the implementation of the CNN classifier, image augmentation is applied again on the batch size of 32. The image augmentation allows the model to train itself with an increased number of convolutional layers and avoids overfitting of the classes. Image augmentation also provides stability in results and avoids overfitting along with increased accuracy. Each one of these is explained in the following sections. Please note the images shown in Figure 4.1 are real and depict a removable external panel on the aircraft body with holes for fasteners. All of these holes are prone to corrosion, which makes them appear quite prominent. The picture is used for training purposes by changing its angles and altering its properties.

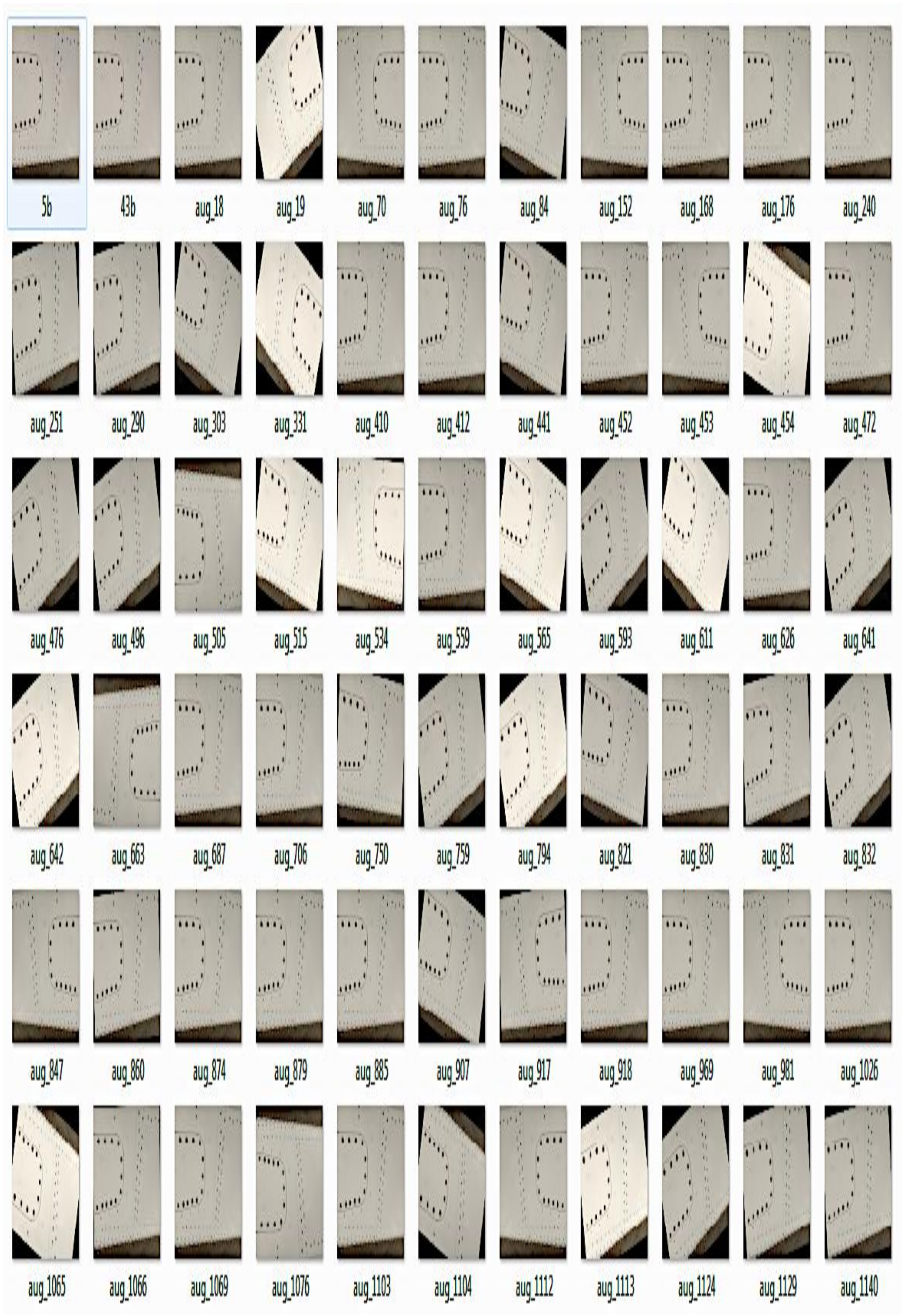


Figure 4.1: Augmented Images of 5b and 43b.

4.2.1 Effect on Accuracy

As image augmentation increases the size of data, therefore, there are a greater number of images for training and testing. So, the model learns more quickly and effectively. This also affects the results in form of training and validation accuracy as shown in Table 4.1 and Table 4.2. The validation accuracy before augmentation on small datasets is 0.44 much smaller in comparison to the validation accuracy of 0.62 after augmentation on a large dataset.

4.2.2 Effect on Stability

Image augmentation stabilizes the model in terms of accuracy gain and loss. The augmented dataset shows a stabilized increase with each epoch in comparison to the actual dataset. This can be shown in Figure 4.2 and Figure 4.3. In Figure 4.2 the accuracy shuffles between lower and higher values randomly and then settles to a single result while in Figure 4.3 the accuracy gradually starts increasing from lower value to higher value and rarely comes down during the process. The results shown in Figure 4.2 and Figure 4.3 are from the default Neural Network implementation. The same pattern is followed when applied with CNN.

4.2.3 Effect on Overfitting

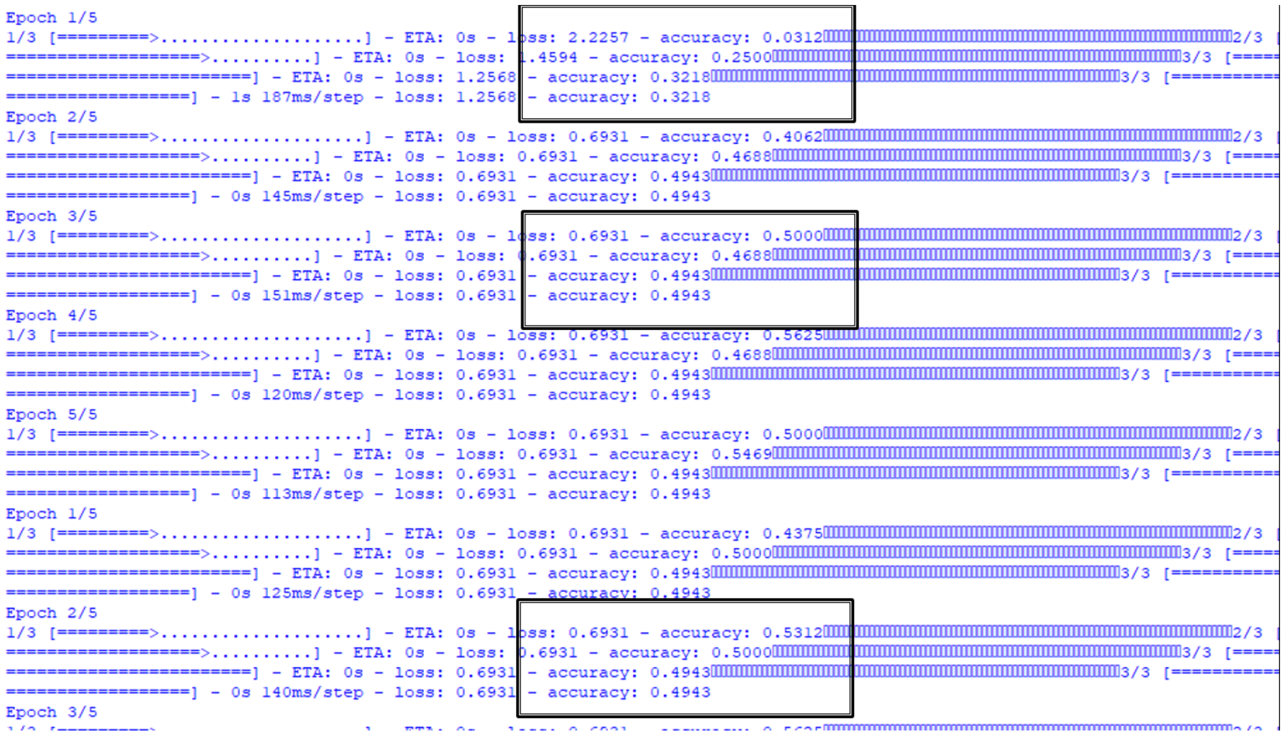
Upon observation of Table 4.1, it is evident that all the values in the "Normal" class are zero. The reason for all the values in the "Normal" class being zero in Table 4.1 is that the model is only trained to recognize the "Faulty" class in the given dataset. When a small dataset is used to train a CNN, the model tends to overfit one class, resulting in high accuracy but limited ability to identify other classes. To address this issue, a supported algorithm such as SVM can be introduced or the dataset can be expanded using augmentation techniques. As shown in Table 4.2, overfitting has been addressed by using augmentation techniques. Table 4.1 contains only 18 support elements in the test dataset, whereas Table 4.2 contains 818 support elements, with the same 80-20 training and testing split.

Table 4.1: Result Table for Small Dataset Epochs=5.

	precision	recall	f1-score	support
Faulty	0.44	1.00	0.62	8
Normal	0.00	0.00	0.00	10
accuracy			0.44	18
macro avg	0.22	0.50	0.31	18
weighted avg	0.20	0.44	0.27	18

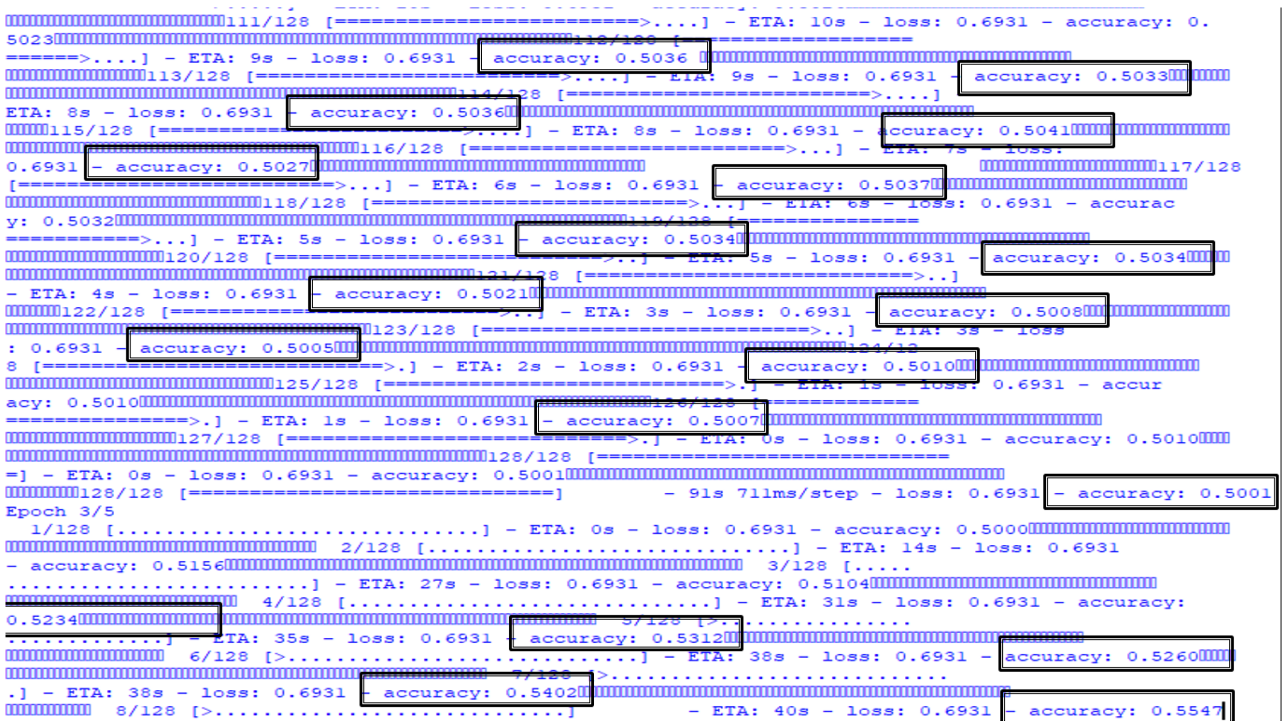
Table 4.2: Result Table for Large Dataset Epochs=5.

	precision	recall	f1-score	support
Faulty	1.00	0.24	0.38	421
Normal	0.55	1.00	0.71	397
accuracy			0.61	818
macro avg	0.78	0.62	0.55	818
weighted avg	0.78	0.61	0.54	818



```
Epoch 1/5
1/3 [=========>....................] - ETA: 0s - loss: 2.2257 - accuracy: 0.0312▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯2/3 [
===================>..........] - ETA: 0s - loss: 1.4594 - accuracy: 0.2500▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [=====
=========================] - ETA: 0s - loss: 1.2568 - accuracy: 0.3218▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [==========
==================] - 1s 187ms/step - loss: 1.2568 - accuracy: 0.3218
Epoch 2/5
1/3 [=========>....................] - ETA: 0s - loss: 0.6931 - accuracy: 0.4062▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯2/3 [
===================>..........] - ETA: 0s - loss: 0.6931 - accuracy: 0.4688▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [=====
=========================] - ETA: 0s - loss: 0.6931 - accuracy: 0.4943▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [==========
==================] - 0s 145ms/step - loss: 0.6931 - accuracy: 0.4943
Epoch 3/5
1/3 [=========>....................] - ETA: 0s - loss: 0.6931 - accuracy: 0.5000▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯2/3 [
===================>..........] - ETA: 0s - loss: 0.6931 - accuracy: 0.4688▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [=====
=========================] - ETA: 0s - loss: 0.6931 - accuracy: 0.4943▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [==========
==================] - 0s 151ms/step - loss: 0.6931 - accuracy: 0.4943
Epoch 4/5
1/3 [=========>....................] - ETA: 0s - loss: 0.6931 - accuracy: 0.5625▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯2/3 [
===================>..........] - ETA: 0s - loss: 0.6931 - accuracy: 0.4688▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [=====
=========================] - ETA: 0s - loss: 0.6931 - accuracy: 0.4943▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [==========
==================] - 0s 120ms/step - loss: 0.6931 - accuracy: 0.4943
Epoch 5/5
1/3 [=========>....................] - ETA: 0s - loss: 0.6931 - accuracy: 0.5000▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯2/3 [
===================>..........] - ETA: 0s - loss: 0.6931 - accuracy: 0.5469▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [=====
=========================] - ETA: 0s - loss: 0.6931 - accuracy: 0.4943▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [==========
==================] - 0s 113ms/step - loss: 0.6931 - accuracy: 0.4943
Epoch 1/5
1/3 [=========>....................] - ETA: 0s - loss: 0.6931 - accuracy: 0.4375▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯2/3 [
===================>..........] - ETA: 0s - loss: 0.6931 - accuracy: 0.5000▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [=====
=========================] - ETA: 0s - loss: 0.6931 - accuracy: 0.4943▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [==========
==================] - 0s 125ms/step - loss: 0.6931 - accuracy: 0.4943
Epoch 2/5
1/3 [=========>....................] - ETA: 0s - loss: 0.6931 - accuracy: 0.5312▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯2/3 [
===================>..........] - ETA: 0s - loss: 0.6931 - accuracy: 0.5000▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [=====
=========================] - ETA: 0s - loss: 0.6931 - accuracy: 0.4943▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯3/3 [==========
==================] - 0s 140ms/step - loss: 0.6931 - accuracy: 0.4943
Epoch 3/5
1/3 [=========>....................] - ETA: 0s - loss: 0.6931 - accuracy: 0.5625▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯2/3
```

Figure 4.2: Destabilized Accuracy Results of Actual Images.



```
111/128 [========================>....] - ETA: 10s - loss: 0.6931 - accuracy: 0.
5023112/128 [==================
======>....] - ETA: 9s - loss: 0.6931 - accuracy: 0.5036 
113/128 [========================>....] - ETA: 9s - loss: 0.6931 - accuracy: 0.5033
114/128 [=========================>....]
ETA: 8s - loss: 0.6931 - accuracy: 0.5036
115/128 [=========================>....] - ETA: 8s - loss: 0.6931 - accuracy: 0.5041
116/128 [==========================>...] - ETA: 7s - loss:
0.6931 - accuracy: 0.5027                                   117/128
[==========================>...] - ETA: 6s - loss: 0.6931 - accuracy: 0.5037
118/128 [==========================>...] - ETA: 6s - loss: 0.6931 - accurac
y: 0.5032119/128 [================
===========>...] - ETA: 5s - loss: 0.6931 - accuracy: 0.5034
120/128 [===========================>..] - ETA: 5s - loss: 0.6931 - accuracy: 0.5034
121/128 [===========================>..]
- ETA: 4s - loss: 0.6931 - accuracy: 0.5021
122/128 [===========================>..] - ETA: 3s - loss: 0.6931 - accuracy: 0.5008
123/128 [===========================>..] - ETA: 3s - loss
: 0.6931 - accuracy: 0.5005124/12
8 [============================>.] - ETA: 2s - loss: 0.6931 - accuracy: 0.5010
125/128 [============================>.] - ETA: 1s - loss: 0.6931 - accur
acy: 0.5010126/128 [==============
===============>.] - ETA: 1s - loss: 0.6931 - accuracy: 0.5007
127/128 [============================>.] - ETA: 0s - loss: 0.6931 - accuracy: 0.5010
128/128 [==============================
=] - ETA: 0s - loss: 0.6931 - accuracy: 0.5001
128/128 [==============================]      - 91s 711ms/step - loss: 0.6931 - accuracy: 0.5001
Epoch 3/5
  1/128 [..............................] - ETA: 0s - loss: 0.6931 - accuracy: 0.5000
  2/128 [..............................] - ETA: 14s - loss: 0.6931
- accuracy: 0.5156  3/128 [.....
.........................] - ETA: 27s - loss: 0.6931 - accuracy: 0.5104
  4/128 [..............................] - ETA: 31s - loss: 0.6931 - accuracy:
0.5234  5/128 [>.................
.............] - ETA: 35s - loss: 0.6931 - accuracy: 0.5312
  6/128 [>.............................] - ETA: 38s - loss: 0.6931 - accuracy: 0.5260
  7/128 [>.............................
.] - ETA: 38s - loss: 0.6931 - accuracy: 0.5402
  8/128 [>.............................]       - ETA: 40s - loss: 0.6931 - accuracy: 0.5547
```

Figure 4.3: Stabilized Accuracy Results of Augmented Images.

4.3 Results

The results are generated in Python 3.8.x environment. A 64-bit version of the software has been used to avoid TensorFlow compatibility issues. Moreover, the model is trained for both datasets. In total three executions are conducted, one for the small dataset and two for the large dataset, and the evaluation matrix is set to be accurate. There are two accuracy measures discussed in the results. The accuracy shown in the tables represents validation accuracy while at some point training accuracy is also discussed and is presented through figures.

4.3.1 On Actual Dataset

The actual dataset contains 43 faulty and 44 normal images. The execution is conducted with one input layer, two convolutional layers, and one output layer. The maximum accuracy on training is 72.97% with a validation accuracy of 44.44%. However, on in-depth analysis, it is observed that it only identifies “faulty” images due to less amount of learning data available. In precise words, the model overfits the “faulty” class. Figure 4.4 shows the decrease in the validation loss with no improvement in validation accuracy. This is due to the fact; it recognizes all the faulty images as faulty as shown in Table 4.1 with the recall of faulty row as 1.00.

4.3.2 On Augmented Dataset

The augmented dataset contains approximately 2000 images for normal and faulty classes. The two executions are conducted with one input layer, three convolutional layers, and one output layer with epochs values of 5 and 10 respectively. The results are shown in Table 4.2 and Table 4.3 for each epoch. Table 4.2 is executed with epochs 5 and shows a validation accuracy of 0.61 0r 61% while Table 4.3 with epochs value of 10 shows a validation accuracy of 0.81 or 81%. It can be observed from the results that more time training vectors are used to update the weights shows positive results in terms of accuracy and loss. However, it is also observed that the maximum values are shown at the midpoint of execution as shown in Figure 5.5. The graph for validation loss comes down till mid and then goes up after mid. The validation accuracy at epochs value of 3 is maximum and validation loss is minimum. The same pattern is observed with epochs repetition of 10 at epochs value of 6. So, to conclude the maximum validation accuracy achieved is with epochs 10 at level 6 of 90.34% while the maximum training accuracy achieved is 98.28%. However, after executions, the maximum values of training accuracy are 98% and validation accuracy of 81.78%.

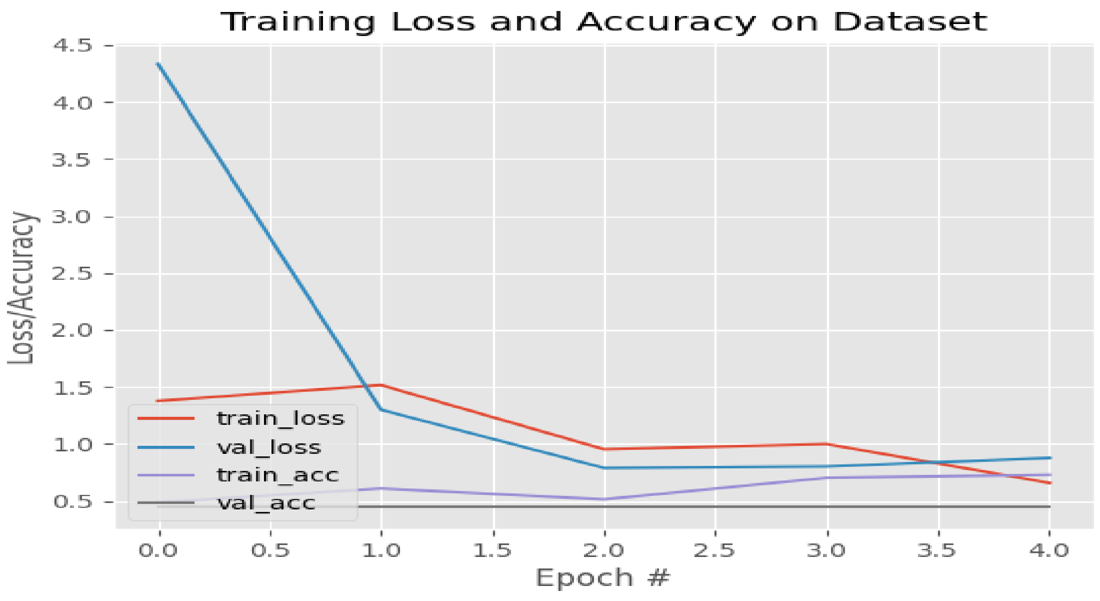


Figure 4.4: Result Graph of Small Dataset.

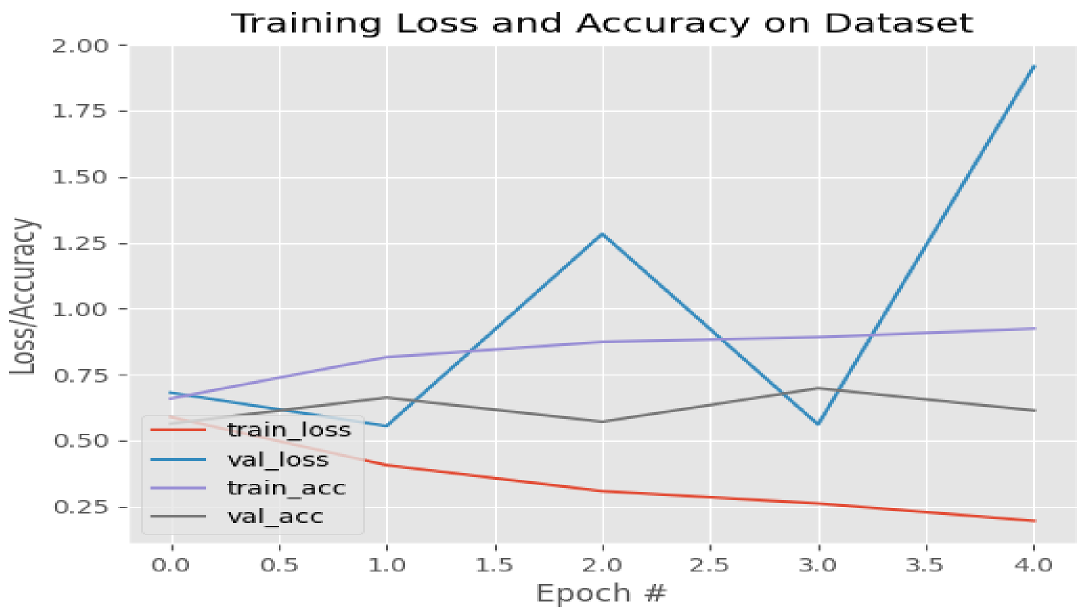
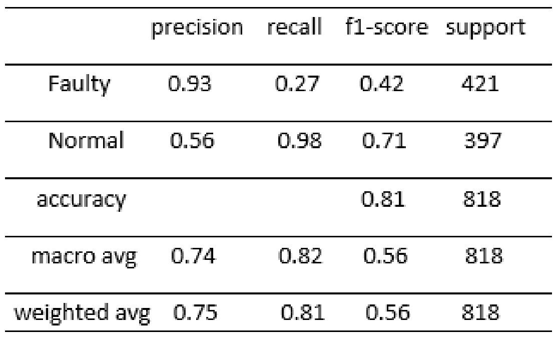


Figure 4.5: Result Graph of Large Dataset.

Table 4.3: Result Table for Large Dataset Epochs=10.



	precision	recall	f1-score	support
Faulty	0.93	0.27	0.42	421
Normal	0.56	0.98	0.71	397
accuracy			0.81	818
macro avg	0.74	0.82	0.56	818
weighted avg	0.75	0.81	0.56	818

4.4 Graphical User Interface (GUI) for the Project

After preprocessing, modeling, and a successful run of the designed code, a GUI-based user interface is developed. With this interface, the user can load the folder of images that the user needs to classify as faulty or normal. Figure 4.6 shows the default interface of the program. Once the user selects the folder, all images in the folder are listed in the side column (Figure 4.7). After this user is just required to click on any image name in the list and the trained model shows it as faulty or normal (Figure 4.8). For this purpose, the preprocessing and model training is stored in the form of pre-compiled files to reduce the complexity of the interface design.

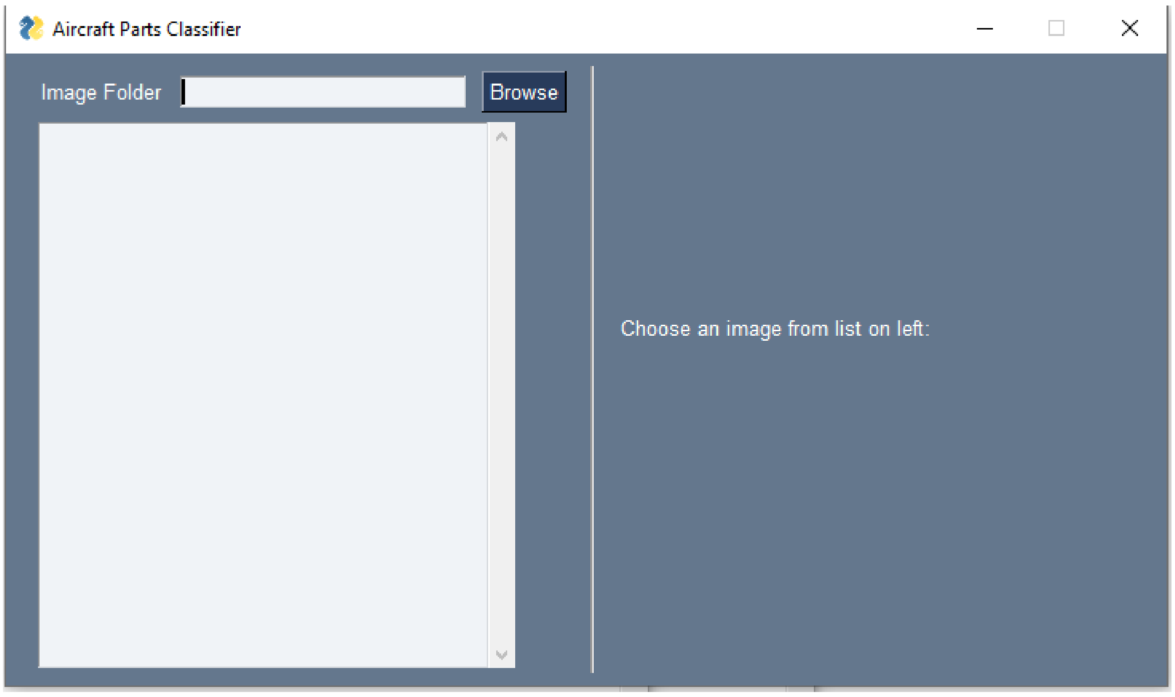


Figure 4.6: Interface Home.

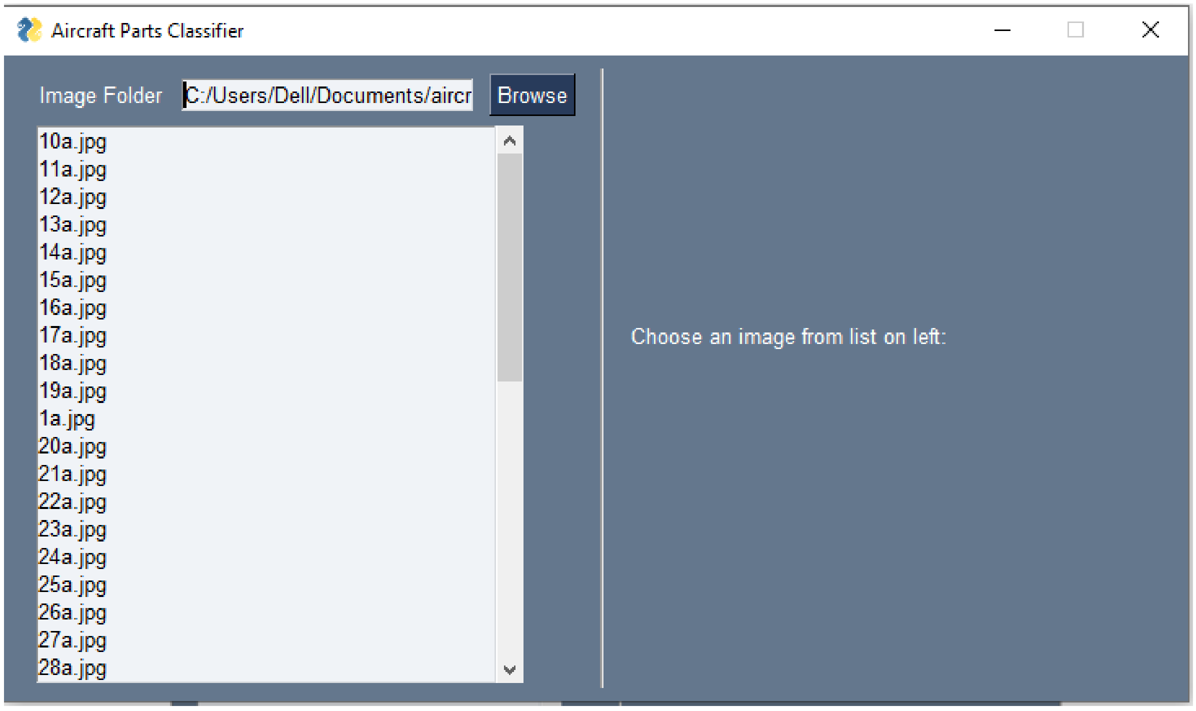
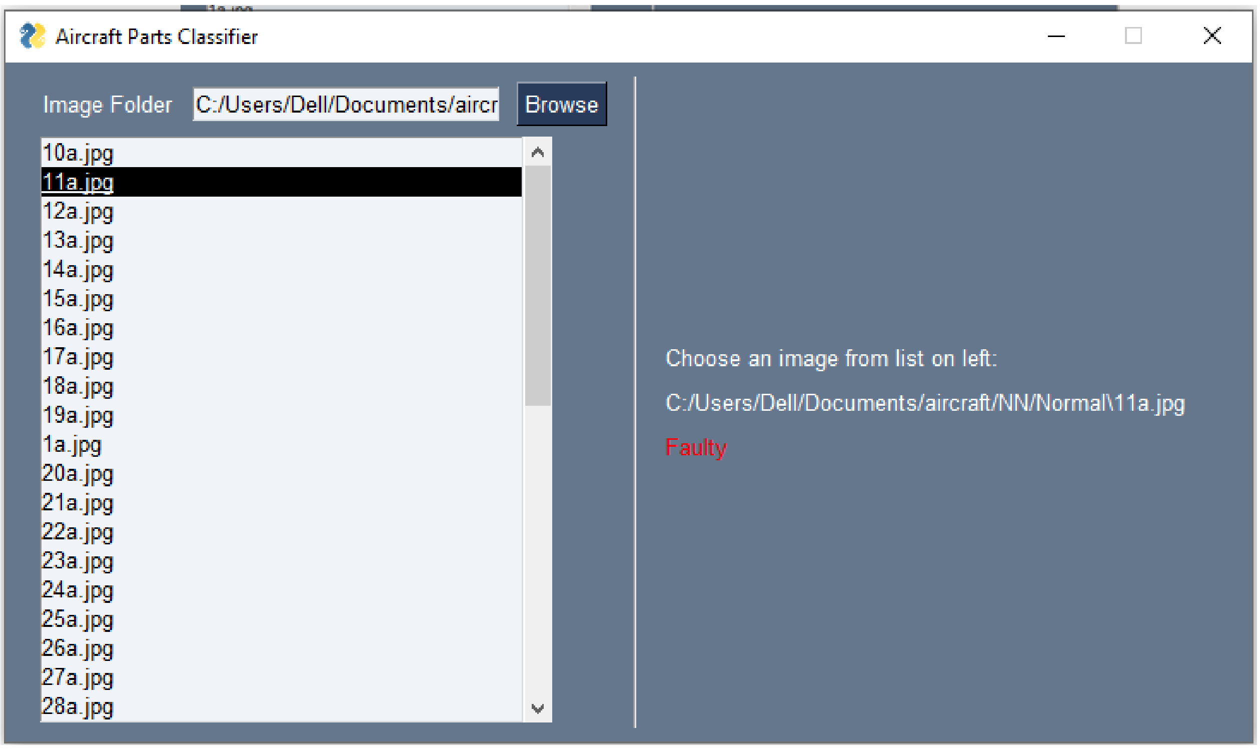


Figure 4.7: Selecting Folder.



10a.jpg
11a.jpg
12a.jpg
13a.jpg
14a.jpg
15a.jpg
16a.jpg
17a.jpg
18a.jpg
19a.jpg
1a.jpg
20a.jpg
21a.jpg
22a.jpg
23a.jpg
24a.jpg
25a.jpg
26a.jpg
27a.jpg
28a.jpg

Choose an image from list on left:
C:/Users/Dell/Documents/aircraft/NN/Normal\11a.jpg
Faulty

Figure 4.8: Classification Interface.

4.5 Comparative Study

Cha, Choi and Büyüköztürk, (2017) proposed a deep learning-based method for crack detection in aircraft structures using Convolutional Neural Networks (CNNs). The authors used a dataset of images of cracked and non-cracked specimens to train and evaluate their model. They compared their CNN-based method with traditional machine learning approaches, such as Support Vector Machines (SVMs) and Decision Trees, and showed that their proposed method achieved higher accuracy and better performance. The CNN-based method achieved an accuracy of 96.7% for detecting cracks in the test dataset, which was significantly higher than the SVM-based method (84.2%) and the decision tree-based method (72.3%). The proposed CNN-based method was also shown to handle complex crack shapes and sizes, making it suitable for practical applications in the aerospace industry.

Miranda *et al.* (2019) proposed a method for automated defect detection and classification on aerospace composite structures using deep learning. The authors used a dataset of high-resolution images of different types of defects in composite structures to train and evaluate their model. The results showed that their proposed method achieved high accuracy in detecting and classifying defects in aerospace composite structures. Specifically, the proposed method achieved an overall detection accuracy of 96.7% and a precision of 97.5% on a test dataset of images. The authors also compared their method with other state-of-the-art approaches for defect detection and classification and showed that their proposed method outperformed them in terms of accuracy and robustness.

Yasuda *et al.* (2021) proposed a method for automatic detection of surface defects on airplanes using deep CNNs. The authors used a dataset of high-resolution images of different types of surface defects on airplanes to train and evaluate their model. The proposed method achieved an overall detection accuracy of 92.1% and a precision of 97.2% on a test dataset of images. The authors also compared their method with other state-of-the-art approaches for surface defect detection and showed that their proposed method outperformed them in terms of accuracy and robustness.

Ding *et al.* (2022) proposed a deep learning-based method for detecting cracks in aircraft structures using a combination of CNNs and Recurrent Neural Networks (RNNs). The authors used a dataset of images of cracked and non-cracked specimens to train and evaluate their model. The proposed method achieved an accuracy of 99.31% for detecting cracks in the test dataset, which was significantly higher than other state-of-the-art methods, such as traditional machine learning approaches and other deep learning-based methods.

In this work, we conducted multiple executions of the model with various epochs, and concluded that the best results were obtained with three convolutional layers and between 5 to 8 epochs, achieving a training accuracy of 98.28%. Our results outperformed the results obtained in the aforementioned papers, except for (Ding *et al.*, 2022), where our results were found to be close to their results. Our study demonstrated the effectiveness of using deep learning-based methods, specifically CNNs, for crack and defect detection in aerospace structures, and highlighted the potential of these methods for improving the efficiency and accuracy of maintenance and inspection processes in the aerospace industry.

Chapter 5: Discussion

This chapter discusses the results of the study aimed at classifying aircraft body images as either faulty or normal. The experiments were conducted using a small actual dataset of 87 images and a large augmented dataset of 4000 images. It was observed that with the small dataset, the Convolutional Neural Network (CNN) overfits for a single class nine out of ten times. However, increasing the size of the dataset can rectify this issue. In the large dataset, tuning the CNN with the correct number of convolutional layers, epoch values, and batch size is crucial for achieving better accuracy and efficiency. The batch size plays a significant role in the algorithms' efficiency as it is not feasible to load all images in memory at once with a large number of images.

Moreover, adding more convolution layers or repeating epochs may not necessarily lead to perfect results. One must decide on the appropriate number of epochs and convolutional layers to avoid overfitting and overlearning the model. In this study, using three convolutional layers and 10 epochs provided better results and efficiency than other combinations.

Furthermore, the study stores the approved model and dataset in files for future utilization. The Keras. TensorFlow library stores the model, while the dataset is first converted to a numpy array and then stored in files. This method eliminates the need to train the model repeatedly and reduces the system's complexity.

The study also employed data augmentation techniques to increase the dataset's size. Data augmentation is a process of generating new training data by applying transformations such as flipping, rotating, or scaling the original images. This helps reduce overfitting and improves the model's generalization ability.

Additionally, the study employed a specific type of CNN architecture that significantly impacts the model's performance. Researchers usually choose the architecture based on the complexity of the task, the dataset size, and the available computational resources. The performance of the model may also vary depending on the quality of the dataset, including the resolution, lighting, and camera angle of the images. Therefore, having high-quality data is crucial for developing an accurate and reliable classification model.

Lastly, the storage of the model and dataset is vital for reproducibility and future use. Researchers should consider using a version control system such as Git to manage the code and data files to ensure they can be easily retrieved and shared with other researchers.

In this dissertation, a CNN-based AI model for detecting faulty aircraft body parts using image processing through ML techniques is proposed. Although the proposed model has been tested and evaluated through experimental results, there are still some issues that could be addressed and explored in future works and theses. While this thesis provides some details about the data augmentation techniques used, it does not investigate other types of data augmentation techniques that could be useful in other applications. However, the information provided can still be helpful for those interested in understanding the data augmentation techniques used in this study and their impact on the model's performance.

Despite the few issues associated with this work, the proposed CNN-based AI model offers a practical solution for detecting faulty aircraft body parts using image processing through ML techniques. The trained model's ability to classify unseen new images is a valuable feature, and the suggestion of using Support Vector Machines (SVM) or Self-Organizing Maps (SOM) as a hybrid approach to improve the system's validation accuracy indicates potential for future research in this area.

In summary, this thesis presents the results of experiments conducted to classify aircraft images using a CNN classifier. Two different datasets were used, one with actual images and the other with augmented images. The results show that the CNN classifier achieved a maximum of 98.28% training accuracy and 81.78% validation accuracy on the augmented dataset. Image data augmentation was used to increase the size of the dataset by flipping, rotating, and skewing the provided images, which resulted in faster and more effective learning. Image augmentation also stabilized the results and avoided overfitting, which was a problem with the small dataset. The use of SVM or expanding the dataset using augmentation techniques can address the issue of overfitting. Overall, the results demonstrate the effectiveness of CNN classifiers in classifying aircraft images, especially when used with augmented datasets. The proposed work presents a promising approach for detecting faulty aircraft body parts using ML techniques.

Chapter 6: Conclusion and Future Work

The primary objective of this thesis was to develop innovative techniques for the inspection and maintenance of aircraft structures. By utilizing image processing and machine learning techniques, we were able to train a CNN-based AI model capable of detecting faulty aircraft body parts. The primary challenge in implementing an ML-based model was obtaining a dataset of aircraft images, which we overcame by using data augmentation. We found that the performance of the model was better when tested on a large dataset than a small dataset. Multiple executions of the model were conducted with various epochs, and it was concluded that between 5 and 8 epochs with three convolutional layers, the best results were obtained in terms of training and validation accuracy, achieving a training accuracy of 98.28%, which outperformed other works existing in the literature.

To demonstrate the potential of our model, we designed a graphical user interface that can classify unseen new images as faulty or normal. We also propose that using SVM or SOM with CNN as a hybrid approach can further improve the validation accuracy of the system in the future. Our study shows that machine learning techniques have the potential to revolutionize aircraft maintenance and safety, and we believe that our research can be applied to other industries as well.

6.1 Future Work

The findings of this study offer a promising avenue for future research and development in the field of aviation maintenance testing. One potential area of exploration is the development of a four-wheeled robot that can assist with aircraft component assessments. This robot would be designed to move autonomously around the aircraft, scanning and capturing images of various parts for structural examination. The robot's framework would be mounted on rotating tires and guided by an infrared (IR) sensor coupled to a microcontroller, allowing for mobility. The framework would serve as the base for the camera, which would be capable of rotating 360 degrees along two axes and providing a comprehensive view of the aircraft.

Further enhancements to the image capturing capability of the robot could be made to allow it to capture images of deeper levels of the aircraft body. One approach to this could involve the addition of robotic hands capable of accessing embedded machines and parts for image analysis.

On the image processing side, there is a need to develop a stable dataset of all body parts, including both faulty and normal images. This dataset would be used to train deep learning models, such as Long Short-Term Memory (LSTM) networks, to enhance the efficiency and performance of the machine. The use of a stable dataset can help to stabilize the model and improve its accuracy. It may also be worthwhile to consider the use of multi-classification techniques to enable the model to identify both faulty images and the nature of the fault in the aircraft body part.

In addition to the development of the robot and image processing techniques, there is a need for the continued exploration and expansion of this research. For instance, the proposed technology could be applied to other areas of the aviation industry, such as the inspection of engines, landing gear, and other critical components. Overall, the automation of maintenance testing using innovative techniques such as those proposed in this study holds great promise for the future of the aviation industry.

References

Amit, K. (2019) *Machine Learning Techniques, Artificial Intelligence and Soft Computing*. Available at: https://doi.org/10.1201/9781315219738-13.

Apo (2016) ‘Cost of Delay Estimates FAA APO100’, *Federal Aviation Administration*, 4, pp. 31–33.

Apo (2020) *Aircraft Certification | Federal Aviation Administration*, *Federal Aviation Administration*. Available at: https://www.faa.gov/aircraft/air_cert (Accessed: 2 June 2023).

AXISCADES Engineering Technologies Limited (2020) ‘Machine Learning & Artificial Intelligence in Aerospace Industry’.

Azizi Oroumieh, M.A. *et al.* (2013) ‘Aircraft design cycle time reduction using artificial intelligence’, *Aerospace Science and Technology*, 26(1), pp. 244–258. Available at: https://doi.org/10.1016/j.ast.2012.05.003.

Ballard, D.H. & Brown, C.M. (1992) ‘Principles of animate vision’, *CVGIP: Image Understanding*, 56(1), pp. 3–21. Available at: https://doi.org/10.1016/1049-9660(92)90081-D.

Bauda, M.A., Bazot, C. & Larnier, S. (2017) ‘Real-time ground marking analysis for safe trajectories of autonomous mobile robots’, in *Proceedings of the 2017 IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics, ECMSM 2017*. Available at: https://doi.org/10.1109/ECMSM.2017.7945887.

Bouarfa, S. *et al.* (2020) ‘Towards automated aircraft maintenance inspection. A use case of detecting aircraft dents using mask r-cnn’, in *AIAA Scitech 2020 Forum*. American Institute of Aeronautics and Astronautics (AIAA). Available at: https://doi.org/10.2514/6.2020-0389.

CAA (2002) ‘An Introduction to Aircraft Maintenance Engineering Human Factors for JAR 66’, *Human Factors*, p. 127.

Cha, Y.J., Choi, W. & Büyüköztürk, O. (2017) ‘Deep Learning-Based Crack Damage Detection Using Convolutional Neural Networks’, *Computer-Aided Civil and Infrastructure Engineering*, 32(5), pp. 361–378. Available at: https://doi.org/10.1111/MICE.12263.

Chan, S., Oktavianti, I. & Puspita, V. (2019) ‘A Deep Learning CNN and AI-Tuned SVM for Electricity Consumption Forecasting: Multivariate Time Series Data’, *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference, IEMCON 2019*, pp. 488–494. Available at: https://doi.org/10.1109/IEMCON.2019.8936260.

Davies, E.R. (2004) ‘Machine Vision, Third Edition: Theory, Algorithms, Practicalities’, *Pattern Recognition Letters* [Preprint].

Dey, N. *et al.* (2020) *Applications of Firefly Algorithm and its Variants*.

DING, M. *et al.* (2022) ‘Visual inspection of aircraft skin: Automated pixel-level defect detection by instance segmentation’, *Chinese Journal of Aeronautics*, 35(10), pp. 254–264. Available at: https://doi.org/10.1016/J.CJA.2022.05.002.

Doerr, B. *et al.* (2013) *Theory of Evolutionary Algorithms (Dagstuhl Seminar 13271), Dagstuhl Reports*. Available at: https://doi.org/http://dx.doi.org/10.4230/DagRep.5.5.57.

Donadio, F., Frejaville, J., Larnier, S. & Vetault, S. (2016) ‘Human-robot collaboration to perform aircraft inspection in working environment’, in *Proceedings of 5th International conference on Machine Control and Guidance (MCG)*.

Ewald, V. *et al.* (2018) ‘Incorporating Inductive Bias into Deep Learning : A Perspective from Incorporating Inductive Bias into Deep Learning : A Perspective from Automated Visual Inspection in Aircraft Maintenance’, *10 th International Symposium on NDT in Aerospace*, (October), pp. 1–9. Available at: https://www.researchgate.net/publication/328334741 (Accessed: 2 June 2023).

Fatima, M. & Pasha, M. (2017) ‘Survey of Machine Learning Algorithms for Disease Diagnostic’, *Journal of Intelligent Learning Systems and Applications*, 09(01), pp. 1–16. Available at: https://doi.org/10.4236/jilsa.2017.91001.

Futterlieb, M. (2017) *Vision Based Navigation in a Dynamic Environment, Undefined*. Available at: http://thesesups.ups-tlse.fr/3722/1/2017TOU30191.pdf (Accessed: 8 June 2020).

Futterlieb, M., Cadenat, V. & Sentenac, T. (2014) ‘A navigational framework combining visual servoing and spiral obstacle avoidance techniques’, in *ICINCO 2014 - Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics*, pp. 57–64. Available at: https://doi.org/10.5220/0005027800570064.

Gonzalez, L.F. *et al.* (2016) ‘Unmanned aerial vehicles (UAVs) and artificial intelligence revolutionizing wildlife monitoring and conservation’, *Sensors (Switzerland)*, 16(1). Available at: https://doi.org/10.3390/s16010097.

Guo, Y., Yu, J. & wu, wei (2020) ‘Research on still image activity recognition based on deep learning’, in Z. Pan and X. Wang (eds) *Eleventh International Conference on Graphics and Image Processing (ICGIP 2019)*. SPIE, p. 53. Available at: https://doi.org/10.1117/12.2557257.

Gupta, V. (2017) *Image Classification using MLP in Keras | LearnOpenCV, Learn OpenCV*. Available at: https://learnopencv.com/image-classification-using-feedforward-neural-network-in-keras/ (Accessed: 26 September 2020).

Ian, M., Elena, V. & Michael, J. (2019) ‘Artificial Intelligence in the Aviation Manufacturing Process for Complex Assemblies and Components’, in *IOP Conference Series: Materials Science and Engineering*. Available at: https://doi.org/10.1088/1757-899X/689/1/012022.

Image Classification (2009) *Encyclopedia of Database Systems*. Available at: https://doi.org/10.1007/978-0-387-39940-9_2800.

‘Journal of intelligent material systems and structures’ (1990) *Materials & Design*, 11(2), p. 108. Available at: https://doi.org/10.1016/0261-3069(90)90079-y.

Jovancevic, I. *et al.* (2015) ‘Automated exterior inspection of an aircraft with a pan-tilt-zoom camera mounted on a mobile robot’, *Journal of Electronic Imaging*, 24(6), p. 061110. Available at: https://doi.org/10.1117/1.JEI.24.6.061110.

Jovančević, I. *et al.* (2016) ‘Matching CAD model and image features for robot navigation and inspection of an aircraft’, in *ICPRAM 2016 - Proceedings of the 5th International Conference on Pattern Recognition Applications and Methods*, pp. 359–366. Available at: https://doi.org/10.5220/0005756303590366.

Karaoğlu, U. *et al.* (2023) ‘Applications of machine learning in aircraft maintenance’, *library.acadlore.com*, 2(1), pp. 77–96. Available at: https://doi.org/10.56578/jemse020105.

Konstantinov, S. V. & Baryshnikov, A.A. (2017) ‘Comparative Analysis of Evolutionary Algorithms for the Problem of Parametric Optimization of PID Controllers’, *Procedia Computer Science*, 103(October 2016), pp. 100–107. Available at: https://doi.org/10.1016/j.procs.2017.01.021.

Kroll, A. (2008) ‘A survey on mobile robots for industrial Inspections’, *Intelligent Autonomous Systems 10, IAS 2008*, pp. 406–414. Available at: https://doi.org/10.3233/978-1-58603-887-8-406.

Latorella, K.A. & Prabhu, P. V. (2000) ‘A review of human error in aviation maintenance and inspection’, *International Journal of Industrial Ergonomics*, pp. 133–161. Available at: https://doi.org/10.1016/S0169-8141(99)00063-3.

Laxman, K.C. *et al.* (2023) ‘Automated crack detection and crack depth prediction for reinforced concrete structures using deep learning’, *Construction and Building Materials*, 370, p. 130709. Available at: https://doi.org/10.1016/J.CONBUILDMAT.2023.130709.

Ma, B. *et al.* (2020) ‘Autonomous deep learning: A genetic DCNN designer for image classification’, *Neurocomputing*, 379, pp. 152–161. Available at: https://doi.org/10.1016/j.neucom.2019.10.007.

Miranda, J. *et al.* (2019) ‘Machine learning approaches for defect classification on aircraft fuselage images aquired by an UAV’, in C. Cudel, S. Bazeille, and N. Verrier (eds) *Fourteenth International Conference on Quality Control by Artificial Vision*. SPIE, p. 10. Available at: https://doi.org/10.1117/12.2520567.

Parker, L.E. & Draper, J. V. (1998) ‘Robotics applications in maintenance and repair’, *Handbook of Industrial Robotics*, p. 1378. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.15.382&rep=rep1&type=pdf (Accessed: 8 June 2020).

Pettersen, K.A. & Aase, K. (2008) ‘Explaining safe work practices in aviation line maintenance’, *Safety Science*, 46(3), pp. 510–519. Available at: https://doi.org/10.1016/j.ssci.2007.06.020.

Richards, R.A. (2002) ‘Application of multiple artificial intelligence techniques for an aircraft carrier landing decision support tool’, in *IEEE International Conference on Fuzzy Systems*, pp. 7–11. Available at: https://doi.org/10.1109/fuzz.2002.1004950.

Schlegl, T. *et al.* (2019) ‘f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks’, *Medical Image Analysis*, 54, pp. 30–44. Available at: https://doi.org/10.1016/j.media.2019.01.010.

Setlur Nagesh, S.V. *et al.* (2020) ‘Image processing using Convolutional Neural Network (CNN) for Region of Interest (ROI) fluoroscopy’, in B.S. Gimi and A. Krol (eds) *Medical Imaging 2020: Biomedical Applications in Molecular, Structural, and Functional Imaging*. SPIE, p. 43. Available at: https://doi.org/10.1117/12.2549242.

Shafi, I. *et al.* (2023) ‘Deep Learning-Based Real Time Defect Detection for Optimization of Aircraft Manufacturing and Control Performance’, *Drones 2023, Vol. 7, Page 31*, 7(1), p. 31. Available at: https://doi.org/10.3390/DRONES7010031.

Siegel, M., Gunatilake, P. & Podnar, G. (1998) ‘Robotic assistants for aircraft inspectors’, *Industrial Robot*, 25(6), pp. 389–400. Available at: https://doi.org/10.1108/01439919810240234.

Sims, N. (2019) *Transforming the Future of Airports With Artificial Intelligence, Machine Learning and Generative Design*. Available at: https://www.burnsmcd.com/insightsnews/tech/transforming-future-of-airports-with-ai (Accessed: 8 June 2020).

Stoean, R. (2020) ‘Analysis on the potential of an EA–surrogate modelling tandem for deep learning parametrization: an example for cancer classification from medical images’, *Neural Computing and Applications*, 32(2), pp. 313–322. Available at: https://doi.org/10.1007/s00521-018-3709-5.

Thorpe, C. *et al.* (1988) ‘Vision and Navigation for the Carnegie-Mellon Navlab’, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3), pp. 362–373. Available at: https://doi.org/10.1109/34.3900.

Verma, A., Kaur, I. & Arora, N. (2016) ‘Comparative analysis of information extraction techniques for data mining’, *Indian Journal of Science and Technology*, 9(11). Available at: https://doi.org/10.17485/ijst/2016/v9i11/80464.

Waxman, A.M. *et al.* (1987) ‘A Visual Navigation System for Autonomous Land Vehicles’, *IEEE Journal on Robotics and Automation*, 3(2), pp. 124–141. Available at: https://doi.org/10.1109/JRA.1987.1087089.

Yasuda, Y.D. V. *et al.* (2021) ‘Automated Visual Inspection of Aircraft Exterior Using Deep Learning’, *Anais Estendidos da Conference on Graphics, Patterns and Images (SIBGRAPI)*, pp. 173–176. Available at: https://doi.org/10.5753/SIBGRAPI.EST.2021.20034.

Zhang, Y. *et al.* (2021) ‘A Deep Neural Network-Based Fault Detection Scheme for Aircraft IMU Sensors’, *International Journal of Aerospace Engineering*, 2021. Available at: https://doi.org/10.1155/2021/3936826.

Zhao, Q. *et al.* (2022) ‘Redundant object detection method for civil aircraft assembly based on machine vision and smart glasses’, *Measurement Science and Technology*, 33(10), p. 105011. Available at: https://doi.org/10.1088/1361-6501/AC7CBD.

Zou, F. (2020) ‘Review of Aero-engine Defect Detection Technology’, in *Proceedings of 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2020*. IEEE, pp. 1524–1527. Available at: https://doi.org/10.1109/ITNEC48623.2020.9084849.

Appendices

Appendix A

Preliminary code for model training

```
import pandas as pd
import numpy as np
import logging, os
import random
import tensorflow as tf
import cv2
from tensorflow import keras
from sklearn.metrics import classification_report
from keras.layers.core import Dense
from keras.layers import InputLayer, Flatten, Dropout
from tensorflow.keras import layers, Input
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.utils import to_categorical
from matplotlib import pyplot as plt
import matplotlib.image as mpimg
from keras.layers import Conv2D, MaxPool2D
from keras.layers import BatchNormalization
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.image import ImageDataGenerator

IMG_HEIGHT=200
IMG_WIDTH=200
EPOCHS = 5
def image_dataset(img_folder):

 img_data_array=[] # Array to store Images
 class_name=[] # Array to Store Corresponding class Names
 #Iterate through directories in img_folder
 for dir1 in os.listdir(img_folder):
 # iterate through files in each directory
 for file in os.listdir(os.path.join(img_folder, dir1)):

 image_path= os.path.join(img_folder, dir1,  file) # concatenating path of directory+Folder and File
 image= cv2.imread( image_path, cv2.COLOR_BGR2RGB) # Aquiring Image
 # Image Preprocessing
 image=cv2.resize(image, (IMG_HEIGHT, IMG_WIDTH),interpolation = cv2.INTER_AREA) # Resizing Image
 image=np.array(image) # Converting Image to Array
```

```
image = image.astype('float32') # Typecasting integer value to Float
image /= 255  # Normalizing Array values between 0 to 1 from 0 to 255
# Image Storing
img_data_array.append(image)
class_name.append(dir1)
print(file)
return img_data_array, class_name

## Image Aquisition of 3 random Normal and Faulty Images
## plt.figure(figsize=(10,10))
## faulty_folder=r'.\NN\faulty'
## normal_folder=r'.\NN\normal'
## for i in range(3):
## file = random.choice(os.listdir(faulty_folder))
## faulty_image_path= os.path.join(faulty_folder, file)
## img1=mpimg.imread(faulty_image_path)
## ax=plt.subplot(1,6,i+1)
## ax.title.set_text(file)
## plt.imshow(img1)
##
## file=file.replace("b.","a.")
## normal_image_path= os.path.join(normal_folder, file)
## img2=mpimg.imread(normal_image_path)
## ax=plt.subplot(1,6,i+4)
## ax.title.set_text(file)
## plt.imshow(img2)
## plt.show()


# Assigning Images with Class name and converting them to normalized form

# creating datset with images and class labels
img_data, class_name =image_dataset(r'.\NN') # D:\NN is the abse folder

# Assigning Unique class name to each folder
target_dict={k: v for v, k in enumerate(np.unique(class_name))}
#Converting Class Name to Numeric Value
target_val=  [target_dict[class_name[i]] for i in range(len(class_name))]

trainX, testX, trainY, testY = train_test_split(np.array(img_data, np.float32),
np.array(list(map(int,target_val)), np.float32), test_size=0.2, shuffle=True,
random_state=42)
trainY = to_categorical(trainY, 2)
testY = to_categorical(testY, 2)
##print("--Images Data--\n\n",img_data)
##print("--Images Corresponding Classes in Text Form--\n\n",class_name)
```

```
##print("--Classes--\n\n",target_dict)
##print("--Images Correponding Classes in Labelled Form--\n\n",target_val)
##print("Test Results\n\n")
##model=tf.keras.Sequential(
##       [
## tf.keras.layers.InputLayer(input_shape=(IMG_HEIGHT,IMG_WIDTH, 3)),
## tf.keras.layers.Conv2D(filters=32, kernel_size=3, strides=(2, 2), activation='relu'),
## tf.keras.layers.Conv2D(filters=64, kernel_size=3, strides=(2, 2), activation='relu'),
## tf.keras.layers.Flatten(),
## tf.keras.layers.Dense(6)
##       ])
## model.compile(optimizer='rmsprop', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
## history = model.fit(x=np.array(img_data, np.float32), y=np.array(list(map(int,target_val)), np.float32), epochs=5)
## history = model.fit(x=tf.cast(np.array(img_data), tf.float64), y=tf.cast(list(map(int,target_val)),tf.int32), epochs=5)

model = Sequential()
model.add(InputLayer(input_shape=(IMG_HEIGHT,IMG_WIDTH, 3)))

# 1st conv block
model.add(Conv2D(25, (5, 5), activation='relu', strides=(1, 1), padding='same'))
model.add(MaxPool2D(pool_size=(2, 2), padding='same'))
# 2nd conv block
model.add(Conv2D(50, (5, 5), activation='relu', strides=(2, 2), padding='same'))
model.add(MaxPool2D(pool_size=(2, 2), padding='same'))
model.add(BatchNormalization())
# 3rd conv block
model.add(Conv2D(70, (3, 3), activation='relu', strides=(2, 2), padding='same'))
model.add(MaxPool2D(pool_size=(2, 2), padding='valid'))
model.add(BatchNormalization())
# CNN block
model.add(Flatten())
model.add(Dense(units=100, activation='relu'))
model.add(Dense(units=10, activation='relu'))
model.add(Dropout(0.25))
# output layer
model.add(Dense(units=2, activation='softmax'))

# compile model
model.compile(loss='binary_crossentropy', optimizer="adam", metrics=['accuracy'])
# fit on data for 5 epochs
dataAugmentation=ImageDataGenerator()

```

```
H=model.fit_generator(dataAugmentation.flow(trainX, trainY, batch_size = 32),validation_data = (testX, testY), steps_per_epoch = len(trainX) // 32,epochs = 5)

# evaluate the network
print("[INFO] evaluating network...")
predictions = model.predict(x=testX.astype("float32"), batch_size=8)
print(classification_report(testY.argmax(axis=1),
predictions.argmax(axis=1), target_names=target_dict))
# plot the training loss and accuracy
N = np.arange(0, EPOCHS)
plt.style.use("ggplot")
plt.figure()
plt.plot(N, H.history["loss"], label="train_loss")
plt.plot(N, H.history["val_loss"], label="val_loss")
plt.plot(N, H.history["accuracy"], label="train_acc")
plt.plot(N, H.history["val_accuracy"], label="val_acc")
plt.title("Training Loss and Accuracy on Dataset")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend(loc="lower left")
plt.show()
```

Appendix B

Preprocessing code

```
import pandas as pd
import numpy as np
from numpy import save,load
import logging, os
import random
import tensorflow as tf
import cv2
from tensorflow import keras
from sklearn.metrics import classification_report
from keras.layers.core import Dense
from keras.layers import InputLayer, Flatten, Dropout
from tensorflow.keras import layers, Input
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.utils import to_categorical
from  matplotlib import pyplot as plt
import matplotlib.image as mpimg
from keras.layers import Conv2D, MaxPool2D
from keras.layers import BatchNormalization
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.image import ImageDataGenerator

IMG_HEIGHT=200
IMG_WIDTH=200
EPOCHS = 5
def image_dataset(img_folder):

img_data_array=[] # Array to store Images
class_name=[] # Array to Store Corresponding class Names
# Iterate through directories in img_folder
for dir1 in os.listdir(img_folder):
#iterate through files in each directory
for file in os.listdir(os.path.join(img_folder, dir1)):

image_path= os.path.join(img_folder, dir1,  file) # concatenating path of
directory+Folder and File
image= cv2.imread( image_path, cv2.COLOR_BGR2RGB) # Aquiring Image
 # Image Preprocessing
 image=cv2.resize(image, (IMG_HEIGHT, IMG_WIDTH), interpolation =
cv2.INTER_AREA) # Resizing Image
 image=np.array(image) # Converting Image to Array
 image = image.astype('float32') # Typecasting integer value to Float
image /= 255  # Normalizing Array values between 0 to 1 from 0 to 255
# Image Storing
img_data_array.append(image)
class_name.append(dir1)
Print (file)
return img_data_array, class_name



#creating datset with images and class labels
img_data, class_name =image_dataset(r'.\NN') # D:\NN is the abse folder

# Assigning Unique class name to each folder
target_dict={k: v for v, k in enumerate(np.unique(class_name))}
#Converting Class Name to Numeric Value
target_val=  [target_dict[class_name[i]] for i in range(len(class_name))]

trainX, testX, trainY, testY = train_test_split(np.array(img_data, np.float32),
np.array(list(map(int,target_val)), np.float32), test_size=0.2, shuffle=True,
random_state=42)

save('trainX.npy', trainX)
save('trainY.npy', trainY)
save('testX.npy', testX)
save('testY.npy', testY)
```

Appendix C

Model Training code

```
import pandas as pd
import numpy as np
from numpy import load
import logging, os
import random
import tensorflow as tf
import cv2
from tensorflow import keras
from sklearn.metrics import classification_report
from keras.layers.core import Dense
from keras.layers import InputLayer, Flatten, Dropout
from tensorflow.keras import layers, Input
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.utils import to_categorical
from  matplotlib import pyplot as plt
import matplotlib.image as mpimg
from keras.layers import Conv2D, MaxPool2D
from keras.layers import BatchNormalization
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.image import ImageDataGenerator

from keras.models import model_from_json

IMG_HEIGHT=200
IMG_WIDTH=200
EPOCHS = 5


trainX=load('trainX.npy')
trainY=load('trainY.npy')
testX=load('testX.npy')
testY=load('testY.npy')

trainY = to_categorical(trainY, 2)
testY = to_categorical(testY, 2)


model = Sequential()
model.add(InputLayer(input_shape=(IMG_HEIGHT,IMG_WIDTH, 3)))

# 1st conv block
```

```
model.add(Conv2D(25, (5, 5), activation='relu', strides=(1, 1), padding='same'))
model.add(MaxPool2D(pool_size=(2, 2), padding='same'))
# 2nd conv block
model.add(Conv2D(50, (5, 5), activation='relu', strides=(2, 2), padding='same'))
model.add(MaxPool2D(pool_size=(2, 2), padding='same'))
model.add(BatchNormalization())
# 3rd conv block
model.add(Conv2D(70, (3, 3), activation='relu', strides=(2, 2), padding='same'))
model.add(MaxPool2D(pool_size=(2, 2), padding='valid'))
model.add(BatchNormalization())
# CNN block
model.add(Flatten())
model.add(Dense(units=100, activation='relu'))
model.add(Dense(units=10, activation='relu'))
model.add(Dropout(0.25))
# output layer
model.add(Dense(units=2, activation='softmax'))

# compile model
model.compile(loss='binary_crossentropy', optimizer="adam", metrics=['accuracy'])
# fit on data for 5 epochs
dataAugmentation=ImageDataGenerator()

H=model.fit(dataAugmentation.flow(trainX, trainY, batch_size = 32),validation_data = (testX, testY), steps_per_epoch = len(trainX) // 32,epochs = 5)

# plot the training loss and accuracy
N = np.arange(0, EPOCHS)
plt.style.use("ggplot")
plt.figure()
plt.plot(N, H.history["loss"], label="train_loss")
plt.plot(N, H.history["val_loss"], label="val_loss")
plt.plot(N, H.history["accuracy"], label="train_acc")
plt.plot(N, H.history["val_accuracy"], label="val_acc")
plt.title("Training Loss and Accuracy on Dataset")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend(loc="lower left")
plt.show()
model_json = model.to_json()
with open("model.json", "w") as json_file:
    json_file.write(model_json)
# serialize weights to HDF5
model.save_weights("model.h5")
```

Appendix D

Interface code

```
# img_viewer.py
from tkinter import *
from PIL import ImageTk,Image
import PySimpleGUI as sg
import os.path
import pandas as pd
import numpy as np
import logging, os
import tensorflow as tf
import cv2
from tensorflow import keras


from keras.models import model_from_json


# First the window layout in 2 columns
json_file = open('model.json', 'r')
loaded_model_json = json_file.read()
json_file.close()
model = model_from_json(loaded_model_json)
# load weights into new model
model.load_weights("model.h5")
IMG_HEIGHT=200
IMG_WIDTH=200

file_list_column = [
    [
        sg.Text("Image Folder"),
        sg.In(size=(25, 1), enable_events=True, key="-FOLDER-"),
        sg.FolderBrowse(),
    ],
    [
        sg.Listbox(
            values=[], enable_events=True, size=(40, 20), key="-FILE LIST-"
        )
    ],
]
# For now will only show the name of the file that was chosen
image_viewer_column = [
    [sg.Text("Choose an image from list on left:")],
    [sg.Text(size=(40, 1), key="-TOUT-")],
    [sg.Text(size=(40, 1), key="-IMAGE-", text_color="LightGreen"),],
]
# ----- Full layout -----
layout = [
    [
 sg.Column(file_list_column),
 sg.VSeperator(),
 sg.Column(image_viewer_column),
    ]
]

window = sg.Window("Aircraft Parts Classifier", layout)
while True:
 event, values = window.read()
 if event == "Exit" or event == sg.WIN_CLOSED:
 break
 # Folder name was filled in, make a list of files in the folder
 if event == "-FOLDER-":
 folder = values["-FOLDER-"]
 try:
 # Get list of files in folder
 file_list = os.listdir(folder)
 except:
 file_list = []

 fnames = [
 f
  for f in file_list
 if os.path.isfile(os.path.join(folder, f))
 and f.lower().endswith((".png", ".gif",".jpg"))
        ]
 window["-FILE LIST-"].update(fnames)
 elif event == "-FILE LIST-":  # A file was chosen from the listbox
 try:
 filename = os.path.join(
 values["-FOLDER-"], values["-FILE LIST-"][0]
            )
 window["-TOUT-"].update(filename)
 img_data_array=[]
 image= cv2.imread( filename, cv2.COLOR_BGR2RGB) # Aquiring Image
 # Image Preprocessing
 image=cv2.resize(image, (IMG_HEIGHT, IMG_WIDTH),interpolation = cv2.INTER_AREA) # Resizing Image
 image=np.array(image) # Converting Image to Array
 image = image.astype('float32') # Typecasting integer value to Float
 image /= 255  # Normalizing Array values between 0 to 1 from 0 to 255
 # Image Storing
img_data_array.append(image)
img=np.array(img_data_array, np.float32)
predictions = model.predict(x=img.astype("float32"))
#print(predictions.argmax(axis=1))
if(predictions.argmax(axis=1)==0):
window["-IMAGE-"].update(value="Faulty", text_color="Red")
Elif (predictions.argmax(axis=1)==1):
window["-IMAGE-"].update(value="Normal", text_color="LightGreen")

except:
pass
window.close()
```

UAE UNIVERSITY MASTER THESIS NO. 2023: 30

This work aims to provide a machine learning based approach for the fault detection in aircraft body part. The main objective of this thesis is to evaluate the performance of CNN model for the classification of outer body parts of aircraft using image processing. Also, this thesis provides a self-collected dataset of aircraft outer images that can be pivot point for the researchers in future.

Noura Almansoori received her Master of Science in Electrical Engineering from the Department of Electrical and Communication Engineering, College of Engineering and her Bachelor of Science in Electrical Engineering from the College of Engineering, United Arab Emirates University, UAE.