Target Identification and Events Analysis of UAV Based Image using Deep Learning

Lencho Ajema Negese

A Thesis Submitted to the Department of Computer Science and Engineering
School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of Master′s

in Computer Science and Engineering

Office of Graduate Studies
Adama Science and Technology University

September 2022
Adama, Ethiopia

Target Identification and Events Analysis of UAV Based Image using Deep Learning

Lencho Ajema Negese

Advisor: Dr. Mesfin Abebe (Phd)

A Thesis Submitted To Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of Master′s

in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

September 2022

Adama, Ethiopia

Declaration

I declare that this Master Thesis entitled "**Target Identification and Events Analysis of UAV Based on Deep Learning**" is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation.

Lencho Ajema	_____	_____
Name of Student	Signature	Date

Recommendation of Advisors

I/we, the advisor(s) of this thesis, hereby certify that I/we have read the revised version of the thesis entitled "**Target Identification and Events Analysis of UAV Based on Deep Learning**" prepared under my guidance by Lencho Ajema submitted in partial fulfillment of the requirements for the degree of Masters of Science in Computer Science and Engineering. Therefore, I/we recommend the submission of the revised version of the thesis to the department following the applicable procedures.

Dr. Mesfin Abebe	_____	_____
Major Advisor	Signature	Date
_____	_____	_____
Co-Advisor	Signature	Date

Approval Page of M.Sc. Thesis

I/we, the advisors of the thesis entitled "**Target Identification and Events Analysis of UAV Based on Deep Learning**" and developed by Lencho Ajema, hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

Dr. Mesfin Abebe	_____	_____
Major Advisor	Signature	Date
_____	_____	_____
Co-Advisor	Signature	Date

Approval of Board of Reviewers

We, the undersigned, members of the Board of Examiners of the thesis by **Lencho Ajema Negese** have read and evaluated the thesis entitled "**Target Identification and Events Analysis of UAV Based on Deep Learning**" and examined the candidate during the open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in Computer Science and Engineering.

_____	_____	_____
Chairperson	Signature	Date
_____	_____	_____
Internal Examiner	Signature	Date
Eyob N.		November 30, 2022
External Examiner	Signature	Date

Finally, approval and acceptance of the thesis is contingent upon submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

_____	_____	_____
Department Head	Signature	Date
_____	_____	_____
School Dean	Signature	Date
_____	_____	_____
Office of Postgraduate Studies, Dean	Signature	Date

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

3D	Three Dimensions
AI	Artificial Intelligence
AIDER	Aerial Image Dataset For Emergency Response Applications
APIs	Application Programming Interface
AUC	Area Under The Roc Curve
CAP	Credit Assignment Path
CNN	Convolutional Neural Networks
DCNNS	Deep Convolutional Neural Networks
DL	Deep Learning
DNN	Deep Neural Networks
DRL	Deep Reinforcement Learning
DRL OR RL	Deep Reinforcement Learning
DTMS	Digital Terrain Models
ERA	Event Recognition In Aerial
FN	False Negative
FP	False Positive
GANS	Generative Adversarial Networks
GCS	Ground Control Station
GPS	Global Positioning System
GPUS	Graphics Processing Units
GRUS	Gated Recurrent Units
ICAO	International Civil Aviation Organization ()
IMU	Inertial Measurement Unit
ISR	Intelligence, Surveillance, And Reconnaissance
IT	Information Technology
LIDAR	Light Detection And Ranging
LSTM	Long Short-Term Memory
MEMS	Micro-Electro-Mechanical Systems

NASA	National Aeronautics And Space Administration
NDRE	Normalized Difference Red Edge
NDVI	Normalized Difference Vegetation Index
PPK	Post-Processing Kinematic
RF	Radio Frequencies
RGB	Red, Green, And Blue
RNNS	Recurrent Neural Networks
RPA	Remotely Piloted Aircraft
RTK	Real-Time Kinematic
SFM	Structure From Motion
TN	True Negative
TP	True Positive
UAS	Unmanned Aerial System
UAV	Unmanned Aerial Vehicle

ABSTRACT

Deep learning is used to learn, track, and discover targets from the data obtained. Deep learning is also used in target identification systems to improve the ability of these systems to identify the position of the targets. Unmanned aerial vehicles (UAVs) – also known as drones – with integrated deep learning can patrol border areas, identify potential threats, and transmit information by analyzing events based on deep learning about these threats to response teams or trigger alarms. Generally, the conditions listed above were what motivated me to deep dive into this research study on event analysis and target identification of UAVs based on deep learning. To contribute valuable research by integrating deep learning and machines like UAVs to solve the problem that happened with the limitation of Intelligence, Surveillance, and Reconnaissance (ISR) operations by human beings, especially in remote and complex areas. Activity that happens in every moment is tedious and complex for a human to analyze, but in this paper, we presented target identification and event analysis of UAVs based on deep learning methods to analyze events continuously from an aerial view using deep learning. The main purpose of this study was to review related works, train publically available aerial datasets and self-collected datasets from UAVs using pre-trained models, and present the result of the best models that performs event analysis, target identification, and classification in real-time, with computation power requirements that can be met with a common laptop computer and android phone contain above android version 11.

Keywords*: Drone, UAV, Deep Learning, Bird View, Target Identification, Event Analysis*

CHAPTER ONE

1 INTRODUCTION

1.1 Background of the Study

Unmanned aerial vehicles (drones) are gaining popularity as a viable technology for infrastructure and environmental monitoring, with a wide range of uses. Computer vision techniques are frequently used in these applications to analyze the data obtained from an onboard camera. These applications include traffic monitoring and vehicle detection for emergency response. The development and availability of big data (structured data versus unstructured data), advancements in computer processing speed and new chip architectures, cloud computing and APIs, and the emergence of data science are influences on artificial intelligence(AI), which includes machine learning and deep learning. In addition, AI techniques are being developed to enhance the accuracy of target identification in complex combat environments. These techniques allow defense forces to gain an in-depth understanding of potential operational areas by analyzing images, videos, and other forms of unstructured information. Additionally, deep learning in target identification systems improves the ability of these systems to identify the position of the targets. Deep learning is also used to learn, track, and discover targets from the data obtained.

Threat monitoring and situational awareness rely heavily on intelligence, surveillance, and reconnaissance (ISR) operations. ISR operations are used to acquire and process information to support a range of military activities. ISR missions can be completed by unmanned systems that can be remotely controlled or dispatched along a predetermined path. Deep learning capabilities for these systems let defense personnel monitor threats, which improves their situational awareness. Unmanned aerial vehicles (UAVs) – also known as drones – with integrated deep learning can patrol border areas, identify potential threats, and transmit information by analyzing events based on deep learning about these threats to response teams or trigger alarms. Thus, using UAVs can improve the protection of military installations and improve the efficiency and safety of military personnel during combat or in off-road situations. The detection and classification of moving objects in the observed area is a crucial use of aerial surveillance for

security purposes. Recent investigations have made either detection, classification, or both within certain limits. In their study (Liu and Mattyus 2015), they accomplished object detection and tracking for vehicles and humans in real-time from low-altitude aerial surveillance. In addition to the detection and tracking, Iwashita et al. also performed classification with 80% accuracy for vehicles and humans, but the operation was not in real-time. Oreifej et al. conducted another similar study where they performed detection and classification for humans but not tracking. In classification, accuracy was 85% and the operation was not in real-time. Although detection, tracking, and classification were achieved in real-time, the camera was stationary and much closer to the moving objects, which made the entire process easier due to higher resolution, larger target objects, and fewer noise sources.

1.2 Statement of the Problem

A challenge for companies with large sites is, that they cannot be everywhere or see everything that's going on. This prevents a smooth operation, security, and safety. When managing large events one of the biggest challenges is monitoring large crowds and security guards on the ground cannot see incidents until it is too late. The advancement in machine learning algorithms, sensors, and IT technologies has opened the doors for UAV applications in many sectors. The main sectors, however, are computer wireless networks, smart cities, military, agriculture, and mining. UAVs are widely used in a lot of daily activities for civilian and military applications the research community is also digging continuously in the area. The activity that happened in every moment is tedious and complex for humans to analyze, but in this paper, we proposed an event analysis algorithm for UAVs to analyze events continuously from an aerial view using deep learning. Based on the analyzed event identification of a target will be made. This research study addressed such problem raised above by answering the research questions listed below.

1.3 Research Questions

Event analysis and target identification of UAVs based on deep learning addressed the questions listed below to give an answer and conclusion about the effectiveness and accuracy of the proposed method by experimenting and implementing the proposed model and different pre-trained deep learning models on the prepared dataset which collected by our self and pre-collected from different researcher by making some adjustment to fit with our dataset.

RQ1: Can we get any gap from related work studied for image recognition from aerial view?

RQ2: Is it possible to identify events from an aerial view image (by UAV) using deep learning models?

RQ3: How can we extract target from events seen from aerial?

RQ4: What is the best result of pre-trained models by implementing transfer learning?

1.4 Objective of the Study

1.4.1 General Objective

The general objective of this study is to identify target and analyze it from aerial image of UAV using deep learning approaches.

1.4.2 Specific Objective

The following are the specific objectives to achieve the objective of the study:

- To review papers and models published previously
- To develop and implement a real-time event analysis system.
- To develop and implement a real-time target identification system
- To implement transfer learning and fine tuning.
- To train and test aerial datasets with pre-trained models.

1.5 Significance of the Study

UAVs have some challenges for object recognition, object identification, and event analysis in uncertain environments. This research was conducted to identify target and event analysis that have a powerful objective to meet for our country, Ethiopia. It will move one step forward in the advancement and use of technology in the deep learning field of study, geospatial and remote sensing.

The advancement of technology in the current world is shifting to the capability and complexity of AI. Especially in the military, using the combinational power of UAVs and deep learning for intelligence, surveillance, and reconnaissance (ISR) is very essential and important to analyze and identify threats or situations that happened from anywhere under the UAV in real-time. This research will play a great role in the advancement of technology in the military by identifying the target and analyzing events that occurred in the area.

1.6 Expected Outcome of the Study

This study will benefit different civilian and military organizations in different ways. Analysis of events will help to predict the next and to control what is happening, like traffic flow and congestion, fire detection analysis, and floods. Especially in the military to fulfill the need for surveillance, intelligence, and reconnaissance, and in target identification, it will play a great role.

Unmanned systems used to carry out ISR missions can either be remotely operated or sent on a pre-defined route. Equipping these systems with AI assists defense personnel in threat monitoring, thereby enhancing their situational awareness. ISR operations are used to acquire and process information to support a range of military activities.

Unmanned aerial vehicles (UAVs) with integrated deep learning can patrol border areas, identify potential threats, and transmit information about these threats to response teams. Using UAVs can thus strengthen the security of military bases and increase the safety and efficacy of military personnel in battle or at remote locations.

1.7 Scope and Limitation of the Study

1.7.1 Scope of the Study

Target identification and event analysis of UAVs based on deep learning will be made depending on the dataset available by adding five events for military purposes because of its computational cost. This study was conducted by collecting data set from Ethiopian Air force UAV department and from different youtube videos by cutting into 5 seconds which later extracted to single frames, we have also used available dataset like ERA (Mou et al. 2020) and AIDER (Kyrkou and Theocharides 2019a) by adding some new data set additionally for our context. Since an unlimited number of events are available in daily activity. It is hard to analyze all because of its cost and complexity.

1.7.2 Limitation of the Study

A different situation that may be considered a limitation will be the adoption and usage of drones, which include serious security threats and weather conditions.

Weak Policies: The adoption of authorization and authentication policies should be strict, preventing unauthorized entities from accessing a drone system and preventing possible insider threats.

Therefore, the applications that control drones must be tested to prevent any security flaw that could be exploited by attackers. Weak level of protection: drone protection systems should include three main phases: detection, correction, and protection, which is not currently the case.

Cloudy and rainy weather may disturb the capability of identification and analysis of events.

1.8 Research Organization

The first chapter serves as an introduction, outlining the many facets of the topic. The remaining five chapters of the thesis are divided into the previous chapter's literature evaluation of pertinent publications and its identification of research gaps. The third chapter outlines the study's methodology. The study's methodology, population, and sample are covered in the first half of the analysis, while the instruments and methods used in the research are covered in the second section. Additionally, the study's methodology is presented. In this chapter, the entire study's design is covered in depth. The fourth chapter of this study provides a brief explanation of the suggested approaches along with a flow diagram. Analyses and results are presented in this chapter by inferring conclusions from the research. The analysis and findings from the data are described and shown in tabular, graphical, or visual form. Finally, we include the findings, recommendations for further research, and conclusion.

CHAPTER TWO:

2 LITERATURE REVIEW AND RELATED WORKS

2.1 Introduction

Various research has been conducted and published in the field of deep learning to classify and analyze aerial events by using different models which are pre-trained on various images and videos. In this chapter, we reviewed the recent literature and related works to identify the gap for further study with the possible solutions in current world scenarios, specifically in object recognition, target identification, and event analysis based on bird view by processing through Deep Learning models. Unmanned aerial vehicles (drones) are gaining popularity as a viable technology for infrastructure and environmental monitoring, with a wide range of uses. Computer vision techniques are frequently used in these applications to analyze the data obtained from an onboard camera. These applications include traffic monitoring and vehicle detection for emergency response.

2.2 Target Identification

(Li et al. 2016) and (Razakarivony and Jurie 2016) presented in their paper about target detection, especially for cars from the aerial view. UAVs are still becoming more and more popular in both military and civilian uses, in addition to personal usage, despite the most current flight control laws. The growing demand for aerial view technologies is being driven by this interest. Especially in a situation when there is a high risk of an emergency, such technologies are essential to the functioning of UAVs. Camera-based solutions are the standard option for target identification and event analysis systems. Due to the cost and weight restrictions placed on UAV payloads, algorithms for target identification and event analysis from a video are required for this, and they may be effectively executed on board. The identification and tracking of objects from a fixed camera have received a lot of investigation, but few studies have focused on target identification and event analysis from unmanned aerial vehicles (UAVs) based on deep learning. In our research, we present a novel method for target identification and event analysis by the camera mounted on a UAV and process the collected data through pre-trained deep learning models.

2.3 Event Recognition

Aerial Image Database for Emergency Response (AIDER) which presented by (Kyrkou and Theocharides 2019a) for recognition of disaster event recognition from aerial view. They used have used 5 class recognition for around 2545 images collected from YouTube videos. Since they may operate in distant and challenging-to-access places, unmanned aerial vehicles (UAVs) with video sensors can provide improved situational awareness for various emergency response and disaster management applications. Additionally, using an embedded platform and deep learning, UAVs can autonomously monitor a disaster-affected area, analyze the image in real time, and send out alerts in the event of various calamities like building collapses, floods, or fire to more quickly mitigate their effects on the environment and the human population. To that aim, the automatic aerial scene classification of catastrophic occurrences from a UAV's perspective is studied by (Kyrkou and Theocharides 2019b). In particular, an application called Aerial Image Database for Emergency Response is introduced, and a comparison of current methods is done.
The multi modal datasets of AU-AIR for object detection (Bozcan and Kayacan 2020) is also another paper presented for detection of image from aerial image. It combines robotics and vision for UAVs with multimodal data from various on-board sensors and advances the creation of robotic and computer vision algorithms for autonomous aerial surveillance. AU-AIR has several features like Object detection in aerial images, >2 hours of raw videos, 32,823 labeled frames, 132,034 object instances, 8 object categories related to traffic surveillance, Frames are also labeled with time, GPS, IMU, altitude, and linear velocities of the UAV

2.4 Unmanned Aerial Vehicle (UAV)

Unmanned Aerial Vehicles (UAVs) are planes that don't have a pilot on board. The Unmanned Aerial System (UAS) is another title for these devices, whereas the International Civil Aviation Organization (ICAO) calls them Remotely Piloted Aircraft (RPA). They are known as drones in the public eye. UAVs can be controlled remotely by a pilot at a ground control station or autonomously by following a pre-programmed flight plan. The history of these systems may be traced back to hobby and professional radio-controlled aircraft, as well as military applications, such as flying over hostile or otherwise dangerous areas. However, in recent years, UAVs have been used in a variety of civil applications, including urban planning,

environmental monitoring, agriculture, governance, utilities, and commercial uses. It can also fly at tremendous altitudes while carrying a significant amount of weight. Drones are commonly used for a variety of tasks, including aerial photography, surveillance, spraying water, pesticides, and surveying fields for leaks in water connections, among others.

These unmanned aerial vehicles may now be flown and controlled using a smartphone. To fly them, though, you must have a license. The government puts severe restrictions on the heights at which they may be flown, the locations where they can be flown, and the reasons for which they can be utilized. UAVs are assisting in the fast replacement of traditional aerial photography (photogrammetry) in the field of remote sensing by producing high spatial resolution aerial photographs (orthophotos) of the earth's surface. UAVs are less expensive to acquire and operate than manned aircraft, which are utilized in traditional photography. They also create photographs with comparable or superior spatial resolution. These and other characteristics make UAVs ideal monitoring devices, allowing for long-term observation of a variety of phenomena.

UAVs can thus play an essential role in monitoring smallholder (rainfed) agricultural systems, where persistent cloud cover makes image acquisition difficult for optical remote sensing systems. Images of important crop growth phases can be captured and analyzed using UAVs to monitor agricultural dynamics.

In current world UAVs are highly changing the game of war almost in all the battle as of report indicate in countries with known operational armed drones are around 28 as reported by Wikipedia(“Unmanned combat aerial vehicle - Wikipedia,” n.d.). For example:

- ✓ Azerbaijan – Bayraktar TB2
- ✓ China – GJ-11, CAIG Wing Loong I, CAIG Wing Loong II, CH-3, CH-4, CH-5
- ✓ Egypt – CAIG Wing Loong, CH-4 Rainbow
- ✓ Ethiopia – Bayraktar TB2(), CAIG Wing Loong
- ✓ France – MQ-9 Reaper
- ✓ India - IAI Eitan, and many more countries in the world are using the power of Drones in different military operations internally and also for external war between countries.

As we have seen from the recent and current history of the world the war situation in different countries, UAVs are playing a great role in changing the complexity of the war. The need of using UAVs among many countries is increasing dramatically from time to time (“Deadly secret:

Electronic warfare shapes Russia-Ukraine war," 2022; "Ukraine Embraces the 'Messy Middle' to Win the Drone War," 2022; (Ozberk 2021)).
In our few next sections, we will discuss the history of UAVs, types of UAVs, and major components of UAVs like cameras used for the collection of data.

The earliest reported usage of a drone UAV was in 1849, when Austrians used unmanned air balloons armed with explosives to attack the city of Venice. Reginald Denny of the United States created the first radio-controlled aircraft, the Radioplane OQ-2, in 1941, during World War II. In 1986, the United States and Israel formed RQ2 Pioneer, a joint company to develop less expensive and more effective drones. Amazon stated in 2014 that it would deploy drone technology in product delivery, dubbed "Prime Air," to make it easier for items to be delivered by air.

2.4.1 Types of UAV

According to the most recent developments in UAV technology, the following are the most common UAV applications: (i) photogrammetry and remote sensing (I Colomina 2014), to obtain information from an image and generate 3D data; (ii) 3D modeling (Wefelscheid et al. 2011), to reconstruct the 3D shape of buildings or areas; (iii) surveillance (Semsch et al. 2009), for both civil and military areas; (iv) inspection (Zhang et al. 2012), particularly when human intervention is potentially dangerous; (v) disaster response UAVs may be utilized simply and swiftly in this situation; (vi) forest and agricultural, as well as geological studies (with thermal and multispectral sensors (Saari et al. 2011) are growing domains in which UAVs can give high-resolution and repeating data essential for monitoring. NASA (2015) suggested a new categorization system that classifies UAV missions into four categories:
UAV land management missions are carried out to acquire geospatial data on specific locations for monitoring or management. They are particularly suitable in dangerous areas (for example, after disasters and emergencies). Their civil applications may include agricultural forestry, fighting fires, geological research, surveying, and mapping. There is no one-size-fits-all answer to all practical challenges, and in each situation, a thorough assessment of the mission's features is required to determine the optimum solution and sensors to be fitted. Fixed-wing and multirotor vehicles have traditionally been the most frequently utilized systems in environmental operations because of their flight endurance and operability.

It's worth noting that when considering a UAV, we must keep in mind that the entire system consists of not only the aerial vehicle but also the ground control station (GCS). In this potentially dangerous position, the pilot must assume direct control of the UAV and land it safely.

UAVs can be classified based on their size, range, and endurance, employing a tier system similar to that used by the military. The following sub-classes can be used to categorize objects based on their size:

- ✓ Very small unmanned aerial vehicles
- ✓ Micro or Nano UAVs
- ✓ Small unmanned aerial vehicles
- ✓ Mini unmanned aerial vehicles
- ✓ Medium unmanned aerial vehicles
- ✓ Large unmanned aerial vehicles

UAVs can also be categorized based on their range and endurance in the air using the following sub-classes created by the US military:

- ✓ Very low-cost close-range unmanned aerial vehicles
- ✓ Close-range unmanned aerial vehicles
- ✓ Short-range unmanned aerial vehicles
- ✓ Mid-range unmanned aerial vehicles
- ✓ Endurance unmanned aerial vehicles

The very small UAV class includes UAVs ranging in size from the size of a giant bug to 30-50 cm in length. The Israeli-US Hunter has a wingspan of 10.2 m and a length of 6.9 m. The Australian Cyber Technology CyberQuad Mini (42x42 cm square) and its most recent iteration, the cyberQuad Maxi, are examples of this class. Classification Range and Endurance are two factors to consider. UAVs with a close range have a range of 50 kilometers and an endurance time of 1 to 6 hours. The US General Atomics Predators A and B, as well as the US Northrop Grumman Global Hawk (Figure 2-1), are examples of this type of UAV.

Figure 2-1 Global Hawk Reconnaissance Drones

Figure 2-2 Tactical Drones

Figure 2-3 Atomics MQ 1B Predator

2.4.2 Components of UAV

Aerial vehicles are complicated systems comprised of hardware and software components. The advancement of electronics has enabled the creation of navigation and control systems that are becoming more widely available on the market.

A UAV's main components are divided into three categories: (i) the aerial platform, which includes the airframe, navigation system, power system, and payload; (ii) the ground control station (GCS), which allows human control from a remote emplacement; and (iii) the communication system, which facilitates communication between the other two components.

The aerial platform is made up of many components whose goal is to allow flying and transport of certain sensors in the air for data collection: The UAV's major structure is its airframe. Its

structure must take into account the weight of the systems on board, particularly the power and communication, and control systems. Furthermore, the airframe must be sufficiently engineered to handle the forces that may occur during flight without deformation or vibration. Fixed wings are generally constructed of polystyrene or plastic, whereas popular multirotor airframes are built of aluminum or carbon fiber (to be lightweight and robust), and the number of arms is a function of the projected payload and the number of engines.

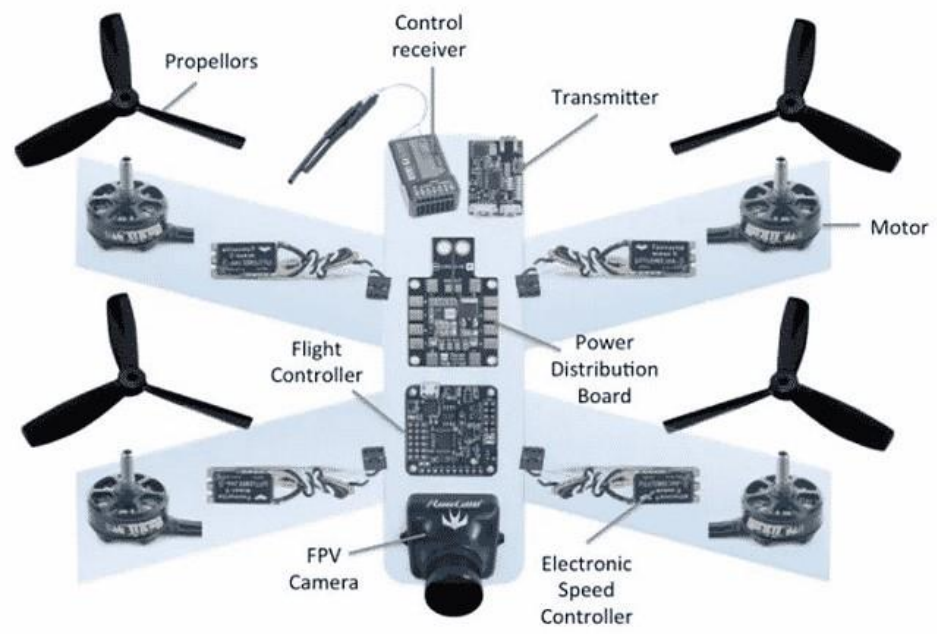


Figure 2-4 components of small drones

Figure 2-5 Bayraktar TB2 UAV(left) and its ground control station(right)

The autopilot, which permits autonomous or semi-autonomous flying through hardware and software components, is the key component of avionics. Flight control, GPS/GNSS, and an inertial system make up the navigation system.
The navigation system's "heart" is flight control. This board is in charge of flight planning and may check the theoretical trajectory against the actual one in real-time. Onboard is a dual-constellation (GPS and GLONASS) system. The payload is made up of sensors or equipment carried by the UAV that are used to collect data. Another important component of current UAVs is the GPS/GNSS board.

A gimbal allows the payload to rotate along one or more axes and is frequently fitted with servos that may modify or stabilize the sensor's orientation. A GCS is often a computer or a tablet capable of planning and controlling the flight. If the UAV is not fully autonomous, the pilot should be outfitted with a remote control that may be utilized in an emergency or to execute the takeoff and landing. Most commercial UAVs include a specialized mission planner, although open-source software can also be used.
The final UAV component is the communication system, which serves as a radio link between the vehicle and the ground. Radio communication is required to command and operate a UAV, as well as to ensure continuous connectivity for emergency operations. Radio frequencies (RF) in the range of 30 MHz to 3 GHz are the most common RF bandwidths used by tiny multirotor.

I. Camera

UAVs are most commonly used for picture and video acquisition. Many commercial navigation systems can now handle picture capture while also storing the UAV's location (GNSS) and attitude (IMU).

II. Thermal detectors

Thermal detectors are frequently used in conjunction with an RGB camera with the same field of view to improve the interpretation of captured information by combining the visible and thermal bands. The applications of these sensors on UAVs are diverse ranging from search and rescue missions (e.g., (Rudol and Doherty 2008)) to precision agriculture to earth science applications and volcanology.

III. Multispectral cameras

Multispectral cameras may record data in the non-visible part of the light spectrum. General-purpose ones, on the other hand, can acquire all of the visible bands. External components, including GNSS antennas and calibration sensors, are also attached to cameras. Visible bands are obtained to undertake multi-band analysis and indices computing. By taking into consideration the incident light circumstances throughout the UAV operation, the panel acquisition guarantees a more precise collection of data.

IV. Light detection and ranging

LiDAR is an active sensor that sends a signal to a target item and measures the duration of flight and intensity of the signal that is returned. A variety of LiDAR models are commercially available for use on a UAV system with sufficient payload and dimension capacity. The first applications of these sensors are forestry and vegetation mapping.

2.5 Deep-learning

Deep Learning is a subfield of machine learning (Anon n.d.-a; Lecun, Bengio, and Hinton 2015; Mandapati et al. 2022) concerned with algorithms inspired by the structure and function of the brain called artificial neural networks. It is a part of a broader family of machine learning methods based on artificial neural networks with representation learning. Deep Learning can be classified into three categories, which is supervised, semi-supervised or unsupervised. Deep-learning architectures such as deep neural networks, deep belief networks, deep reinforcement learning, recurrent neural networks, and convolutional neural networks have been used in fields such as computer vision, speech recognition, natural language processing, machine translation, bioinformatics, drug design, medical image analysis, climate science, material inspection, and board game programs, producing results that are comparable to and in some cases superior to traditional methods(Deng and Yu 2013a, 2013b; Lecun et al. 2015; Santosh, Das, and Ghosh 2022; Tan et al. 2018).

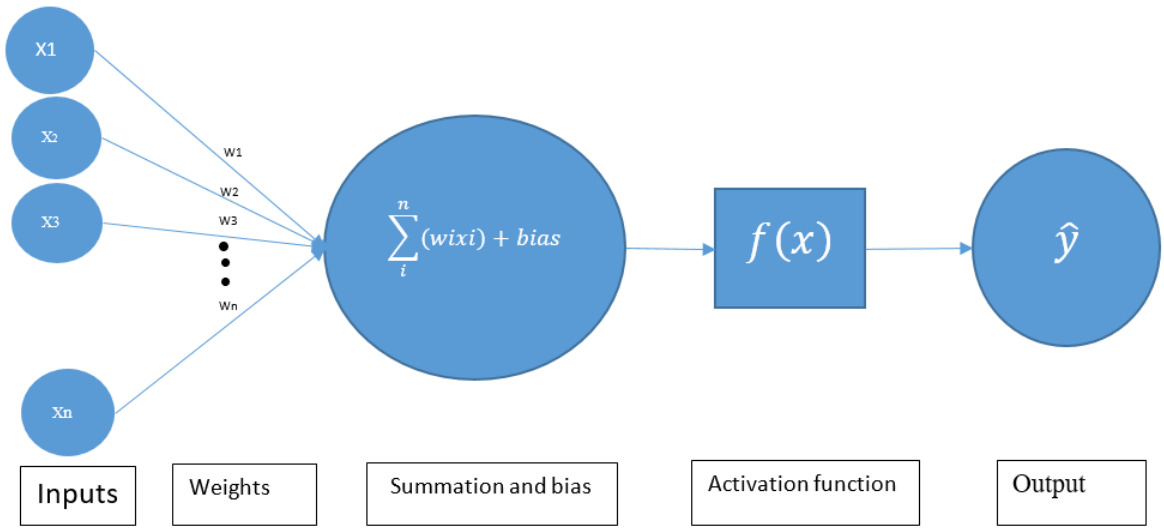


Figure 2-6 Single Neural network (perceptron)

The adjective "deep" in deep learning refers to the use of multiple layers in the network. Early work showed that a linear perceptron cannot be a universal classifier, but that a network with a nonpolynomial activation function with one hidden layer of unbounded width can. Deep learning is a modern variation that is concerned with an unbounded number of layers of bounded size, which permits practical application and optimized implementation while retaining theoretical universality under mild conditions. In deep learning the layers are also permitted to be heterogeneous and to deviate widely from biologically informed connectionist models, for the sake of efficiency, trainability, and understandability, whence the "structured" part.

Deep learning systems have a substantial credit assignment path (CAP) depth. The CAP is the chain of transformations from input to output. CAPs describe potentially causal connections between input and output. For a feedforward neural network (Figure 2-6 Single Neural network (perceptron), the depth of the CAPs is that of the network and is the number of hidden layers plus one (as the output layer is also parameterized). For recurrent neural networks, in which a signal may propagate through a layer more than once, the CAP depth is potentially unlimited.(Anon n.d.-c; Tsantekidis, Passalis, and Tefas 2022). No universally agreed-upon threshold of depth divides shallow learning from deep learning, but most researchers agree that deep learning involves CAP depth higher than 2. CAP of depth 2 is a universal approximator in the sense that it can emulate any function. Beyond that, more layers do not add to the function approximator ability of the network. Deep models (CAP >

2) can extract better features than shallow models and hence, extra layers help in learning the features effectively.

Deep learning architectures can be constructed with a greedy layer-by-layer method. Deep learning helps to disentangle these abstractions and pick out which features improve performance. For supervised learning tasks, deep learning methods eliminate feature engineering, by translating the data into compact intermediate representations akin to principal components, and derive layered structures that remove redundancy in representation.

Deep learning algorithms can be applied to unsupervised learning tasks. This is an important benefit because unlabeled data are more abundant than labeled data. Examples of deep structures that can be trained in an unsupervised manner are deep belief networks.

2.5.1 Classification of DL approaches

Unsupervised, semi-supervised, and supervised DL approaches are the three primary categories of deep learning. Additionally, deep reinforcement learning (DRL), often known as RL, is a form of learning approach that is typically classified as semi-supervised (and sometimes unsupervised) learning techniques.

2.5.1.1 Deep supervised learning

This method works with data that has been labeled. The environment has a collection of inputs and resulting outputs $(xt, yt) \sim \rho$ $\hat{y}_t = f(x_t$ when evaluating such a strategy. For example, if the input is $xt^{u}(\hat{y}_t, y_t$, the smart agent will guess and provide a loss value. The agent then updates the network settings several times to achieve a better approximation for the desired outputs. Following a successful training session, the agent gains the capacity to extract appropriate answers to questions from the surroundings. Recurrent neural networks (RNNs), convolutional neural networks (CNNs), and deep neural networks are examples of supervised learning approaches for DL (DNNs). Gated recurrent units (GRUs) and long short-term memory (LSTM) techniques are also included in the RNN category. The capacity to gather data or produce a data output using prior information is the key benefit of this technology. The downside of this strategy is that when the training set lacks samples that should be in a class, the decision boundary may be overstrained. Overall, this approach is less complicated than previous high-performance learning strategies.

2.5.1.2 Deep semi-supervised learning

The learning process in this approach is based on semi-labeled datasets. In certain cases, generative adversarial networks (GANs) and deep reinforcement learning (DRL) are used in

the same way as this approach. Additionally, RNNs, such as GRUs and LSTMs, is used for partially supervised learning. One of the benefits of this method is that it reduces the quantity of labeled data required. On the other hand, one of the downsides of this method is that it may provide inaccurate choices due to irrelevant input features contained in training data. One of the most well-known applications of semi-supervised learning is the text document classifier. Semi-supervised learning is appropriate for text document classification tasks because of the difficulty of getting a large number of tagged text documents.

2.5.1.3 Deep unsupervised learning

This approach allows the learning process to be implemented in the absence of labeled data (i.e. no labels are required). The agent discovers the unidentified structure or relationships in the input data by learning the significant attributes or inner representation necessary. Unsupervised learning techniques such as generative networks, dimensionality reduction, and clustering are typically included. Restricted Boltzmann machines, auto-encoders, and GANs are some of the most recently created DL components that have done well on non-linear dimensionality reduction and clustering problems. RNNs, which include GRUs and LSTM methods, have also been used in a variety of applications for unsupervised learning. Unsupervised learning's key weaknesses are its inability to give precise data sorting information and its computational complexity. Clustering is one of the most often used unsupervised learning techniques.

2.5.1.4 Deep reinforcement learning

Reinforcement learning is based on interacting with the environment, whereas supervised learning is based on sample data presented. Google Deep Mind created this approach in 2013. Many improved strategies based on reinforcement learning were developed as a result. For example, if the input environment samples: $xt \sim \rho$, the agent predicts, $y^t = x(ft)$, and the agent's received cost$c_t \sim P(c_t \mid x_t, \hat{y}_t)$, where P is the unknown probability distribution, the environment will ask the agent a question. It returns a noisy score as a result. Semi-supervised learning is a term used to describe this process. Several supervised and unsupervised approaches have been developed based on this principle. This learning is substantially more difficult to accomplish than typical supervised procedures since the reinforcement learning methodology lacks a basic loss function. Furthermore, there are two key differences between supervised and reinforcement learning: first, there is no complete access to the function, which necessitates optimization, implying that it should be queried via

interaction; and second, the state being interacted with is based on an environment, with the input xt based on the previous actions.

The type of reinforcement learning that must be undertaken to complete a task is determined by the space or scope of the challenge. DRL, for example, is the best method for optimizing problems with multiple parameters. Derivative-free reinforcement learning, on the other hand, is an approach that works effectively for situations with few parameters. Business strategy planning and robots for industrial automation are two examples of reinforcement learning applications. The fundamental disadvantage of Reinforcement Learning is that parameters can affect learning speed. The following are the key reasons for using Reinforcement Learning:

- ✓ It aids you in determining which activity yields the biggest benefit over time.
- ✓ It aids you in determining which situations necessitate action.
- ✓ It also allows it to choose the optimum strategy for achieving high rewards.
- ✓ Reinforcement Learning also includes a reward function for the learning agent.
- ✓ Reinforcement Learning cannot be used in all situations, such as when there is enough data to address the problem using supervised learning approaches.

Reinforcement Learning is time-consuming and computationally intensive. Particularly if the workspace is vast.

2.6 Transfer Learning

Deep CNNs, which give ground-breaking help for many classification issues, have been discovered to be widely used in recent research. Deep CNN models, on the whole, need a lot of data to work well. The absence of training data is a typical problem connected with employing such models. Gathering a big amount of data is a difficult task, and there is currently no effective answer. The TL approach (Zoph, Vasudevan, Shlens, & Le, 2018), which is particularly effective in dealing with the lack of training data issue, is now used to handle the undersized dataset problem. TL works by feeding a vast amount of data into a CNN model. The model is then fine-tuned for training on a small dataset of requests in the following stage.

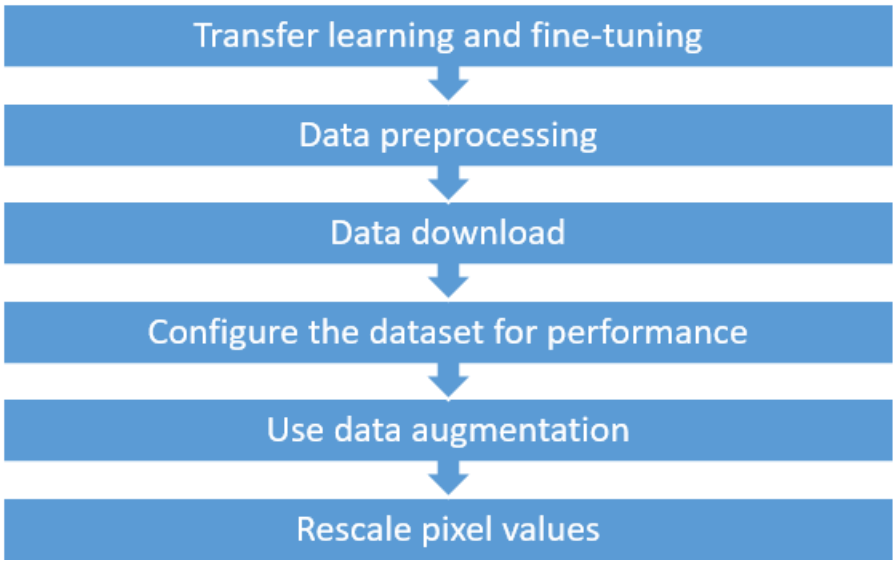


Figure 2-7 Data processing in transfer learning

During the training phase, the DL network learns the bias and weights while using a large amount of data. These weights are then used in other networks to retrain or test a comparable innovative model. As a result, rather than having to start from scratch, the innovative model may pre-train weights.

For image recognition, many CNN models were already trained on huge datasets like as ImageNet, such as (Iandola et al., 2016), GoogleNet(Szegedy et al. 2014), and ResNet(He et al. 2016). These models may then be used to recognize a new activity without having to start from the beginning. Aside from a few learning elements, the weights remain the same. These models are especially effective when data samples are few. There are several advantages to using a pre-trained model. To begin with, training complex models on vast datasets necessitates expensive computer resources. Second, training big models can take a long time, up to many weeks. Finally, a model that has been pre-trained can help with network generalization and convergence.

Using pre-trained models to solve a research problem: A large number of photos are required to train a deep learning technique. As a result, achieving good results is difficult in these conditions. When a large quantity of data is available, deep convolutional neural networks (DCNNs) with several layers may achieve great results in image classification or recognition applications, with performance sometimes exceeding that of a human (Pan 2016). However, large datasets and suitably generalized DCNN models are required to prevent overfitting difficulties in such applications. The dataset size for training a DCNN model has no upper limit. However, due to over or under-fitting difficulties, the accuracy of the model becomes insufficient if the used model has fewer layers or if a short dataset is used for training.

Models with fewer layers have low accuracy because they can't make use of the hierarchical characteristics of large datasets. Obtaining adequate training data for DL models is tough. Gathering labeled datasets is expensive in medical imaging and environmental science.

Retrieving the necessary training data and rebuilding the models is either too expensive or impossible in many remote sensing applications. There are frequently insufficient training data that adequately reflect the actual change information of ground objects, particularly for the change detection job. Therefore, it's critical to minimise the time and effort needed to gather new training data. Transfer learning or knowledge transfer between task domains might be a solid option in the situation. The capacity to take information from one or more source tasks and apply it to a new or target task is referred to as transfer learning. Transfer learning is the process of enhancing the target predictive function by employing the corresponding knowledge, provided a source domain with a relevant source task and a target domain with a corresponding task.

There are now two main strategies being used in transfer learning research. The first method is training a new shallow model based on the outputs of one or more layers of a network (such as AlexNet or resnet-101) trained on a different task and utilising them as general high-dimensional feature detectors. The second method is more complex and entails fine-tuning the network that has already been trained on generic photos. As a result, in addition to replacing the final layer (used for classification and regression), the earlier layers are also kept in place. Hou et al. transported CNNs that had already been trained on a sizable natural image data set, such as ImageNet, to an RS domain, using the previous method. They specifically adjust the VGG-16 to conform to their optical RS pictures on an aerial image dataset (AID) in order to improve outcomes. The ResNet-101 network was used by Venugopal et al. as a pre-trained model, and the parameters were adjusted based on a dilated convolutional neural network (DCNN) that recognises differences between the two pictures. The categorised result is then identified as unaltered and modified regions from the final feature map. To address the issue of change detection in optical aerial photos, Zhang et al. put forth a novel approach based on a deep Siamese semantic network trained with an enhanced triplet loss function. Due to the difficulties of directly training the Siamese network, a DeepLabv2 model that had been pre-trained on a large-scale picture data set (such as the PASCAL VOC 2012 dataset) was transferred to the network. Based on this methodology, the network has produced equivalent performance with less computational expense. The four steps that make up this change detection approach are: The input bi-temporal pictures are first preprocessed using histogram matching in order to apply a radiant correction to the two

coregistered images. The deep Siamese semantic network is then fed the preprocessed pair of pictures to produce two feature maps. After that, bilinear interpolation is used to apply a scaling operation to two semantic feature maps. The Euclidean distance between semantic feature maps is then calculated to create a distance map. The distance map is then split using a straightforward threshold segmentation technique, which produces the final change detection result. Dual learning-based Siamese framework (DLSF), an innovative hybrid end-to-end framework for change detection from very high resolution (VHR) pictures, was suggested by Fang et al. Two concurrent streams—dual learning-based domain transfer and Siamese-based change decision—make up this framework. The second approach aims to learn a decision strategy to determine the changes in the two domains, respectively, whereas the first path aims to reduce the domain discrepancies between two paired pictures and keep the intrinsic information by translating them into each other's domain. By lowering the disparity in distribution between the two domains, Yang et al. have applied the idea of change from the source domain to the target domain. In their model, the pretraining phase consists of two tasks: supervised change detection using the U-Net architecture in the source domain and a reconstruction network without labels in the target domain. The final layers specific to each job are trained individually, but the lower levels are shared across the two tasks. After pretraining, the change detection network is adjusted for the target domain using trustworthy labels selected from a CD map. Although the job of detecting changes in sea ice requires just a little amount of training data, Gao et al. employed a huge data set to train a transferred multilayer fusion network (MLFN) and an optimization technique to fine-tune the network parameters.

2.7 Related Works

Unmanned aerial vehicles (UAVs) have become an attractive area of research increasingly in recent years (A Bhardwaj 2016, Anon n.d.-b; C Gomez 2016; F Chiabrando 2013; F Clapuyt 2016; Guo, Chen, and Zhao 2021; Leira et al. 2021; P Boccardo 2015; S Amici 2013; Silvagni et al. 2017; Stöcker et al. 2017; Sun, Boukerche, and Wu 2017). Many researchers have been studying UAVs to utilize as an ideal platform for civilian tasks or military tasks, such as inspection, delivery, reconnaissance, or surveillance like the mobile robots based on autonomous navigation, real-time path planning, and object recognition (Fisher 2004; Oh et al. 2011).

A. I. Khan and Y. Al-Mulla presented The UAV has advantages of flying at low altitude, small size, high resolution, lightweight, and portability. The UAV and machine learning has

immense scope in scientific research. This study has synthesized research on UAVs and machine learning implementation. The blended UAV and machine learning research have not yet attained maturity. The present study found that researches in this area are sporadic and the bulk of them relates to the computer/wireless network, smart cities, military, agriculture, mining, and wildlife statistical analysis. The random forest and support vector have been incomparably used in UAVs. The USA and the Asia Pacific account bulk of the UAV research and usage shares. There are security and privacy concerns due to the proliferation of unregistered commercial UAVs. Future research in this direction expects to propose a model for detecting and identifying unregistered consumer UAVs. The study is a work in progress. In the future, trained machine learning models will be developed for recognizing objects in the UAV and satellite imagery(Leira et al. 2021).

Paper presented by C. P. C. Chanel, F. Teichteil-Konigsbuch, and C. Lesire discussed, The use of unmanned aerial vehicles (UAVs) is growing rapidly across many civil application domains, including real-time monitoring, providing wireless coverage, remote sensing, search and rescue, delivery of goods, security and surveillance, precision agriculture, and civil infrastructure inspection. Smart UAVs are the newest major advancement in UAV technology, promising to open up new possibilities across a variety of applications, particularly in civil infrastructure in terms of decreased costs and hazards. More than $45 billion in market value of UAV utilisation is anticipated to be dominated by civil infrastructure. They discussed the difficulties associated with UAV civil applications in their article. Additionally, they go on current research directions and offer predictions for upcoming UAV applications. In addition, they outline the major difficulties associated with charging, avoiding collisions and swarming, as well as networking and security-related difficulties. They explore open research issues and derive high-level insights on how these challenges might be approached based on our analysis of the most recent literature. In their presentation, E. Unlu, E. Zenou, N. Riviere, and P.-E. Dupouy noted that the commercial unmanned aerial vehicle (UAV) market, commonly known as the "drone" sector, has grown significantly over the past few years, making these gadgets very accessible to the general people. Because these gadgets have the potential to pose major risks whether intentionally or accidentally, this phenomena has quickly aroused security concerns. In recent years, academics and business have put out a number of ideas to safeguard crucial areas. Due to its resilience, computer vision is frequently used to identify drones autonomously as opposed to other suggested techniques like RADAR, acoustics, and RF signal analysis. We see that deep learning techniques are preferred among these computer vision-based methods due to their

potency. We describe a target recognition and tracking system in this study that makes use of a stationary wide-angle camera and a lower-angle camera mounted on a revolving turret. We provide a combined multi-frame deep learning detection method that overlays the wide-angle static camera's frame with the zoomed camera's frame to effectively employ memory and time. This method reduces the cost of a resource-intensive detection algorithm by simultaneously performing the initial identification of small-sized aerial invaders on the main picture plane and their detection on the zoomed image plane.

In addition to this, we present the integrated system including tracking algorithms, deep learning classification architectures, and protocols.

2.8 Summary

Due to the bird's-eye view viewpoint, the extremely complex backdrops, and the varied appearances of things, object recognition in aerial view is an active yet difficult topic in computer vision. Methods using horizontal suggestions for common item recognition frequently produce mismatches between the region of interest and objects, especially when recognizing densely packed objects in aerial photos. The final item classification confidence and localization accuracy are frequently out of alignment as a result of this.

In this research, we offer a novel method for target identification and event analysis using a camera mounted on a distinct UAV. Using a pre-trained deep learning model, we first extract recognizable features from the scene. We identify the spatio-temporal characteristics of each moving item, and we then categorize those possible targets depending on how they move relative to the backdrop. Tracking with the Kalman filter improves performance. As a consequence, the candidate detections are consistently timed. On video datasets collected by a UAV, the algorithm was verified.

Table I: Related work summary

Author and paper	Dataset	Type of Task	Data Source	Video	# Classes	# Samples	Year
(Shu et al. n.d.) Joint Inference of Groups, Events and Human Roles in Aerial Videos	UCLA Aerial Event dataset	human-centric event recognition	self-collected (actor staged)	✓	12	104	2015
(Barekatain et al. 2017) Okutama-Action: An Aerial View Video Dataset for Concurrent Human Action Detection	Okutama-Action dataset	human action detection	self-collected (actor staged)	✓	12	-	2017
(Kyrkou 2020) AIDER (Aerial Image Dataset for Emergency Response Applications)	AIDER dataset	disaster event recognition	Internet	✗	5	2545	2019
(Mou et al. 2020)A Dataset and Deep Learning Benchmark for Event Recognition in Aerial Videos	ERA dataset	general event recognition	YouTube	✓	25	2864	2020
(Bozcan and Kayacan 2020) AU-AIR: A Multi-modal Unmanned Aerial Vehicle Dataset for Low Altitude Traffic Surveillance	AU-AIR	multi modal object detection	Self-collected	✓	8	32,823	2020
TIEA OF UAV based on DL(self)	TIEA OF UAV based on DL(ours)	general event recognition	self-collected, YouTube, and Dataset	✓	30	-	2022

CHAPTER THREE

3 MATERIALS AND METHODS

3.1 Chapter Overview

The methodology, mathematical formulation of the evaluation metrics and necessary materials for this research are described in this chapter. This chapter goes through study and implementation to be the best algorithm that performs event analysis, target identification, and classification, with computation power requirements that can be met with a common laptop computer.

3.2 Dataset Preparation

There are many machine learning and deep learning researches are done by many researcher in the field of computer vision and machine learning using structured, semi-structured and unstructured data from all over the world. One of the major influencer of deep learning to be researched at this time is availability of large dataset and computation power for training and evaluation of the model. Here we describe how we prepared our dataset for training and testing of different pre-trained model. Since our research depend on data obtained from aerial view, we used data collected by UAVs which has fully integrated system of camera, and communication media for remote connection. In this research we have used two kind of datasets for our entire research, first previously collected dataset by computer vision and deep learning community, which set as free for other researcher to fill the gap under this area. Secondly self-collected datasets were used. In this research we will follow process steps as shown in Figure 3-1 over all steps of the process.

- ✓ Image and video data were collected using a UAV (Figure 3-2 UAVs for data collection) and also a video from YouTube which recorded by UAV.
- ✓ Already collected data was used from data sets like ERA (Mou et al. 2020) and AIDER (Kyrkou and Theocharides 2019a).
- ✓ Data were analyzed using the power of jupyter notebook which programmed by python with help of different libraries and APIs freely available in Google Colab and GPU'S of tensorflow.
- ✓ The evaluation was conducted using a comparison with the previous algorithms by applying the concept of transfer learning and fine tuning.

- ✓ AU-AIR dataset is the first multi-modal UAV dataset for object detection.(Khan and Al-Mulla 2019).

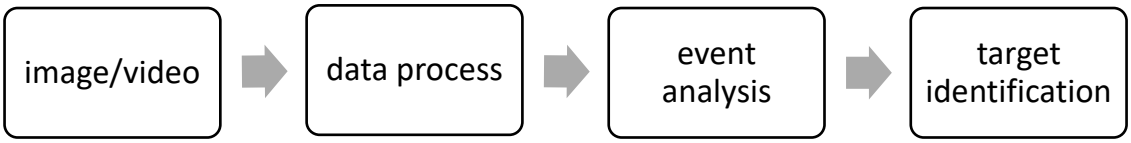


Figure 3-1 over all steps of the process

Figure 3-2 UAVs for data collection

3.2.1 Pre-collected datasets

Pre-collected datasets are a datasets which collected by researcher who interested in the field and they put together those data by collecting from different sources like UAV (drone) and YouTube video of aerial scene. In our research we have used pre-collected dataset ERA and AU-AIR.

ERA (Event Recognition in Aerial videos) (Mou et al. 2020) consists of 2,864 videos each with a label from 25 different classes corresponding to an event unfolding in 5 seconds. The 25 categories were grouped as Security, Disaster, Traffic, Productive activity, Social activity, Sport, and Non-vent. The ERA dataset is designed to have a significant intra-class variation and inter-class similarity and captures dynamic events in various circumstances and at dramatically various scales.

The multi modal datasets of AU-AIR for object detection (Bozcan and Kayacan 2020) is also another paper presented for detection of image from aerial image. It combines robotics and vision for UAVs with multimodal data from various on-board sensors and advances the creation of robotic and computer vision algorithms for autonomous aerial surveillance. AU-AIR has several features like Object detection in aerial images, >2 hours of raw videos,

32,823 labeled frames, 132,034 object instances, 8 object categories related to traffic surveillance, Frames are also labeled with time, GPS, IMU, altitude, and linear velocities of the UAV

3.2.2 Self-collected data

We collected images and videos using UAV and sent them to data annotators. Each annotator will be asked to localize image snippets that depict specific events and categorize them to the specific event. There were three different annotators with payments who were responsible for validating the collected image during the annotation procedure. The first annotator generates primary annotations, to begin with. Then annotations from the first round are sent to the second annotator for tune-ups. Finally, the third annotator screen all annotated image and remove images with quite similar contents. Moreover, they were asked to double-check whether these images were taken by UAVs.

3.3 Data Analysis

The Data set collected using different methods like previously composed and self-collected datasets using UAVs which are taken from aerial view are analyzed using Google collaborator and GPU'S installed high speed with high-performance computers.

Google Colaboratory or "Colab" (Colaboratory, n.d.) is a shorthand for a Google Research product. Machine learning, data analysis, and teaching are three areas where Colab excels because it lets anybody create and run arbitrary Python code using a web browser. Technically speaking, Colab offers unrestricted access to computer resources, including GPUs, and is a hosted Jupyter notebook service. Graphics processing units (GPUs), originally developed for accelerating graphics processing, can dramatically speed up computational processes for deep learning. They are a crucial component of a contemporary artificial intelligence infrastructure, and new GPUs have been created and tuned especially for deep learning. Many deep learning models can be trained faster thanks to graphics processing units (GPUs). It is computationally expensive to multiply matrices in order to train models for applications like image classification, video analysis, and natural language processing. These operations might benefit from the GPU's massively parallel architecture. On a single processor, training a deep learning model that requires extensive computation activities on very big datasets might take days. However, you may cut training time in half, from days to hours, if your application is built to offload those activities to one or more GPUs.

3.4 Prediction and Evaluation Metrics

The evaluation criteria used in DL tasks are critical in getting the best classifier. They are used in two steps of a typical data categorization procedure: training and testing. During the training stage, it is used to improve the classification algorithm. This means that the evaluation measure is used to discriminate and pick the best answer, such as a discriminator that can produce an extra-accurate forecast of incoming classifier evaluations. For the time being, the evaluation metric is used to quantify the efficiency of the constructed classifier in the model testing step utilizing concealed data, such as an evaluator. As specified in Eq. 1, TN and TP are the numbers of successfully categorized negative and positive cases, respectively. Furthermore, the terms FN and FP refer to the number of misclassified positive and negative cases, respectively. The following are some of the most well-known assessment measures.

The prediction model needs to be evaluated by model evaluation techniques to ensure that the model is compatible with the dataset and works well on new unseen input data. The purpose of model performance evaluation is to estimate the overall accuracy of a model based on unseen/ external sample data (Raschka 2018). The performance evaluation method is divided into two categories; holdout and Cross-validation (Singh n.d.).

3.4.1 Holdout

In this method, the largest data set is randomly divided into two sub-categories: The training set is a subset of the dataset to build predictive models.

The validation set is a subset of the dataset to evaluate model performance developed at the training level. Provides a test platform to fine-tune the model parameters and select the best model. Not all modeling algorithms require a validation set.

Test sets are a subset of a dataset to evaluate the future performance of a model. If a model fits the training set much better than it fits the test set, over-fitting is probably the cause.

On the other hand, the holdout is dependent on only one train-test split. That makes the hold-out technique score dependent on how the dataset is divided into train and test sets. (Allibhai n.d.) When there is a large dataset, the holdout method is better.

3.4.2 Cross-validation

Cross-validation is one of the performance evaluation methods for evaluating and comparing models by dividing data into partitions: one is used to train a model and the rest is used to test or validate the model. To avoid over-fitting, the cross-validation technique is the most widely

used evaluation method (Stone 1974). Different types of cross-validation can be used as an evaluation method.

K-fold cross-validation: in the k-fold, the first samples are randomly divided into k equal-sized sub-samples. Of the k sub-samples, one subsample remains as the validation data set for testing the model, and the remaining k − 1 sub-sample serves as the training dataset (Brownlee n.d.). The average of k scored accuracy is called CV accuracy and served as a performance metric of the model.

Stratified K-Fold Cross Validation**:** Because K-fold cv randomly modifies the data set and divides it into folds, the chances of getting very unbalanced folds are high, which makes model training to be biased. The stratified form of k-fold CV, which conserves the imbalanced class distribution in each fold is called stratified k-fold CV, and It forces class distribution in each split or divide of the data to match the distribution in a complete training set (Hussain Mujtaba n.d.). In this study stratified k-fold cross-validation technique was used because the data collected contains imbalanced class distribution.

3.4.3 Classification Metrics

Since classification algorithms do not produce the same results, predictive classification model evaluation metrics are needed to measure model performance. There is a different kind of classification model evaluation (M and M.N 2015). Of these, precision, recall, confusion matrix, and f1-score are used in this study with cross-validation. The followings are the fundamental terms in performance measure:

True positive (TP): is the condition when both actual value and predicted value are true.

True negative (TN): is the condition when both the actual value of the data point and the predicted are False.

False-positive (FP): These are the cases when the actual value of the data point was False and the predicted is true.

False-negative (FN): are the cases when the actual value of the data point was true and the predicted is False.

1. Accuracy: Determines the proportion of correctly predicted classes to the total number of samples assessed (Eq. 1).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

(1)

2. Sensitivity or Recall: Used to calculate the percentage of correctly identified positive patterns (Eq. 2).

$$Sensitivity = \frac{TP}{TP + FN}$$

(2)

3. Specificity: Used to calculate the percentage of correctly identified negative patterns (Eq. 3).

$$Specificity = \frac{TN}{FP + TN}$$

(3)

4. Precision: Used to compute the positive patterns in a positive class that is properly predicted by all predicted patterns (Eq. 4).

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

(4)

5. F1-Score: Determines the harmonic average of recall and accuracy rates (Eq. 5).

$$F1score = \frac{2(Precision * Recall)}{Precision\ +\ Recall}$$

(5)

6. J Score: Youdens J statistic is another name for this metric. The metric is represented by Eq.6.

$$Jscore\ =\ Sensitivity + Specificity\ -\ 1$$

(6)

7. This measure relates to the potential of a false alarm ratio as determined in Eq. 7.

$$FPR\ =\ 1\ -\ Specificity$$

(7)

8. Area AUC (Area under the ROC Curve) is a standard ranking statistic. It's used to compare learning algorithms and to build an ideal learning model. The AUC value, unlike probability and threshold measures, reveals the whole classifier ranking performance. The AUC value for a two-class issue is calculated using the formula below. (Eq. 8)

$$AUC = \frac{Sp - \frac{np(nn+1)}{2n}}{np\ nn} \quad (8)$$

Where Sps is the total number of positive ranked samples. The numbers nn and np represent the number of negative and positive samples, respectively. When compared to accuracy metrics, the AUC value has been empirically and theoretically validated, making it extremely useful for finding an optimal solution and evaluating classifier performance via classification training.

The AUC performed admirably when it came to discrimination and assessment processes. When distinguishing a large number of produced solutions, however, the AUC computation is largely cost-effective for multiclass problems. Furthermore, according to the Hand and Till AUC model, computing the AUC takes O(|C|2nlogn) and in Provost and Domingo's AUC model, it takes O(|C|nlogn).

3.5 Tools for Data Preparation

The dataset that were collected are single frame and short size videos which described in section 3.2 Dataset Preparation. To train and test our pre-trained deep learning models like VGG16, VGG19, MobileNet, DenseNet, ResNet. After training our datasets through different deep learning models for single frame and videos in tensorflow available freely we prepared well document for our research in Microsoft office 2016 word document. Microsoft Office OneNote Microsoft Office also includes Word, Excel, Publisher, and PowerPoint in addition to OneNote. It is a well-known application program created for study, note-taking, and information storing. A program that enables us to edit words by copying text from a file printer or image and pasting it into our notes.

MS-PowerPoint: it is a comprehensive presentation graphics suite. It provides us with everything required to create a presentation that looks polished. Word editing, outlining, sketching, graphing, and presentation management capabilities are all available in PowerPoint, all of which are simple to use and pick up.

3.5.1 Implementation Tools

Working with the finest machine learning algorithms might be just as crucial as selecting the appropriate equipment. As a result, the suggested model has been implemented using the following tools.

TensorFlow is an open-source toolkit for statistical and predictive analytics workloads, deep learning, and large-scale machine learning. This kind of technology facilitates the process of obtaining data, offering forecasts at scale, and improving future outcomes, which speeds up and simplifies the implementation of machine learning models by developers.

Okay, so what does TensorFlow actually do? For tasks like handwritten digit classification, picture recognition, word embedding, and natural language processing, it can train and use deep neural networks (NLP). Any programmer may be enhanced with the code from its software libraries to aid in the learning of these activities.

Applications for TensorFlow may be executed on either standard CPUs (central processing units) or GPUs (higher-performance graphics processing units). Due to the fact that TensorFlow was created by Google, it also utilizes TPUs created by that firm with the express purpose of accelerating TensorFlow tasks.

3.5.2 Tools for programming

There are several highly well-liked machine learning languages, including Python and others. To create the models, Python was utilized as the programming language. Python is a well-known language with top-notch machine learning and data analysis packages. It is also a high-level programming, object-oriented, general-purpose interpreted, and interactive language.

3.5.3 Tools for data analytics and visualization

Pandas is a Python data analysis framework that improves modelling and analytics. By transforming JSON, CSV, SQL database, and TSV data files into data frames, Excel, or SPSS tables with rows and columns, Pandas makes analysis simpler.

A Python machine learning package for high-quality visualizations is called Matplotlib. A Python two-dimensional (2D) graphing package is called Matplotlib. With just a few lines of code, Matplotlib enables you to create production-quality visuals. The display of ML data by plotting.

Jupyter notebook: features for group collaboration. A free interactive computing online application is the Jupyter Notebook. The functionality of a notebook is extensive, and there are many possible uses for it. While creating the suggested model, we utilized this application to write, execute, and process data using a Python program.

3.5.4 Frameworks

NumPy is a Python extension package for use in scientific computing. It supports matrices and multidimensional arrays. Data in ML may be represented as arrays. Numerous types of databases may be rapidly and simply integrated with NumPy. While transforming the data into a vector and reshaping the vector to fit the vector with the suggested model, we made use of this package.

Easy-to-use machine learning framework for a variety of industries: Scikit-learn. Scikit-learn is a free program. SciPy (Scientific Python), NumPy, and matplotlib are the foundations upon which the Python machine learning library is constructed. To divide the dataset into train, validation, and test data for the proposed model, we utilised this package.

3.5.5 Hardware Tools

The tools which are discussed above in this section have been deployed on a personal computer equipped with a Processor 11th Gen Intel(R) Core(TM) i5-1135G7 CPU @ 2.40GHz, 2300 Mhz, 4 Core(s)a processor Installed RAM 8.00 GB 64-bit operating system, an x64-based processor with installed NVidia GeForce MX350 graphics. 1Terabyte hard disk storage capacities. The operating system is Windows 11 Home.

3.6 Summary

The power of Jupyter Notebook, which was created in Python with the aid of many publicly accessible libraries and APIs in Google Colab and tensorflow, was used to analyze the data. We used data acquired by UAVs since our research depends on information received from aerial views (Unmanned Aerial Vehicles). It progresses the development of computer vision and robotic algorithms for autonomous aerial surveillance.

In the realm of remote sensing, UAVs are aiding in the quick replacement of conventional aerial photography (photogrammetry). They create orthophotos—high spatial resolution aerial images—of the earths surface. They are excellent monitoring tools because they enable continuous observation of a wide range of occurrences.

CHAPTER FOUR

4 PROPOSED MODEL OF TARGET IDENTIFICATION AND EVENT ANALYSIS USING DEEP LEARNING

4.1 Chapter Overview

The design and overview of the proposed solution have been covered in this chapter. The proposed method for target identification and event analysis of UAV based on deep learning prediction using pre-trained model would be designed and developed in order to solve the problems and answer the questions raised in chapter one, following to fill the gaps mentioned in the literature review. The proposed solution comprises the way of creating datasets and several types of deep learning algorithms and model assessment approaches that are utilized to evaluate models.

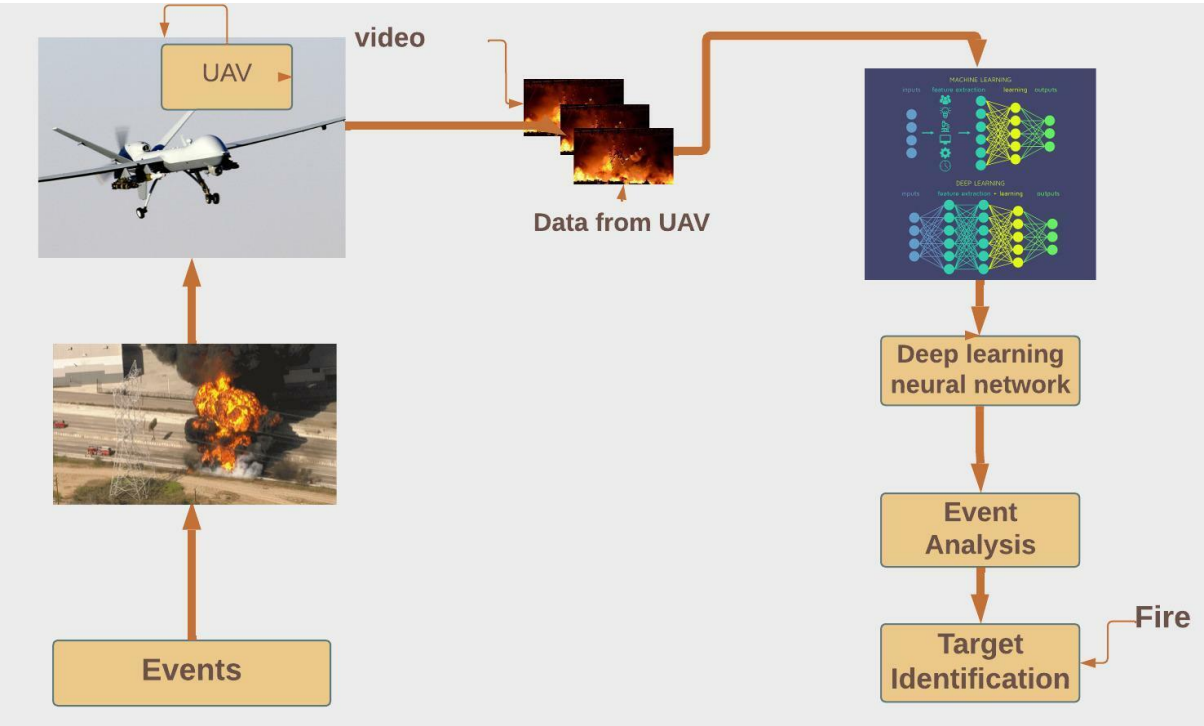


Figure 4-1 Shows the overall diagram of the target identification and event analysis

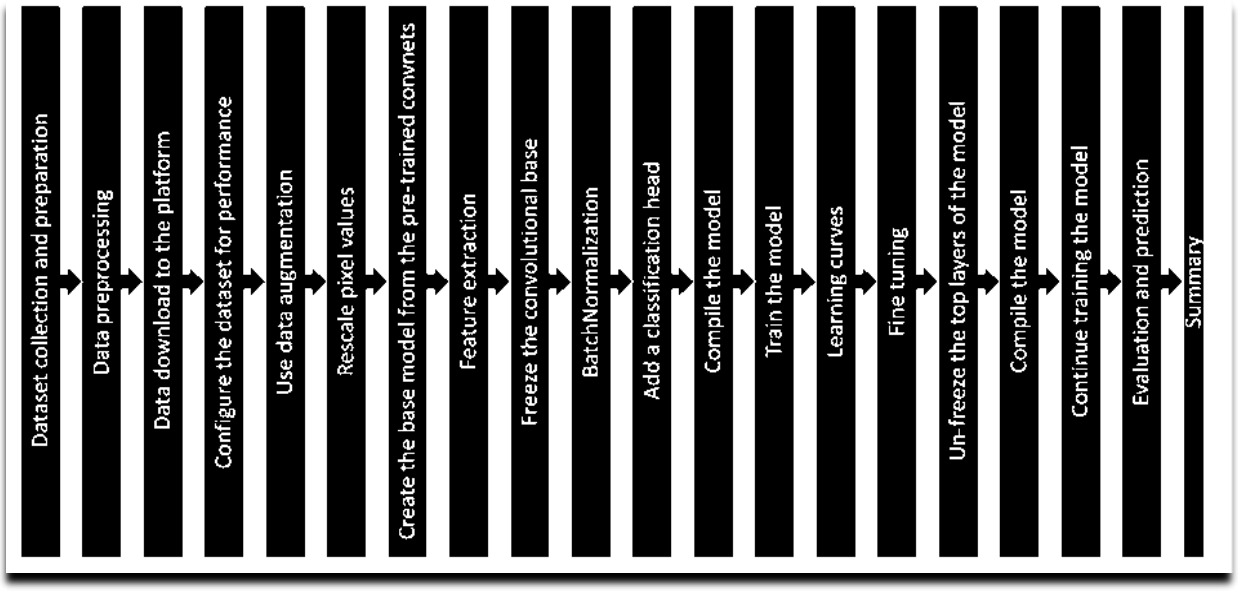


Figure 4-2 Overall flow chart of the proposed model

As shown in the Figure 4-2 above of flow chart model we have trained all our datasets with pre-trained model. We classified the flow chart in to four main parts, first dataset collection and preparation, second feature extraction, third transfer learning and the fourth is fine tuning and evaluation.

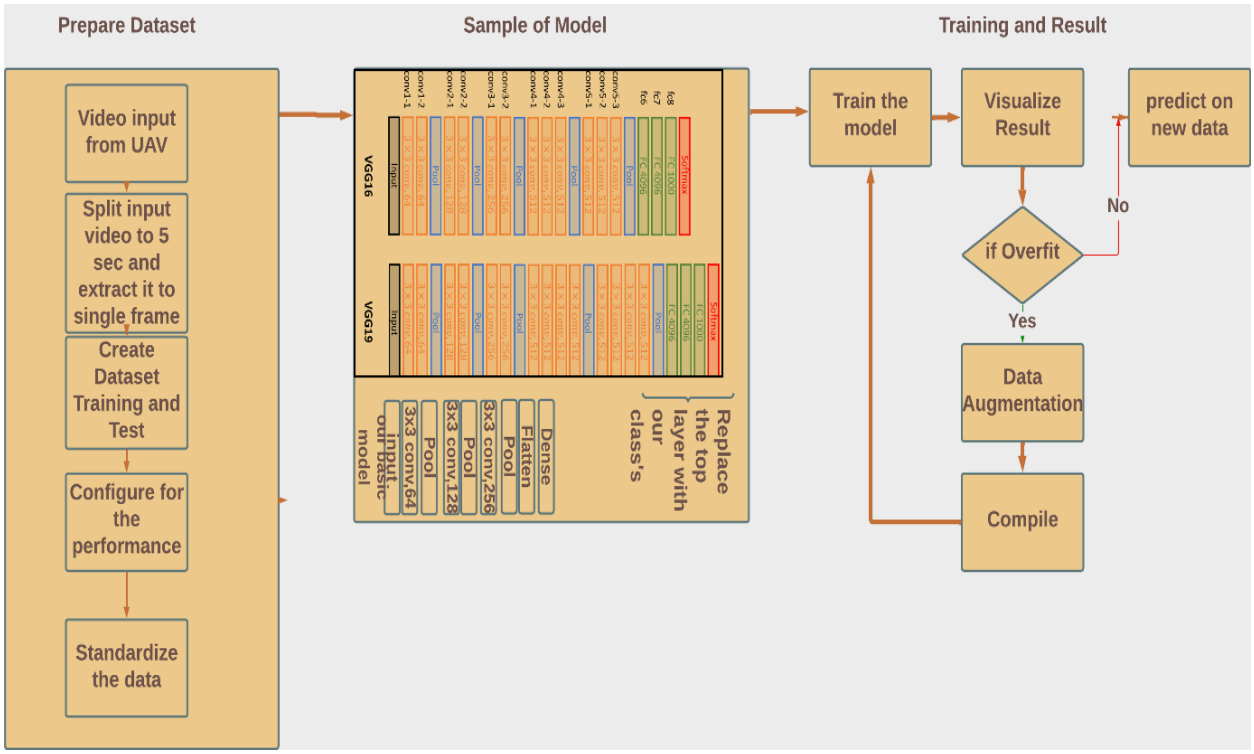


Figure 4-3 overall Architecture

4.2 Dataset

In this research paper we have collected 3036 images from different source like Ethiopian Air Force UAV department and pre collected dataset, which categorized into 32 class's listed below. We collected 1786 images by our self and 1250 images were used from previously collected dataset.

The following were some of the datasets we utilized for this research: Armed Group, Artillery Howitzer, Baseball, Basketball, Boating, CarRacing, Concert, Conflict, Constructing, Cycling, Fire, Flood, Harvesting, Landslide, Military Camp, Mudslide, NonEvent, Parade Protest, Party, Plowing, PoliceChase, Post Earthquake, Religious Activity, Running, Soccer, Swimming, Tank movement, Traffic Collision, Traffic Congestion, military parade, and military vehicle were all taken from the ERA dataset. Before applying data augmentation, this dataset contained 3036 images from 32 classes that were taken from an aerial view. However, after applying data augmentation on zoom, flip, rotation, and contrast, the total number of the dataset was multiplied by 4, giving us 12,144 images for training and testing our model. Our dataset needs to be split into training and test categories, each 80% by 20%, in order to train the model. While the test set only includes 2428 picture samples, the training set has 9715 image samples. For maximum dataset diversity, these data were collected from several sites across the globe. Each image in this collection was shot at a resolution of 640 x 640 and then resized for various models in accordance with their specifications, for example. We reduced the size of the Vgg16 model's input to 224 by 224 for the first layer.

Figure 4-4 Dataset class's

4.3 Model Selection

Using model selection, we selected a suitable deep learning model from the available options. Different deep learning models were tested for performance during the model selection

process, and the deep learning model that performed the best was chosen. On a separate test set, the performance of the chosen deep learning model was assessed (model assessment). The target model's weights were initialized with a transfer dataset to perform transfer learning. The target model's layers were therefore already trained to recognize objects. These objects, however, were not those of the intended task (aerial images). Therefore, using our training set of categorized photos, the target model was enhanced.

Deep convolutional neural networks serve as the foundation networks for deriving abstract feature maps from input data. Base networks are common deep learning architecture building blocks that can be used with different datasets for image categorization. In this study, the base networks ResNet-101, Inception-ResNet-v2 (Szegedy et al. 2017), and NASNet are evaluated (Zoph et al. 2018).

The image was altered by applying the following realistic geometric distortions (Zoph et al. 2018). Only the horizontal flip of the augmentation, which reflects the image along the central axis, was used in our baseline augmentation. The baseline augmentation was the only augmentation experiment that did not include a horizontal flip or at least one further augmentation. Since nighttime images are in the grayscale and can be detected using daytime images, we hypothesized that converting RGB to grey might improve recognition for nighttime images. Brightness, contrast, hue, and saturation are additional color enhancements that alter the related visual property. Additionally, the augmentation of color distortion significantly altered the image's brightness, hue, contrast, and saturation.

Grid search was used to choose the learning rate from a range between 0.1 and 0.0001. (Kornblith, Shlens, and Le 2019). In particular, the following learning rates—0.03, 0.003, 0.0003 (baseline), and 0.00003—were employed (Nasirahmadi et al. 2019). Finally for our training data model we selected 0.0001 learning rate to train the model and for testing purpose.

The following hardware and software were employed in our tests. Our open source deep learning software framework was TensorFlow6 v1.12.0, and the object detection task was carried out using the TensorFlow object detection API7 (Huang et al. 2016) NVidia GeForce MX350 graphics cards, with the Nvidia driver version 396.54 for personal computer and we used all our training in google collaborator Python 3 Google Compute Engine backend (GPU). One GPU processed around half of the experiments due to the parallel processing of the experiments. The option "num classes" was set to 31 in each trial.

4.4 Dataset Preprocessing

As shown in the figure above we begin by compiling a list of the 30 most often occurring events from aerial view to create our taxonomy. Additionally, we created a category called "non-event" that includes video devoid of specific events in order to test if models are capable of differentiating between events and regular videos. You can see the 30 classes that make up our dataset.

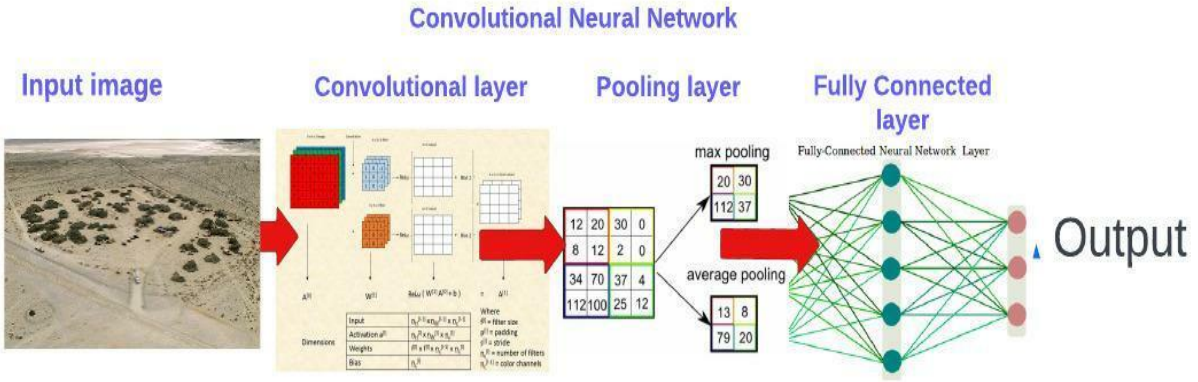
We search YouTube by parsing the video metadata and crawling the search engine to produce a collection of candidate videos for each category. These candidate films are then collected for further classification. Thus, we amass 1,120 lengthy films. Please take note that we have set the keywords "drone" and "UAV" to exclude any videos that are not from UAVs. The videos are then downloaded and sent to data annotators. Each annotator is expected to choose and localize 5-second video clips from lengthy candidate videos that portray particular occurrences.

Three distinct paid annotators are in charge of certifying edited video throughout the annotation process. Primary annotations are first created by the first annotator. The second annotator will then revise the first round's annotations. Finally, the third annotator screens all created 5-second videos and filters out those that have content that is too similar. Additionally, they are requested to confirm that UAVs did not capture these videos. The entire time spent on annotation is about 290 hours.

Figure 4-5 military camp aerial image

4.5 Convolutional Neural Network (CNN)

An input layer and an output layer are parts of a convolutional neural network (CNN). A convolutional layer, a pooling layer, and a fully connected layer are sandwiched between these two layers. For the complex collection of models, additional layers may be employed. The representation of data and weights is based on matrix-vector multiplication.



CNN only matches the part of the input image (feature) that matches. The feature extraction process performed by the CNN is shown by a 3x3 grid. Max polling was initiated after the convolution process was finished, which will cause the image stack to become smaller. For the maximum poling, the window size is specified together with the stride. The window is

then smoothly filtered across the image. Each feature map's dimensionality is decreased using max pooling. In convolutional neural networks, rectified linear units are typically used to carry out the normalisation step (ReLU). ReLU converts the filtered image's entire negative value to zero. The model's non-linearity property is enhanced via the ReLU method. The convolutional neural network, or classifier, showed a fully connected layer in the following layer.

4.5.1 Convolutional layer

The computation to extract relevant information from picture data occurs in this CNN building piece. Learnable filters, commonly referred to as kernels, make up each convolutional layer. A filter is a two-dimensional array of weights that is moved over the receptive fields of the image to determine whether or not a feature is present, as opposed to a coloured image, which may be thought of as a three-dimensional matrix of pixels. When a filter is applied to an image, the dot product is calculated using the input pixel and filter weights. The filter matrix is commonly 3x3 (but the size might vary). Following the filter's n-pixel shift, or "stride," the dot product of the input pixels and filter weights is calculated. Convolution is the procedure that keeps going till the input matrix is finished. An activation map or feature map is a matrix that is produced by each dot product. The feature map includes the feature data that was taken from the image by the chosen filter.

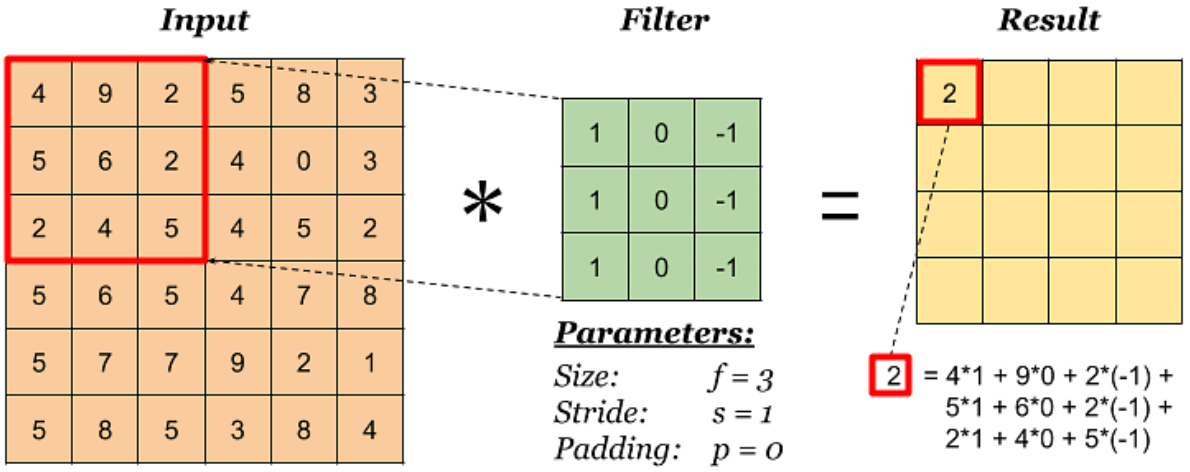


Figure 4-6 Convolutional layer input n_H6 x n_W6 with 3 x 3 filters with stride 1

As seen in the graphic above, an image sub-matrix of the same size is extracted from the 3x3 kernel or filter. The output matrix stores the dot product of the matrices. As the stride in the previous example was 1, the filter then advances by 1 pixel before performing the dot product computation once more.

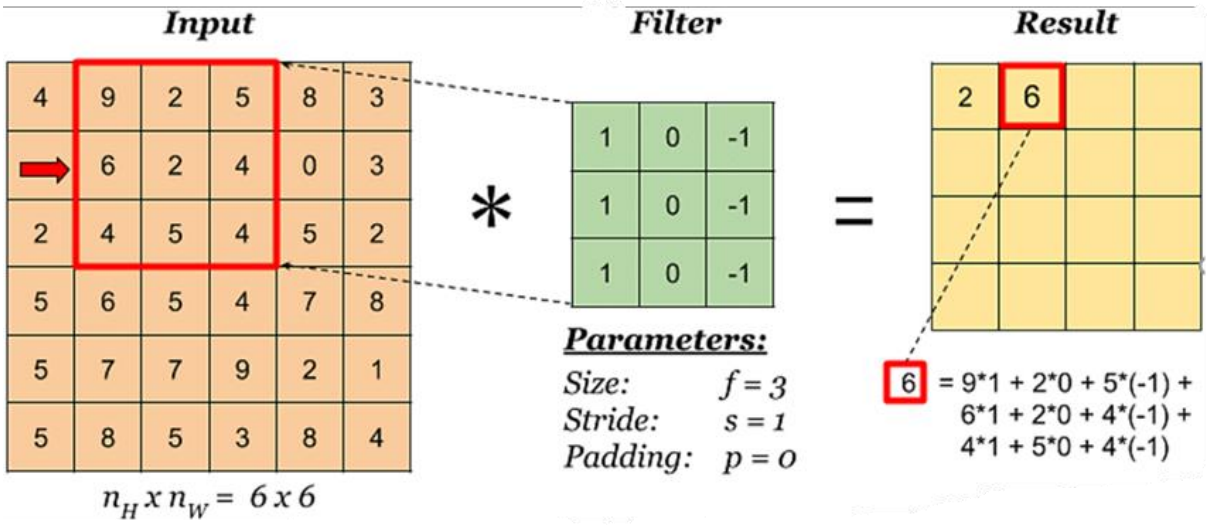


Figure 4-7 Input nH6 x nW6 with 3 x 3 filters with stride 1

The entire matrix will be traversed in a similar manner, and a feature map will be produced.

Any integer can be used as the stride's value, and a larger stride produces a smaller output matrix.

Additionally, if the filter does not completely fit the input picture matrix, some parts of the filter may migrate across the image matrix outside of the image. Take the following case, for instance.

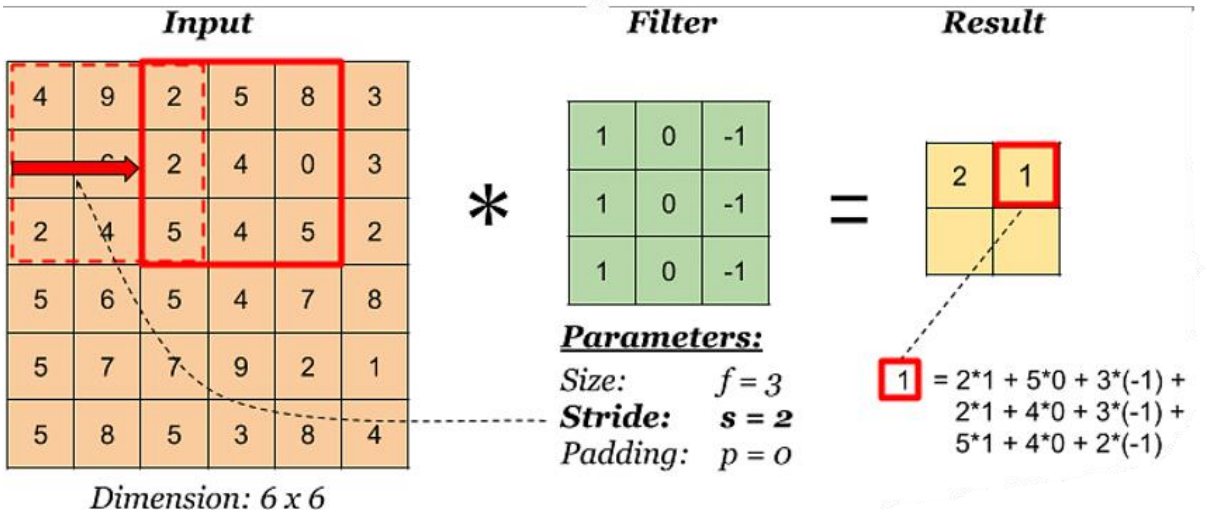


Figure 4-8 Input n_H6 x n_W6 with 3 x 3 filters with stride 2

Here, the filter stride is two. The final column of the filter won't have any picture pixels to interact with in the subsequent iteration when the filter tries to advance by two steps. Zero padding enters the picture at this point. To prevent such circumstances, the input matrix can be padded with elements having zero values. Padding comes in three different forms:

1. Valid Padding - In this scenario, if the filter is outside the input matrix, the final convolution is discarded or skipped.

2. Same Padding - In this instance, the padding makes sure that the input and output layers are both the same size.

3. Full Padding - As the name implies, the output size is increased by adding a border of zeros to the input matrix.

Padding also prevents convolutional layers from needlessly shrinking the height and width of the feature map because of filters. Otherwise, the feature maps' dimensions would decrease in deep CNNs. After convolution, the feature map is transformed using ReLU, or Rectified Linear Unit, to provide nonlinearity.

4.5.2 Pooling Layer

By limiting the number of parameters, this layer reduces the dimensionality of the input, and it also uses a filter to do this. The sole distinction is that pooling layers aggregate the input pixels instead of utilizing a matrix with weights like convolutional layers do. Although some data is lost during this process, there are advantages to using pooling layers, such as a reduction in feature complexity, which makes CNN less prone to overfitting and speeds up calculations. Pooling layers come in two different varieties:

1. Max Pooling - The output matrix saves the input pixel with the highest value.
2. Average Pooling - The output matrix is saved with the average of the input pixels.

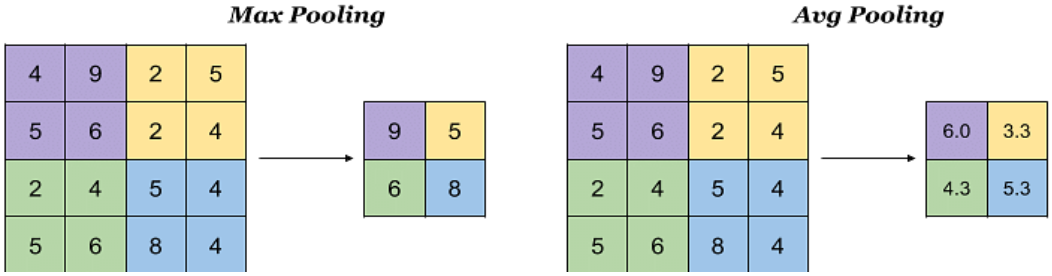


Figure 4-9 Max pooling and average pooling

4.5.3 Fully-Connected Layer

We can see that the feature map does not connect to every pixel value in the input image, indicating that the convolutional layer is only partially connected. Only the sub-matrix where a filter is used is connected to it. However, we also have something called a fully-connected layer, in which every node is linked to every preceding node. Three-dimensional vectors are produced as the output from the other two levels in the feature maps. For the final classification task, you can flatten them to produce a single-dimensional vector that is sent to this fully-connected layer (or layers). To estimate the probabilities of classes for the final prediction, this layer employs the softmax activation function.

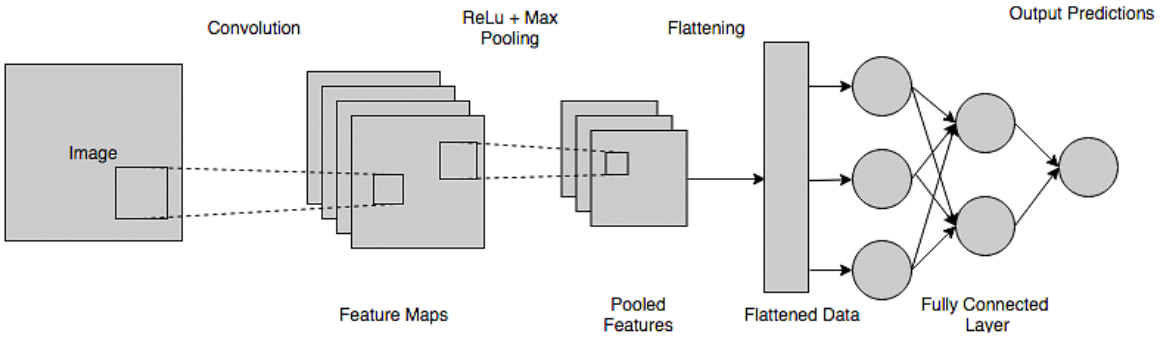


Figure 4-10 Fully-connected layer

In this study, Google is used for image categorization and prediction utilising transfer learning. Colab is another name for Google Colaboratory, a cloud-based service. Jupyter Notebook is the foundation of Colab, which uses it to perform deep learning and machine learning operations.

The Google Colab offers runtime GPU access without charge. The Google collaborative is helpful for GPU-centric applications like computer vision. According to MobileNet's overall architecture, the initial layer of the system is depthwise convolution, which is used for simple filtering that uses one convolutional filter per input channel. Similar to the first layer, the second layer consists of pointwise convolutions, commonly known as 1X1 convolutions. Utilizing a second layer in the MobileNet, linear combination computing of the input channels is employed to create additional features. ReLU is utilised as the activation function due to its robustness in low precision computation. The intermediate expansion layer filters features that are the source of non linearity using lightweight depthwise convolutions. The 32 filter initial fully linked convolution layer in MobileNetV2 is followed by 19 further bottleneck layers. There are a total of 16 and 19 layers available, respectively, according to the numbers in VGG 16 and VGG 19. The architecture for VGG 16 and VGG 19 is depicted in, respectively,

The first two layers employ a convolution layer with 3X3 filters, and in these two levels, the convolution layer employs 64 filters, resulting in a final product of 224X224X64. 2conv layers and 64 filters are used in the (CONV64) expression. The filters employ the same convolutions and are 3 3 with a stride of 1. This architecture's pooling layer will lower a volume's height and breadth from 224 x 224 x 64 to 112 x 112 x 64. Next, employ a further two convolutional layers with 128 filters. A pooling layer is included, resulting in a new dimension of 56 X 56 X 128. 256 filters were applied to 2conv layers. And last, 7 X 7 X 512 into a Fully Connected Layer (FC) with 4096 units, obtaining one of 1000 classes in a softmax output. A unique type of network known as a residual neural network has been

developed in artificial neural networks. The residual network frequently makes use of skip connections. Another name for skip connections is shortcut jump over layer.

```
#Since our dataset is catagorical we use class names as a label
#This code will list class names
class_names = train_dataset.class_names
plt.figure(figsize=(10, 10))
for images, labels in train_dataset.take(1):
  for i in range(9):
    ax = plt.subplot(3, 3, i + 1)
    plt.imshow(images[i].numpy().astype("uint8"))
    plt.title(class_names[labels[i]])
    plt.axis("off")
```

Figure 4-11 Sample for collected dataset.(a)

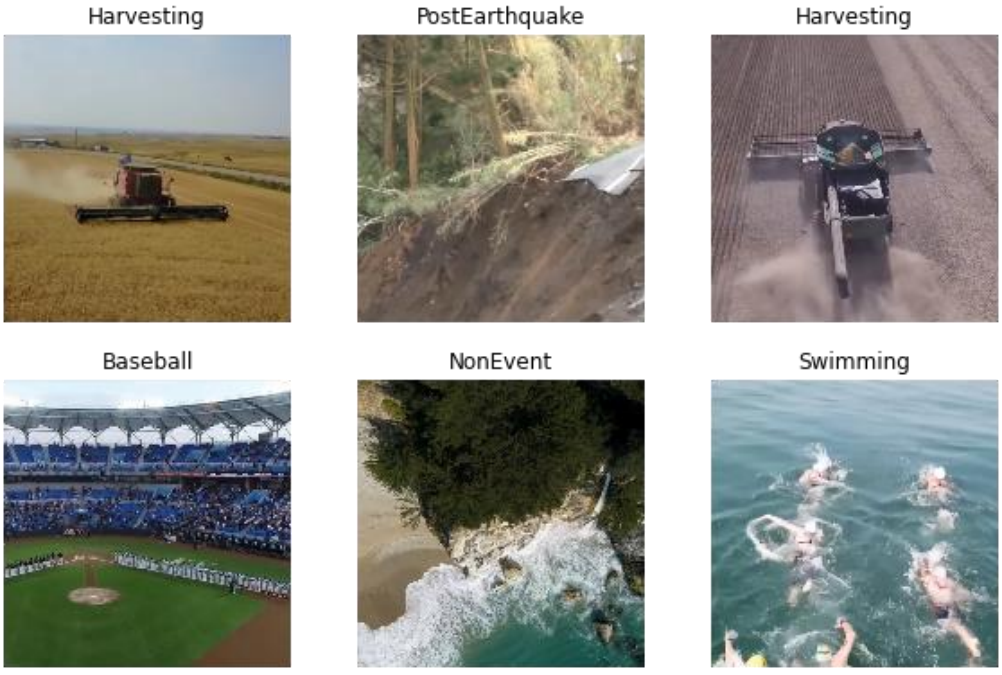


Figure 4-12 Sample for collected dataset.(b)

4.6 Models for Single-Frame Analysis.

First, we discuss single-frame classification models, which only use one video frame—in this case, the middle frame—as the input to networks. The following are the single-frame pre-trained models in use: ResNet-50(Szegedy et al. 2017), ResNet-101(Szegedy et al. 2017), ResNet-152(Szegedy et al. 2017), MobileNet(Howard et al. 2017), DenseNet-121(Huang et al. 2016), DenseNet-169(Huang et al. 2016), DenseNet-201(Huang et al. 2016), and NASNet-L(Zoph et al. 2018) are some examples of ResNet models.

4.6.1 Overview of VGGNet

The Visual Geometry Group, which is a part of Oxford University's Department of Science and Engineering, goes by the full moniker VGG. From VGG16 to VGG19, a succession of convolutional network models that can be used for face recognition and picture classification have been made available. Understanding how the depth of convolutional networks impacts the precision and precision of large-scale picture categorization and recognition was the original goal of VGG's study on convolutional network depth. CNN (Deep-16) , a small 3x3 convolution kernel is used in all layers in order to reduce the number of network layers and avoid using too many parameters.

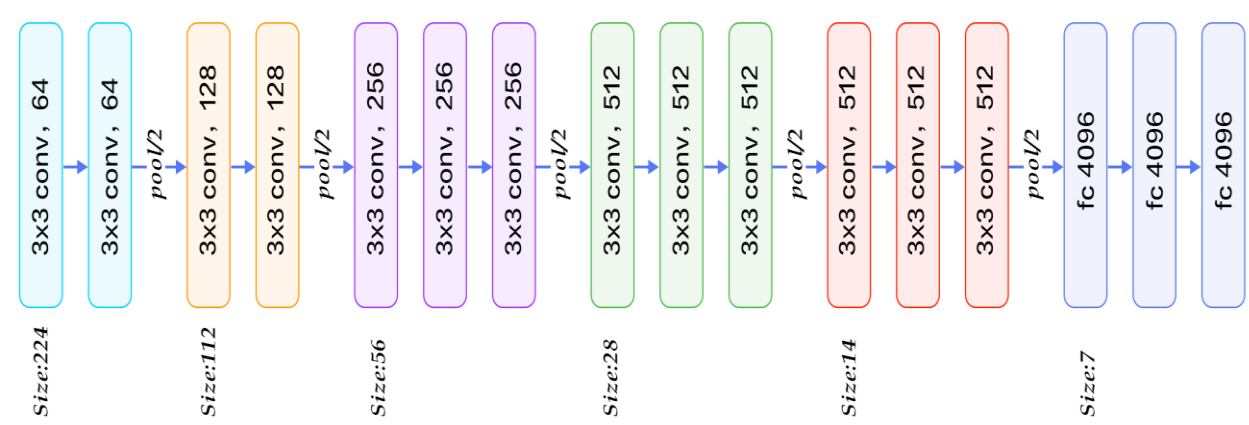
4.6.2 The network structure

A 224x244 RGB size image is set as the VGG's input. The training set image's average RGB value is determined for each image, and the image is then fed into the VGG convolution network. The convolution step is fixed and a 3x3 or 1x1 filter is employed. There are three VGG fully connected layers, ranging in value from VGG11 to VGG19 depending on how many convolutional and fully connected layers are present overall. 8 convolutional layers and 3 fully connected layers make up the minimum VGG11 architecture. There are 16 convolutional layers in the maximal VGG19. +3 completely interconnected layers The VGG network also does not have a pooling layer, a total of 5 pooling layers, spread under the various convolutional layers, behind each convolutional layer. VGG19 has 19 layers, while VGG16 has 16 layers. The final three fully connected layers contain several identical VGGs. Five sets of convolutional layers make up the overall structure, which is followed by a MaxPool. The five sets of convolutional layers differ in that an increasing number of cascaded convolutional layers are present.

	Layer	Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	224 x 224 x 3	-	-	-
1	2 X Convolution	64	224 x 224 x 64	3x3	1	relu
	Max Pooling	64	112 x 112 x 64	3x3	2	relu
3	2 X Convolution	128	112 x 112 x 128	3x3	1	relu
	Max Pooling	128	56 x 56 x 128	3x3	2	relu
5	2 X Convolution	256	56 x 56 x 256	3x3	1	relu
	Max Pooling	256	28 x 28 x 256	3x3	2	relu
7	3 X Convolution	512	28 x 28 x 512	3x3	1	relu
	Max Pooling	512	14 x 14 x 512	3x3	2	relu
10	3 X Convolution	512	14 x 14 x 512	3x3	1	relu
	Max Pooling	512	7 x 7 x 512	3x3	2	relu
13	FC	-	25088	-	-	relu
14	FC	-	4096	-	-	relu
15	FC	-	4096	-	-	relu
Output	FC	-	32	-	-	Softmax

Figure 4-13 Fine tuned vgg16 summary table

The convolution kernel in AlexNet is 7 * 7 in size, and each convolutional layer only comprises one convolution. Each convolution layer in the VGGNet has two to four convolution processes. The pooling kernel is 2 * 2, the step size is 2, the pooling kernel is 3 * 3, and the convolution step size is 1. Reducing the size of the convolution kernel and increasing the quantity of convolution layers is the most evident improvement to VGGNet.



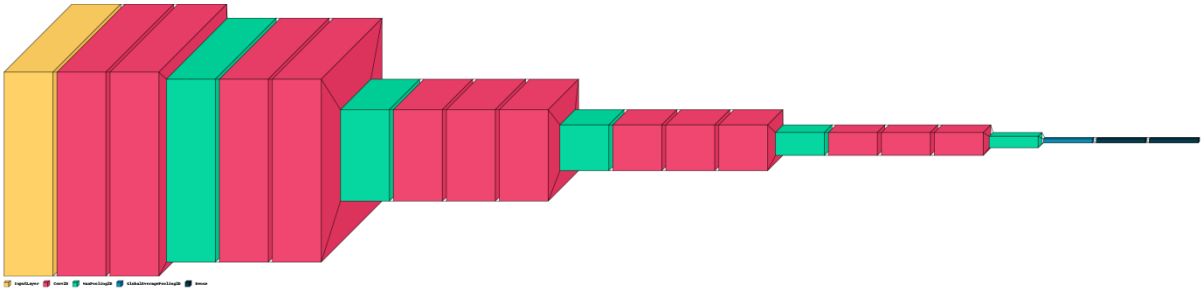


Figure 4-14 Fine Tuned Block Diagram of Vgg16

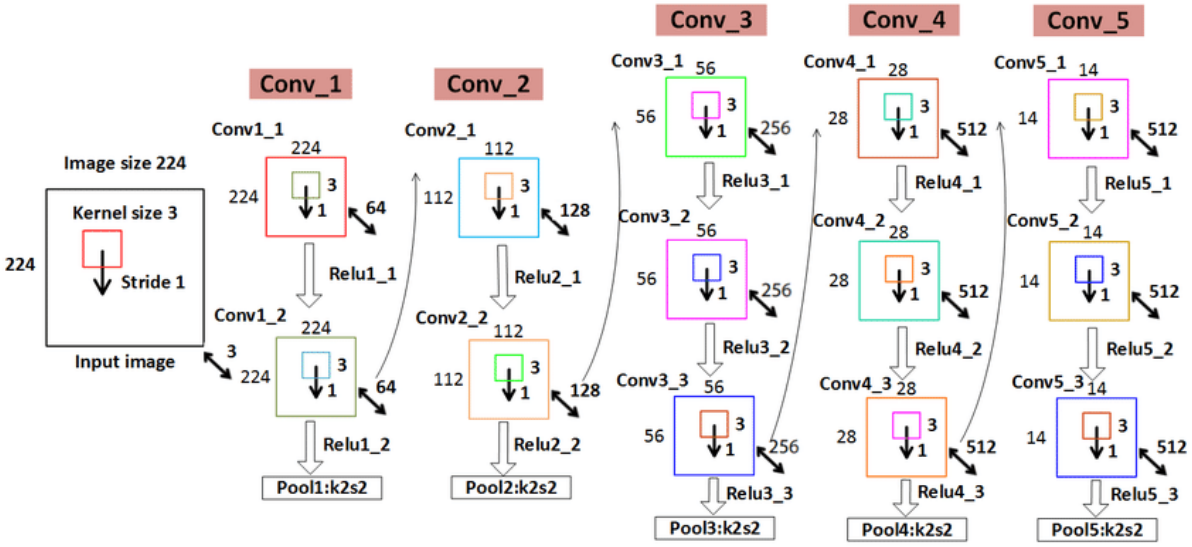


Figure 4-15 Detail process of convolutional layer

A 5 * 5 receptive field is similar to two successive 3 * 3 convolutions, and a 7 * 7 receptive field to three. The decision function is made more discriminative by employing three ReLu layers rather than one when using three 3 * 3 convolutions instead of one 7 * 7 convolution, in addition to decreasing parameters. The input and output, for instance, are all C channels. While 1 convolutional layer using 7 * 7 requires 7 * 7 * C * C, 3 convolutional layers using 3 * 3 require 3 (3 * 3 * C * C) which equals 27 * C * C. This can be thought of as decomposing the 7 * 7 convolution into three 3 * 3 convolutions through some sort of regularization.

The primary purpose of the 1 * 1 convolution layer is to make the decision function more non-linear while maintaining the convolution layer's receptive field. ReLu introduces nonlinearity even though the 1 * 1 convolution procedure is linear.

4.7 Training

A stochastic gradient descent SGD + momentum (0.9) with momentum is used for optimization. A 256-piece batch is used.

Regularization: Weight decay is 5e-4 and L2 regularization is employed. Dropout occurs at p = 0.5, after the first two fully connected layers.

We hypothesize that the VGGNet can converge in less cycles despite being deeper and having more parameters than the AlexNet network for two reasons: first, the deeper structure and smaller convolutions introduce implicit regularization; and second, there are certain layers of pre-training.

Initialization of parameters: For a shallow A network, parameters are initialized at random, the bias is initialized to 0, and the weight w is sampled from the range N (0, 0.01).

The first three fully connected layers and the first four convolutional layers are initialized with the A network's settings for deeper networks. Later on, it was shown that it could also be initialized directly without the use of previously trained parameters.

In each SGD iteration, each rescaled image is randomly cropped to produce an input image with a resolution of 224 * 224. The cropped image is also randomly rotated horizontally and RGB color shifted to improve the data set.

4.7.1 Summary of VGGNet improvement points

1. A shallower network and a smaller 3 * 3 convolution kernel are used. In comparison to the field of vision of a 5 * 5 convolution kernel, the stack of two 3 * 3 convolution kernels and The field of view of a 7 * 7 convolution kernel is equal to the stack of three 3 * 3 convolution kernels. This allows for the possibility of having fewer parameters (3 stacked 3 * 3 structures have only 7 * 7 structural parameters (3 * 3 * 3 / (7 * 7) = 55%); nonetheless, they have more parameters. The non-linear transformation improves CNN's capacity for feature learning.

2. A 1 * 1 convolution kernel is added to the VGGNet's convolutional structure. Non-linear transformation is used to boost the network's expressive power and decrease computation, without changing the input or output dimensions.

3. To hasten the convergence of training, first train a basic (low-level) VGGNet A-level network, and then initialise the subsequent sophisticated models using the weights of the A network.

4.8 Models for analysis of video.

In order for these models to learn temporal information from movies, they require many video frames as input. The following is a summary of the many video categorization models.

C3D[1]. For our tests, we used the Sport1M dataset and the UCF101 dataset to train two C3D4 networks with pre-trained weights (see C3Dy and C3Dz in Table III, respectively).

ResNet P3D[2]. On the Kinetics dataset and the Kinetics-600 dataset, we train two 199-layer P3D ResNet5 models (P3D-ResNet-199) with pre-trained weights (see P3Dy- ResNet-199 and P3Dz-ResNet-199 in Table III, respectively).

I3D[3]. We train two I3D6 models whose backbones are both Inception-v1 (I3D-Inception-v1) with pre-trained weights on the Kinetics dataset and Kinetics+ImageNet, respectively, to evaluate the performance of I3D on our dataset (see I3Dy-Inception-v1 and I3Dz-Inception-v1 in Table III).

TRN[4], in our experiments, we choose the Inception architecture as the framework and train TRNs7 with 16 multi-scale interactions. In particular, we test two Inception architectural variations: BNInception and Inception-v3.

With pre-trained weights from the Something-Something V2 dataset (TRNy-BNInception in Table III) for the first, and the Moments in Time dataset for the second, we initialise the models (TRNz-Inception-v3 in Table III).

4.9 Summary

We designed and trained new model with 3 convolutional layer, 3 Max_pooling layer, 1 Flatten layer with 2 dense layer, the previously trained model is reused through transfer learning. The information learned from the prior task is used in transfer learning. The three areas are image classification, image prediction, and natural language processing where transfer learning is most frequently applied. For the dataset we collected, we used transfer learning in this study to offer target identification and event analysis. The transfer learning models employed in this study are ResNet-50(Szegedy et al. 2017), ResNet-101(Szegedy et al. 2017), ResNet-152(Szegedy et al. 2017), MobileNet(Howard et al. 2017), DenseNet-121(Huang et al. 2016), DenseNet-169(Huang et al. 2016), DenseNet-201(Huang et al. 2016), and NASNet-L(Zoph et al. 2018) are some examples of ResNet models used for single frame. Google Colab notebook has been utilised for the image classification and prediction. The findings are tested on a Google Colab notebook because the system's performance is dependent on the GPU system.

[1] https://github.com/tqvinhcs/C3D-tensorflow

[2] https://github.com/zzy123abc/p3d

[3] https://github.com/LossNAN/I3D-Tensorflow

[4] https://github.com/metalbubble/TRN-pytorch

The number of parameters used by MobileNetV2 is lower than that of other trained models. When compared to other trained models using the ImageNet dataset, ResNet50's accuracy is higher. The paper's restriction is that only transfer learning from MobileNet, MobileNetV2, VGG16, VGG19, and ResNet50 has been used to determine the accuracy of transfer learning. The outcome demonstrates that MobileNetV2 performance is generally superior than other pre-trained models, and it also requires less disc space. Transfer learning may be utilised in the to improve and achieve the best accuracy.

CHAPTER FIVE:

5 EXPERIMENTATION AND IMPLIMENTATION

5.1 Introduction

The implementation and experimentation of the proposed solution will be thoroughly covered in this chapter. To solve the issue and respond to the research questions posed in chapters one, sections 1.3 and 1.4, several experiments have been conducted. The initial set of tests sought to determine the best method for gathering data. The produced dataset would then be imported into the Tensorflow and made available for experimentation. We have been creating data that is appropriate for deep learning models using Python code and a variety of data pretreatment methods. In the last batch of tests, the chosen deep learning model will be trained using chosen feature extraction, and its performance will be assessed using model performance assessment.

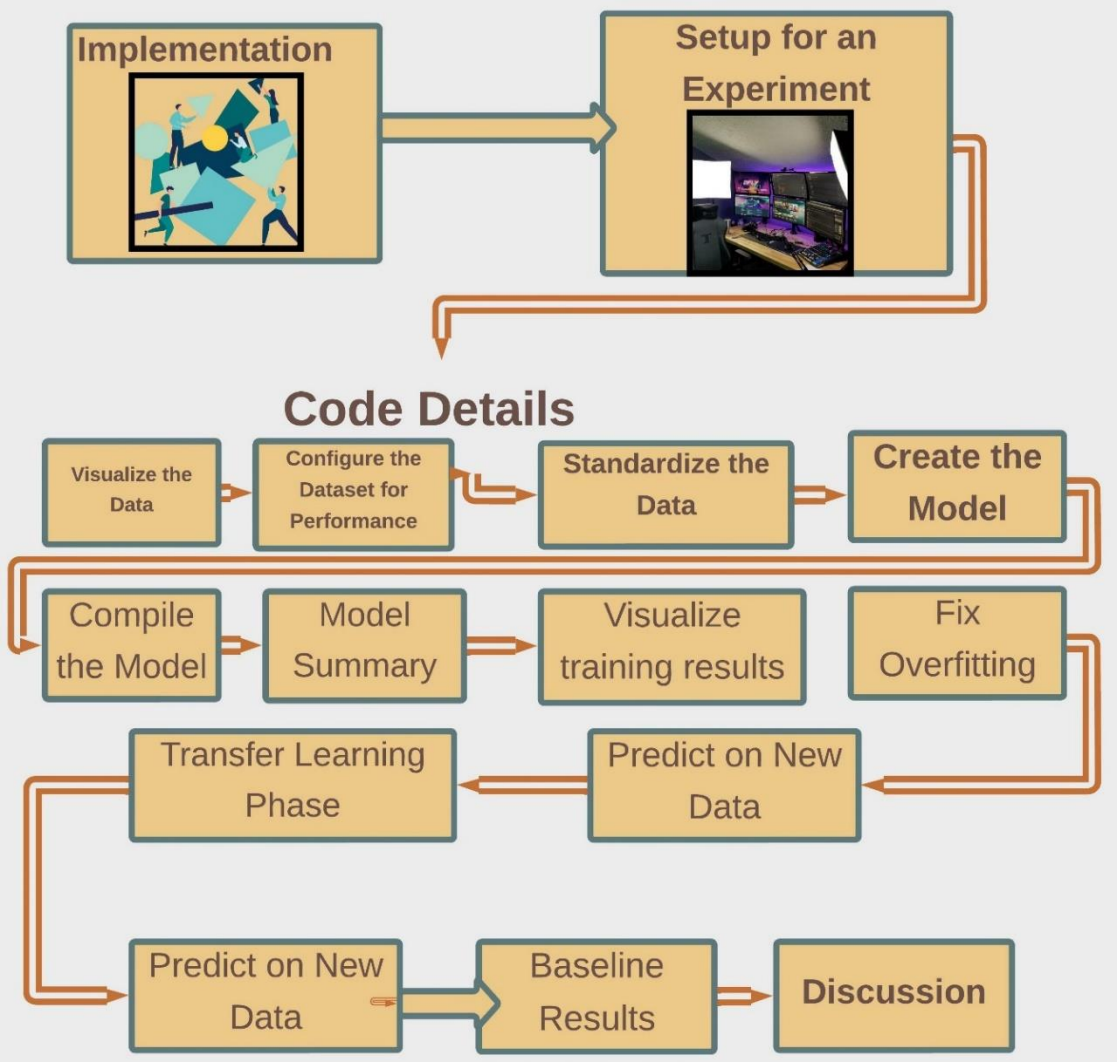


Figure 5-1 over all block diagram of experiment and implementation

5.2 Implementation

In this study, we implemented the suggested solution using a variety of development tools and packages. We developed our model using the Python programming language, as we covered in chapter three section 4.3. Data analysis and visualization were done using Pandas and mat-plot-lib.

5.3 Setup for an Experiment

Data and assessment metrics for the experimental setup. We abide by the following two guidelines regarding the division of training and test sets: 1) Movies taken from the same lengthy film are allocated to the same set, and 2) there should be about equal numbers of training and test videos for each class. This split technique can assess a model's ability to generalize because video clips from the same lengthy movie typically have comparable attributes (such as backdrop, lighting, and resolution). In our dataset, we have included our

training/test split. Displays the statistics of the training and test samples. 10% of the training instances are chosen at random to serve as the validation set throughout the training phase. We employ per-class precision and total accuracy as assessment criteria to evaluate models thoroughly.

5.4 Code Details

Utilize a Keras utility to load dataset uploaded to google drive in zip type from google collaborator. Using the useful tf.keras.utils.image dataset from directory tool, we load these photos from disc. This will lead us from a disk-based image directory to a tf.data. Using only a few lines of code, a dataset. We used a training-validation split, using 80% of the photos for training and 20% for validation.

Set up the loader's parameters as follows:

```
import matplotlib.pyplot as plt
import zipfile
import pathlib
zipreff=zipfile.ZipFile('/content/drive/MyDrive/Dataset1.zip')
zip_data=zipreff.extractall('sample_data/Lend1')
data_dir ='sample_data/Lend1/SingleFrames/Tra'
val_dir='sample_data/Lend1/SingleFrames/Test'
```

```
data_dir = pathlib.Path(data_dir)
val_dir = pathlib.Path(val_dir)
image_count = len(list(data_dir.glob('*/*.png')))
batch_size = 32
img_height = 224
img_width = 224
train_ds = tf.keras.utils.image_dataset_from_directory( data_dir,
image_size=(img_height, img_width),
batch_size=batch_size)
val_ds = tf.keras.utils.image_dataset_from_directory(
val_dir,
image_size=(img_height, img_width),
batch_size=batch_size)
class_names = train_ds.class_names
print(class_names)
```

output

['Baseball', 'Basketball', 'Boating', 'CarRacing', 'Concert', 'Conflict', 'Constructing', 'Cycling', 'Fire', 'Flood', 'Harvesting', 'HeavyWeapon', 'Landslide', 'MilitaryCamp', 'MilitaryDrill ', 'MilitaryShow', 'MilitaryTraining', 'Mudslide', 'NonEvent', 'ParadeProtest', 'Party', 'Ploughing', 'PoliceChase', 'PostEarthquake', 'ReligiousActivity', 'Running', 'Soccer', 'Swimming', 'TankMovement', 'TrafficCollision', 'TrafficCongestion']

5.4.1 Visualize the Data

Here are the first twenty images from the training dataset:

```
#Here are the first 20 images from the training dataset taken
#randomly as a sample
plt.figure(figsize=(10, 10))
for images, labels in train_ds.take(1):
  for i in range(20):
    ax = plt.subplot(4, 5, i + 1)
    plt.imshow(images[i].numpy().astype("uint8"))
    plt.title(class_names[labels[i]])
    plt.axis("off")
```

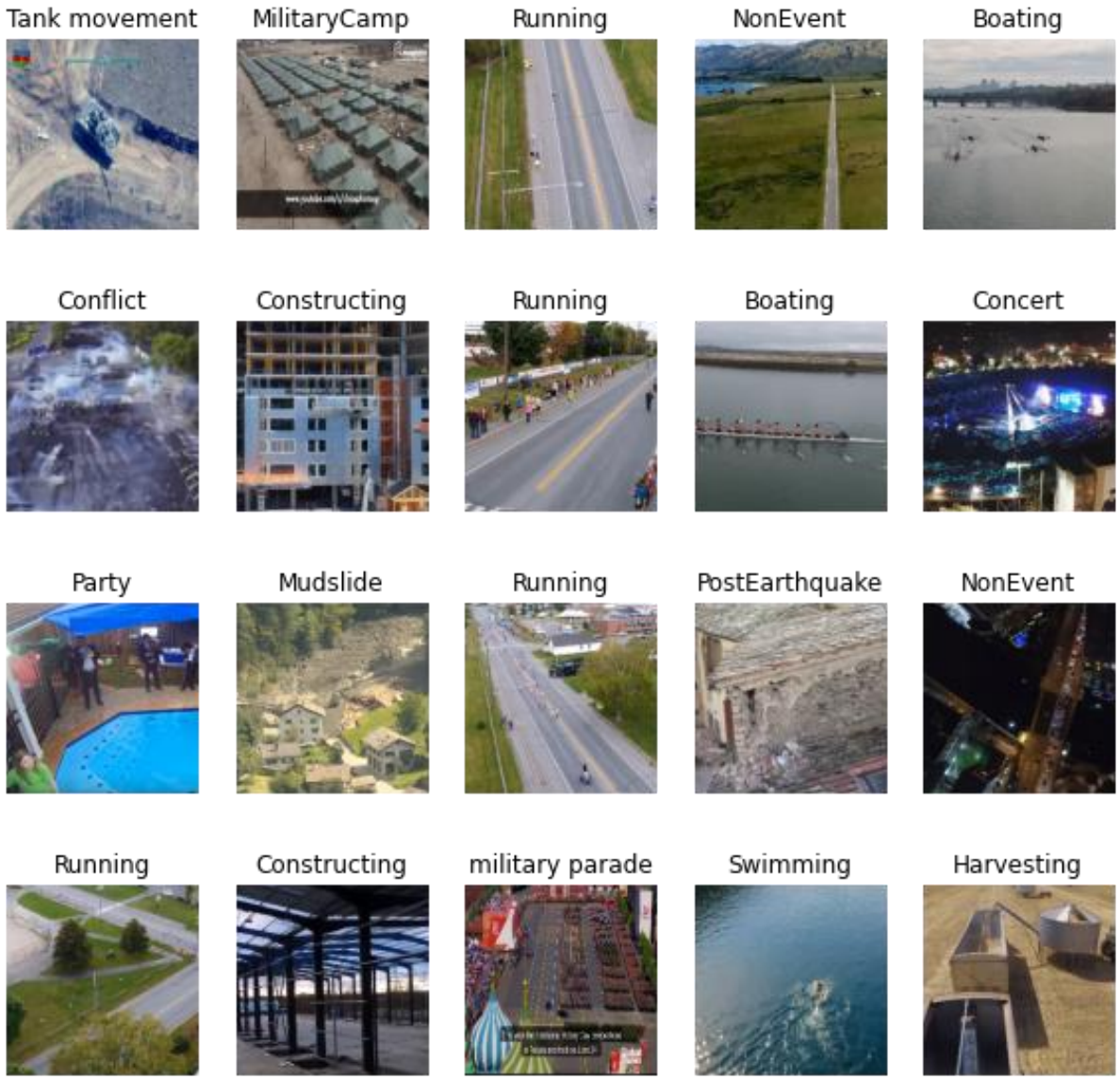


Figure 5-2 Visualize the data

The following code will pass these datasets to the Keras Model.fit method for training later in this study.

```
for image_batch, labels_batch in train_ds:
  print(image_batch.shape)
  print(labels_batch.shape)
  break
```

The image_batch is a tensor of the shape (32, 224, 224, 3). This is a batch of 32 images of shape 224x224x3 (the last dimension refers to color channels RGB). The label_batch is a tensor of the shape (32,), these are corresponding labels to the 32 images.

5.4.2 Configure the Dataset for Performance

To use buffered prefetching, so you can yield data from disk without having I/O become blocking. These are two important methods you should use when loading data:

Dataset.cache keeps the images in memory after they're loaded off disk during the first epoch. This will ensure the dataset does not become a bottleneck while training your model. If your dataset is too large to fit into memory, you can also use this method to create a performant on-disk cache.

5.4.3 Standardize the Data

The RGB channel values are in the [0, 255] range. This is not ideal for a neural network; in general we should seek to make our input values small.

Here, we will standardized values to be in the [0, 1] range by using tf.keras.layers.Rescaling:

```
normalization_layer = layers.Rescaling(1./255)
```

There are two ways to use this layer. We can apply it to the dataset by calling Dataset.map:

```
normalized_ds = train_ds.map(lambda x, y: (normalization_layer(x), y))
image_batch, labels_batch = next(iter(normalized_ds))
first_image = image_batch[0]
# Notice the pixel values are now in `[0,1]`.
print(np.min(first_image), np.max(first_image))
```

5.4.4 Create the Model

The Keras Sequential model consists of three convolution blocks (tf.keras.layers.Conv2D) with a max pooling layer (tf.keras.layers.MaxPooling2D) in each of them. There's a fully-connected layer (tf.keras.layers.Dense) with 128 units on top of it that is activated by a ReLU

activation function ('relu'). This model has not been tuned for high accuracy; the goal of this tutorial is to show a standard approach.

```
num_classes = len(class_names)
model = Sequential([
  layers.Rescaling(1./255, input_shape=(img_height, img_width, 3)),
  layers.Conv2D(16, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(32, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(64, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Flatten(),
  layers.Dense(224, activation='relu'),
  layers.Dense(num_classes)
])
```

5.4.5 Compile the Model

We choose the tf.keras.optimizers.Adam optimizer and tf.keras.losses.Categorical Cross entropy loss function. To view training and validation accuracy for each training epoch, pass the metrics argument to Model.compile.

```
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

5.4.6 Model Summary

View all the layers of the network using the Keras Model.summary method:

```
model.summary()
```

Train the model

Train the model for 10 epochs with the Keras Model.fit method:

```
epochs=10
history = model.fit(
  train_ds,
  validation_data=val_ds,
  epochs=epochs
)
```

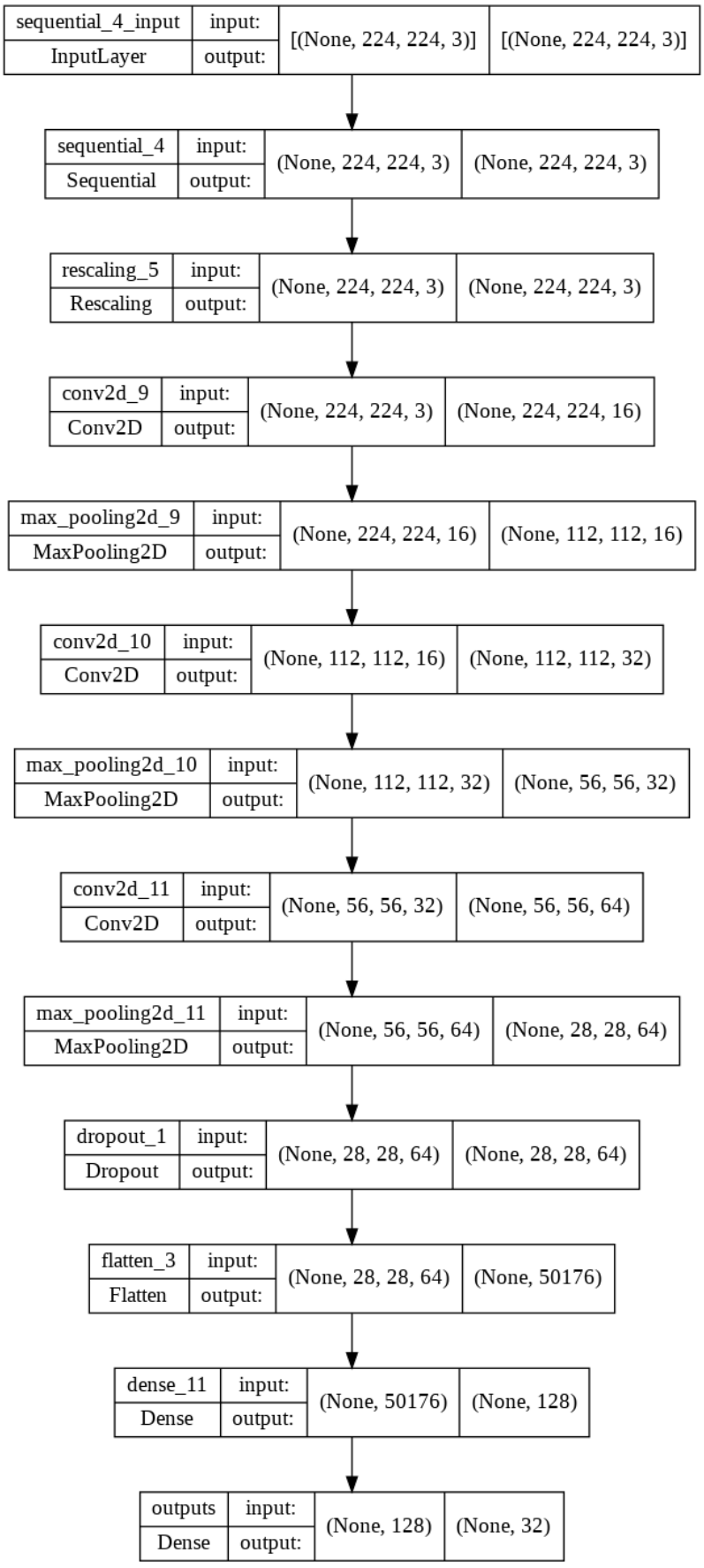


Figure 5-3 our model summary

```
model.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 rescaling_4 (Rescaling)     (None, 224, 224, 3)       0

 conv2d_6 (Conv2D)           (None, 224, 224, 16)      448

 max_pooling2d_6 (MaxPooling  (None, 112, 112, 16)     0
 2D)

 conv2d_7 (Conv2D)           (None, 112, 112, 32)      4640

 max_pooling2d_7 (MaxPooling  (None, 56, 56, 32)       0
 2D)

 conv2d_8 (Conv2D)           (None, 56, 56, 64)        18496

 max_pooling2d_8 (MaxPooling  (None, 28, 28, 64)       0
 2D)

 flatten_2 (Flatten)         (None, 50176)             0

 dense_5 (Dense)             (None, 224)               11239648

 dense_6 (Dense)             (None, 32)                7200

=================================================================
Total params: 11,270,432
Trainable params: 11,270,432
Non-trainable params: 0
_________________________________________________________________
```

Figure 5-4 Model summary

```
[21] epochs=10
     history = model.fit(
       train_ds,
       validation_data=val_ds,
       epochs=epochs,
       callbacks=[tensorboard]
     )
```

```
Epoch 1/10
37/37 [==============================] - 26s 379ms/step - loss: 3.2822 - accuracy: 0.1187 - val_loss: 2.8656 - val_accuracy: 0.1493
Epoch 2/10
37/37 [==============================] - 2s 50ms/step - loss: 2.7777 - accuracy: 0.1934 - val_loss: 2.5149 - val_accuracy: 0.2672
Epoch 3/10
37/37 [==============================] - 2s 45ms/step - loss: 2.4020 - accuracy: 0.3113 - val_loss: 1.9902 - val_accuracy: 0.4792
Epoch 4/10
37/37 [==============================] - 2s 45ms/step - loss: 1.9502 - accuracy: 0.4453 - val_loss: 1.5712 - val_accuracy: 0.5963
Epoch 5/10
37/37 [==============================] - 2s 45ms/step - loss: 1.2410 - accuracy: 0.6514 - val_loss: 1.0452 - val_accuracy: 0.6989
Epoch 6/10
37/37 [==============================] - 2s 45ms/step - loss: 0.7325 - accuracy: 0.7981 - val_loss: 0.3651 - val_accuracy: 0.9118
Epoch 7/10
37/37 [==============================] - 2s 45ms/step - loss: 0.3709 - accuracy: 0.9033 - val_loss: 0.3099 - val_accuracy: 0.9067
Epoch 8/10
37/37 [==============================] - 2s 45ms/step - loss: 0.3012 - accuracy: 0.9296 - val_loss: 0.1945 - val_accuracy: 0.9542
Epoch 9/10
37/37 [==============================] - 2s 45ms/step - loss: 0.1119 - accuracy: 0.9788 - val_loss: 0.0747 - val_accuracy: 0.9796
Epoch 10/10
37/37 [==============================] - 2s 45ms/step - loss: 0.0685 - accuracy: 0.9830 - val_loss: 0.0386 - val_accuracy: 0.9924
```

Figure 5-5 Training result

5.4.7 Visualize training results

Create plots of the loss and accuracy on the training and validation sets:

```
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs_range = range(epochs)
plt.figure(figsize=(8, 8))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, acc, label='Training Accuracy')
plt.plot(epochs_range, val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')
plt.subplot(1, 2, 2)
plt.plot(epochs_range, loss, label='Training Loss')
plt.plot(epochs_range, val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
```

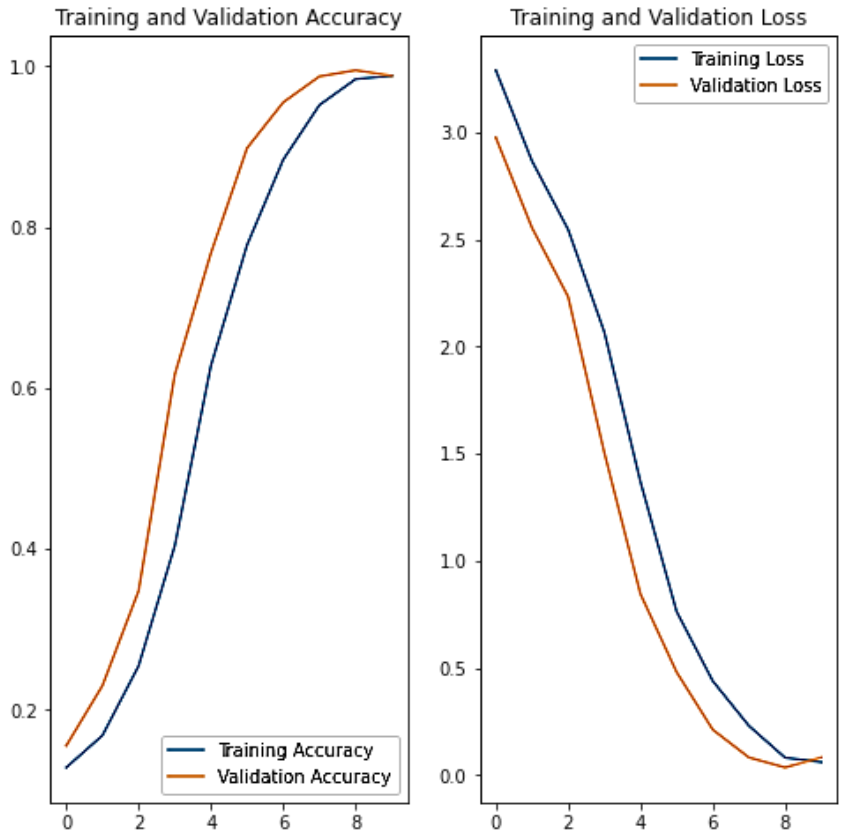


Figure 5-6 Visualize training results

The figures demonstrate that the model has only managed to obtain about 98% accuracy on the validation set, and that the differences between training accuracy and validation accuracy are significant. We get best validation output in our collected dataset, i.e we don't need data augmentation and dropout to solve overfitting.

5.4.8 Overfitting

According to the aforementioned plots, validation accuracy increases linearly over time, whereas training accuracy plateaus at about 98%. Additionally, there is a small change in accuracy between training and validation, which is an indication of there is no overfitting.

When there are few training instances, the model may occasionally pick up on undesirable noises or details, which have a negative effect on how well it performs on future cases. So in our case there is no overfitting problem. It implies that the model will have no trouble generalising to a different dataset. There are several strategies for preventing overfitting during the training phase. By using data augmentation and incorporate dropout into our model, but as we have seen from the plot above we didn't face this problem we are not expected to augment our data.

Using data augmentation and droupout in our scenario have negative impact.

5.4.9 Predict on New Data

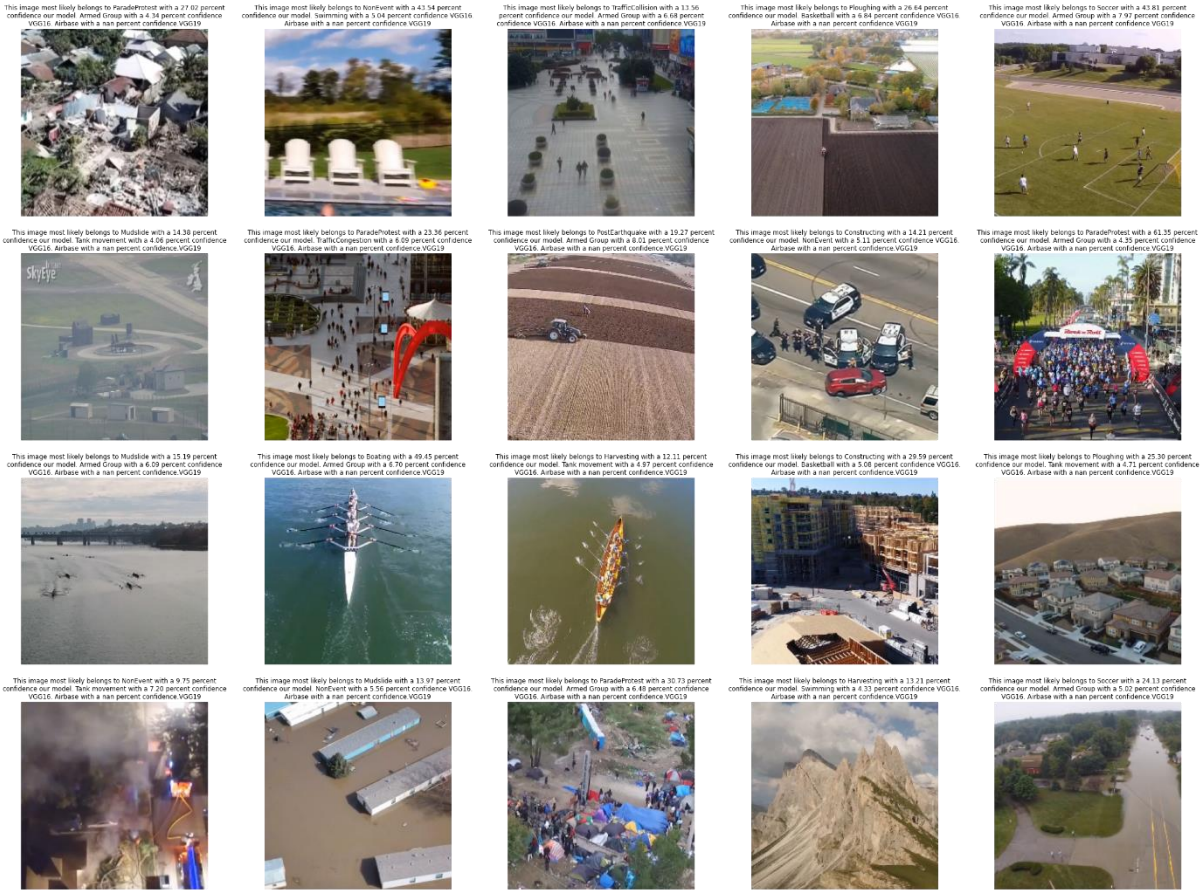


Figure 5-7 Pridiction result in our model

5.4.10Transfer Learning Phase

Only the new fully connected layer was trained using the features taken from the pre-trained models during the Transfer Learning phase.

Import Keras libraries

The pre-trained ImageNet models VGG16, VGG19, ResNet50, InceptionV3 and Xception are supported by Keras.

The GlobalAveragePooling2D layer offers an average pooling for spatial data, and the Dense layer is the layer of the densely linked neural network.

SGD (Stochastic Gradient Descent), RMSprop, Adagrad, Adadelta, Adam, Adamax, Nadam, and TFOptimizer are just a few of the optimizers supported by Keras (for Tensorflow).

In order to better manage memory, the ImageDataGenerator module generates batches of tensor image data with real-time data augmentation.

```
# import pertinent libraries
import os
import sys
import datetime
import glob as glob
import numpy as np
import cv2
# [Keras Models]
# import the Keras implementations of VGG16, VGG19, InceptionV
3 and Xception models
# the model used here is VGG16
from keras.applications.vgg16 import VGG16, preprocess_input
from keras.applications.vgg19 import VGG19, preprocess_input
from keras.models import Model
from keras.layers import Dense, GlobalAveragePooling2D
from keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.utils import array_to_img, img_to_array,
 load_img
from tensorflow.keras.optimizers import SGD
import tensorflow
#from scipy.interpolate import make_interp_spline
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

5.5 Baseline Results

Table II and Table III, respectively, present quantitative findings from single-frame classification models and video classification models. As we can see, DenseNet-201 outperforms Inception-v3 by just 0.2% and obtains the greatest performance in the single-frame classification challenge with an OA of 62.3%. TRNz-Inception-v3 excels at the video classification task and achieves an OA of 64.3%. It is intriguing to note that when Table II and Table III are compared, the best-performing event analysis model achieves the highest OA. This shows the value of utilizing temporal cues in event recognition from aerial films. Additionally, we display some predictions for the top two single frame. Inception-v3 and DenseNet-201 classification network architectures, as well as the top two video classification network architectures (I3Dz-Inception-v1 and TRNz-Inception-v3), all baselines with high confidence ratings are able to correctly identify frames and videos with discriminative event-relevant properties, such as backed-up traffic on a highway and smoke billowing from a neighborhood. In addition, TRN's high-scoring predictions for spotting plowing and parade/protest demonstrate how effectively utilizing temporal information aids in

differentiating events with slight interclass differences. Further, we aware that extreme weather conditions can cause classification errors nighttime and snowy scene frames and movies.

5.6 Discussion

To the best of our knowledge this study is first in our country by studying images and analyzing events from the aerial using deep learning. In our study we considered different military events to analyze it and identify it as a target. This research study has been studied by considering Ethiopian Air force UAV department as abase for the motivation. Throughout this research work we studied different research questions listed under section 1.3 and experimented it by starting from collection of dataset, then select appropriate models for experimentation process by considering different scenarios which listed under section 4.3 that lead us to this discussion part. Here is the discussion for the research questions listed under section 1.3.

RQ1: Can we get any gap from related work studied for image recognition from aerial view?
As we have mentioned in chapter two there are different study has been made which related to this research study from which we get the gap and worked on it. Unlike the dataset of ImageNet in computer vision there is no much datasets for training the model abundantly.
RQ2: Is it possible to identify events from an aerial view image (by UAV) using deep learning models? To answer this question we used pre-trained models and build our sequential neural network model to identify 32 events from aerial view images using deep learning models like vgg16, vgg19, Inception-v3, resenet50 101 152, mobilenet, Densenet, and nasnet. The result obtained from this study show that possibility of identification of targets recorded from aerial. Even though there is possibility to identify targets from aerial scene this area need more study because of the nature of the image gained from the aerial view, which contains much information as its angle get wide and wider.
RQ3: How can we extract targets from events seen from aerial? For extracting the target first we need to give inputs as the required target. After accepting the required target our model starts to analyze events and classify them into its category and check if the target we need is available and give some message if the target is identified.
RQ4: What is the best result of pre-trained models by implementing transfer learning? by Training pre-trained model with our dataset we got result as show in the Table II Performance of Single-Frame Classification Models Test SetTable II and Table III which we marked the best result by bold font.

Table II Performance of Single-Frame Classification Models Test Set

Models	post-earthquake	flood	fire	landslide	mudslide	traffic collision	traffic congestion	harvesting	ploughing	constructing	police chase	conflict	baseball	basketball	boating	cycling	running	soccer	swimming	car racing	party	concert	parade/protest	religious activity	Armed Group	Artillery howitzer	military parade	military Vehicle	Military Camp	Tank movement	non-event	OA
VGG-16	46.3	59.1	53.6	38.8	56.1	30.8	**76.2**	62.7	65.4	69	70	44.4	61	56.2	69.4	39.6	33.3	**87.5**	62	32	**73.5**	56.7	47.8	64.6	50	57.1	58.1	65.2	**80.9**	7.4	30.4	51.9
VGG-19	45.5	56.4	70.2	48.1	47.1	33.3	50	57.1	58.1	65.2	**80.9**	7.4	66.7	55.9	66.7	35.8	57.1	67.3	55.6	26.7	53.3	54.4	43.6	50	55.4	64.6	77.3	73.7	76.5	50	31.1	49.7
Inception-v3	62.9	76.1	**88**	44.7	54.7	**48**	55.4	64.6	77.3	73.7	76.5	50	72	61.2	73.7	70.2	**90**	80	61.7	60	66.7	47.7	52.2	62.2	50	**77.4**	72.9	63.8	68.6	62.5	45.5	62.1
ResNet-50	65.5	69.8	77.4	40	51.9	40.6	50	**77.4**	72.9	63.8	68.6	**62.5**	83.3	52.2	71.4	77.4	28.6	73.5	54.3	50	61.5	49.4	46	48.9	48.7	65.8	78	69.5	64.6	55	38.9	57.3
ResNet-101	59.6	82.9	79.2	34.5	43.8	18.8	48.7	65.8	78	69.5	64.6	55	76.1	57.7	82.2	**90.5**	61.5	73.3	58.2	31.6	51.2	49.5	47.1	**64.7**	58.5	61.9	75.6	58	59.3	57.1	36.2	55.3
ResNet-152	67.3	68.2	78.8	45.2	46.4	38.9	58.5	61.9	75.6	58	59.3	57.1	79.5	56.9	77.8	63.4	75	74.4	56.1	30.8	61.9	44.7	48.6	52.8	52.6	66.2	66.7	67.2	70.6	50	37	56.1
MobileNet	**72**	70.8	78	**57.5**	**61**	43.6	52.6	66.2	66.7	67.2	70.6	50	74.5	59.7	76.4	54.7	72	64.8	52.9	56.2	65	44.4	54.5	61.5	58.2	71.1	78	70.2	73.5	48	52.5	61.3
DenseNet-121	58.6	71.4	82.8	54.5	51.6	38.1	58.2	71.1	78	70.2	73.5	48	85	68.4	**86.7**	65.3	57.1	75.4	61.7	52.9	68.3	52.3	**66.7**	47.8	59.5	71.6	87.2	**80.4**	76.6	53.8	43.3	61.7
DenseNet-169	70	**82.9**	71.9	45.2	40.2	36.7	59.5	71.6	**87.2**	**80.4**	76.6	53.8	**91.4**	65	67.7	76.9	63.6	75	**63.2**	57.1	59.1	60	55.4	60.9	62.3	71.6	**85.4**	71.2	77.1	47.1	39.7	60.6
DenseNet-201	69.9	80.4	84.5	52.2	48.1	43.2	62.3	71.6	85.4	71.2	77.1	47.1	87.8	63.6	79.6	69.8	47.8	65	58	43.8	61	**60.9**	55	60.8	78	57.5	61	43.6	52.6	66.2	42.1	**62.3**
NASNet-L	60	50	77.2	41	50.9	46.9	50	68	77.8	82.7	78	61.5	82.6	**74.5**	78	75	62.2	69	54.5	**70**	69.2	44.6	58.7	55.9	**82.8**	54.5	51.6	38.1	58.2	**71.1**	41.7	60.2

Table III Performance of Video Classification Models: Test Set

Model	post-earthquake	flood	landslide	mudslide	traffic collision	traffic congestion	harvesting	ploughing	constructing	police chase	conflict	baseball	basketball	boating	cycling	running	soccer	swimming	car racing	party	concert	parade/protest	religious activity	Armed Group	Artillery howitzer	military parade	military vehicle	Military Camp	Tank movement	non-event	fire	OA
C3D+	23.1	24.3	19.5	32.9	7	15.5	27.5	36.1	45.5	50	18.2	40.9	37	47.5	20.6	12	58.3	36.2	16.7	25.8	38.2	37.8	27.5	38.5	42.3	31.1	40	51.9	11.1	29.6	30.9	30.4
C3D++	27.9	56.5	10.2	23.9	8.3	38.5	42.3	31.1	40	51.9	11.1	45.7	48.9	41.9	13.6	9.3	41.9	38.2	18.2	17.4	32	28.1	35.8	37.8	**77.4**	70.8	62	**81.6**	22.2	28.5	32.7	31.1
P3D ResNet-199	43.6	65.9	35.5	48.7	20	37.8	77.4	70.8	62	**81.6**	22.2	66.7	63.1	55.4	35.6	35.3	76.2	57.4	40	54.5	37.5	38.7	47.8	40.8	56.9	67.4	**71.4**	57.9	50	37.4	66.7	50.7
P3D ResNet-199	72.4	76.3	24.5	38.2	35.6	40.8	56.9	67.4	**71.4**	57.9	50	70.4	**78.8**	71.7	47.1	60	79.5	68.1	40.9	59.1	37	**49.1**	55.9	55	61.5	50	53.3	73.2	50	37.9	84.8	53.3
I3D Inception-v1	40.4	63.5	22.6	46.3	17.6	55	61.5	50	53.3	73.2	50	75	69.4	60.7	61.9	53.3	70.8	52.5	50	57.1	**50.7**	40.3	49	52.2	67.1	66.7	54.2	64.8	57.9	35.8	68.9	51.3
I3D Inception-v1	60	68.1	29	**60.4**	**51.5**	52.2	67.1	66.7	54.2	64.8	**57.9**	**85**	61.9	**86.4**	**75**	44.4	77.6	64.1	65.2	53.7	50	47.8	65.1	**66.7**	68.1	**77.4**	52.4	70.5	46.8	43	65.7	58.5
TRN BNInception	**84.8**	71.4	51.2	50	46.8	**66.7**	68.1	77.4	52.4	70.5	46.8	64.5	67.7	84	56.1	55.2	83.3	**72.9**	61.1	62	48.9	44.6	62.8	24.5	38.2	35.6	40.8	56.9	**67.4**	**51.1**	82.5	62
TRN BNInception	69.2	**87.8**	**65.8**	60	44.1	58.3	**78.1**	**90.7**	70.8	73.3	28.6	83.3	72.7	73.7	60	**66.7**	73.6	70.6	**63.6**	**65.1**	47.7	42.7	**65.1**	22.6	46.3	17.6	55	61.5	50	47.9	**88.9**	**64.3**

CONCLUSIONS AND FUTURE WORK RECOMMENDATION

Conclusion

For completely identifying events in the real world from UAV recordings, we utilized a dataset called TIEA. The events in the TIEA dataset span a variety of sizes and contexts. Numerous deep network findings are reported using two different methods: single-frame classification and video classification. The experimental findings indicate that this is a challenging problem for the area of remote sensing, and the suggested dataset presents a fresh obstacle to the development of models that can comprehend what occurs on the globe from an aerial perspective. Our dataset has the potential to be used in further tasks using current or new annotations in the future, including video retrieval, multi-attribute learning for comprehending aerial events, and temporal event localization in lengthy movies. We point out that this study could benefit the field of computer vision as well.

Future work recommendation

Even while these research study have been successful, there are still some difficult circumstances which left for future work. These instances have in common the difficulty in recognizing event-relevant characteristics, such as human activities, which leads to failures to notice these occurrences. In conclusion, event detection in aerial video is still a difficult task that might be made easier by better understanding discriminative features and utilizing temporal signals. We would like to suggest that as a future work in this field, despite being the largest dataset for event analyzing in aerial videos to our knowledge, the TIEA dataset is still somewhat small in comparison to other video classification datasets used in computer vision. Consequently, the model training process faces the little data problem.

The issue of developing unbiased models on an unbalanced dataset is brought on by the lopsided distribution across distinct classes.

This dataset has high intra-class variance and inter-class similarity because events occur in diverse contexts and are recorded at varied sizes.

REFERENCE

A Bhardwaj, L. Sam, F. Akanksha, J. Martín-Torres, R. Kumar. 2016. "UAVs as Remote Sensing Platform in Glaciology: Present Applications and Future Prospects." *Remote Sens Environ* 175:196–204.

Allibhai, Eijaz. n.d. "Hold-out vs. Cross-Validation in Machine Learning." Retrieved July 21, 2021 (https://medium.com/@eijaz/holdout-vs-cross-validation-in-machine-learning-7637112d3f8f).

Anon. n.d.-a. "Deep Learning Model - an Overview | ScienceDirect Topics." Retrieved May 3, 2022 (https://www.sciencedirect.com/topics/computer-science/deep-learning-model).

Anon. n.d.-b. "Learning-Based Collision-Free Coordination for a Team of Uncertain Quadrotor UAVs - ScienceDirect." Retrieved April 30, 2022 (https://www.sciencedirect.com/science/article/abs/pii/S1270963821006374).

Anon. n.d.-c. "Recurrent Neural Networks - ScienceDirect." Retrieved August 2, 2022 (https://www.sciencedirect.com/science/article/pii/B9780323857871000105).

Anon. n.d.-d. "Welcome To Colaboratory - Colaboratory." Retrieved August 2, 2022 (https://colab.research.google.com/?utm_source=scs-index).

Barekatain, Mohammadamin, Miquel Marti, Hsueh Fu Shih, Samuel Murray, Kotaro Nakayama, Yutaka Matsuo, and Helmut Prendinger. 2017. "Okutama-Action: An Aerial View Video Dataset for Concurrent Human Action Detection." *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* 2017-July:2153–60. doi: 10.48550/arxiv.1706.03038.

Bozcan, Ilker, and Erdal Kayacan. 2020. "AU-AIR: A Multi-Modal Unmanned Aerial Vehicle Dataset for Low Altitude Traffic Surveillance." *Proceedings - IEEE International Conference on Robotics and Automation* 8504–10. doi: 10.48550/arxiv.2001.11737.

Brownlee, Jason. n.d. "K-Fold Cross-Validation." Retrieved July 30, 2021 (https://machinelearningmastery.com/k-fold-cross-validation/).

C Gomez, H. Purdie. 2016. “UAV-Based Photogrammetry and Geocomputing for Hazards and Disaster Risk Monitoring – a Review.” *Geoenvironmental Disasters* 3:1–11.

Deng, Li, and Dong Yu. 2013a. “Deep Learning: Methods and Applications.” *Foundations and Trends in Signal Processing* 7(3–4):197–387. doi: 10.1561/2000000039.

Deng, Li, and Dong Yu. 2013b. “Deep Learning: Methods and Applications.” *Foundations and Trends in Signal Processing* 7(3–4):197–387. doi: 10.1561/2000000039.

F Chiabrando, AM Lingua, M. Piras. 2013. “Direct Photogrammetry Using UAV: Tests and First Results.” *ISPRS Arch* 1:W2.

F Clapuyt, V. Vanacker, K. Oost. 2016. “Reproducibility of UAV-Based Earth Topography Reconstructions Based on Structure-from-Motion Algorithms.” *Geomorphology* 260:4–15.

Fisher, R. 2004. “The Pets04 Surveillance Ground-Truth Data Sets.” 1–5.

Guo, Yaohua, Gang Chen, and Tao Zhao. 2021. “Learning-Based Collision-Free Coordination for a Team of Uncertain Quadrotor UAVs.” *Aerospace Science and Technology* 119:107127. doi: 10.1016/J.AST.2021.107127.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. “Deep Residual Learning for Image Recognition.” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 2016-December:770–78. doi: 10.1109/CVPR.2016.90.

Howard, Andrew G., Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.”

Huang, Gao, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2016. “Densely Connected Convolutional Networks.” *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017* 2017-January:2261–69. doi: 10.48550/arxiv.1608.06993.

Hussain Mujtaba. n.d. “Types of Cross Validation.” Retrieved July 30, 2021 (https://www.mygreatlearning.com/blog/cross-validation/).

I Colomina, P. Molina. 2014. “Unmanned Aerial Systems for Photogrammetry and Remote Sensing: A Review.” *ISPRS Journal of Photogrammetry and Remote Sensing* 92:79–97.

Khan, Asharul Islam, and Yaseen Al-Mulla. 2019. “Unmanned Aerial Vehicle in the Machine Learning Environment.” *Procedia Computer Science* 160:46–53. doi: 10.1016/J.PROCS.2019.09.442.

Kornblith, Simon, Jonathon Shlens, and Quoc v. Le. 2019. “Do Better ImageNet Models Transfer Better?” 2661–71.

Kyrkou, Christos. 2020. “AIDER (Aerial Image Dataset for Emergency Response Applications).” doi: 10.5281/ZENODO.3888300.

Kyrkou, Christos, and Theocharis Theocharides. 2019a. “Deep-Learning-Based Aerial Image Classification for Emergency Response Applications Using Unmanned Aerial Vehicles.” *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* 2019-June:517–25. doi: 10.48550/arxiv.1906.08716.

Kyrkou, Christos, and Theocharis Theocharides. 2019b. “Deep-Learning-Based Aerial Image Classification for Emergency Response Applications Using Unmanned Aerial Vehicles.”

Lecun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. “Deep Learning.” *Nature* 521(7553):436–44. doi: 10.1038/NATURE14539.

Leira, Frederik S., Håkon Hagen Helgesen, Tor Arne Johansen, and Thor I. Fossen. 2021. “Object Detection, Recognition, and Tracking from UAVs Using a Thermal Camera.” *Journal of Field Robotics* 38(2):242–67. doi: 10.1002/ROB.21985.

Li, Jing, Dong Hye Ye, Timothy Chung, Mathias Kolsch, Juan Wachs, and Charles Bouman. 2016. “Multi-Target Detection and Tracking from a Single Camera in Unmanned Aerial Vehicles (UAVs).” *IEEE International Conference on Intelligent Robots and Systems* 2016-November:4992–97. doi: 10.1109/IROS.2016.7759733.

Liu, Kang, and Gellert Mattyus. 2015. “Fast Multiclass Vehicle Detection on Aerial Images.” *IEEE Geoscience and Remote Sensing Letters* 12(9):1938–42. doi: 10.1109/LGRS.2015.2439517.

Mandapati, Sridhar, Seifedine Kadry, R. Lakshmana Kumar, Krongkarn Sutham, and Orawit Thinnukool. 2022. “Deep Learning Model Construction for a Semi-Supervised Classification with Feature Learning.” *Complex & Intelligent Systems*. doi: 10.1007/S40747-022-00641-9/TABLES/3.

M, Hossin, and Sulaiman M.N. 2015. “A Review on Evaluation Metrics for Data Classification Evaluations.” *International Journal of Data Mining & Knowledge Management Process* 5(2):01–11. doi: 10.5121/ijdkp.2015.5201.

Mou, Lichao, Yuansheng Hua, Pu Jin, and Xiao Xiang Zhu. 2020. “ERA: A Dataset and Deep Learning Benchmark for Event Recognition in Aerial Videos.”

Nasirahmadi, Abozar, Barbara Sturm, Sandra Edwards, Knut Håkan Jeppsson, Anne Charlotte Olsson, Simone Müller, and Oliver Hensel. 2019. “Deep Learning and Machine Vision Approaches for Posture Detection of Individual Pigs.” *Sensors 2019, Vol. 19, Page 3738* 19(17):3738. doi: 10.3390/S19173738.

Oh, S., A. Hoogs, A. Perera, N. Cuntoor, C. C. Chen, J. T. Lee, S. Mukherjee, J. K. Aggarwal, H. Lee, L. Davis, E. Swears, X. Wang, Q. Ji, K. Reddy, M. Shah, C. Vondrick, H. Pirsiavash, D. Ramanan, J. Yuen, A. Torralba, B. Song, A. Fong, A. Roy-Chowdhury, and M. Desai. 2011. “A Large-Scale Benchmark Dataset for Event Recognition in Surveillance Video.”

Ozberk, Tayfun. 2021. “Turkey Plans to Deploy Attack Drones from Its Amphibious Assault Ship.” *Defense News*.

Pan, Weike. 2016. “A Survey of Transfer Learning for Collaborative Recommendation with Auxiliary Data.” *Neurocomputing* 177:447–53. doi: 10.1016/J.NEUCOM.2015.11.059.

P Boccardo, F. Chiabrando, F. Dutto, FG Tonolo, AM Lingua. 2015. “UAV Deployment Exercise for Mapping Purposes: Evaluation of Emergency Response Applications.” *Sensors* 15(7):15717–37.

Raschka, Sebastian. 2018. “Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning.” 49.

Razakarivony, Sebastien, and Frederic Jurie. 2016. “Vehicle Detection in Aerial Imagery : A Small Target Detection Benchmark.” *Journal of Visual Communication and Image Representation* 34:187–203. doi: 10.1016/j.jvcir.2015.11.002.

Rudol, Piotr, and Patrick Doherty. 2008. “Human Body Detection and Geolocalization for UAV Search and Rescue Missions Using Color and Thermal Imagery.” *IEEE Aerospace Conference Proceedings*. doi: 10.1109/AERO.2008.4526559.

S Amici, M. Turci, S. Giammanco, L. Spampinato, F. Giulietti. 2013. “UAV Thermal Infrared Remote Sensing of an Italian Mud Volcano.” *Adv Remote Sens* 2(December):358–64.

Santosh, KC, Nibaran Das, and Swarnendu Ghosh. 2022. “Deep Learning Models.” *Deep Learning Models for Medical Imaging* 65–97. doi: 10.1016/B978-0-12-823504-1.00013-1.

Shu, Tianmin, Dan Xie, Brandon Rothrock, Sinisa Todorovic, and Song-Chun Zhu. n.d. *Joint Inference of Groups, Events and Human Roles in Aerial Videos*.

Silvagni, Mario, Andrea Tonoli, Enrico Zenerino, and Marcello Chiaberge. 2017. “Multipurpose UAV for Search and Rescue Operations in Mountain Avalanche Events.” *Geomatics, Natural Hazards and Risk* 8(1):18–33. doi: 10.1080/19475705.2016.1238852.

Singh, Divya. n.d. “Predictive Model Performance Evaluation.” Retrieved June 22, 2021 (https://medium.com/@divyacyclitics15/what-is-predictive-model-performance-evaluation-8ef117ae0e40).

Stöcker, Claudia, Rohan Bennett, Francesco Nex, Markus Gerke, and Jaap Zevenbergen. 2017. “Review of the Current State of UAV Regulations.” *Remote Sensing* 9(5):459. doi: 10.3390/rs9050459.

Stone, M. 1974. “Cross-Validatory Choice and Assessment of Statistical Predictions.” *Journal of the Royal Statistical Society: Series B (Methodological)* 36(2):111–33. doi: 10.1111/j.2517-6161.1974.tb00994.x.

Sun, Peng, Azzedine Boukerche, and Qiyue Wu. 2017. “Theoretical Analysis of the Target Detection Rules for the UAV-Based Wireless Sensor Networks.” *IEEE International Conference on Communications*. doi: 10.1109/ICC.2017.7997348.

Szegedy, Christian, Sergey Ioffe, Vincent Vanhoucke, and Alexander A. Alemi. 2017. “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning.” *31st AAAI Conference on Artificial Intelligence, AAAI 2017* 4278–84.

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2014. “Going Deeper with Convolutions.” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 07-12-June-2015:1–9. doi: 10.48550/arxiv.1409.4842.

Tan, Chuanqi, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. 2018. “A Survey on Deep Transfer Learning.” *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 11141 LNCS:270–79. doi: 10.1007/978-3-030-01424-7_27.

Tsantekidis, Avraam, Nikolaos Passalis, and Anastasios Tefas. 2022. “Recurrent Neural Networks.” *Deep Learning for Robot Perception and Cognition* 101–15. doi: 10.1016/B978-0-32-385787-1.00010-5.

Zoph, Barret, Vijay Vasudevan, Jonathon Shlens, and Quoc v. Le. 2018. “Learning Transferable Architectures for Scalable Image Recognition.” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 8697–8710. doi: 10.1109/CVPR.2018.00907.