COGNITIVE SCIENCE BASED REAL TIME THREAT WARNING SYSTEM USING DEEP LEARNING AND COMPUTER VISION

A Thesis Submitted

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

IN

COMPUTER SCIENCE AND ENGINEERING

By

ANURAG SINGH
17SCSE301004

Supervisor

Dr. Naresh Kumar
Professor

GALGOTIAS UNIVERSITY
UTTAR PRADESH
INDIA

2022

Galgotias University
Uttar Pradesh
School of Computing Science & Engineering

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis, entitled "**Cognitive Science Based Real Time Threat Warning System Using Deep Learning And Computer Vision**" in fulfillment of the requirements for the award of the degree of **Doctor of Philosophy in Computer Science Engineering** and submitted in Galgotias University, Greater Noida is an authentic record of my own work carried out during a period from August 2017 to July 2022 under the supervision of Dr. Naresh Kumar.

The matter embodied in this thesis has not been submitted by me for the award of any other degree of this or any other University/Institute.

Mr. Anurag Singh

17SCSE301004

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

(Dr. Naresh Kumar)

Supervisor
SCSE

Galgotias University
Uttar Pradesh
School of Computing Science & Engineering

CERTIFICATE

This is to certify that **Mr. ANURAG SINGH (Reg. No.17SCSE301004)** has presented his pre-submission seminar of the thesis entitled **“COGNITIVE SCIENCE BASED REAL TIME THREAT WARNING SYSTEM USING DEEP LEARNING AND COMPUTER VISION”** before the committee and summary is approved and forwarded to School Research Committee of Computing Science & Engineering, in the Faculty of Engineering & Technology, Galgotias University, Uttar Pradesh.

Dean – PhD & PG

The Ph.D. Viva-Voice examination of **ANURAG SINGH** Research Scholar, has been held on ____________

Supervisor External Examiner

DEDICATION

This thesis is dedicated to My Family

My Parents

My Wife

My Brothers

ACKNOWLEDGEMENTS

Undertaking this PhD has been a truly life-changing experience for me and it would not have been possible to do without the support and guidance that I received from many people.

I would like to first say a very big thank you to my supervisor Dr. Naresh Kumar for all the support and encouragement he gave me, during both the long months I spent undertaking my research work. Without his guidance and constant feedback this PhD would not have been achievable.

Many thanks also to my previous guides Prof. (Dr.) Tapas Kumar & Dr. Jeevanandam convinced me during our many discussions to apply computation into the research work which has opened various new avenues in the field.

I gratefully acknowledge the support received towards my PhD from the administration of Galgotias University, including Hon’ble Chancellor sir Mr. Suneel Galgotia, Hon’ble CEO sir Mr. Dhruv Galgotia, Vice-Chancellor Ma’am Prof.(Dr.) Preeti Bajaj and all fraternity of the Galgotias university.

I greatly appreciate the support received through my students here at the university, thank you to my beloved students for making the process of data collection comfortable.

I am indebted to all my friends and family who patiently supported my long working hours and who were always so helpful in numerous ways.

I would also like to say a heartfelt thank you to my in-laws, parents and my extended family members always believing in me and encouraging me to follow my dreams. And finally, a very special thank you to my wife, Mamta Singh, for her invaluable advice and feedback on my research and for always being so supportive for my work and my quirks. Also, in helping me enormously, especially with the mammoth task of doing the final formatting and printing of this thesis.

Anurag Singh

PREFACE OF THESIS

It is critical for many countries to ensure public safety in detecting and identifying threats in a night, commercial places, border areas and public places. Humans, animals, and forests are extremely valuable components of our ecosystem but are consistently surrounded by a variety of threats. For instance, forest fires and large fire flames present immense risks as they can damage residential areas, forests, defence systems, and industries. While fires in the early stages can be identified by smoke detectors, sensors, and human assistants, these measures usually take too long and have a high false rate in detecting the fire flames, their range and size. Majority of past research in this area has focused on the use of image-level categorization and object-level detection techniques. As an X-ray and thermal security image analysis strategy, object separation can considerably improve automatic threat detection when used in conjunction with other techniques. In order to detect possible threats, the effects of introducing segmentation deep learning models into the threat detection pipeline of a large imbalanced X-ray and thermal dataset were investigated. In our proposed system, we established a novel deep learning and computer vision-based threat detection model using transfer learning model by optimizing the hyper-parameters, which includes the learning rate of the model. We further optimized the batch size and mini-batch gradient to improve detection and classification of these types of threats in real-time with greater accuracy and size in a dynamic environment so that the system is capable of making decisions without human assistance. Our deep learning model was trained on a Tesla K 80 graphic Processing Units with 2496 cores of Compute Unified Device Architecture (CUDA) and Video Random Access Memory (VRAM) with 12GB Graphics Double Data Rate (GDDR5). The training set was obtained from surveillance cameras, and the model was trained for classification using Mask Recurrent Convolutional Neural Network (RCNN). By applying the transfer learning technique on the Mask RCNN and Faster RCNN models, we significantly improved the detection and classification accuracy compared to traditional techniques. Specifically, our proposed technique achieved a bounding box score accuracy of mAP=89.2% and a classification rate of 86.2%.

for an intersection over the union (IoU) of 0.9 on our custom training and testing dataset using the transfer learning technique. Based on these promising results, our threat warning system can reduce fictitious alarms and more effectively save human lives compared to traditional sensors and vision methods.

In order to get the final results, we combined the two models i.e Faster R-CNN with Mask RCNN into a single detection pipeline using the transfer learning technique, which outperforms baseline and end-to-end instance segmentation methods using less number of the practical dataset, with mAPs ranging from 94.88 percent to 91.40 percent helps in detecting the person with guns, knives, pliers to avoid cross border threats.

TABLE OF CONTENTS

List of Figures

List of Tables

List of publications

Conferences

- A. Singh and J. Jotheeswaran, "Cognitive science based inclusive border management system," *2018 Majan International Conference (MIC)*, Muscat, 2018, pp. 1-5.doi: 10.1109/MINTC.2018.8363158-“**Scopus”**
- Singh A., Jotheeswaran J. (2018) P300 Brain Waves Instigated Semi Supervised Video Surveillance for Inclusive Security Systems. In: Ren J. et al. (eds) Advances in Brain Inspired Cognitive Systems. BICS 2018. Lecture Notes in Computer Science, vol 10989. Springer, Cham-“**Scopus”**

Journals

- Indonesian Journal of Electrical Engineering and Computer Science, Vol. 27, No. 2, August 2022, pp. 1007~1015, ISSN: 2502-4752, DOI:10.11591/ijeecs.v27.i2.pp1007-1015” ThreatNet: Advanced threat detection, region-based convolutional neural network framework**-**“**Scopus**”
- Anurag S, Naresh K, Tapas K. A Real-Time Deep Transfer Learning-Based Classification and Social Distance Alert Framework Based on Covid-19. Int J Cur Res Rev. 2nd Wave of COVID-19: Role of Social Awareness, Health and Technology Sector, June, 2021, 86-92, http://dx.doi.org/10.31782/IJCRR.2021.SP203-**“Scopus”**
- Singh A., Jotheeswaran J. (2018) Hybrid video surveillance systems using P300 based computational cognitive threat signature library," 2018 Brain Inspired Cognitive Architecture (BICA), Prague, Czech Republic 2018. https://doi.org/10.1016/j.procs.2018.11.115-“**Scopus**”

List of Abbreviations and Symbols Used

Abbreviation	**Full Form**
HCI	Human Computer Interface
GPU	Graphical Processing Unit
TPU	Tensor Processing Units
CNS	Central Nervous System
EEG	Electroencephalogram
AI	Artificial Intelligence
CNN	Convolutional Neural Network
RELU	Rectified Linear Activation function
FCN	Fully Connected Network
LSTM	Long short-term memory (LSTM)
RNN	Recurrent Neural Network
GAN	Generative Adversarial Network
RBFNS	Radial Basis Function Network
MLPS	Multi Layer Perceptron
CUDA	Compute Unified Device Architecture
API	Application Programming Interface
ILSVRC	ImageNet Large Scale Visual Recognition Challenge (ILSVRC)
ANN	Artificial Neural Network
DNN	Deep Neural Network
DBN	Deep Belief Network
VGG	Visual Geometry Group
MNIST	Modified National Institute of Standards and Technology
HRNET	High-Resolution Net,
CV	Computer Vision
IOU	Intersection Over Union
OD	Object Detection

HOG	Histogram Oriented Gradient
RCNN	Recurrent Convolutional Neural Network
SSD	Single Shot Detection
FCNN	Fully Connected Neural Network
YOLO	You Only Look Once
ML	Machine Learning
TL	Transfer Learning

CHAPTER:1

INTRODUCTION

CHAPTER 1

1. INTRODUCTION

1.1. Introduction

Surveillance systems are vital in every country's border areas, and border patrolling is the most dangerous of all cross-border activities, leading to higher running expenses and casualties. Border patrolling still requires security personnel and soldiers despite the use of sensors and cameras equipped with a Global Positioning System (GPS), Geographic Information system (GIS), and wireless access to sites via Unmanned Aerial Vehicles (UAVs). Researchers and security organizations began designing more advanced security systems that reduce human lives and ammunition waste by eliminating false alarms and improving detection precision on a target. Numerous nations began developing the Comprehensive Integrated Border Management System (CIBMS), a comprehensive border patrolling solution. It functions based on close security and suggests constructing a new radar system in border zones that will provide a 120-degree image of the surrounding area to the control center. When the command center gathers information on an attempted infiltration, the specialist cameras at the border rapidly adjust their recording settings to capture photos of terrorists trying to slip into the country. The primary factor that contributes to the success of employing this method is the reduction in the number of false alerts and consequently missed hits caused by automatic weapons. In a cross-border environment, the automatic identification of anomalous cross-border activities can be utilized to locate a probable threat, such as person with weapons, tossing a bomb and doing anything else dangerous. The Computer vision domain in this area is redefining and creating the revolutionary systems for a number of security applications that can be used in any place either in public or border areas. Surveillance cameras can now be found almost anywhere, including offices, banks, hospitals, parking lots, shopping malls, and airports. This is done to maintain relative peace and to reduce the crime rate. According to the World Population Review, poverty, unemployment, age, law enforcement, and other factors plays a role in a country's overall crime rate, which varies greatly from country to country. Given this environment and the continual advancement of technology, it's simple to see why incorporating computer vision into security and surveillance systems is so appealing.

In Chapter 1, We have described the overview of the Computer Vision and Artificial Intelligence-based areas used in carrying out the research and overcoming the issues, as well as the recommended approaches, which will be introduced in building and utilizing our own trained deep learning-based threat detection model.

1.2. Computer Vision

Computer vision is an area of both artificial intelligence (AI) and machine learning (ML) that aims to give computers the ability to "see" exactly the same thing that people do. Deep learning and neural network technologies have found applications in a broad number of fields, and the security sector is one of those fields. As a result, computer vision has seen a massive revolution as a direct consequence of these technologies. Computer vision appears to be a simple problem that anyone, including young children, can solve. However, due to the fact that we have a weak understanding of biological vision as well as the complexities of visual perception in an ever-changing and virtually infinitely variable physical world, this topic is still largely unresolved. The reason for this is because of the complexity of the environment. This thesis offers a basic introduction to the study of computer vision as a topic of research.

Following in this thesis, we are going to gain knowledge in the following areas:

- The history of, and divergence from, image processing in the field of computer vision.
- To what extent does the difficulty of computer vision stem from.
- Common goals or challenges in computer vision research.
- Artificial Intelligence and its related algorithms used for such issues.

1.2.1. Computer Vision and Image Processing

Computer vision differs significantly from image processing as image processing is the act of creating a new image from an existing one. This is typically accomplished by simplifying or refining the information that is present inside the image. Image processing is sometimes referred to as digital image manipulation. It is a subfield of digital signal processing that is not concerned with deciphering the information contained in an image. Image processing, such as picture pre-processing, may be required by a computer vision system to be applied to raw data.

Image processing examples include:

- The process of standardizing the photometric qualities of an image, such as its brightness and color.
- Cropping an image, such as arranging a photograph such that an object is centered in it.
- Eliminating digital artifacts from a photo, such as low-light digital artifacts and other types of digital artifacts.

The majority of the photos that we use in our research come from closed-circuit television, often known as CCTV, cameras. These cameras are installed across cities as part of a monitoring system designed to make the communities safer. It's an effective deterrence against vandals and criminals and a useful source of information for police investigations. There are numerous advantages for both the government and local companies when it comes to city monitoring, including the facilitation of the planning of large-scale events, the administration of traffic systems, etc. Cameras positioned in busy places, for instance, have been shown to decrease crime and increase foot traffic to nearby businesses. In addition, authorities have the ability and usually exercise it to subsidize businesses that provide security cameras in particular locations. Cameras installed around a city may seem like a good idea, but they can cause invasions of privacy and even physical harm. One must remember that surveillance cameras are not aimed at homes or workplaces. In addition, bad recordings or general hardware damage can occur if cameras are not set up properly, if extreme weather occurs, or if an accident occurs in the outdoors.

1.2.2. AI-powered surveillance

Computer Vision with AI-powered surveillance solutions are rather common, and there are several examples of their use. The following is a list of the most common applications and benefits of this technology.

1.2.2.1. Reduction in crime

It is reasonable to assume that the presence of security cameras in public spaces will reduce the incidence of criminal activity and vandalism. Does it not make sense to commit a crime in which you have a chance of being caught? On the other hand, closed-circuit television cameras serve just as "viewers" in the event of an accident, therefore they could not be enough protection. Instead, when paired with artificial intelligence programs for face recognition or weapon detection, these cameras can be an effective preventive tool. An artificial intelligence-powered CCTV system is designed to "see" guns, ski masks, criminals, and suspicious behavior. When a threat is detected, the camera can activate the alerts and send warning signals to law enforcement or security staff, preventing negative consequences. Naturally, the use of CCTV in conjunction with other crime-prevention and deterrence measures, such as increased illumination, armed guards, and defensible space, is going to produce the best possible results. In general, security cameras aid in the prevention of crime and the reduction of its severity by supplying investigators with pre-incident notifications and post-incident video evidence.

1.2.2.2. Person detection

Face recognition AI models provide computers the ability to differentiate between individuals. As a consequence of this, when a closed-circuit television camera and a facial recognition model are linked, the police are able to utilize an image of the offender and have the CCTVs inform them when a match is detected. Because of this, police enforcement and other government agencies will be able to take preventative actions and respond to incidents even before they take place.

The same line of reasoning should be utilized when searching for lost children as well as adults who are disoriented due to cognitive diseases such as Alzheimer's disease, forgetfulness, epilepsy, or dementia. Again, technologies such as real-time video analysis and face recognition make it much simpler to identify people.

1.2.2.3. COVID-19 compliance

Throughout the COVID-19 epidemic, numerous workplaces have refitted their existing surveillance cameras with ai - powered algorithms to monitor compliance with health guidelines, detect people wearing masks, monitor the frequency with which people wash their hands, and even recognize coughing.

1.2.2.4. Facial authentication in healthcare

It's no secret that the healthcare industry is struggling with patient identification. Between 2013 and 2015, there were 8000 instances of the wrong patient being treated at one of the nation's 200 hospitals, with around 9% of those errors resulting in damage or death. The causes of death include providing incorrect medication based on incorrect health data or failing to follow the patients' diet. Facial authentication is a technique used to protect patients from being misidentified during treatment. Providing AI-powered surveillance in hospitals also provides for fraud detection and authorisation validation.

1.2.2.5. Biometrics analysis

The term "biometrics" refers to any human trait, physical or behavioral, that can be used for digital identification and authentication. Banks, terminals, public buildings, and handheld devices all make extensive use of biometric authentication methods like fingerprint and facial recognition, as well as speech and typing cadence analysis.

1.3. How does Computer Vision Work?

To self-train and analyze visual input, computer vision relies heavily on pattern recognition techniques because data is widely available and corporations are ready to share it, deep learning professionals can utilize it to improve the process's accuracy and speed. Previously, machine learning algorithms were employed for computer vision applications; however, deep learning approaches have grown as a better answer for this sector. For example, machine learning techniques necessitate a massive quantity of data as well as active human monitoring in the initial phase to ensure that the findings are as precise as feasible. In contrast, deep learning is based on neural networks and employs examples to solve issues. It self-learns by recognizing common patterns in instances using labeled data.

1.3.1. What is the significance of computer vision?

We are being inundated with photographs of all types, from self-portraits to landscapes. Internet trends reports that daily image uploads exceed 1.8 billion. One can only hazard a guess at the final tally if one takes into account the photographs kept on phones. Each day, we watch over 4,146,600 hours of YouTube and send over 103,447,520 spam emails. Once more, that's just a piece of the puzzle; the internet of things, mass media, and communication are also major factors. Analysing and comprehending this plethora of visual material is essential. Trained computers to "see" these photos and movies is made easier with the aid of computer vision.

In addition, everyone may get online today because of improved connectivity. Young people are more vulnerable to "toxicity" and abuse when using the internet. Computer vision not only automates many tasks, but also ensures the moderation and monitoring of visual information on the web. Indexing is a crucial part of online content curation. It's simple to classify the vast amount of information available online because it often takes only two forms: text and either images or sounds. Algorithms in computer vision can read and catalog images. Google and YouTube, two of the most popular search engines, both use computer vision to quickly and accurately analyze uploaded images and videos before include them in results. By doing so, they not only give consumers with useful information but also shield them from harmful "toxicity" seen in some online communities.

1.3.2. Computer Vision's Future

Computer vision is a rapidly evolving technology that has piqued the interest of many industries. In the future, it will be able to handle a greater range of content. The domain already has a consistent market of 2.37 million US dollars and is predicted to increase at a CAGR of 47 percent until 2023. With the quantity of data we generate every day, it is only natural for machines to use that data to create solutions.

Once computer vision experts have solved the domain's existing challenges, we can expect a reliable system that automates content moderation and monitoring. With Google, Facebook, Apple, and Microsoft investing in computer vision, it's only a matter of time before it dominates the worldwide market. Improve your skills in this area to take advantage of the disruptive economy.

1.3.3. How can computer vision help the world?

Computer vision can benefit the world in a variety of ways:

- In an emergency, unmanned aerial vehicles can provide supplies.
- Facial recognition for public safety and military applications
- Gesture recognition to raise red flags against miscreants in public locations Optical character recognition for text processing Disaster management
- Automatic Decision support systems with more accuracy in surveillance systems.

  Thus, we conclude this brief overview of computer vision.

1.4. Artificial Intelligence: Deep Learning with Computer Vision

1.4.1. Deep Learning and the Human Brain: A cognitive Approach

Cognitive neuroscientists first presented a number of brain development hypotheses in the early 1990s, and others have since expanded on these ideas (particularly, neocortical development). They were used in computing models, which established them as the precursors of deep learning systems. Along the same lines as deep learning models, the self-organization of the brain is helped along by a wave of nerve development factor, which is analogous to the neural networks that are utilized in these developmental models. At this point, deep learning can be thought of as an attempt to imitate the brain of a baby. The brain of a newborn is like a sponge, and it learns via repetition. It takes some time for the web of neural networks in it to grow and be able to infer or deduce several things from a single batch of training data. Deep learning has the potential to let artificial intelligence reason, interpret, evaluate, and deduce on its own in the not-too-distant future.

Deep learning uses artificial neural networks that are made to look like the networks in the brain. As data flows through this artificial mesh, each layer examines a different element of the input, filters outliers, identifies known things, and generates the final result. It has been hypothesized that deep learning operates in a pattern similar to that of the brain. Yes, the human brain operates in a manner not dissimilar to that of a computer, but on a much more complex scale. The human brain has a significantly more complicated web of varied neurons, with each node performing a distinct function. Our comprehension of things is substantially superior.

1.4.2. Anatomy and Function of the Human Brain.

What exactly is a brain?

The brain is a complicated organ in our bodies that affects everything from cognition to memory to emotion to touch to motor skills to vision to respiration to temperature to hunger to almost every other activity. The Central Nervous System(CNS) is comprised of the individual's brain as well as the spinal cord that originates from the brain.

What exactly is the brain comprised of?

An average adult's brain weighs approximately 3 pounds and is composed primarily of fat (60 percent). Water, protein, carbohydrates, and salts make up the remaining 40% of the body's weight. The brain, in and of itself, is not a muscular structure. It is made up of a variety of cells, including neurons, glial cells, and blood vessels.

What exactly are grey and white matter?

The brain's grey and white matter are two separate parts of the central nervous system. Rather than referring to the white matter as the brain's innermost layer, scientists use the terms "grey matter" and "grey matter." Grey and white matter are arranged in opposite directions in the spinal cord compared to the cerebral cortex.

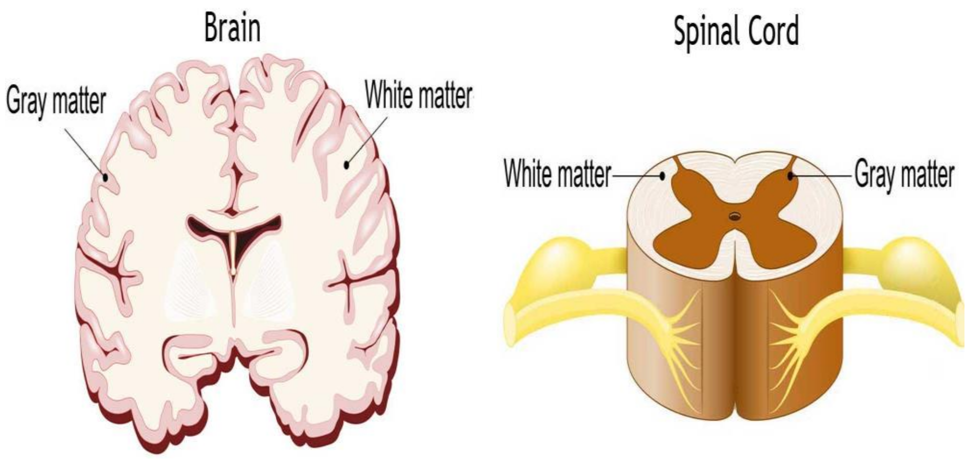


Figure 1.1 Inside & Outside of the Brain

Neuron somas (round core cell bodies) make up the majority of grey matter, whereas myelinated axons (long stems that connect neurons) make up the majority of white matter (a protective coating). On some scans, the two appear as unique colours due to the changes in the composition of neuron sections discussed in Figure 1.1.

1.4.3. How Does it Work?

The human brain has around one trillion neurons, which are linked to other neurons to form a neural network. Let us investigate how the human brain and Artificial Neural Network function.

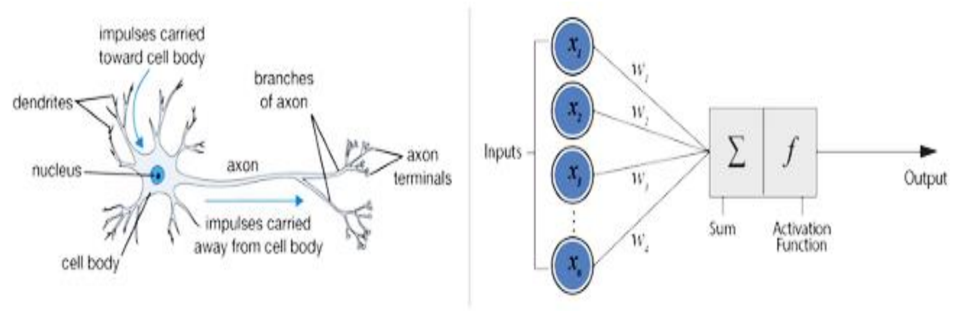


Figure 1.2 Biological Neuron Vs Artificial Neuron

The components of a human neuron are illustrated in Figure 1.2. The preceding diagram illustrates how dendrites receive electrical impulses, transmit them down an axon, and discharge them through axon terminals. It consists of the aforementioned layers and stages that make up an ANN. The very first layer of the prediction process, which also serves as the point at which inputs for the prediction are obtained. The second layer, referred to as the hidden layer, comprises an activation function that modifies and sums our inputs. The third layer is our output layer, which is where we observe our neuron's observable output in the required format. This final element suggests that if we expect a yes or no answer, we have not received any other type of answer, such as a number. The connections of neurons provide the true power of neural networks, both biological and artificial.

As a part of our nervous system, the brain transmits as well as receives chemical and electrical impulses. It is possible for human brain to understand all of the many signals that govern the various activities in your body. There are some that make you feel fatigued, while others that make you feel apprehensive or nervous. Most of the body's messages are stored in the brain, but some are sent to distant extremities via its enormous network of nerves and the spine, which are connected to each other by a network of nerves that extends from head to toe. This is accomplished through the use of billions of neurons in the central nervous system (nerve cells).

1.5. Artificial Neural Networks to Deep Learning

Artificial neurons, which are sometimes referred to as nodes, are used to construct a neural network, which is structured in the same manner in which a human brain performs its functions. In this particular instance, there are three layers of nodes placed one on top of the other discussed in Figure 1.3.

- The layer of input
- The output layer
- The hidden layer(s)

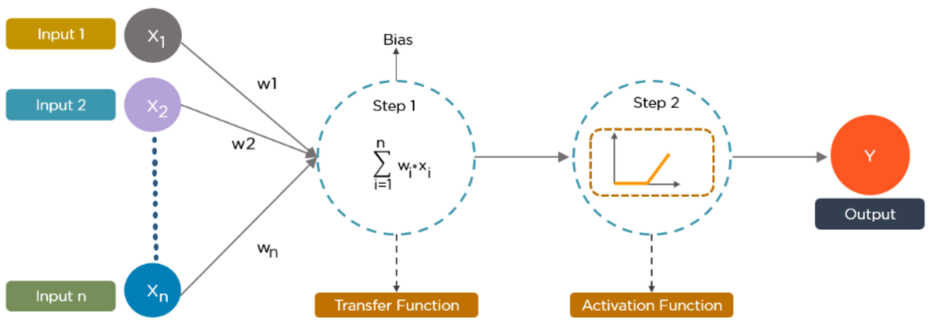


Figure 1.3 Artificial Neural Network

The family of artificial intelligence algorithms known as "deep learning" has undergone a revolution in recent years as a result of this transformation. It was previously impossible for computers to achieve human-like performance in a wide variety of difficult cognitive activities; however, because to recent advancements in algorithm design, this is now possible. Despite their connectionist foundation, recent advances in deep learning models have been driven mostly by practical engineering needs. The cognitive benefits of deep learning models seem to apply regardless of their application. This is an example of the biological process of evolution, in which a previously unsuitable physiological system is adapted to do the new task. We've tried to explain why cognitive science has to start taking deep learning seriously, and we've explained why the time is now. In the beginning, the pathway of deep learning advancement from the connectionist attempt is investigated, which displays great consistency as well as contrasts. Then, beginning with vision and moving on to language, it will be described how deep learning models may be advantageous for a wide variety of cognitive domains, particularly those in which it has achieved performance levels comparable to those of humans.

1.5.1. Artificial Neural Networks potential drawbacks

In spite of advances in the application of cognitive models to artificial intelligence, the goal of mimicking human thought remains unmet. " It takes a lot of training data for algorithms like neural networks, for example, to be able to predict what will happen in the future with similar data. Even so, they are only able to identify the specific problem area they were trained in.

This is a completely different way of thinking from what we have in our human brains. The human brain makes generalizations about new events using context and more limited experience that even the most powerful cognitive models today cannot achieve. We still don't know everything about how the human brain works, even with the most cutting-edge research. However, even with this basic understanding, it is still a big leap to transfer human brain processes into computer programmes.

When it comes to research and development, High-level interpretation of digital images or movies is the subject of Computer Vision, a well-established field that deals with this issue. AI is being used to solve many modern-day social problems through automation, including object identification, smart homes, surveillance systems and other similar applications. Systems and concepts derived from human visual and cognitive intelligence are capable of simulating and replicating complex judgments made by the human brain, which is capable of making complicated decisions.

1.5.2. A brief overview of Deep Learning

Deep learning, a subtype of machine learning in which artificial neural networks recreate the inner workings of the human brain to analyze data, find patterns, and inform decision making, has made tremendous progress in the field of artificial intelligence in recent years. Deep learning can now perform unsupervised learning from unstructured or unlabeled data, extending on what the human brain is capable of. This information, often known as "big data," may be collected from many different places online and offline.

These data sources are so vast that it may take humans decades to comprehend and extract vital information, yet deep learning allows models to detect things, recognize speech, translate language, and make decisions at fast speeds. Many firms see the immense potential that can be gained by tapping this wealth of information and are increasingly integrating AI systems powered by deep learning to gain a competitive advantage through data and automation. Lets see how deep learning can work on surveillance area.

Surveillance security is a time-consuming and difficult activity. In this thesis, we have created a system to automate the work of video surveillance analysis and detection system. We have evaluated the video footage in real time to detect any unusual activity such as violence, theft and threats. There is a lot of study being done in the market about video surveillance, and the function of CCTV videos has expanded significantly. CCTV cameras are strategically placed across the area for surveillance and security.

1.5.3. Deep Learning in the surveillance area

Deep learning algorithms for deep surveillance have advanced during the last decade. These developments have demonstrated an important trend in deep surveillance and promise a significant increase in efficiency. Deep surveillance is commonly used to identify theft, detect violence, and predict the likelihood of an explosion. We should reduce the computing and storage problems that need deep learning in order to get over these obstacles. The human brain can be utilized as a model for increasing efficiency, specifically by emulating the brain's early development period for deep learning. This is comparable to how we looked to the human brain for inspiration when developing AI.

One of the primary areas in which most governments are working is to automate border and local area monitoring systems. The biggest and most common risks for all nations come from the border areas that they share with their neighbors. India shares borders with Pakistan, Bangladesh, Sri Lanka, Nepal, China, and Bhutan, among others. Over the previous seven decades, India has faced serious difficulties such as cross-border terrorism, illegal migration, narcotics trafficking, and arms smuggling. Every year, hundreds of military and civilians are killed in India as a result of these challenges. Researchers and scientists are working on automation-based technologies to tackle these obstacles and concerns. It is difficult for soldiers to keep watch on borders around the clock, seven days a week, in all kinds of weather, from the bitter cold of Siachin winters to the ferocity of storms and the scorching heat of desert summers. There are several forms of hazards, such as Covid-19, forest fires, and natural calamities, which result in deaths.

For many years, researchers have been working on artificial intelligence, machine learning-based algorithms that replicate and imitate human behaviors such as adding a caption to images, autonomous automobiles, games, and so on. Artificial intelligence is the way of the future which is nearly a distinct being affecting every technology. Machine learning is one of the underlying scientific principles that underpins this phenomenon, and deep learning is the engine that drives the research forward. Deep learning outperforms humans' capacity to process and progress massive amounts of data in image analysis, face recognition, autonomous driving, and other areas because to a flood of data and the introduction of faster GPUs and TPUs. To prevent these deaths, different Human-Computer Interaction (HCI)-based technologies that function without human aid and with greater precision have been introduced.

1.6. Problem Definition

1.6.1. Motivation & Goal

The motivation behind carrying out this research work is to automate the surveillance systems because many human and animals are losing their life due to false detection and alarm. The goal is to analyze and improve the training, computation time, classification and tagging performance of different threats using fewer training datasets is the aim of this research. A deep-learning model hyperparameter optimization for real-time item detection and tracking of any form of risk will be used to achieve this goal.

In order to accomplish this thesis's overarching goal, the following objectives have been outlined.:

- To detect and recognise real-time threats by employing appropriate deep learning models for using a smaller training dataset and making full use of transfer learning methodology for multidomain usage and performance of the selected deep learning models in terms of categorization.
- First, we will train a deep learning model on our data, and then we will make adjustments to the model's hyperparameters in order to get the model to perform as accurately as possible.
- In this work, we compare and visualize how several cutting-edge algorithms perform in a categorization task.
- Making use of transfer learning in addition to many other tactics in order to enhance and optimize the speed.

Prior to delivering our ideas and solutions, we have provided a formal description of the situation which are the major challenges in computer vision domain.

1.6.2. Image Classification

The following are the major issues in picture classification:

1.6.2.1.Intra-Class Variation

The variance between photographs of the same class is referred to as intra-class variation. Having chairs of various types in our dataset is an example of intra-class variation. The weapon might be "Gun" "Fires" "Type of Guns," "type of fires," and so on.

1.6.2.2.Scale Variation

This is a typical issue in picture categorization. Scale variation is the use of numerous images of the same thing at different sizes.

1.6.2.3.View-Point Variation

We have perspective variation, which means that an item may be oriented/rotated in several dimensions in terms of how it is photographed and recorded in an image. Whatever angle we catch the image of the weapon, it is still a weapon.

1.6.2.4.Occlusion

There are several items in the image that we want to categorise but cannot see entirely. Their main role is obscured by other items. The image of the threat is provided, however, observe how it is hidden behind the cover, out of sight. It is not entirely visible, but our image classification algorithm should be able to detect and label it as a threat.

1.6.2.5.Illumination

Our picture categorization system should be able to manage variations in illumination as well. Both of these photographs are of the same cup, but with different pixel intensities. Our picture categorization system should be able to deal with lighting variations. So, if we offer our image classification system a picture of the same item with varying brightness levels (Illumination), the system should be able to assign the same label.

1.6.2.6.Background Clutter

It signifies that there are many items in the image and it is difficult for the viewer to find the specific object. These photos have a lot of "noise." But, we are only interested in one specific thing in the image; however, owing to all of the "noise," it is difficult to identify the specific object. It's a challenging assignment for people, so picture how difficult it is for a machine with no semantic knowledge of the image.

Why should we be concerned about intra-class variation? Deep classification networks outperform on closed datasets. They are not, however, intended to work well in open sets. It is feasible to examine object classes that have never been seen before in training in open sets. A classification network trained on imagenet, for example, would never have seen a goldfish. Unseen objects are reliably predicted to belong to one of the N-known training classes using classification networks. We do, however, want our network to recognise them as out-of-training distribution classes. The current best strategy for doing this is to use one-vs-all classifiers on the penultimate layer embeddings. If all of the one-vs-all classifiers reject a test picture, it is expected to be out-of-distribution. These classifiers would perform better in one-vs-all scenarios if the embeddings in the same class are compact and discriminative, i.e. contain less intraclass variation.

Both the process of face identification and face verification have received a significant amount of research, with the latter garnering the majority of the focus. Research on face recognition has almost exclusively focused on closed-set methods, which are based on the assumption that all of the inspecting photographs included in evaluation include the identities of people enrolled in the gallery. This closed-set assumption, however, cannot be established since in real systems only a fraction of probe sample IDs are registered in the gallery. Instead, they need to assume that there is an open collection of probe samples and be able to reject or ignore those that match identities that are unknown to them. One such significant application is face recognition. The subject identities of test images are often not available in the training data in this scenario.

Organization of the thesis:

The five sections of the thesis are described in greater depth below.

Chapter 1: Introduction:

The introduction part will discuss the importance of the work, gaps, and description of research work to be carried out on different types of threats.

Chapter 2: Literature Survey:

This chapter will provide a brief description of the research works that have been carried out by various researchers in the fields of Computer Vision and Deep

Learning technology. Additionally, the field in which we are carrying out our research work, which is object detection, will be briefly discussed along with a variety of different approaches.

Chapter 3: Classification Of Threat-Detection Images: Training From Scratch Or Transfer Learning:

This chapter will contain the detailed proposed methodology along with comparative analysis of the advance models. This chapter will describe the results and discussion of the observations made in regard to the objectives of the study.

Chapter 4: Performance & Result Analysis:

This chapter will contain the Experimental description, Dataset and methodology adopted for carrying out the experiment. This chapter will describe the results and discussion of the observations made in regard with the objectives of the study.

Chapter 5: Summary and Conclusion:

The findings is summarized, and the overall conclusion of the work is presented in this chapter. This chapter will also explore the limits of the study, as well as make some recommendations and look ahead to the future of the research.

CHAPTER:2

LITERATURE SURVEY

CHAPTER 2

2. LITERATURE SURVEY

2.1. Introduction

In the Chapter 1, we have discussed about the introduction to computer vision-based surveillance and problems related to this area. Whereas, in this chapter, we went through a brief examination regarding computer vision and deep learning by making use of neural networks, artificial neural networks (ANNs), convolutional neural networks (CNNs), activation functions. Following these principles, the chapter digs into the Artificial Intelligence based algorithms i.e Deep Learning and Computer Vision based object detection architecture, demonstrating how it varies from object tracking before going into the commonly used object detection models and how they have evolved over time as computer vision incorporates within Machine Learning. During the middle of the 1980s and the early 1990s, there was a decline in interest in research pertaining to artificial intelligence and machine learning. As a result, the majority of the field's advancements were splintered into subfields such as natural language processing, image recognition, and robotics. It is necessary to have a substantial amount of data on hand in order to perform computer vision. It repeats the data processing process until it discovers variations, at which time it is able to recognize individual photographs. In order to train a computer system to detect automated threats, it must be fed enormous volumes of photographs based on threats and other materials connected to threats. This enables the system to distinguish the many types of threats and grasp the differences between them.

To train a computer about the context of visual input, a technique known as machine learning makes use of algorithmic models. If the model is fed with a sufficient amount of data, the computer will "look" at the data and learn to differentiate between different types of images. Algorithms allow the computer to learn on its own rather than having to be programmed to recognize an image. A convolutional neural network (CNN) is a type of network that helps a machine learning or deep learning model "see" by segmenting images into pixels that are tagged or labeled. It performs convolutions with the labels in order to make predictions based on what it is now "seeing," and it does this by concatenating the labels (an operation in mathematics performed on two functions that results in the production of a third function). Convolutions will be performed by the neural network, and the accuracy of its predictions will be evaluated by a series of iterations, and this process will continue until the predictions begin to be realized. After that, it is able to detect or see images in a manner that is analogous to that of humans.

A CNN, much like a person looking at an image from a distance, first recognizes sharp corners and basic shapes and then, as it goes through iterations of prediction, it fills in the blanks with information. A CNN is used to comprehend individual images. In the same manner as it is used in audio applications, a recurrent neural network, or RNN, is used in video applications to assist computers in understanding how images in a series of frames are associated to one another.

An image classification process is referred to as object detection, whereas an object detection procedure is referred to as object classification [1][2]. It has been used in consumer electronics as well as human-computer interaction, visual computing, robotics, and security. Deep learning has been the dominant method for computer vision (ILSVRC) which focuses on image classification and object identification improvements in this section as of 2012. This thesis provides a quick introduction to the fundamental notions before delving into the most cutting-edge image categorization and object identification algorithms. Finally, we look at open-source object identification frameworks and the trade-offs between various deep learning object detection methods in terms of speed and accuracy. Deep learning approaches have the ability to achieve cutting-edge results on tough computer vision problems such as image categorization, object detection, and face recognition.

2.2. Artificial Intelligence and Neural Network

2.2.1. Artificial Neural Network

To start on deep learning, let us understand the basics of Artificial Neural Network (ANN), which is an olfactory system and other biological systems that can be mimicked like a human brain to make decisions and classifications. Similar to the brain's neurons and synapses, the central processing units (CPUs) in brain-like neural networks are linked via neural synapses. The transmission of signals (properties) from one neuron to the next is made possible by weighted neural connections [3]. An artificial neural network, rather than being coded with task-specific rules, learns via observation and experimentation. They may then apply what they've learnt from studying handwritten instances of the numbers 0 through 9 to identify numbers in further, unseen data. [4].

Three layers are present in an ANN: the input layer, the output layer, and one or more hidden layers. There are fewer restrictions on output than there are on input. Figure 2.1 hides two levels that aren't immediately apparent. The information contained within the first hidden layer is communicated to the second hidden layer, where it is used to train the third hidden layer on the fundamentals of the topic at hand. Using the inputs from the previous layer, a second layer, for example, can learn more complicated and abstract features. Further layers in a network may learn more complicated features in a similar fashion. When confronted with challenging issues such as picture categorization, object recognition, other tasks of a comparable type, one possible solution is to employ a network with multiple layers.

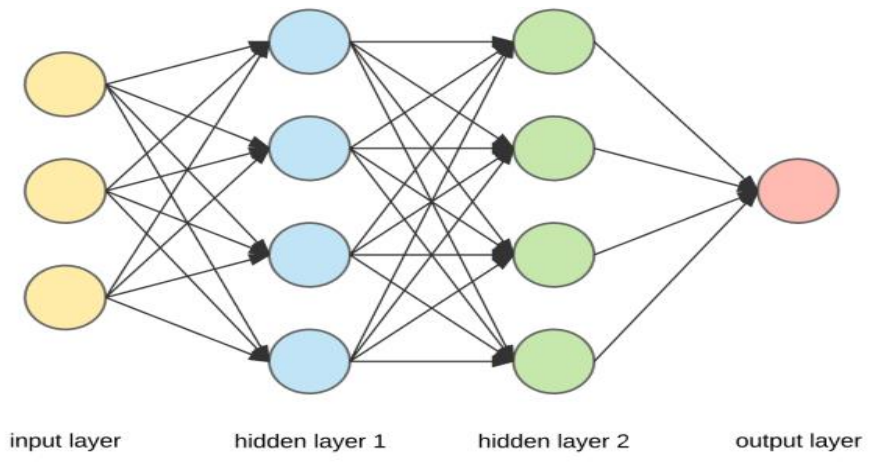


Figure 2.1 ANN Architecture

2.2.2. Deep Learning Architecture

Deep Learning is a branch of machine learning and artificial intelligence that studies exceedingly complicated artificial neural networks and then applies them to the training of extremely complex computer models. A multi-tiered model known as a Deep Neural Network is the most prevalent type of artificial neural network. The depth of a deep neural network refers to its ability to store a specific amount of information. It is uncommon for deep networks to have more than two layers of buried neurons between the input and output levels. As this illustrates, deep networks are extremely complex. Images, speech recognition, language translation, and other areas have all seen significant progress due to DNN [5, 6]. DNN have a far wider range of applications than only pattern recognition and translation. It is possible to learn features (representations) from a wide range of data sets using deep learning approaches rather than using task-specific algorithms [7]. As per deep learning concepts, it is possible to learn on your own rather than any assistance. Systems must rely on supervised learning in order to profit from deep learning in the actual world [8,9]. Because deep learning systems can be taught on more data, Andrew Ng et.al [10] asserts that they are scalable in comparison to traditional learning methods that are limited in their ability to improve performance as shown in Figure 2.2.

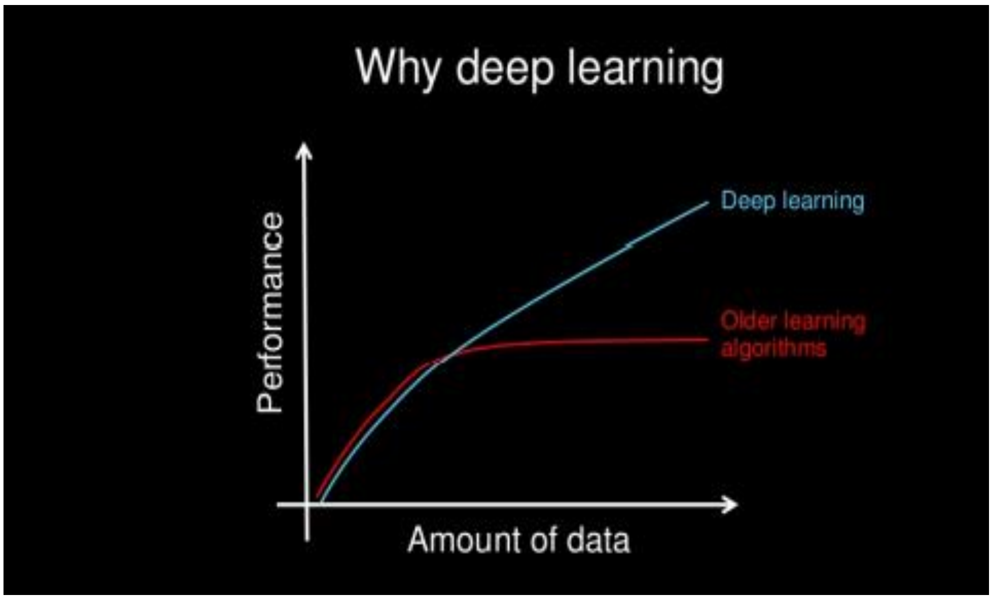


Figure 2.2 Performance of Traditional learning methods against Deep Learning

When we talk about "feature learning," we're referring to automated feature extraction from raw data. There are several degrees of complexity and abstraction that a deep neural network must master in order to perform properly.

Deep learning also involves the creation of hierarchical feature sets. The primary layers are in charge of passing down lower-level features to the secondary levels. The successive layers are constructed on top of one another using the higher-level features that are derived from the lower-level data. Deep learning models learn critical functions without manual-created features [11, 12, 13]. They can learn in a hierarchical fashion in a manner that is analogous to how the human brain works, and are hence suitable for use in a diverse array of applications despite their low cost. To overcome this issue Convolutional Neural Network (CNN) came into picture to resolve over the traditional methods.

2.2.3. Convolutional Neural Network

There have been many different deep learning architectures that have made use of CNNs. CNNs, on the other hand, begin with multiple convolutional layers rather than just one. In addition to image categorization, object recognition, voice recognition, CNN is frequently employed for a vast array of many purposes. As a direct result of AlexNet's outstanding performance in the ImageNet Challenge in 2012, CNN has received broad acclaim among academics and practitioners working in the domain areas of computer vision. Let's have a look at one of primary components belongs to CNN architecture that were covered in figure 2.3:

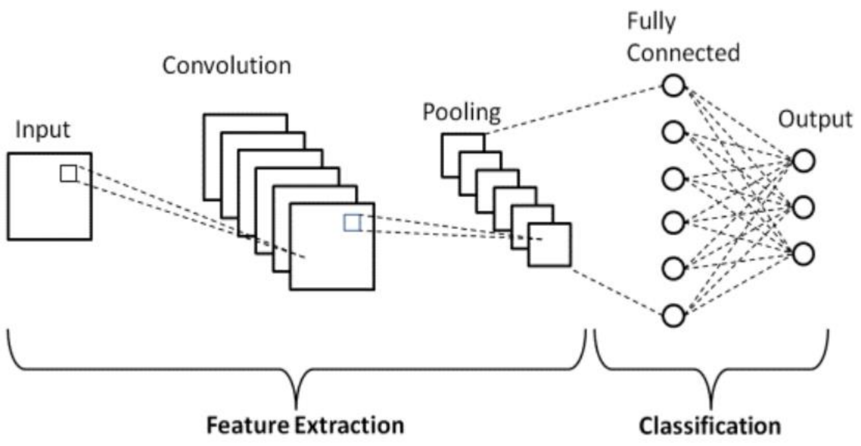


Figure 2.3 Basic architecture of CNN

- Convolution layers
- Pooling layers

- Fully connected layers
- Softmax function

Two fully linked layers conduct the final transformation for each image after a convolutional and pooling layer. The softmax function is used as the final step in the multiclass classification. Let us discuss the layers in brief.

2.2.3.1. Convolution Layers

A CNN cannot be constructed without the convolution layer, which is the first and most critical component. Neurons in convolution layers filter out the information they receive in this layer. Sequentially sliding filters across the image is used to create feature maps. Convolution is a term used to describe this sliding process [14]. Figures 2.4 and 2.5 show a 3x3 convolution filter applied to a 6x6 feature map as input to the previous section's earlier depiction of the process (kernel). A 4x4 feature map is generated as a result of swiping the filter across the entire input.

3	2	5	1	8	6
1	4	6	4	7	3
8	9	1	5	3	7
2	4	5	2	9	2
3	1	6	1	5	8
9	3	7	3	2	5

(a) Input feature map

1	0	0
0	1	1
1	1	1

(b) Filter

Figure 2.4 6x6 input feature map and 3x3 filter

3x1	2x0	5x0	1	8	6
1x0	4x1	6x1	4	7	3
8x1	9x1	1x1	5	3	7
2	4	5	2	9	2
3	1	6	1	5	8
9	3	7	3	2	5

(a) Convolution process

31	27	25	26
22	21	30	27
27	24	24	30
28	24	23	25

(b) Output feature map

Figure 2.5 Convolution process of 6x6 input with 3x3 filter

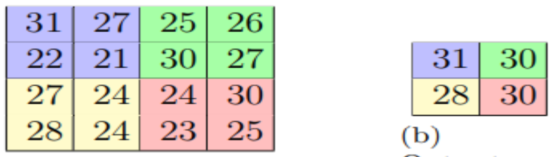
The above image discuss in Figure 2.5, shows the vectors and matrices can be extended to represent data using tensors, which are higher-dimensional extensions. A neural network is trained using tensors, which are data arrays of various dimensions and ranks.

During the training phase, deep learning requires massive data sets that are organised in an ad hoc manner. When dealing with multi-dimensional data sets such as tensors, compactness is a benefited. Neural networks employ a tensor representation of the input to predict the output.

In the CNN diagram discussed in Figure 2.2, the first layer of the network is a convolutional layer block. Each layer receives the input image or feature map and acts as a separate filter to learn individual feature mappings from the image or feature map. Each layer is a filter. Using three convolution layers and an image as input, the following formulas can be used: There are three distinct properties in each layer's feature map: colors, edges, and forms. There are three dimensions (height/width/depth) of output from each convolution block (height, width, and depth). Figure 2.5 estimates the number of convolution layers from an image's length and breadth.

2.2.3.2. Pooling Layers

When employing a convolutional approach, the pooling layer often follows the convolutional layer. A pooling layer [15] reduces each feature map's dimension by two. The final image's file size is minimized by the use of feature map subsampling and compression. Pooling and convolutional layers, on the other hand, produce results that are still primarily three-dimensional. There has been a reduction in the breadth and height, but the depth has not changed. It's possible to combine different types of pools into one large one. Full utilization of the available resources is a common tactic. As far as practical applications go, max-pooling [16, 18] is the approach of choice. When using max-pooling, the highest value in a feature map is selected and it is illustrated in Figure 2.6 that how to maximize pooling with a 2x2 window and a stride of 2. The length of a stride is based on the count of pixels traversed in each step by the filter or window. The most significant advantage of pooling is the reduction in parameters and, consequently, in training time.



(a) Input feature map

(b) Output feature map after max-pooling

Figure 2.6 A function of max-pooling layer with 2x2 window and stride 2

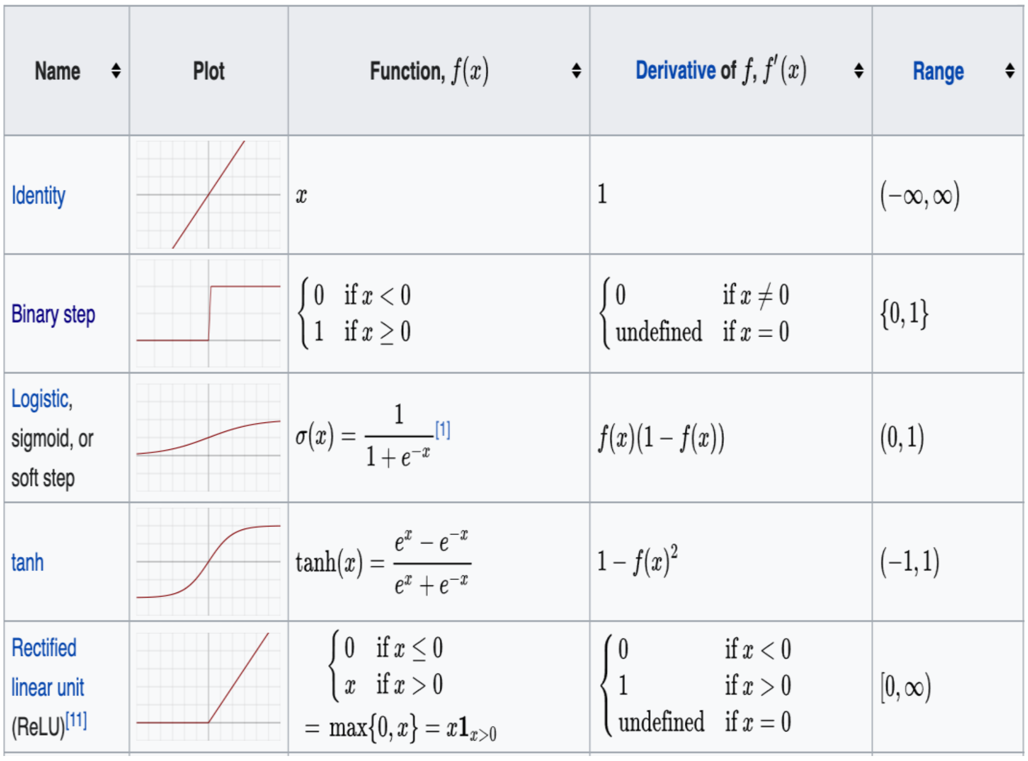
2.2.3.3. Activation Functions

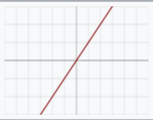
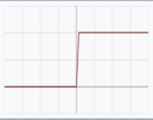
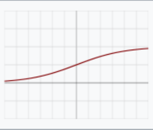
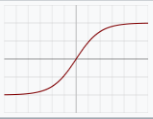
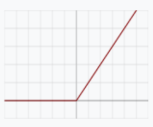
Finally, in the CNN model, the activation function is an important component to consider. The activation approach generates a weighted total and then adds bias to specify how much a neuron should be stimulated. The activation function's goal is to add nonlinearity to a neuron's output. Any continuous and sophisticated link between network variables can be taught and studied using this type of connection. To put it another way, it tells the network which model information should and should not be transmitted. Nonlinearity is injected into the network as a result. Many activation functions, such as the ReLU, Softmax, TanH, and Sigmoid, are extensively employed in computer science and engineering. Each of these functions has a certain purpose. The sigmoid and softmax functions are recommended for a CNN model for binary classification; however, the softmax function is typically used in a CNN model for multi-class classification.

One loss function that can be used to reduce error in a CNN is the softmax loss function. For binary classification detection, the softmax function is indispensable as it provides information on the likelihood that we properly predicted one class over the others. When using the softmax function to identify multiclasses, one-dimensional vector probabilities for all classes are generated. As a result, the target group is the most likely to be chosen. We know that neurons in a neural network work in line with their weight, bias, and activation function. When training a neural network, the output error is used to adjust the network's neuron weights and biases. The process is called back-propagation.

Back-propagation is made possible by activation functions due to the fact that the gradients necessary to update the weights and biases are provided concurrently with the error. Without an activation function, a neural network is equivalent to a linear regression model. As the activation function makes nonlinear changes to the input, the system is able to acquire knowledge and perform increasingly difficult tasks.

Table 1.1 Activation Functions with parameters



Name	Plot	Function, $f(x)$	Derivative of f, $f'(x)$	Range
Identity		x	1	$(-\infty, \infty)$
Binary step		$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$\begin{cases} 0 & \text{if } x \neq 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$	$\{0, 1\}$
Logistic, sigmoid, or soft step		$\sigma(x) = \frac{1}{1+e^{-x}}$[1]	$f(x)(1-f(x))$	$(0, 1)$
tanh		$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$1 - f(x)^2$	$(-1, 1)$
Rectified linear unit (ReLU)[11]		$\begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ $= \max\{0, x\} = x\mathbf{1}_{x>0}$	$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$	$[0, \infty)$

2.2.4. Different type of Networks

Over the last ten years, several CNN layouts have been proposed [15, 16]. A well-thought-out model's design is vital, and it is useful for a wide range of application types. CNN's infrastructure has changed significantly since the network's debut in 1989. This category includes reformulating the structure, regularizing the data, optimizing the parameters, and making other minor changes. However, the massive performance boost seen in CNN can be largely attributed to the rearranging of processing units and the addition of new frames.

Recent advances in CNN design have been made possible by researchers who took advantage of network depth.

2.2.4.1. AlexNet

LeNet's work on deep neural networks created the groundwork for deep convolutional neural networks (CNNs). (See Figure 2.7.) Previously, CNNs could only recognize handwritten numbers, limiting their use to a small set of image classes. A great majority of academics recognize AlexNet as a forerunner in the field of deep CNN architecture due to its groundbreaking work in image identification and classification [20]. AlexNet, which boosted CNN learning skills by raising the depth of the CNN and adding additional parameter optimization algorithms, was introduced for the first time and addressed in [20]. AlexNet was created to help CNN learn better. Figure 2.8, which can be viewed here, depicts AlexNet's architecture in a more reduced form.

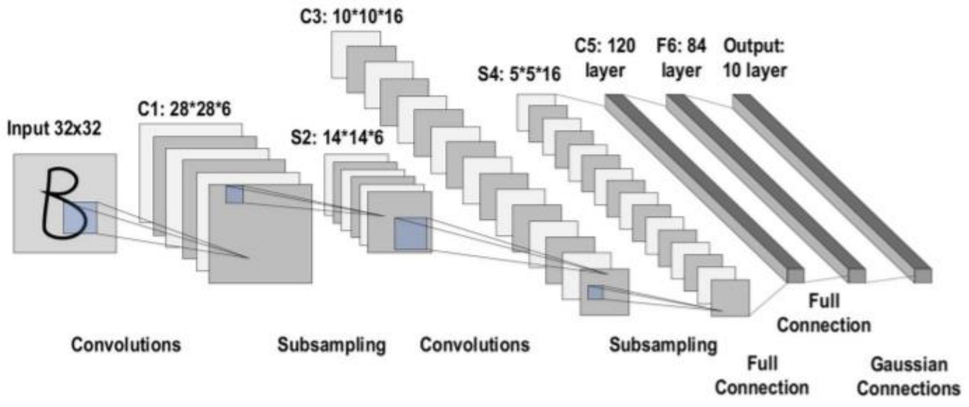


Figure 2.7 LeNet Architecture

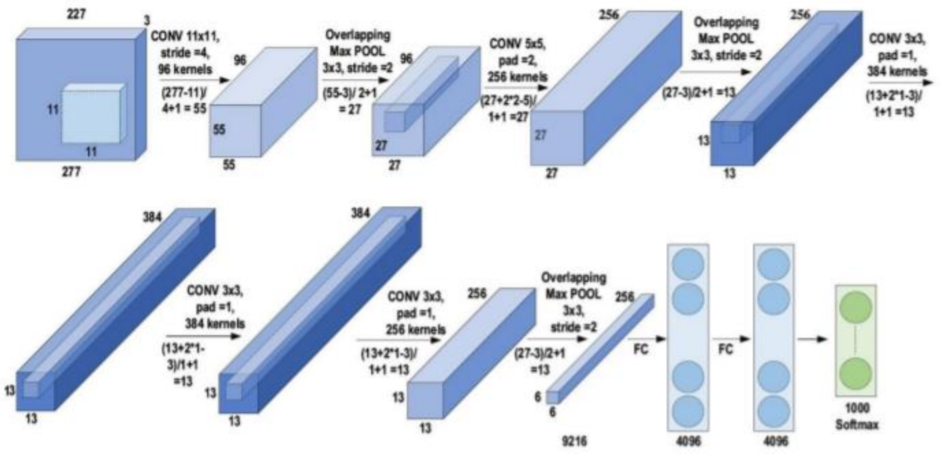

227
3
11
11
277
CONV 11x11, stride =4, 96 kernels
(277-11)/ 4+1 = 55
96
55
55
Overlapping Max POOL 3x3, stride =2
(55-3)/ 2+1 = 27
96
27
27
CONV 5x5, pad =2, 256 kernels
(27+2*2-5)/ 1+1 =27
256
27
27
Overlapping Max POOL 3x3, stride =2
(27-3)/2+1 =13
256
13
13
CONV 3x3, pad =1, 384 kernels
(13+2*1-3)/ 1+1 =13
384
13
13
CONV 3x3, pad =1, 384 kernels
(13+2*1- 3)/1+1 =13
384
13
13
CONV 3x3, pad =1, 256 kernels
(13+2*1-3)/ 1+1 =13
256
13
13
Overlapping Max POOL 3x3, stride =2
(27-3)/2+1 =13
256
6
6
9216
FC
4096
FC
4096
1000 Softmax

Figure 2.8 AlexNet Architecture

When it was first being developed, the deep CNN had a learning capability that was severely limited due to the constraints of the hardware. In order to circumvent these hardware restrictions, the training of AlexNet was carried out in parallel on two NVIDIA GTX 580 graphics processing units. Furthermore, as previously indicated, the quantity of feature extraction steps in AlexNet was extended from 5 in LeNet to 7 in order for the CNN to respond to a broader variety of image classifications. In spite of the fact that it can help enhance generalisation across a broad spectrum of image resolutions, depth proved to be the most significant limitation in our investigation. The problem was solved by Krizhevesky and his colleagues by applying the theory developed by Hinton [21]. According to Krizhevesky and his colleagues' description, during the training process, it was discovered that the system's learnt properties were improved by making random sweeps across numerous transformational units. ReLU [23] can be used as a non-saturating activation function to accelerate convergence. This avoids the problem of vanishing gradients, which is the root of the problem. Local response normalisation and overlapping subsampling were used to limit the degree of overfitting. Another modification that was made to enhance the functionality of the older networks was the implementation of large-size filters in the early levels of the network (5 x 5 and 11 x 11 respectively).

2.2.4.2. Network-in-Network

This network model included two novel notions [25], which differed slightly from the previous models in that they were not previously considered. The first was to employ a large number of layers of convolutional perceptual processing. When performing these convolutions, a 1x1 filter is used, which allows for the addition of additional nonlinearity to the networks to be included. Additionally, this allows for the expansion of the network depth, which can then be regularised through the use of dropouts. The bottleneck layer in DL models is where this concept is most commonly used, and it is referred to as such. There are two revolutionary concepts here: first, the GAP can be used in place of an FC layer, reducing the number of model parameters that must be implemented. The network architecture of the organisation is also significantly altered by GAP. A low-dimensional feature vector can be generated from a large feature map using

GAP without having to shrink the original [26, 27]. As depicted in Figure 2.9, the network's organisational structure.

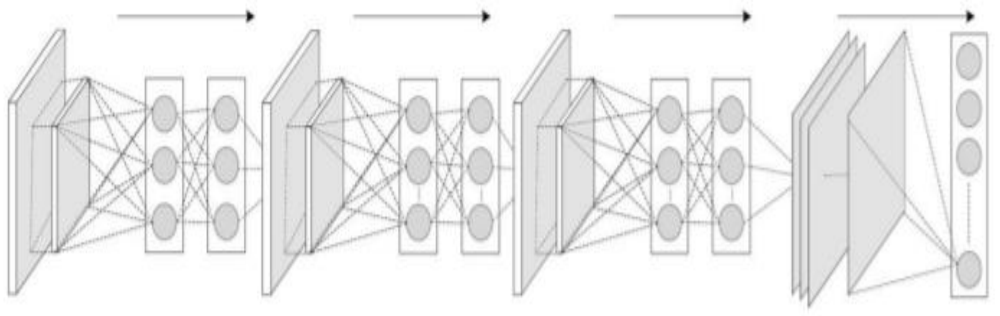


Figure 2.9 The architecture of network-in-network

2.2.4.3. ZefNet

Since CNN's learning method was mostly trial and error before 2013, it was difficult to tell exactly what the upgrade was expected to accomplish before its implementation. With complicated images, Deep CNN's performance was hindered because of this issue. DeconvNet was introduced in 2013 [28] to solve the issue (a multilayer de-convolutional neural network). To appropriately display a network of nodes, ZefNet, the term given to this technology, was invented. Analysing the firing patterns of neurons in the network allowed for the creation of a network activity visualisation [29]. Auto-encoders (AEs) were also evaluated by presenting the image's generated classes utilising the output neurons generated by AE's neurons. Using DenconvNet, a forward-pass CNN-like performance is achieved when the convolutional and pooling layers are reversed. For each level, the internal feature representation is interpreted by the brain as a picture form, which is then translated back into convolutional output and used to create the image form. ZefNet depended on a number of assumptions to maintain track of the learning schematic as it moved through the training stage. Results also revealed a flaw in the model's abilities that needed to be corrected. DeconvNet was employed on AlexNet to experimentally show this concept. Because of this, only a tiny fraction of the network's neurons were active at the first and second levels, while the bulk of the network's neurons went into sleep mode. This result also revealed that the attributes acquired through the second layer contained aliasing objects. Following the publication of these results, Zeiler and Fergus made adjustments to the topology of their neural networks. Scientists have optimised parameter values and reduced stride and filter sizes while keeping the distinctive qualities of the two initial convolutional layers as a result of their work. The network's performance has improved as a result of this restructuring of CNN's topology. Visualizing the features can be utilised to uncover design problems and make necessary parameter changes, according to a new organisation.

2.2.4.4. Visual geometry group (VGG)

Following the demonstration of CNN's utility in image identification, Simonyan and Zisserman offered a simple and efficient design strategy for CNN's architecture. This ground-breaking design was the result of the labour of a business called Visual Geometry Group (VGG). To recreate the network representational capacity linkages in greater depth, nineteen additional layers were added than in ZefNet [31] and AlexNet [32]. However, ZefNet was a trailblazing network in the 2013-ILSVRC competition, claiming that small-sized filters might increase the performance of a CNN. VGG replaced the previous layers of 5 x 5 and 11 x 11 filters in ZefNet with a layer of 3 x 3 filters. Surprisingly, our investigations shown that assigning small-size filters in parallel can provide the same outcomes as doing it alone. These small-size filters were discovered to have receptive field efficiency comparable to large-size filters, or higher (7 x 7 and 5 x 5). The use of small-size filters was also advantageous because the number of parameters used was reduced, lowering computing complexity. With these discoveries came the birth of a new research trend at CNN: the usage of small-size filters for research. VGG additionally modifies the complexity of the network by putting 1 x 1 convolutions in the middle of each convolutional layer. The features are then classified into linear groups based on how they are presented. In the network tuning procedure, the max pooling layer [33] and padding are used to maintain spatial resolution. The VGG algorithm performed wonderfully in picture classification and localisation. Due to its greater depth, more uniform structure, and user-friendliness they did not win the competition, but it has got attention to get first place in 2014-ILSVRC. VGG, on the other hand, has a prohibitively large computational cost due to the algorithm's utilisation of about 140 million parameters. Figure 2.10 depicts the network's organisational structure.

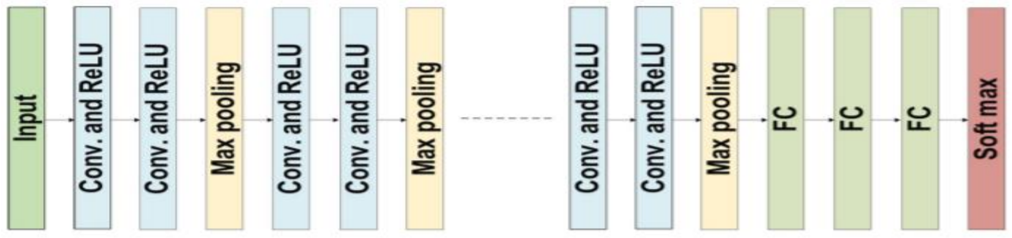


Figure 2.10 VGG Architecture

2.2.4.5. GoogleNet

GoogleNet won the ILSVRC competition (also known as Inception-V1). GoogleNet's architectural design prioritizes accuracy and low computation costs. Because it uses merge, transform, and split algorithms for feature extraction, it has developed a revolutionary inception block (module) concept for CNNs. Figure 2.11 depicts the conception block's architecture.

This design employs a variety of filter sizes to gather both channel and spatial information at varying spatial resolutions (5x5, 3x3, and one-dimensional). This NIN design was utilized to replace GoogLeNet's layers with microneural networks [35]. The GoogLeNet merge, transform, and split ideas were used to solve a problem involving various learning methods of variations emerging in a single label of different photos. We employed GoogLeNet to improve the learning capacity and efficiency of CNN parameters. It restricts computation by using a bottleneck layer of an unweighted 1 x 1 convolutional filter before applying big-size kernels. GoogleNet circumvented the problem of redundant data storage by adopting sparse connections. To reduce costs, it took the most direct path possible. Take care to only connect some of the inputs to the outputs. Rather than using an FC layer as the final one, a GAP layer was deployed to cut down on the total number of interconnections. After some fine-tuning, we were able to reduce the total number of parameters from 40,000,000 to 5,000,000. RmsProp optimization and batch normalization were two further examples of required consistency [36]. Convergence was also aided by the availability of supplemental learners. GoogleNet had a catastrophic flaw because to its complicated architecture and the requirement for constant correction as it expanded and matured. Outside of the representational confusion, GoogleNet has some other concerns. One of these was an early layer cutback, which occasionally resulted in vital data being erased.

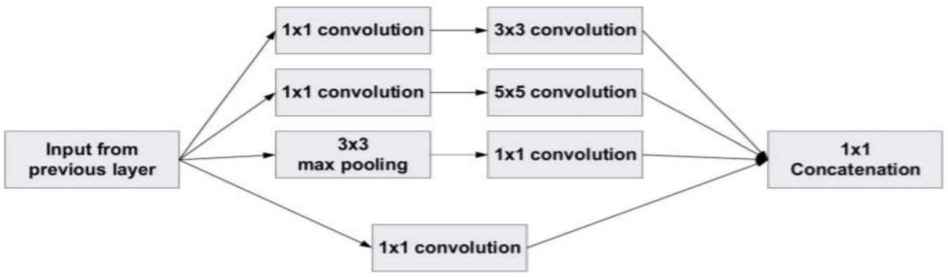


Figure 2.11 Structure of Google Block

2.2.4.6. Highway Network

Increasing the network's depth can improve performance. As a result, network training becomes more difficult. It is possible for dense networks to have gradient back-propagation errors with low gradient values because of the inclusion of many layers in deeper networks. It was for this reason that Srivastava et al. proposed a unique CNN design termed the Highway Network in 2015 [37]. For this method, cross-connectivity is a key component. It is possible for the Highway Network layer to freely transfer information thanks to two gating units within the layer. LSTM-based RNNs were used in the gate mechanism [38, 39].

By providing a regularization effect, which combines the data from the prior k layers with the data from the subsequent k layers, the gradient-based training of the deeper network is made simpler. The deeper network was trained using the data from the previous k layers. It is now possible to utilise the SGD method to train networks with more than 100 layers, such as one that has 900 levels. Thin and deep topologies are less resistant to failure since their convergence is higher than that of a 50-layer Highway Network. [41] found, on the other hand, that a conventional network's performance suffers when more than ten hidden layers are implemented. Even 900-layer Highway Network converges significantly faster than a basic network of the same size, according to this study.

2.2.4.7. ResNet

For their work on ResNet (Residual Network), He et al. [42] received the ILSVRC 2015 award. With this network, they hoped to create an ultra-deep one that would not be affected by the vanishing gradient issue (vanishing gradient problem). ResNet has been developed in a variety of forms with various numbers of layers (starting with 34 and increasing to 1202). ResNet50, the most prevalent form of the network, has one layer of FC and 49 convolutional layers. There were 25.5 million network weights and 3.9 million MACs on the network as a whole. In order to overcome the difficulty of training a deeper network in 2015, Highway Nets used the bypass pathway concept, which was first used in ResNet (Figure. 2.12). To over power the difficulty of training a deeper network, the bypassing of pathway concept was first employed in Highway Nets. In Figure 2.12, you can see how this is done. A typical feedforward network has been merged with a residual connection in the center of this network.

The (x_l - 1)th outputs of the layer before it, the residual layer outputs, are those provided by the preceding layer (x_l - 1).

F(x_l - 1) is obtained through the combination of many techniques [convolution with varying-size filters, batch normalization, and activation functions like ReLU (xl - 1)]. A mathematical expression of the final residual output is denoted by the letter xl.

$$\mathrm{xl} = \mathrm{F}(\mathrm{xl} - 1) + \mathrm{xl} - 1 \qquad (1)$$

The residual network contains a number of fundamental residual blocks that are interconnected. Depending on the residual network design [42], different operations are performed in the residual block. Unlike the highway network, short-cuts within layers allowed for parameter-free and data-independent cross-layer communication between different layers in ResNet. The layers that show when a gated highway shortcut is closed indicate functions that aren't residual. The ResNet network, on the other hand, never closes individuality shortcuts, and residual information is always transmitted. ResNet's shortcut connections (residual links) are also capable of eliminating gradient decreasing difficulties because they promote deep network convergence. In order to win the 2015 ILSVRC, the team used more than 150 depth layers, VGG eight iterations and twenty times as much as AlexNet. VGG, on the other hand, has a lower computational complexity even though it has a greater depth.

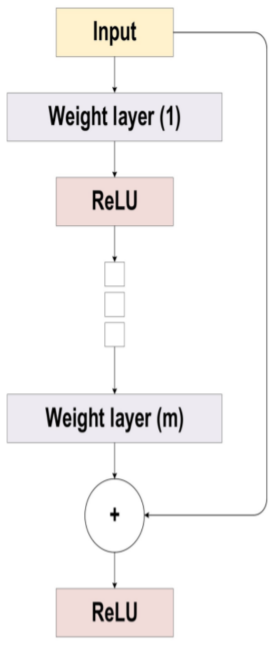


Figure 2.12 Resnet Architecture

2.2.4.8. Inception: ResNet and Inception-V3/4

Inception-ResNet and V3/4, which were enhanced versions of Inception-V1/2, were provided by Szegedy et al. [43,44]. When designing Inception-V3, reduced computational costs were a significant priority, yet extensive network generalization was not compromised [45]. As a result of these advancements, cross-channel correlation and classic convolution are now practically similar. Previous NIN design work by Lin et al. made advantage of the 1x1 filter potential. [45] used a similar technique in response. Inception-V3 maps the input data into 3 or 4 smaller spaces than the original input spaces using the 1 x 1 convolutional approach. These smaller areas are mapped using conventional 5x5 or 3x3 convolutions. As an alternative, Szegedy et al. in Inception-ResNet use the residual connection in place of the filter concatenation. [45,46]. Evidence suggests that Inception-generalization ResNet's abilities are on par with those of Inception-(Inception-4 V4's with residual connections) (with larger breadth and depth but no residual connections). Training the Inception network can be considerably accelerated by releasing residual connections.

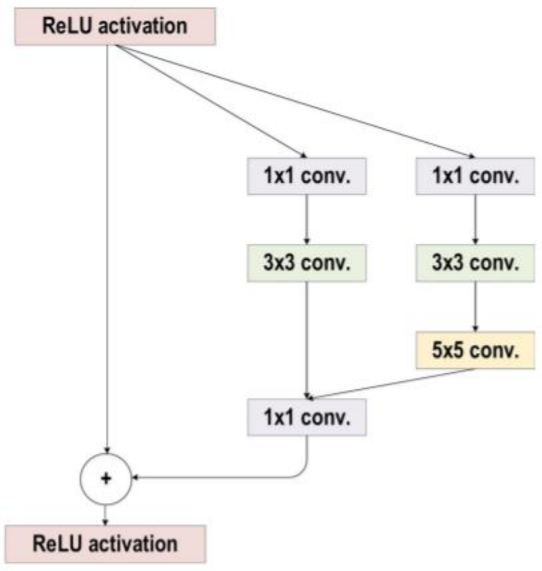


Figure 2.13 Inception-ResNet Architecture

2.2.4.9. DenseNet

DenseNet, like ResNet and the Highway network, was presented as a solution to the problem of vanishing gradients [47,48]. Despite the fact that many levels supply little or no information, ResNet appears to visually save information by maintaining individuality changes, which may be a flaw in the system. ResNet contains a large number of weights because every layer has its own set of parameters. A better solution could be found because to DenseNet's cross-layer connections. A feed-forward approach was used to connect the various layers of the network. Previous layers' feature maps were chosen as input for the new ones. DenseNet has l(l+1)/2 direct connections, whereas normal CNNs have l connections in the preceding and current layers. Using DenseNet's cross-layer depth-wise convolutions, one can see how deep convolutions work. By concatenating the previous layers' features, DenseNet is able to distinguish between new and conserved data. Due to DenseNet's thin layer structure and increasing number of feature mappings, the model is more expensive to implement. Loss functions allow each layer of the network to directly access gradient information in order to improve network performance. Additionally, it has a stabilizing impact, minimizing the likelihood of overfitting to a particular job or training program.

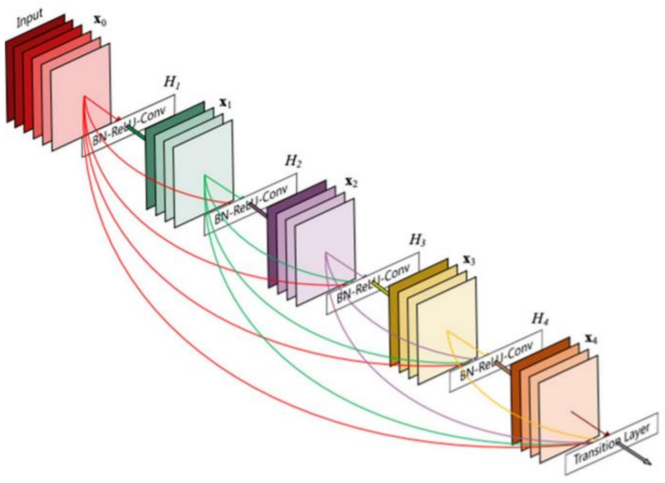


Figure 2.14 DenseNet Architecture

2.2.4.10. CapsuleNet

Conventional feature detectors are ineffective when it comes to identifying the properties of objects and delivering results that are well-behaved when it comes to object recognition.

Due to a number of CNN limitations, CNN does not assess the correlations between specific features such as orientation, size, and perspectives. Due to the CNN's inability to take into consideration the position of numerous components (such as the mouth, eyes, nose, and so on), it may inaccurately recognise a face image with many components (e.g. a face with multiple components, such as a face with multiple components). It is possible to think of a neuron that contains the property of probability in it in addition to its other qualities like its size, direction, viewpoint and so on.

Among the data that these neurons/capsules may detect is the presence of a face. Multiple stacks of capsule nodes are used to create the capsule network. Nodes in the CapsNet encoding unit fall into one of three types (the old version of the CapsNet). For each image in the MNIST architecture, there are 256 9x9 filters stride 1 and another 256 9x9 filters stride 1 contained within the same image. In the end, there are a total of 20 feature maps. A revaluated convolution layer now produces an 8-dimensional vector in the first capsule layer rather of a scalar in the traditional manner. It is necessary to apply Stride 2 in the first convolution layer in order to begin the image processing pipeline. As a result, the output file is 20 9/2 + 1=6 kilobytes large. Capsules measuring 32x8x6x6 were made by stacking 8x32 filters together in a rectangular shape (32 for groups, 8 for neurons, and 6x6 for neuron size).

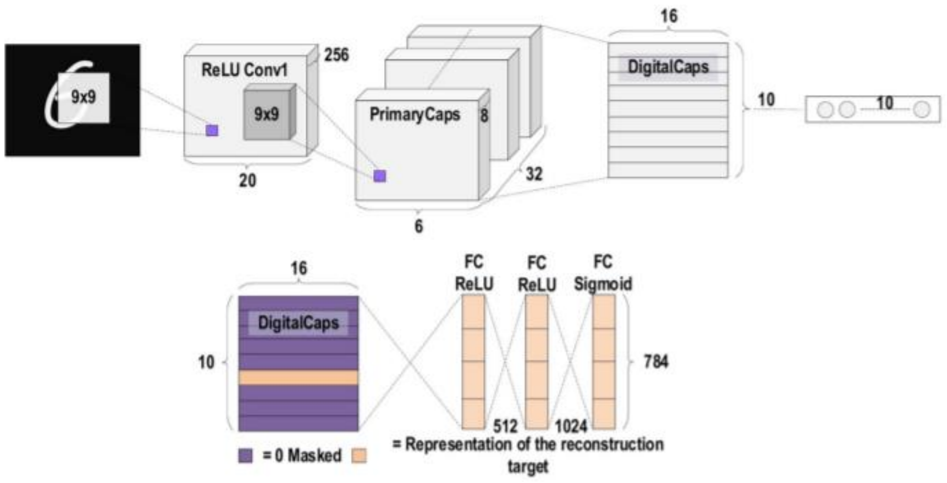


Figure 2.15 CapsuleNet Architecture

Figure 2.15 depicts the CapsNet encoding and decoding technology in great detail (right). Utilize max-pooling layers in CNNs for translation shift management. For instance, if a feature is still within its maximum pooling period, it is detectable. The method's ability to identify overlapping characteristics is extremely valuable for detection and segmentation tasks, which explains why it is so prevalent in these fields. A specific cost function is utilized in the training phase of conventional CNNs in order to determine how accurate the network's predictions are and when there is no longer any difference in weight between two neurons, neuronal activity will stop expanding and the system will return to normal. The dynamic routing of an agreement is based on feature parameters, as opposed to a single size given with all of its cost functions in mind. See [51] for more information about this architectural concept. You may discover more about this architectural design at [52]. This innovative CNN architecture improves the accuracy of distinguishing handwritten digits on the

MNIST. However, this design is recommended for segmentation and detection approaches rather than classification [52, 53].

2.2.4.11. High-resolution network (HRNet)

Semantic segmentation, item recognition, and human posture prediction require high-resolution representations of the scene in order to be accomplished. Via frameworks like VGGNet and ResNet, In order to encode the image as a representation with a lower resolution, a connected series of convolutions ranging from high to low resolution is utilized. A low-resolution representation needs to be used as a starting point for the creation of a high-resolution representation. All high-resolution representations are able to be brought up to date by utilizing a network that is known as a High Resolution Network (HRNet). This network stands out for two reasons: its sheer scale and the breadth of its user base. In order to get the best possible resolution, a variety of convolutions are combined. There is also a lot of information sent between resolutions on a very regular basis. An even more accurate and realistic portrayal of the world and a deeper semantic representation are also achievable. For example, HRNet can be used to identify and categorize items, but this isn't the only use case. HRNet provides a more secure foundation for computer vision issues. Figure 2.16 depicts the HRNet design.

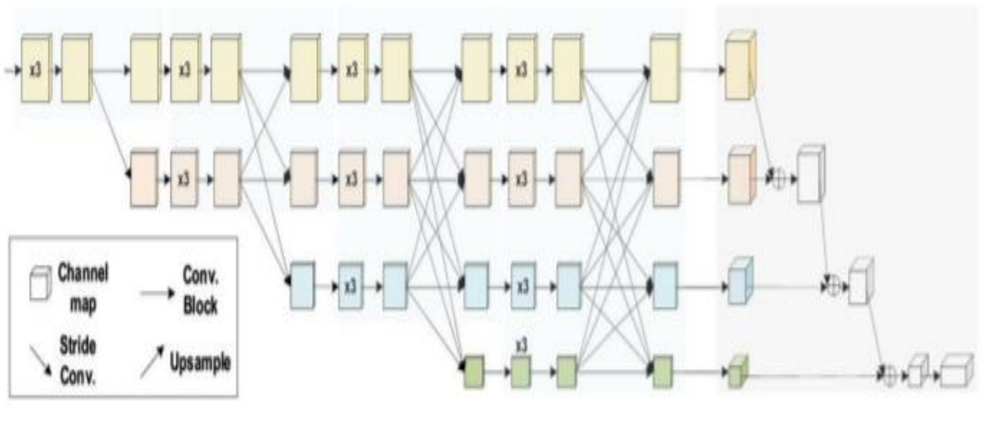


Figure 2.16 HRNet

2.2.5. Convolutional Neural Network Training: Step-by-Step Instructions (CNN)

To train a CNN, a large number of tagged images are required. There are numerous patterns (features) in images that the network can learn by seeing many of them, and as a result of all of that learning, it can make accurate predictions. The labelled set contains three distinct datasets: one for training, one for verification, and one for testing. The weights of the model are adjusted, and a predictive model is constructed using training data, which is then used in prediction. The hyperparameters of the system are only adjusted with the help of the validation set. The trained model is also subjected to an objective evaluation with the help of the test set. During training, the filter weights are adjusted using a technique known as backpropagation. Backpropagation includes the forward pass, loss function calculation, backward pass, and weight updating (in that order).

Each training session starts with the weights being assigned at random. A training image is initially transmitted through the network as part of the forward pass and is then utilized to generate an output image in the next stage. When comparing projected and actual outputs, a loss function is constructed depending on the difference. Following a study of the loss function, weights can be modified to reduce the loss function. This process is also known as "reverse pass." At this moment, the weights of each participant are updated in accordance with the prior activities. To keep track of your weight, use the following equation:

$$\mathrm{w} = \mathrm{wi} - \eta \frac{\mathrm{dL}}{\mathrm{dW}}, \tag{2}$$

$$\text{where } \mathrm{w} = \text{New weight}, \mathrm{wi} = \text{Initial weight},$$

$$\eta = \text{Learning rate}, \frac{\mathrm{dL}}{\mathrm{dW}} = \text{Derivative of loss function} \tag{3}$$

The η parameter is known as the learning rate. In the training process, the learning rate is an important factor that is determined by the user [54]. Because the weights are assigned at random at the beginning of the experiment, the learning rate is generally kept high [55, 56]. With a faster learning rate, the model will be able to converge in less time, which means it will be able to minimise the loss function more quickly. Learning occurs in iterations, and each iteration consists of the four processes listed above. We run a predetermined number of learning iterations in order to reduce the loss function to its smallest possible value, which is the ultimate goal of the training process. The generated model is then compared to the test dataset to ensure it is accurate. After discussion about the type of networks now we will discuss and focus more on the Computer vision based approaches.

2.3. Object Recognition, Object Detection, Localization & Classification

2.3.1. Literature Review

In computer vision, for example, image processing techniques are employed (CV). An additional benefit is the generation of high-dimensional data that is congruent with the natural environment. This is why computer scientists and engineers are working on CV research in attempt to create the most human-like computer vision system that they can. While this may be true for some images, for others, images can be represented mathematically in number of ways, i.e symbolically or numerically (statistical, geometrical and learning). Computer vision (CV) technology has numerous applications, including pattern recognition, medical imaging, industrial control systems, and physics. This discipline's subfields include event and Object Detection (OD), tracking, object recognition, scene reconstruction, picture restoration, video applications, and motion estimation [58].

Object detection is required for CV applications such as video stream identification, categorization, event detection, and object search (OD). It has become increasingly difficult to recognize an object of interest in a moving movie or a series of photographs thanks to smartphone and other gadget applications. Individuals, places, structures, or anything else of interest might be recognized for future research and decision-making [59].

Recognizing things in digital images is one of several jobs that go under the umbrella term "object recognition" in computer vision. Estimation of class is a term used in image classification to describe the process of establishing the class of a particular item inside a picture. "Object localization" refers to the process of identifying and outlining the boundaries of one or more items in an image. Several computer vision programmes make use of it. As a result of the employment of a technique known as object detection, these two goals are brought together.

As a result, we can differentiate three computer vision tasks:

- **Image Classification:** It's the act of determining the type or class of an object based on its appearance in a photograph.

Input: Photographs, for example, are single-object pictures.

Output: A class name (e.g. one or more integers that are mapped to class labels).

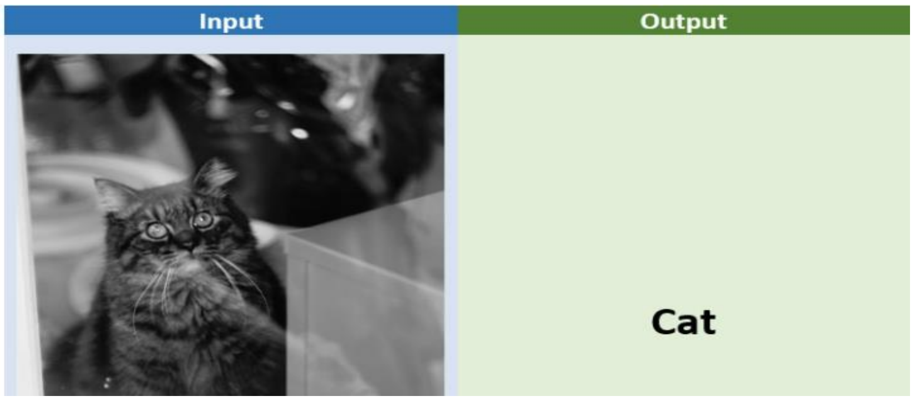


Figure 2.17 Image Classification

- **Object Localization:** A bounding box can be used to indicate the location of items in a photograph.

Input: A photograph, for example, is a picture that contains one or more objects.

Output: Boxes with a perimeter of at least one (e.g. defined by a point, width, and height).

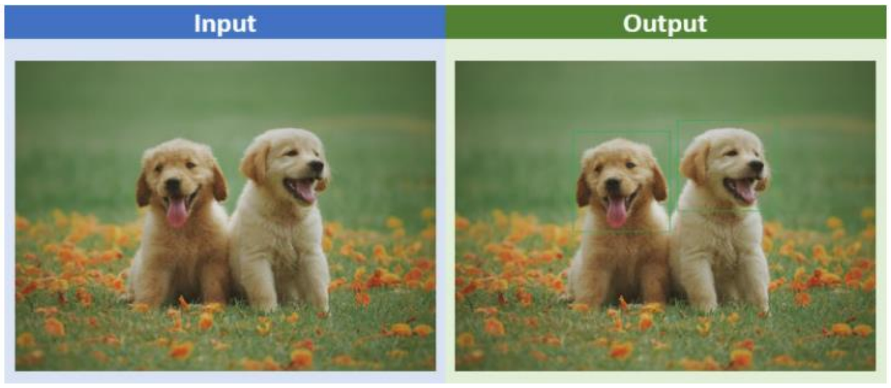


Figure 2.18 Image Localization

- **Object Detection:** Find things in a picture using a box and the types or classes of objects that are found.

Input: A photograph, for example, is a picture that contains one or more objects.

Output: As many or as few boundaries as you like, each with its own class label (such as "Width" or "Height").

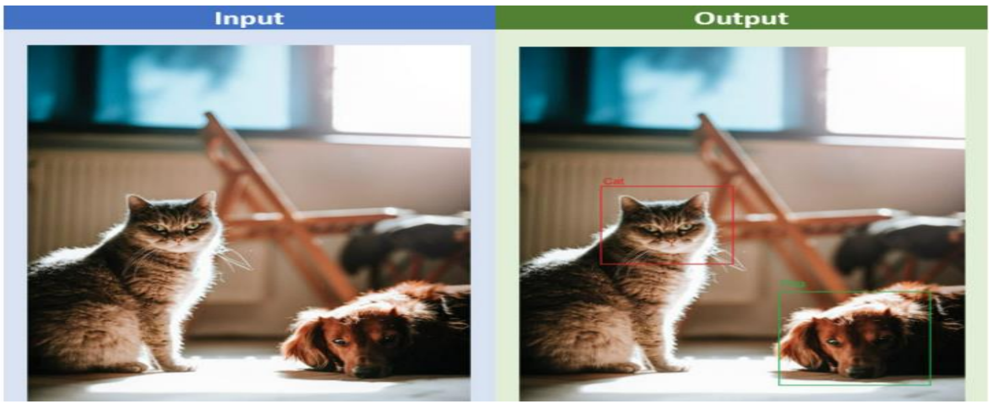


Figure 2.19 Object Detection

In a nutshell, optical object identification is still a computationally demanding and challenging problem to solve. In order to resolve this issue, the retina generates an infinite number of two-dimensional pictures of every object on Earth. These pictures are constantly changing depending on the viewer's viewpoint, direction, location, lighting conditions, and backdrops [60], which is a constant state of flux. It is possible to attribute the majority of these differences to the distortion of non-rigid visual properties. Another characteristic that exacerbates the situation is the presence of intra-class variations linked to form. The OD quality acquired by the CV is determined by the trade-off between time and precision (such as detection rate, error rate, time and precision metrics). The Figure 2.20 shows the complete flow chart of the computer vision task in which the object is first recognized and then followed by the image classification and image localization. Once the above activities gets completed it performs the object detection. After successful completion of object detection, the image segmentation process will come into the picture, which basically do the image segmentation for the region based images and detect the images based on the desired image as per classes and labels.

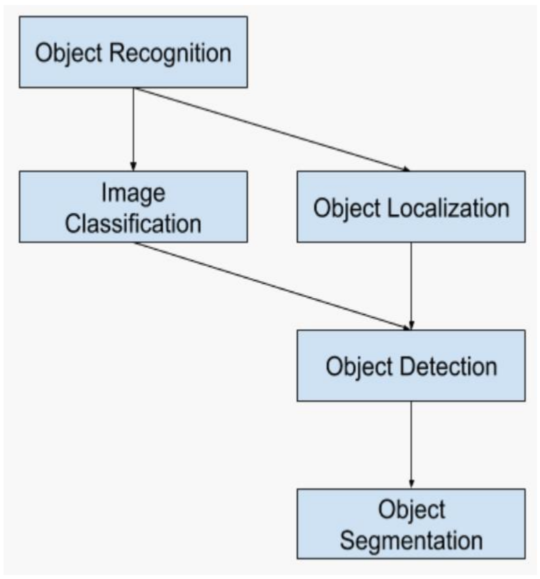


Figure 2.20 Overview of Object Recognition Computer Vision Tasks

2.3.2. Object Localization

In addition to image classification and object recognition, other image recognition tasks include object localization. In spite of common usage, "object detection" & "object localisation" are distinct processes. It is important to note that the concepts of picture categorization and image localisation are distinct.

Image localization, a subset of CNN vision techniques, is used to find objects in images. In order to forecast classes, these techniques use discrete integers. There are four continuous numbers used in object localization: x/y coordinates (height and width), height and width. Convolutional neural networks are the starting point for CNN-based classifiers (CNNs). In order to support a variety of applications and processing resources, many of these networks may contain a few to 100 layers (for example, ResNet 101). There are so many materials to keep track of that it's difficult to keep track of them all. This is followed by one or two additional levels that are completely connected before the CNN layers arrive at the final one. Whether or whether an object appears in the final image is determined by the output layer, the last one before rendering. What might happen in the following scenario? In a single shot, an algorithm can detect up to 100 distinct items. Each number in the last layer, which spans a distance of 100, represents the probability of an object showing up in a photograph.

Except for the output layer, all aspects of an image localization algorithm are the same. Probability values might range from 0 to 1 depending on the classification technique employed in the final layer. Regression problems necessitate that localization methods return four positive real values as their output. A box is drawn around the subject object by using this set of four numbers.

2.3.2.1. The loss function for object localization

Almost all machine learning algorithms aim to produce predictions as close to perfect as feasible given the input data. The loss function is used by every supervised machine learning algorithm to guide the process of learning, which includes modifying weights or parameters to improve performance. In this case, any regression loss function that can be applied to an N-dimensional array will work because object localization is a regression problem. The losses of L1 distance encompass, but are not limited to, the losses of L2 distance, the loss of Huber distance, and so on. The L2 distance loss method is widely used in both industry and academics for measuring distance loss.

2.3.2.2. L2 distance

In some circles, the L2 distance is also referred to as the Euclidean distance. Using point cartesian coordinates, the Pythagorean theorem can be used to calculate the distance between two points in N-dimensional space. This method is applicable to any N-dimensional space with a positive dimension.

To understand the idea of L2 distance, consider two points P and Q in three dimensions. P and Q are expressed in cartesian coordinate systems as (p1, p2, p3) and polar coordinate systems as (q1, q2, q3), respectively. The distance among these two points can be calculated using the following formula:

$$\text{Distance}\,(P, Q) = \text{sqrt}\,((p1 - q1)^{2} + (p2 - q2)^{2} + (p3 - q3)^{2}) \qquad (4)$$

This method is more effective when the difference between anticipated and true values is smaller in terms of L2. Its purpose is to reduce the Euclidean distance between anticipated and true values as much as possible.

2.3.2.3. Image Localization Algorithm Evaluation

It is vital to test data points that have never been observed before while developing machine learning models for application in the actual world. In assessing the efficacy of various models based on real-world data, it serves the same purpose as previously as an assessment metric. The IoU, a commonly used metric for evaluating image localisation quality, can be utilised to do so. Intersection over Union is a term used to describe this idea (IoU). When computing IoU, both the current and predicted bounding boxes are taken into account.

An IoU measure measures the amount by which a given ground truth bound box and its anticipated bounding box actually overlap, whereas a union measure measures the amount by which that overlap actually occurs between the two parties. Both bounding boxes are added together, and their combined total area is the union.

It's normally between 0 and 1 when it comes to IoU For a perfect performance, which is a value of 1, it means that the prediction and the real-world values are indistinguishable. When the junction region is greater, IoU is more likely to be closer to one.

2.3.2.4. Various IoU Losses for Faster and More Accurate Object Detection

BBR (bounding box regression) is used by the Object Localization module to locate objects.

Bounding Box Regression

For anticipating the location of a target item, bounding box regression is a common method in object recognition systems. It aims to improve the predictability of the position of a bounding box. When the predicted bounding box and the ground truth bounding box coincide sufficiently, bounding box regression uses IOU-based losses.

Researchers recommend either a better structural backbone [63, 64] or a better technique for obtaining trustworthy local information [61] from data to improve the performance of deep neural network-based applications.

Intersection over Union

In order for IoU loss to be effective, the anticipated and true bounding boxes must both overlap. When there are no overlapping instances, IOU loss does not result in a shifting gradient. Due to the enormous quantity of IOUs, the IOU loss has a slow convergence rate.

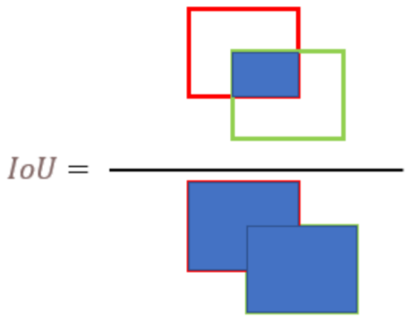


Figure 2.21 The predicted bounding box is red, whereas the ground truth bounding box is green.

A negative result indicates that the test failed since the ground truth boxes did not overlap with the predicted boxes when IoU loss was expected.

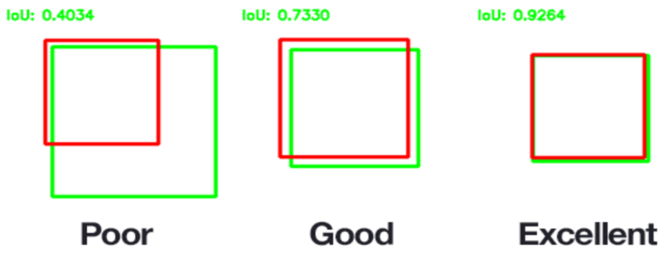
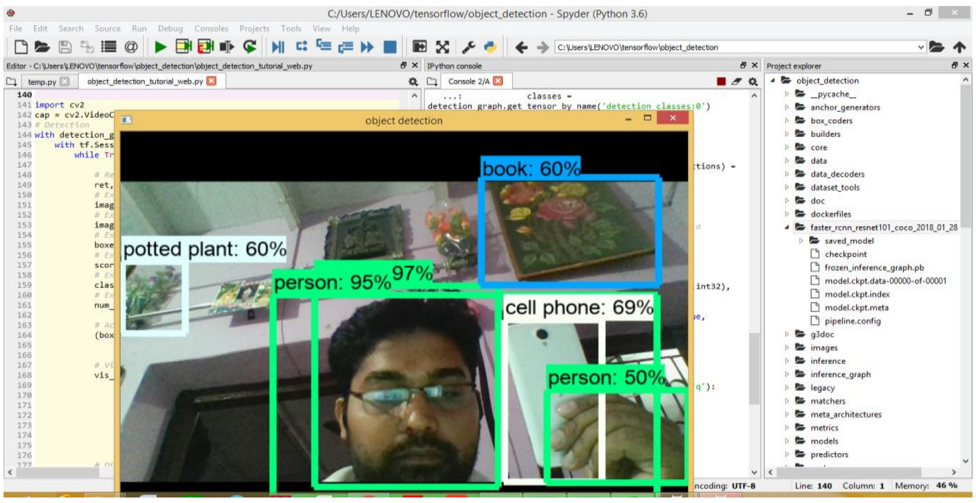


Figure 2.22 Goodness measure of IOU

2.3.3. Object Detection

The three core features of a complete optical character recognition system are feature extraction, object recognition, and object localisation. Each of these factors influences the overall performance of any OD system, regardless of its type. Previous study had mostly concentrated on a single aspect of the OD system. Supplemental variables like as scale and rotation might produce data imbalance, which can have a negative impact on the OD system's performance. Furthermore, the performance of the OD system could be enhanced by employing a novel learning descriptor with rapid and unique invariant properties that can be used to define the environment. As ensemble-based learning develops and advances, it becomes less prone to data imbalance. Furthermore, any changes to the detector used to detect the initial interest spots may result in better overall detection performance.

Face recognition and object class detection in images (for example, faces, automobiles, people, and so on) are two applications of object detection (OD) [63-64]. Item recognition, on the other hand, is the process of categorising an instance of an object into a specific class. Detecting objects in images can be accomplished by combining CV and image processing techniques [61]. Furthermore, machine learning could be applied to enhance an Object Detection system. Object detection, video object recognition and tracking, picture identification, image restoration, image retrieval, camera positioning, and 3D scene reconstructions are just some of the many uses for this technology[62][63].


C:/Users/LENOVO/tensorflow/object_detection - Spyder (Python 3.6)
object detection
book: 60%
potted plant: 60%
person: 95%
97%
cell phone: 69%
person: 50%
object_detection
faster_rcnn_resnet101_coco_2018_01_28
frozen_inference_graph.pb

```
import cv2
cap = cv2.VideoC
# Detection
with detection_g
    with tf.Sess
        while Tr
```

```
classes =
detection_graph.get_tensor_by_name('detection_classes:0')
```

Figure 2.23 Object Detection

During the object detection process, the following actions must be carried out on a regular basis:

1. Input is provided by a visual medium, such as a photograph or a video.
2. Divide or segment the input visual into portions or regions by segmenting or dividing it.
3. Separate each section and treat it as a single image while working on it.

4. These photos were analysed using our Convolutional Neural Network (CNN).
5. Following classification, we can combine all of the photos to obtain the original input image, as well as the items that have been identified and their corresponding labels.

2.3.4. Feature Extraction

Photographic features can be used to identify an object more quickly by looking for a specific portion or pattern. It is easy to identify a square because it has the characteristics that make it easy for us to recognise it as such: four corners and four edges. Some of a feature's characteristics include its corners and edges, as well as any notable features such as ridges or depressions.

Traditional Feature Detection Techniques: A Sneak Peek at What They Look Like:

- For detecting corners in a scene, a Gaussian window function is used, and this method is known as Harris Corner Detection.
- The Researchers improved the corner detection approach by updating the scoring mechanism used in the Harris Corner Identification algorithm..
- Scale-Invariant Feature Transform- Unlike the preceding two techniques, this one is scale invariant.
- Simplified Robust Features (SURF)- As the name implies, this is a simplified version of the SIFT algorithm.
- Accelerated Segment Test (FAST) characteristics- When compared to SURF, this is a significantly faster corner detecting algorithm.
- In order to describe binary robust independent elementary features, one can use this feature descriptor in conjunction with any other feature detector (BRIEF). By converting floating point integers to binary strings, this method reduces the amount of memory needed for descriptors.

Extraction of Deep Learning Features: A Peek Behind the Scenes

CNNs have the ability to learn the characteristics of a given task, which enables them to take the place of traditional feature extractors. CNNs are also superior to traditional feature extractors when it comes to discovering complicated characteristics that more accurately reflect a picture.

As an example, here are a few:

- **For example, SuperPoint can detect and describe interest points on its own.**

The Researcher' fully convolutional neural network can find interest points and descriptors that are similar to SIFT in just one forward pass. With a VGG-style encoder and two decoders, you can pull out the features you want.

- **D2-Net, a joint description and detection of local features, is a trainable CNN that can be used for this purpose**

One convolutional neural network is proposed by the researcher as a dense feature descriptor and a feature detector.

- **LF-Net: Learning Local Features from Images**

The researchers suggests an end-to-end training strategy and a sparse-matching deep architecture for training on image pairs containing relative pose and depth maps. They use the initial image as a reference after running their detector to pinpoint the peak activity. Changing the weights and rerunning on the second image yields a response map with clean maxima in the right places.

- **Deep Learning-Based Image Feature Matching**

For feature matching, they employ a Deep CNN model based on the patch image. Future feature matching could be made easier with the introduction of a graph neural network that transforms places of interest into features that can be more easily identified. Deep learning feature extraction algorithms are more resistant to scaling, occlusion, and distortion than traditional computer vision approaches, however this does not mean that traditional computer vision techniques are no longer useful.

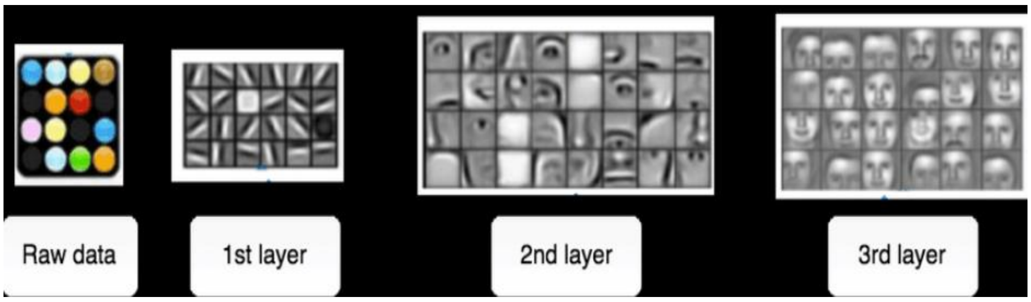


Figure 2.24 Feature hierarchy discovered by a deep learning network applied to facial images

2.3.5. Object Segmentation

The method of giving a particular class to each pixel value in an image is referred to as object segmentation or picture segmentation. New papers and algorithms are being developed on a daily basis to increase the performance of these intelligence systems' object segmentation approaches. In order to get a basic knowledge of the problem of image segmentation, you can benefit greatly from reading this article.

Real-world applications:

1. **Face segmentation:** Computer vision systems can benefit from semantic segmentation for tasks like facial expression detection, age recognition, and determining a person's gender and ethnicity. Semantic segmentation, which splits facial features including the mouth, chin, nose, eyes, and hair into distinct attributes, enables these tasks.
2. **Medical imaging:** Image segmentation is utilised in medical diagnostics, particularly in the evaluation of X-rays and MRI images. Semantic segmentation systems can assist in classifying significant portions of a picture, making diagnostic examinations easier and simpler.
3. **Self-Driving:** Autonomous driving is a very difficult undertaking that involves real-time sensing, analysis, and adaptation. Semantic segmentation is used to recognise things like as other automobiles and traffic signs, as well as areas such as road lanes and walkways. In autonomous driving, instance

segmentation is used to segment individual autos, pedestrians, signs, and so on.

4. **Satellite image processing:** Aerial or satellite photographs cover a huge region of land and include numerous items. Semantic segmentation has been used to analyse land utilisation, regions affected by deforestation, agricultural land analysis, and so on.

Segmentation vs. Detection: What's the difference?

Each class in the image will have its own bounding box, which can only be created by employing Object Detection models. However, because the bounding boxes are rectangular or square in shape, it will not reveal anything about the object's shape. Image segmentation will generate pixel-by-pixel masks for each item, allowing you to learn about the object's granular characteristics.

Image Segmentation & its types:

1. Semantic Segmentation
2. Instance Segmentation

In semantic segmentation, a label is assigned to every one of the pixels in an image. In contrast, categorization assigns a single title to the entire image. Semantic segmentation treats many objects from the same class as a single entity. instance segmentation, on the other hand, treats many objects of a single class as distinct entities (or instances). In most cases, instance segmentation is more difficult than semantic segmentation.

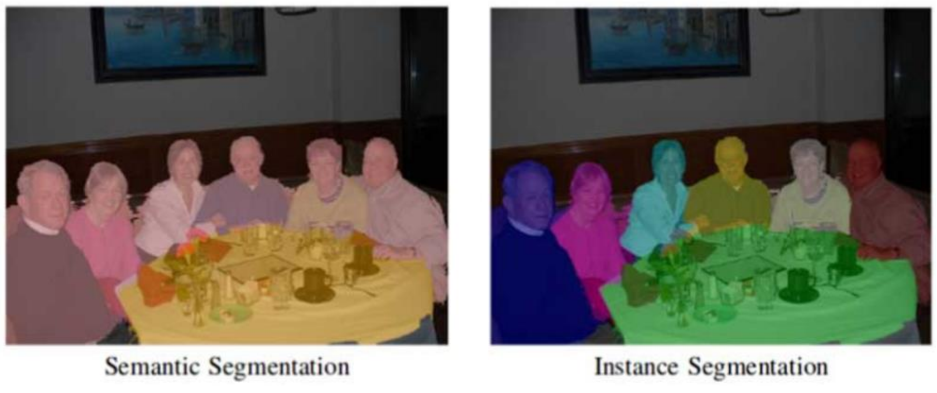


Figure 2.25 Instance Segmentation Vs Semantic Segmentation

2.3.6. Object Detection Algorithms

As self-driving cars have risen to prominence, a multidisciplinary field known as computer vision has grown rapidly in recent years (since CNN covered it extensively). Another important aspect of computer vision is the ability to identify objects. Pose evaluation, vehicle recognition, and surveillance can all benefit from object detection. Unlike classification algorithms, object detection algorithms draw a bounding box over an object of desire in order to locate it within an image.

If you're trying to detect objects in a picture, you may not be able to build a single bounding box; instead, there may be a number of separate bounding boxes that reflect different aspects of the image. Prior to the emergence of Deep Learning, object detection progressed in two distinct stages:

Traditional object detection was used up until 2014.

1. The Viola-Jones Detector (2001), for example, laid the groundwork for the development of standard object detection algorithms.
2. To begin with, there is the HOG Detector (2006), a computer vision and image processing feature descriptor introduced in 2006 for object recognition.
3. DPM (2008), who was the first to use bounding box regression in a statistical analysis.

Most important one-stage object detection algorithms

1. YOLO (2016)
2. SSD (2016)
3. RetinaNet (2017)
4. YOLOv3 (2018)
5. YOLOv4 (2020)
6. YOLOR (2021)

2.3.6.1.Histogram of Oriented Gradients (HOG)

Introduction

A HOG was used in an early approach of recognizing items. It was originally made available to the public in 1986. This method wasn't widely used in computer vision applications until 2005, despite incremental advancements over the following ten years. In order to recognize objects in photos, HOG makes use of a feature extractor. HOG's feature descriptor can be used to extract only the most important information from a given image, while discarding the rest of it. Using a feature descriptor, an image's overall size is transformed into an array or feature vector. As a result, using HOG's gradient orientation method, a picture's most significant elements can be detected and highlighted.

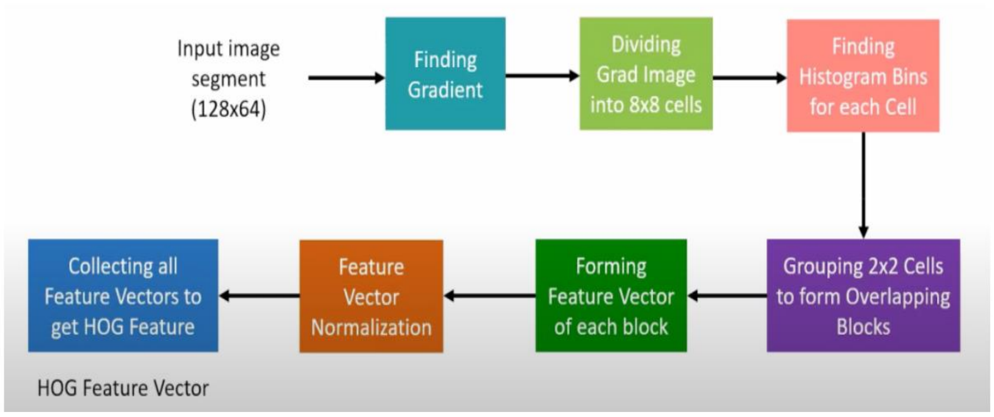

Input image segment (128x64)
Finding Gradient
Dividing Grad Image into 8x8 cells
Finding Histogram Bins for each Cell
Grouping 2x2 Cells to form Overlapping Blocks
Forming Feature Vector of each block
Feature Vector Normalization
Collecting all Feature Vectors to get HOG Feature
HOG Feature Vector

Figure 2.26 HOG Architecture

Before we can understand how HOG operates, we need to first comprehend what it is and how it operates. Accordingly, the gradient histogram is generated by taking into account both vertical and horizontal values while creating the feature vectors for each pixels in an image. The value of a pixel can be deduced from its horizontal and vertical surroundings using gradient magnitude and gradient angle. We'll focus on a small section of a bigger image, as indicated in the illustration. When it comes to calculating gradients, the entire image is separated into 88 gradient representations, each of which is used. We can create a histogram for a given area by using the 64 gradient vectors to divide each cell into angular bins. Using this strategy, 64 vectors can be condensed down to just nine values.

After determining the size of the 9-point histogram values (bins) for every cell, we may determine whether or not to construct overlaps for blocks of cells. The necessary process in creating a HOG feature are to build feature blocks, normalize feature vectors and aggregate related feature vectors.

HOG's Achievements

1. Creating a feature description to be used in object detection.

2. Possibility of using support vector machines (SVMs) to detect objects with high precision.

3. Utilization of a sliding-window method for determining location.

Consider the following:

1. **Limitations:** There were numerous flaws with HOG, which was innovative in the early stages of object identification, but it was not without its merits.
2. It takes a long time for complicated pixel processing in photos, and it is unsuccessful in some object recognition scenarios with smaller regions.
3. HOG is a good baseline for comparing the efficacy of different object detection methods. Regardless, HOG is an effective method for numerous applications in the field of object and facial identification.
4. Due to its smooth borders, one of the most common applications of HOG is pedestrian detection. Detection of certain items is one of the other general uses.

2.3.6.2.Region-based Convolutional Neural Networks (R-CNN)

For the purpose of object detection, RCNN have taken the role of the more traditional methods of HOG and SIFT to extract the most crucial characteristics from the data, we use R-CNN models with selected features (often about 2000 features). Using a selective search algorithm, it is possible to determine which extractions are most important and then use that information to compute the procedure.

Working of R-CNN

Using the selective search technique, we can generate multiple image subsegments at once and pick the best possible entries for our task, making it easier to zero in on the most important features. This indicates that the greedy technique can be used to combine smaller segments into larger segments frequently in order to combine the most effective entries.

After the background check has been finished effectively, we must extract essential attributes and generate appropriate projections for these new prospects. CNN can be used to build an n-dimensional feature vector (2048 or 4096 dimensional) to extract features by using a pre-trained CNN model.

Accurate picture predictions are made using R-CNN, which then labels the box associated with the image. For each activity, the bounding box classification is tweaked using a regression model that takes into account where more research should be conducted.

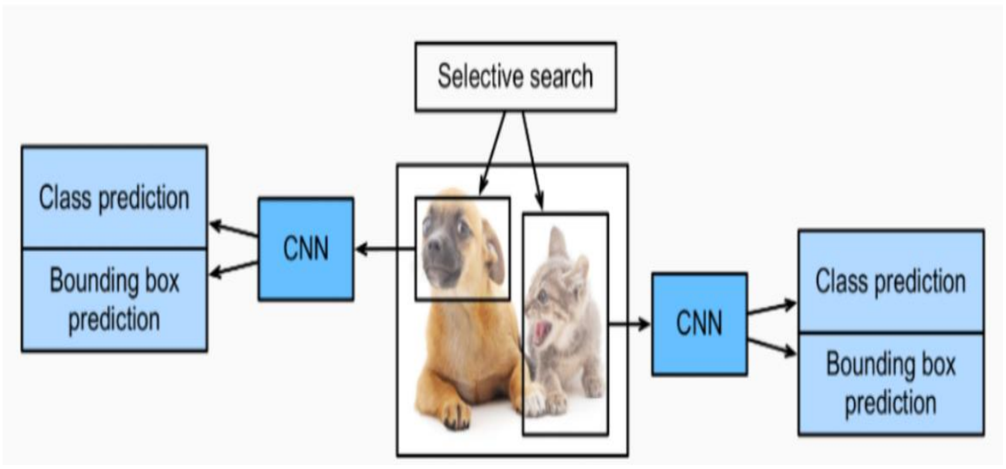


Figure 2.27 RCNN Architecture

Issues with R-CNN

1. Pre-trained CNN models give good feature extraction results, but current methodologies are incredibly slow when it comes to extracting all area proposals and, ultimately, the best regions.
2. The R-CNN model has a number of drawbacks, including its slow training rate and extended prediction time. As a result of the solution's reliance on extensive computer resources, the procedure is more readily implementable. As a result, the entire architecture may be judged to be extremely expensive.
3. Due to the inability to make modifications in the introductory phase, poor candidate selections may frequently occur. This may cause a multitude of problems in the trained model.

Consider the following:

1. **When to Use R-CNN:** The performance of object identification models should be compared to the R-CNN and HOG object detection techniques as a starting point. R-most CNN's recent version is generally recommended because picture and object predictions typically take longer than expected.
2. **Example usage cases:** It is possible to use R-CNN to address a wide range of object detection problems. The use of drone cameras to keep an eye on objects, image recognition software to recognize writing, and Google Lens to identify objects are all examples. Please see the following page for more information.

2.3.6.3.Faster R-CNN

In spite of its ability to execute object identification calculations and give acceptable results, R-CNN has certain fundamental flaws, particularly in terms of speed. To overcome R-CNN issues, some difficulties had to be dealt with more swiftly. CNN's shortcomings were remedied with Fast R-CNN. Instead of examining each segment separately, we employ pre-trained Convolutional Neural Networks. To generate the output of a fully connected layer, RoI pooling employs two pre-trained models and a selective search strategy. The Faster R-CNN network, an upgraded version of the Fast R-CNN, is the topic of this article. In terms of performance speed, the Faster R-CNN model exceeds its predecessors by a large amount. In contrast to the R-CNN and Fast R-CNN models, the Faster R-CNN technique generates region recommendations using a superior network rather than a selective search algorithm. The region proposal network (RPN) employs a diverse set of photos of varying sizes to produce high-quality results.

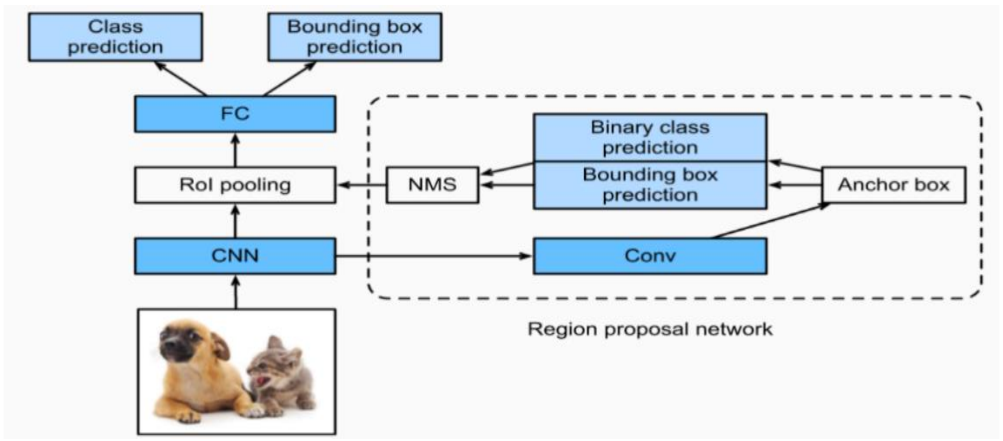


Figure 2.28 Fast RCNN object detection Algorithm

As opposed to taking 10 milliseconds to process a photo, the regional proposal network does it in just 10 microseconds! The convolutional layer of the network can provide feature mappings for each pixel. Feature maps' anchor boxes come in many shapes and sizes. A bounding box is constructed for each anchor box based on its expected binary class.

Non-maximum suppression must be used to remove superfluous data from feature maps. Non-maximum suppression is the only difference between Fast R-CNN and this approach.

Consider the following:

1. **Limitations:** The degree of time delay in proposing distinct items is a key constraint of the Faster R-CNN approach. The design of a system can have an impact on its performance.
2. **When To Use Faster R-CNN?:** The prediction time is quicker than with other CNN algorithms. Compared to R-CNN, Fast R-CNN takes about 2 seconds to predict things in a picture, while Faster R-CNN offers the perfect answer in just 0.2 seconds.
3. The R-CNN approach provides use case examples that are faster, but the R-CNN approach also provides faster results. Faster R-CNN, on the other hand, allows us to accomplish these tasks faster and more successfully.

2.3.6.4.Single Shot Detector (SSD)

An approach known as the single-shot detector for multi-box predictions is an excellent example of a technique that is particularly successful for real-time object identification. Although excellent prediction accuracy may be achieved with Faster R-CNN approaches, a real-time job running at an average frame rate of 7 frames per second is substantially slower than desirable.

An increase in frame rate roughly five times greater than that of the Faster R-CNN model alleviates this issue by improving performance. Multi-scale features and default boxes have taken the place of the old region proposal network.

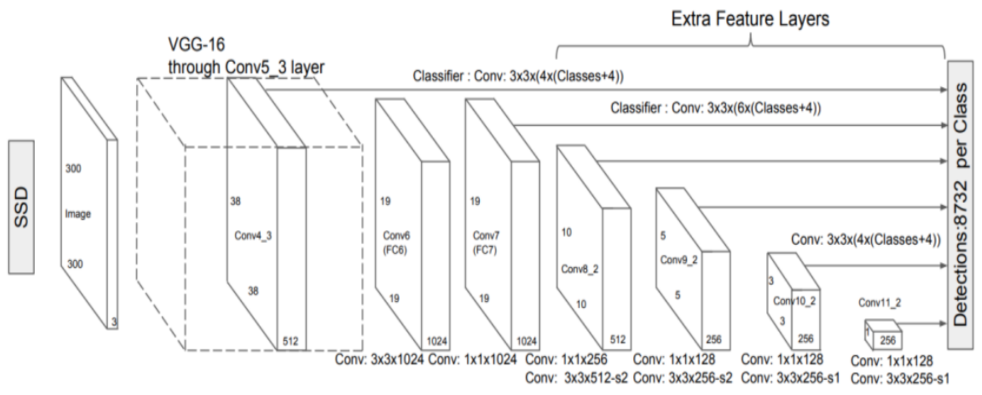


Figure 2.29 SSD Architecture

In the architecture of the single-shot multibox detector, there are three major components. It is necessary to select all of a single-shot detector's critical feature maps before it can begin to extract features from them. It is worth noting that there are no additional layers in this particular architectural zone, which is made up entirely of convolutional layers. As soon as any and all relevant feature maps have been extracted, the process of identifying heads can get underway. A fully convolutional neural network (FCNN) is also included at this level of sophistication.

The second stage of detecting heads is not responsible for deciphering the meaning of a picture's semantic significance, to be clear. However, when it comes to generating the most accurate bounding maps for all of the features, this is not the primary concern. Non-maximum suppression layers are used to lower the error rate caused by repeated computations of the bounding box after the two fundamental phases have been computed.

SSD Constraints

1. In spite of the SSD's improved performance, image resolution is reduced to a lesser standard.
2. For small-scale objects, the SSD architecture often outperforms the Faster R-CNN.

Consider the following:

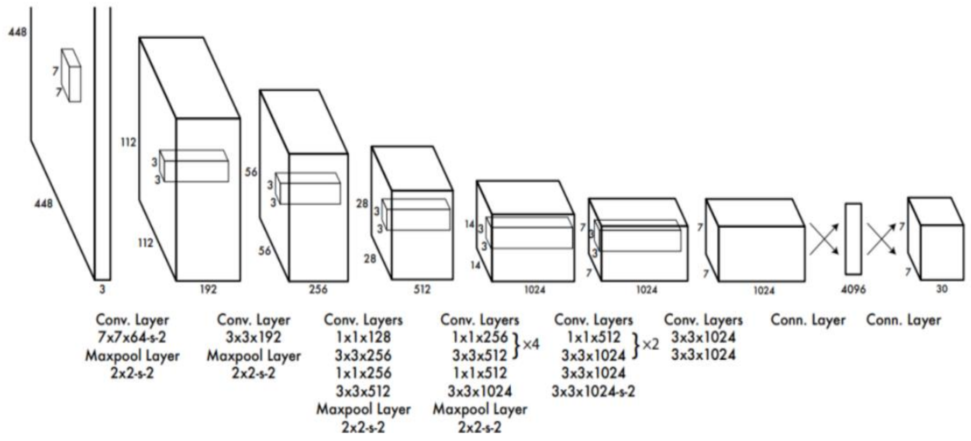
1. There are times when it's better to use the single-shot detector. When it comes to larger objects, where precision is less important, using a single-

shot detector makes sense because it allows us to make faster predictions on a picture. For more accurate estimates of small and precise goods, however, other methods must be investigated.

2. Many datasets, including PASCAL VOC, COCO and ILSVRC can be used to train and test the Single-shot detector. This means that they can pick up on larger objects such as humans and other living things such as tables and chairs.

2.3.6.5.YOLO (You Only Look Once)

If you're looking for the most extensively utilized models and object detection algorithms, look no further than "You Only Look Once" (YOLO). To find object detection algorithms on Google, the YOLO architecture is the most common concept that pops up. YOLO manifests itself in a variety of ways. We have gone into more detail about each of these manifestations in the following sections. In addition to having one of the most advanced neural network architectures available, YOLO's neural network architecture allows it to achieve excellent accuracy while also processing data at a high rate overall. Most of its appeal stems from the speed and precision with which it performs its functions.



448
7
7
448
3
112
3
3
112
192
56
3
3
56
256
28
3
3
28
512
14
3
3
14
1024
7
3
3
7
1024
7
7
1024
4096
7
7
30
Conv. Layer 7x7x64-s-2 Maxpool Layer 2x2-s-2
Conv. Layer 3x3x192 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x128 3x3x256 1x1x256 3x3x512 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x256 3x3x512 }×4 1x1x512 3x3x1024 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x512 3x3x1024 }×2 3x3x1024 3x3x1024-s-2
Conv. Layers 3x3x1024 3x3x1024
Conn. Layer
Conn. Layer

Figure 2.30 You Only Look Once (YOLO) Architectur*e*

In order to detect objects, the YOLO architecture relies on three fundamental concepts.

This model's speed and accuracy are unmatched by any other object detection algorithm, and that's because of the three strategies listed above. The YOLO model is based on the idea of leftover blocks. For this picture, they used the 7 × 7 blocks that were left over from the first construction plan.

For each of these grids, a particular prediction is prepared based on the location of these grids. Using the second method, each forecast's center point is taken into account when determining the size of the bounding boxes. The classification tasks for each grid work well, but it is more challenging to identify the bounding boxes for each forecast. As a last resort, the intersection of unions (IOU) method is used to find the best bounding boxes for an item identification task.

The Benefits of YOLO

1. Even in real-time, YOLO's calculation and processing speed outpaces most other training approaches and object recognition algorithms.
2. In addition to its quick processing speed, the YOLO technique achieves overall high accuracy by reducing background mistakes present in other approaches.
3. An efficient architecture allows YOLO to learn and understand a large number of items in little time.

YOLO's Limitations

4. The inability to notice tiny things in an image or video due to a decreased recall rate.
5. Due to the restrictions of bounding boxes, it is impossible to identify two items that are extremely near to each other.

Consider the following:

1. **When Should You Use YOLO?** There are many ways to recognize objects in images and videos, but the YOLO architecture is the most often used method for real-time object detection. Depending on the hardware, it can achieve excellent accuracy in most real-time processing processes while maintaining a reasonable speed and frame rate.

2. Other significant use cases of the YOLO architecture include vehicle detection, animal identification, and person detection, in addition to object detection on a range of products.

2.4. Hyperparameters

One of the most challenging aspects of machine learning is determining the ideal model complexity. Models with a great deal of complexity can make excellent use of the data, but the unobserved information is not appropriately generalized (overfitting). To get all the information from the data may be difficult with a simple model (underfitting). There is a direct correlation between model performance and hyperparameter values in the context of deep or machine learning. Therefore, hyperparameter exploration aims to search through a vast number of different hyperparameter configurations in order to find the one that gives the maximum overall performance. Most hyperparameter research is done manually, and it takes a long time, because there is a large search area and each configuration is expensive to analyze.

These two sorts of parameters are employed in machine learning models. They are: Hyper-parameters are any and all variables that a user can specify at his or her discretion prior to beginning the training. Modelling parameters (also known as learnable parameters) are parameters that the model learns during training.

Model parameters are used to convert input data into desired outputs, whereas hyperparameters are used to explain the structure of the model that is currently being used to translate data. The most extensively used learning algorithms have a set of hyperparameters that must be defined before any training can occur. These hyperparameters are known as learning parameters. Different training algorithms employ hyperparameters in different ways; others, such as the ordinary least squares approach, do not even require the usage of a hyperparameter. In terms of the time it takes to train a model, hyperparameters can have a significant impact. The sort of hyperparameter that is best suited for the situation at hand must be considered while selecting one. Parameters and hyperparameters are distinguished by dimensionality.

These are the machine learning hyper-parameters linked to network structures:

1. The algorithm's input and output are separated by a voluminous number of tiers. The extent to which additional hidden layers can increase accuracy varies depending on the situation.
2. **Dropout:** This is a regularisation strategy that reduces overfitting and improves validation accuracy. It also displays how much of each epoch's neurons should be removed at random to prevent overfitting.
3. **Weight initialization:** Depending on the activation function for each layer, different weight initializations are best employed. It is critical to specify initial weights for the first forward pass.
4. **Activation functions:** These are used to incorporate nonlinearity into models. Nonlinear prediction bounds must be learned using deep learning models.The following strategies are used to optimise hyperparameters:

Types of Search

1. **Manual search:** Historically, hyperparameters were determined through trial and error. It is still done in the same way, with experienced operators speculating on parameter values that can result in high accuracy in deep learning models. It is simple and effective in the hands of skilled operators, but it is not a scientific method. There is always a search for better, more automated solutions.
2. **Grid search:** Using a grid search instead of manually adjusting is a major advantage. For each hyper-parameter, the model is automatically trained to evaluate multiple values for that parameter. Grid searches, for example, can be used to train models automatically for 10-100 sample batch sizes in 20-step increments.
3. **Random search:** It has been shown that testing randomised hyperparameter values can be more successful than both manual and grid search. Rather than focusing on potential locations, it is preferable to select random test values from the whole issue space.
4. **Bayesian optimization:** This approach approximates the training model by varying hyperparameter values. It monitors the output provided by each set of parameter values for the model and keeps sampling until it obtains a list of hyperparameter value sets.

2.4.1. Learning Rate

The weights of a neural network cannot be calculated using analytical approaches. To compute the weights, a process known as stochastic gradient descent must be utilized, which will be detailed in full later in this article.

Stochastic gradient descent is an optimization approach that uses training dataset instances to calculate the error gradient and then backpropagation to make any necessary modifications to the model weights. Our neural network's weights are distributed according to the gradient of our loss function based on a hyperparameter called the learning rate. In this case, it indicates how often the neural network's stored knowledge has to be reloaded. It is possible to train a neural network to achieve a beneficial conclusion while yet maintaining a relatively moderate learning rate, as seen in Figure 2.31.

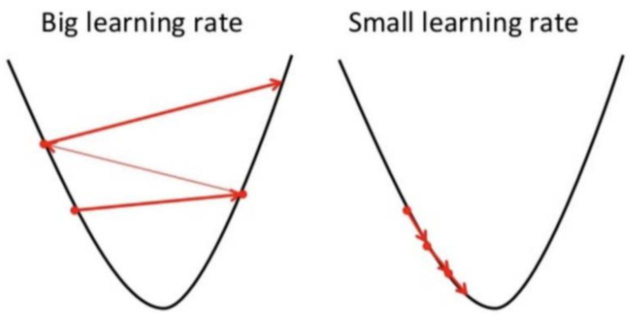


Figure 2.31 A slow learning rate causes the model to converge to the global minimum loss.

Less rapid changes in learning rates cause more training epochs to be required for smaller learning rates, while larger learning rates result in fewer training epochs being required. Higher learning rates, on the other hand, often lead to weights that are less than ideal.

Using adaptive learning rates, the training algorithm can keep track of how well the model is doing and adjust the learning rate accordingly.

The model's performance may be plateauing if it slows down the rate at which it acquires new information. In some cases, the learning rate can be reduced by a factor of two or an order of magnitude. Additional learning may be necessary if performance does not improve as expected.

Neural networks with variable learning rates often outperform those with fixed learning rates.

new weight = old weight - (learning rate * gradient) (5)

2.4.2. Momentum

An optimization approach known as gradient descent with momentum is a version of this process. Only two of the algorithm's many goals are to reduce the number of function evaluations needed to find an optimal solution or produce a better final result. A major drawback of using gradient descent is that it causes the search space to jump around unpredictably, which is frustrating. To find a minimum, for example, the gradient of the individual points (groups of parameters) you locate during your search may lead to a downhill or uphill movement.

For optimization situations where the general trend or structure of the search space is more essential than specific gradients along the route, this could hinder progress in search.

Prior to implementing the update equation, hyperparameters determine how much history (momentum) the equation has, allowing it to gather momentum. Values in the range of 0.0–1.0 that are as close to the maximum as possible are preferred (e.g. 0.8, 0.9, or 0.99 being commonly used). If our momentum is zero, we are descending in a straight line. It is necessary to break up the gradient descent update equation into two components: one component calculates position change, and another component updates old location to the new location of the gradient descent update equation.

The step size is multiplied by the point's gradient to arrive at the parameter change.

change_x = step_size * f'(x) (6)

Simply deducting the change from the current point gives you the new location; this is all that is required.

x = x – change_x (7)

To maintain momentum, remember the position change and apply it to the position change calculation in the subsequent calculation.

When analyzing changes throughout time, keep the following in mind When comparing changes across time, the update at the current iteration or time (t) includes the modification used at the previous time (t-1), as shown in the table below:

change_x(t) = step_size * f'(x(t-1)) + momentum * change_x(t-1) (8)

After that, the position is updated in the same way it was before.

x(t) = x(t-1) – change_x(t) (9)

Over the course of the search, the quantity and direction of change in position accrue, and the amount and direction of change are proportional to the size of the momentum hyperparameter.

To illustrate this, a high momentum value shows that the prior update has a substantial impact on the current update, while a low momentum value (e.g., 0.2) suggests that the previous update has very little influence on the current one.

2.4.3. Dropout

The Dropout layer is a mask in neural networks that removes specific neurons' contributions to the following layer while maintaining the functionality of all other neurons in the network.. The input vector can be subjected to a Dropout layer, which removes some of the vector's properties, or a hidden layer, which removes some of the hidden neurons' features.

CNN training benefits from dropout layers because they prevent the data from being overfit. It's important to have the initial set of training data because it has a big effect on the next sets. These characteristics would be lost if further samples or batches were collected:

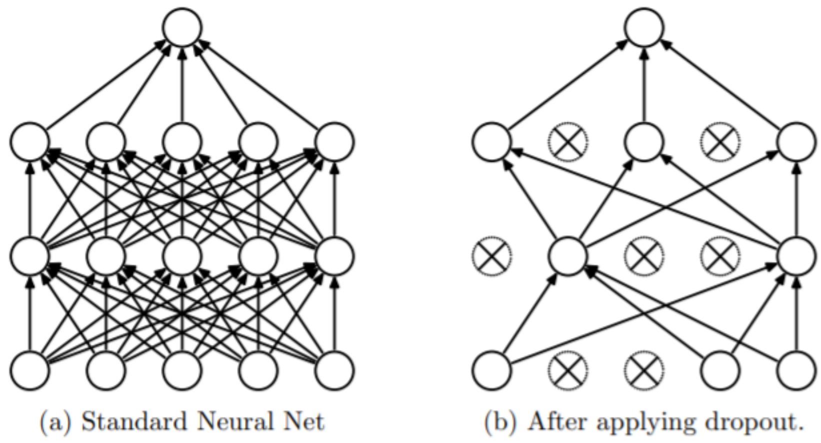


Figure 2.32 Dropout Neural Net Model.

2.4.4. Batch Size

The initial random weights and learning algorithm setup are crucial for training a deep neural network with tens of layers. For example, if weights are changed after each mini-batch, it is possible that the distribution of inputs to deeper layers of the network may change. This might lead to the learning process being unable to stop following a moving item. The consequent shift in the distribution of inputs to network layers is referred to as a "internal covariate shift".

Standardizing the inputs to each mini-batch of data is the goal of the training method known as batch normalization, which is utilized in extremely deep neural networks. When this happens, the learning process is stabilized, and the number of training iterations needed to create deep neural networks is cut down by a significant amount.

2.5. Effectiveness of Transfer Learning on Threat Detection

2.5.1. Training data

Given that it incorporates representation learning as well, DL is especially data-hungry [68,69]. For DL to be effective, a significant quantity of data must be gathered in order to create a performance model that is well behaved; when additional data is gathered, it may be possible to create a model that is even more appropriately behaved. The vast majority of the time, the information at hand is sufficient to build an accurate performance model. DL is not always necessary [70], for example, when the data are insufficient. There are three approaches suggested for resolving this problem successfully. The first phase in the learning process involves using the idea of transfer learning after gathering information from professions that are similar.

On this issue, three different perspectives have been expressed. Putting the transfer-learning concept into reality comes first, followed by data collection and analysis from related activities. While the provided data may not have an immediate influence on the real data, it may help to improve how the original input data was represented and how its mapping function operated. As a result, the overall efficiency of the model improves. When you don't have enough data to start with, you can alter an existing model that has already been trained rather than starting from scratch. Several transfer mechanisms are included in the DLP methodology [72, 73]. In the second alternative, for instance, already-existing data might be augmented. Because the image label is typically unaffected, it is possible to apply picture translation, mirrored pictures, and rotation to supplement image data. When utilizing this method, bioinformatics data must be handled with the utmost care. When enzyme sequences are reproduced, they don't always match the original sequence exactly. The third technique for expanding the training set uses simulated data. If the issue is well-known, a physical process emulator may be constructed in some circumstances. All the data required to repeat the desired outcomes will be included in the finished product. Data processing for DL-based simulation is extensively illustrated in the reference [75].

2.5.2. Overview: What is transfer learning?

A data scientist can use the notion of transfer learning to adapt a machine learning model to a new problem. As an example of this form of learning, people might share their skills. Learn how to drive other two-wheeled vehicles is much easier if you've already mastered cycling. Some of the same models for self-driving trucks could be used for self-driving vehicles.

Use this method to train machine learning models such as deep learning and reinforcement learning. In a new job, the model's training will be put to the test. A typical machine-learning strategy differs from a transfer learning approach in the following way:

2.5.3. Using a pre-trained model

The following is an explanation of how transfer learning operates:

Choose a source model from the list below: The source model must already be trained in order to transfer knowledge from one model to another. The alterations made to the source model are applied to the target model. In many instances, the training data for the target model may differ from the training data for the source model. Many factors should be organized before delivering information. Train the source model in the following ways to get the desired model: After the source model has been changed, the target model is achieved by working your way up from the source model as a starting point.

2.5.4. Developing a new model

The decision to build a new model for the purpose of applying their expertise to the main goal may be justified in certain circumstances. For example, separate models are required to recognise trucks and buses in photographs, but there is insufficient data available. Then you have the option of creating a new model that is the first to recognise automobiles. The model that has been created can be explicitly trained with the data that has been provided and used as a starting point for detecting trucks and buses.

2.5.5. When to use transfer learning?

Prior to using a model trained on a new problem, data scientists must use a model trained on a previous problem. Therefore, they are unable to apply what they have learned in the classroom to real-world scenarios. Transfer learning, on the other hand, yields better results in less time when put into practise. Applied transfer learning can be used by data scientists in the following situations.

A. There is not enough data

There are times when machine learning models can't be trained because of a lack of data. To avoid poor results, data scientists should start with pre-trained models that have already been tested.

B. There is not enough time to train

Some machine learning models are difficult to train and take an excessive amount of time before they are capable of performing their functions effectively. Pre-trained models can be used by data scientists who lack the time to build their own models or who have too many machine learning tasks to complete. Instead of creating a new model, you can save time by working on the existing one.

2.5.6. What are the main benefits?

According to the research in machine learning , Transfer Learning has the following advantages:

- When using other methods of learning, you must construct a model from scratch, which requires no prior knowledge. Because of transfer learning, you have a stronger starting point and are able to complete tasks at a higher level without undergoing formal training.
- A faster rate of learning is achieved through transfer learning because the problem has already been taught for a comparable task. This results in a faster rate of learning during training.
- Increased accuracy after training: Transfer learning allows a machine learning model to converge at a higher performance level after training because it has a better starting point and a faster learning rate. This results in more accurate output after training.

- Faster training: Since it employs a pre-trained model, it is feasible to attain the target performance much faster than traditional learning methods.

There may be little difference in the performance of transfer learning versus standard learning models. A target model cannot be built until transfer learning has had an opportunity to impact it.

2.5.7. Examples for use of transfer learning is shown below:

2.5.7.1.Technologies

- Image recognition applications can benefit from transfer learning. Occasionally, a dog identification paradigm can be used to recognize cats.
- Natural language processing (NLP) transfer learning applications are among the most common. This data can be used to train additional models, such as the pre-trained AI models stated earlier, to anticipate the next word in a sequence based on previously acquired phrase structure.
- A German speech recognition model can be built on top of an artificial intelligence model developed for English-language speech recognition.

2.5.7.2.Industries

- Driverless Cars
- Self-driving trucks may be able to benefit from the same paradigm as self-driving cars, according to researchers.
- A wide range of phenomena can be detected using transfer learning. For example, a model that has been trained to recognize other vehicles on the road can be used while driving autonomously to detect motorcycles or buses.

2.5.7.3.Gaming

- Chess players can put their go strategy to good use. For instance, instead of developing new models from scratch, AlphaGo's expertise could be applied to other video games.

2.5.7.4.Healthcare:

- The similar physical properties of EMG muscle signals and EEG brainwaves for gesture recognition allow for the transfer of learning between the two types of signals.
- Using medical imaging as another example of transfer learning. As an illustration, an image analysis model trained on MRI scans can be applied

to CT scans as a starting point. In the field of medical image recognition, transfer learning, according to Google, has had little impact on algorithmic performance.

- Spam filtering: Using an artificial intelligence model trained for email classification, it is possible to filter spam from emails..

AlexNet and ResNet are two examples of open-source trained models that data scientists can experiment with. Starting point for further development when dealing with machine learning challenges.

2.5.8. What are the best practises in this area?

- Pre-trained datasets, which are publicly available in a wide number of subjects, can save you a lot of time. Instead of generating new models from scratch, using existing source models could improve the robustness of the final model. This will save you time and prevent you from encountering new difficulties while building your target model.
- Be aware of the model from which you drew your inspiration: Check to see if your source model is compatible with your target model. To achieve your goal model, it may take longer if your source model is more difficult or irrelevant.
- Why transfer learning may not be necessary for larger datasets: With larger datasets and more complex tasks, transfer-learning may not have the desired effect. Starting with a random set of weights, classical learning gradually fine-tunes them until they reach a point of convergence. Transfer learning can begin with a pre-trained model, but larger datasets necessitate more iterations, making your original weights ineffective.
- If the source model is too similar to the target model, overfitting may occur in jobs with a small amount of data (for example, identifying cats and dogs). Avoiding this issue can be accomplished by altering the training rate, freezing some layers from the source model, or adding linear classifiers to the final model before it is used in final modelling.

2.5.9. Techniques used in Transfer Learning

Deep CNNs, which have the potential to revolutionize classification, have grown in popularity in recent years. Deep CNN models frequently require a big amount of data. The most prevalent issue when employing these models is a shortage of data to train on, and there is currently no answer to the challenge of acquiring a high amount of data points. The TL approach has been utilized to address the lack of training data issue [74, 75]. Using this method, a CNN model must be trained on a large amount of data before it can be put to good use. Using a few user-supplied training records, the model is tuned.

Let us understand the TL technique with the help of an example. It's a great technique to teach TL if students and teachers can interact. The first step is to do an extensive study on the topic [76]. Thus, the teacher organizes a "lecture series" so that the material is disseminated over a prolonged period of time. In a word, the instructor's role is to transmit knowledge to the pupil. Knowledge is passed down from one person to another, usually from one expert to another (student). As a result, in order to successfully train the deep learning network, as much data as possible must be input into the network throughout the training process. Using these parameters, a new model is built and put to the test to determine if it performs as well as the original.

Weights can be pre-trained instead of being trained from the beginning, as a result. The TL technique's conceptual diagram is depicted in Figure 2.33. The layer1 has large amount of data labels which consist of source label, source model and source data. The knowledge is transferred to the next layer which has target model, target labels and target data using which we will get our desired results. Thus the knowledge transfer from one source data to another source data with weights transferred from 1 layer to another layer helps in deploying the transfer learning approach.

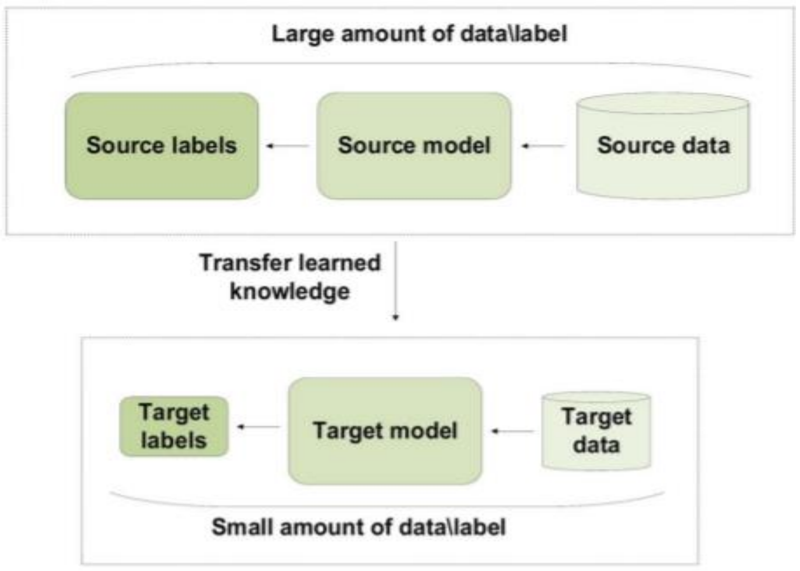

Large amount of data\label
Source labels
Source model
Source data
Transfer learned knowledge
Target labels
Target model
Target data
Small amount of data\label

Figure 2.33 Transfer Learning Architecture

1. Existing trained models include: CNN models like AlexNet [77], GoogleNet [78], and ResNet [79] have been trained on big datasets like ImageNet to make them better at recognizing images.. Using these models, a wide number of jobs may be recognized quickly and easily without having to start over from scratch each and every time. With the exception of a few learned traits, the weights are also always the same. When there aren't many data points to work with, these models are very helpful. Using a model that has already been trained can be helpful in many ways. In order to train complex models on big data sets, you need to use expensive computer resources, which can be too expensive. Second, it can take a few weeks or even longer to train big models, depending on how big they are. Lastly, a model that has already been trained can help networks become more general and speed up the process of convergence.
2. Using pre-trained models to solve a research problem: To train a deep learning approach, a large number of pictures are needed. In these circumstances, it is conceivable but difficult to accomplish well. Multilayered deep convolutional neural networks (DCNNs) may effectively categorize and recognize pictures in the presence of large amounts of training data [72,80]. They can even outperform humans in some situations. In order to avoid problems with overfitting, these applications necessitate a big dataset and DCNN models with acceptable generalization capabilities.

The quantity of data that can be used to train a DCNN model has no upper limit. While this may seem counterintuitive, the accuracy of the model will be compromised by over- or under-fitting concerns if the model is developed with fewer layers or a limited dataset for training. Because of their lack of ability to take use of the hierarchical properties of huge datasets, models with fewer layers are less accurate. Deep learning models necessitate a substantial amount of training data, which can be challenging to get. Acquiring labeled datasets in domains such as medical imaging and environmental science, for example, is prohibitively expensive [72]. Furthermore, the majority of crowdsourcing employees are unable to provide correct annotations on medical or biological imaging due to their insufficient grasp of medical and biological terminology and concepts. To recognize such photos, machine learning (ML) researchers typically rely on field specialists, which is an expensive and time-consuming approach. The sheer volume of labels needed to build robust deep networks looks to be practically difficult to produce. The latter problem has recently been alleviated by the use of TL. Even though TL has been shown to boost the accuracy of several pattern recognition and computer vision tasks [81, 82], there is a fundamental difficulty with the source data type of the TL compared to the target dataset that must be resolved. [80] For example, training CNN models with the ImageNet dataset, which consists of natural photos, enhances their ability to classify medical images. Consequently, because natural photographs differ so greatly from raw medical images, the model's performance isn't enhanced as a result. Furthermore, TL from other domains has been demonstrated to have no appreciable impact on medical imaging task performance, since lightweight models trained from scratch outperform standard ImageNet-transferred models. [84] As a result, in some circumstances, using pre-trained models is not an option. Several researchers used the same-domain TL method in 2020 and achieved excellent results [84-86]. Similar photos are utilized as examples for training in the same-domain TL approach. Two approaches are to train the model on diverse chest X-ray images, then fine-tune and train it on COVID-19 diagnosis images. In [86] it has more information on same-domain TL as well as directions for performing the fine-tuning approach.

2.5.9.1. Data augmentation techniques

When the goal is to increase the quantity of data accessible while avoiding overfitting, approaches to data augmentation can be used [87,88,89]. These are data-space approaches to situations with limited information. Data augmentation refers to a set of approaches for improving the properties and amount of training datasets. As a result, when these strategies are implemented, the performance of DL networks considerably improves. Following that, we'll go through a few other methods for adding information to data.

1. **Flipping:** Flipping the axis vertically is less common than the axis horizontally, although it is still a common technique. Flipping has been found to be advantageous in ImageNet and CIFAR-10 datasets. In addition, implementing it is a cinch. For text-recognition datasets, the transformation is not label-preserving as previously stated (such as SVHN and MNIST).
2. It is common practice in the business to encode digital picture data using a dimension tensor (height width colour channels) because it doesn't necessitate any new hardware, a different approach to expanding the channel's color space is very convenient to implement. In order to achieve color augmentation, only one channel of a certain color, such red, green, or blue, must be isolated. It's easy to turn a picture that just uses one color channel into something completely different by simply dividing the matrix and introducing additional double zeros from the other two color channels. You may also alter the image's brightness using a few simple matrix operations on the RGB values. Creating an image-representing color histogram can yield even better color augmentations. Histograms, like those found in photo-editing software, can also be manipulated to change their intensity levels.
3. **Cropping:** Cropping an important portion of each image is used as a specialized processing step when dealing with combined height and width dimensions in picture data. It's also possible to get the same effect as translations without using a translation tool, thanks to random cropping, Translating an image keeps its spatial dimensions while randomly cropping reduces the image input size [from (256, 256) to (224, 224)], which is why random cropping is superior to random translations. Translations are more

efficient than a random selection of crops. Depending on the cropping reduction threshold selected, the label-preserving transformation may or may not be handled.

4. **Rotation:** A photograph can be rotated from 0 to 360 degrees around an axis to create rotation augmentations. Some of these parameters, such as the rotation degree, have a big impact on how well rotation enhancements work in practice. Digit identification jobs benefit greatly from the ability to rotate small quantities of data (from 0 to 20 degrees). Data labels cannot be maintained following a transformation with a higher rotational degree.
5. **Translation:** Shifting the image in order to avoid positional bias in image data is a key alteration. Every photo in a dataset should be in the center, just as a model's dataset should consist solely of photographs in the center while being tested. After converting the initial images in a specific direction, the leftover area should be filled with Gaussian or random noise, or a constant value such as 255 or zero seconds, as suitable. This padding guarantees that the spatial dimensions of the image are preserved after augmentation.
6. The application of noise augmentation This method involves inserting an arbitrary value matrix into the equation. It is customary to utilize a Gaussian distribution to build this type of matrix. Moreno-Barea et al. [90] examined the noise injection on nine datasets in their investigation. They were retrieved from the repository at the University of California, Irvine [91], which can be accessible here. By introducing noise into photographs, the CNN can learn more robust features.

As a result of this, geometric modifications can be used to generate solutions that are extremely well-behaved in the presence of training data biases. Potential bias sources can be used to identify the distribution of testing and training data, respectively. There are positional biases in facial recognition datasets, for example, because all faces must be perfectly centred within the frames. Because of this, geometric translations are the most effective option. Besides being simple to implement, geometric translations are effective at eliminating positional biases because they can do so. You can start with simple operations like rotation and horizontal flipping before progressing to more complex operations in image processing software.

These transformations have several disadvantages including a longer learning curve, higher computing costs, and a larger memory requirement. For example, cropping or translation must be manually checked to make sure that the picture label is not affected by these changes. Testing biases go far beyond simple positional and transitional shifts, which is an important consideration. As a result, deciding whether or not to make geometric changes is not an easy decision.

2.5.9.2.Imbalanced Data

The over-representation of positive samples in the biological data frequently leads to imbalances in the data. In contrast to regular X-ray images, COVID-19-positive images have a substantially bigger volume. Unbalanced data can lead to surprising results when used to train a DL model.

The methods described here can be put to good use in resolving the issue at hand. Before anything further, the loss must be assessed and the outcome predicted in accordance with the correct criteria. It should be possible for the model to deal with both tiny and big classes of observations due to the unbalanced data. As a result, the model should use the AUC as a resultant loss and the [95] criterion to determine the outcome. Weighted cross-entropy loss should be used if cross-entropy loss is still desired, as this ensures that the model will perform well when it comes to small classes. During the model training process, large classes can be downsampled and tiny classes can be upsampled simultaneously. The construction of a model for each level of the hierarchy in a biological system is made possible by a hierarchical label space, according to Ref. [96]. Unbalanced data has been studied and documented in detail in relation to the DL model. To help alleviate the situation, the most often employed strategies were compared. Keep in mind, however, that these procedures are not geared to address biological concerns.

2.5.9.3.Interpretability of data

Examine DLs if you want to know how different methods work. There are a number of ways to look at this. Bioinformatics [97] is one such area where it is widely believed that DL must be understood and evaluated in order to extract the network's important motifs and patterns. The diagnostic and prediction findings of a trained DL model, as well as ways to improve their accuracy, must be fully understood in order to raise the certainty of prediction outcomes [98]. Because of this, the model relies on the results of the checks and balances.

A weighted score for each section of the example can accomplish this goal, as seen in the following example. Back-propagation techniques and perturbation-based approaches are used in this method [99] to correct the problem. A small percentage of the input is changed, and the impact on the model's output is tracked using perturbation-based methodologies [100,101]. Despite its technical complexity, understanding this notion is simple. A signal from the output layer propagates back into input layer to assess the relevance score of different input portions in back-propagation-based approaches. Procedures like this one are more involved. According to [102], these methods have been proven to work. Depending on the situation, the model's interpretability may be viewed in a different light.

2.5.9.4.Uncertainty scaling

The final prediction label cannot always be determined solely by a DL approach to prediction; each query's confidence score is also required. Calculating the confidence score takes into account a model's confidence in its predictions. In every application, a high confidence score is essential because it guards against people falling for predictions that are based on inaccurate or misleading information. According to some researchers, confidence scores in biology may reduce the time and resources required to verify false predictions. Scaling uncertainty can be a useful tool for evaluating the accuracy of computer-aided diagnosis [104, 105] and machine learning-based illness detection in the healthcare industry. Because multiple DL models may result in overconfident predictions, direct-softmax DL's output frequently does not place its probability score on an appropriate scale [106,107].

2.5.9.5.Catastrophic forgetting

As interference with previously learnt information, the process of incorporating new information into a simple DL model is defined. The following is a probable scenario example: For instance, if a model trained to classify one thousand species of flowers is tweaked solely for this new class, its performance with other classes will decline. In terms of logic, biology and many other disciplines deal with the ongoing collection and regeneration of data. Fortunately, the solution to this conundrum is as straightforward as training a brand-new model employing both old and new data sources. In addition to being time-consuming and computationally intensive, this method results in an undesirable unstable representation of the original data. There are now three basic types of machine learning algorithms [113, 114] that can be used to solve the human brain problem based on neurophysiological theories.

2.5.9.6.Model compression

Due to the vast number of variables to examine and train, DL models require a substantial amount of memory and processing power [115, 116] before they can be implemented in production. Data-intensive tactics have been applied to improve outcomes in the sectors of healthcare and environmental science. These limits limit the application of DL in devices with limited processing capability in the healthcare industry, which is especially crucial. Diverse technologies for monitoring human health and the volume of data have gotten much more complex and massively larger [117,118], needing greater processing and resources [119,121].

To address the computational problems of deep learning, parallel processing techniques such as FPGAs and GPUs [119,120] have been created. [119–120] There are numerous imaginative compression strategies for deep learning models, all of which aim to reduce the computing load of the models from the start. Approaches are divided into four groups, In the first class, the number of superfluous parameters (parameters that have no impact on model performance) is decreased. Deep compression is one form of "parameter trimming" [121] strategies. When a smaller model is trained in the second class using the information collected from the larger model, this is known as knowledge distillation. In the third class, smaller convolution filters are utilized to reduce the number of parameters[124]. Lower-rank factorization is used to compute the information parameters required for final class preservation [125].These techniques for model compression are the best examples of their respective classes. In addition, a more thorough explanation of the circumstance was offered.

2.5.9.7.Overfitting

Since there are so many significant factors and their complex interrelationships, DL models are prone to data overfitting. As a result, the model's ability to perform on the test dataset is impaired. This problem, on the other hand, affects a wide range of responsibilities, not just one. When proposing DL approaches, it is critical to thoroughly examine the repercussions. Overfitting concerns can be solved by using implicit bias in the training phase, according to study [128,129]. Despite the fact that the issue of overfitting must be addressed, it is not a top priority at this time. The problem can be classified into three categories based on the DL approaches that can be used to counteract overfitting. In addition to weight decay [130], batch normalization [131], and dropout (126) the model's architecture and parameters are all affected by these. [130] Weight decay is one of the lesser known second-class techniques Strength decay [130] is the most common regularization method in deep learning and is used by most machine learning systems. Challenges 2 addresses data corruption and augmentation, two common problems with models. A lack of training data may lead to overfitting, as seen in the illustration, because the learned distribution deviates from the genuine distribution. There are many ways to enhance the amount of training material available. Data corruption that has been relegated to the margins can only help the solution by offering new information. A discussion of how the model performs will wrap up this session with a class on the results. To ensure that the model is consistent, a new method penalizes outcomes that are too confident. Regularization of RNN and CNN networks can be achieved using this strategy. A lack of training data can lead to overfitting since the learned distribution deviates from the genuine distribution, as seen in the illustration. Additional information is added to training data via data augmentation. The solution is only made better by minimizing data corruption. It's up to the final class of the model to deal with the model's output, of course. [133] Overconfident outputs are penalized by a newly developed approach as part of the model regularization process. RNNs and CNNs have both been regularized using this approach.

2.5.9.8.Vanishing gradient problem

The vanishing gradient problem occurs when utilizing backpropagation and gradient-based learning approaches with artificial neural networks (ANNs). This is a problem that arises throughout the training process and is distinct from other issues. As a result, the weights of the neural network are recalculated based on the current weight and the partial derivative of the error function throughout each training iteration. If the gradient is too tiny, further training will be impossible and the neural network will stop working, this weight update will not take place. It is possible to control a large input space by using the sigmoid function in conjunction with other activation functions, such as the sinusoidal one. The sigmoid function's derivative will be exceedingly small because of the large variation in the input. As long as these activations are only used by a few layers in a shallow network, this is not an issue. Additional layers during training reduces the gradient, yet the network is able to perform the desired tasks. Using back-propagation, neural network gradients can be calculated. Identifying each layer of the network and the network derivatives within it is done in reverse order, starting at the outermost layer and working inwards. First step derivatives are multiplied by how many network layers there are in this stage. The hidden layers can be activated by multiplying N tiny derivatives with an activation function, such as the sigmoid function. A significant decrease is seen at the top of the hierarchy as a result of this change. Consequently, it is impossible to rapidly change the biases and weights of the early layers of neural networks during the training phase due to the modest gradient. Due to the fact that these early layers are often required to recognize the most crucial elements of the incoming input, this scenario reduces the network's overall accuracy as well However, activation functions may help alleviate this issue. They don't have the ability to condense the input field into a small area. Since the ReLU [137] does not generate a small derivative, which is frequently used in mathematics, it is the most preferable method for mapping X to max. As an additional option, the layer for batch normalization [138] might be used. The derivative disappears when a large input space is reduced to a small one. This was made clear in the past. Batch normalization prevents the expression |x| from exceeding the sigmoid function's outer boundaries by simply normalizing the input data. This problem can be fixed. When applying the normalisation approach, this ensures that the derivative will be large enough to be employed in later phases. It is also possible to address this problem with the help of graphics processing units (GPUs). As a result, a large number of deeper network levels can undertake regular back-propagation at a rate that is faster than the time needed to discover the vanishing gradient problem. [139].

2.5.9.9.Exploding gradient problem

Problems with gradients and with vanishing points are incompatible. Error gradients with statistical significance are created during back-propagation [140,141]. As a result of the latter, the network's weights will undergo enormous shifts, making the system unstable. Consequently, the model will be unable to learn as efficiently as it once was in the future. During back-propagation, the gradient increases exponentially because of the network's repeated compounding of gradients. Consequently, weight values can be extremely large and may even overflow, resulting in NaN values being generated. Listed below are a few examples of potential remedies:

1. There are several different weight regularisation procedures that can be used.
2. The architecture of the network model is being redesigned.

2.5.9.10. Under specification

An under specification [142] bug was discovered on January 1, 2020, by computer specialists from all across the world. Machine learning models, particularly deep learning models, typically perform badly in real-world applications such as computer vision, medical imaging, natural language processing, and medical genomics. The unfavorable outcomes are the result of a lack of specificity. As shown, even minor changes can have a significant influence on a model's solution and predictions in various deployment domains. Underspecification can be remedied in a variety of ways. "Stress tests" are used to analyze how effectively a model performs on real-world data and to identify potential weaknesses in the system. Because of the model's proneness to error, a thorough knowledge of the technique is required. In the words of the researchers, "developing stress tests that are well-matched to the application criteria and deliver acceptable "coverage" of anticipated failure modes is a huge task." A rethinking of some applications may be necessary as a result of this decrease in accuracy. It's imperative to pay close attention to this issue, as it affects applications like as medical imaging and self-driving automobiles.

2.5.9.11. Evaluation metrics

There are many important considerations to keep in mind when developing a classifier for a DL task. Training and testing are two common methods for categorizing data. It is feasible to enhance training accuracy by optimizing the classification algorithm. Evaluative measures are useful for selecting the best answer, assuming that a discriminator exists that can give an unusually accurate forecast for the incoming evaluations for a specific classifier. Model testing phases such as evaluator during model testing using hidden data and the evaluation metric are the only places where the effectiveness of a constructed classifier can currently be tested. Equation 20 says that there are a certain amount of cases that fall into the negative or positive categories. The number of false-positive and false-negative cases is one way to define FN and FP. This is followed by a discussion of the most popular assessment tools currently accessible.

1. Accuracy: It is possible to calculate the percentage of correctly predicted classes compared to the total number of samples tested.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{9}$$

2. Sensitivity or Recall: The percentage of successfully classified positive patterns can be calculated using this function.

$$Sensitivity = \frac{TP}{TP+FN} \tag{10}$$

3. Specificity: The percentage of correctly identified negative patterns is calculated using this formula.

$$Specificity = \frac{TN}{FP+TN} \tag{11}$$

4. Precision: For each anticipated pattern in a given positive class, we compute the positive patterns that were correctly predicted.

$$Precision = \frac{TP}{TP+FP} \tag{12}$$

5. F1-Score: It is used to calculate the harmonic average of the recall and accuracy rates.

$$F1_{score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (13)$$

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

$$J_{score} = Sensitivity + Specificity - 1 \quad (14)$$

7. False Positive Rate (FPR): Essentially, this statistic measures the likelihood of a false alarm ratio being encountered.

$$FPR = 1 - Specificity \quad (15)$$

8. The Area AUC is a common ranking type statistic in the ROC Curve categorization. In order to come up with the optimum learning model for a certain case, it is critical to compare several learning algorithms. As compared to probability and threshold measurements, the AUC value provides information about how well a classification system ranks overall. The AUC for a two-class issue can be calculated using the formula below.

$$AUC = \frac{S_P - n_p(n_n + 1)/2}{n_p n_n} \quad (16)$$

Sp denotes the sum of all positively rated samples. The numbers n_n and n_p represent the number of negative and positive samples, respectively. In comparison to accuracy metrics, the AUC value has been empirically and theoretically confirmed, making it incredibly valuable for finding the best solution and monitoring classifier during classification training.

When it came to the discrimination and assessment processes, the AUC performed admirably. However, when distinguishing a large number of produced solutions, the AUC computation is largely cost-effective for multiclass problems. In addition, the time complexity for computing the AUC is $O(|C|^2 \, n \log n)$ for the Hand and Till AUC model and $O(|C| \, n \log n)$ for the Hand and Till AUC model.

2.6. Problem Statements identified

The suggested research addresses these problems after brief study in literature survey:

- Improving the effectiveness of image feature extraction.
- Enhancing the efficiency with which picture features are categorized.
- In order to circumvent ML's difficulties while dealing with large datasets.
- The goal is to find the most effective DL model for threat detection.
- Optimizing the model's hyper-parameters for better accuracy.
- To avoid having to train several machine learning models from scratch to fulfill similar tasks, saving time and resources.
- Use the weights from the pre-trained models to initialize the weights of the new model.

2.7. Summary

In this Chapter we have briefly discussed about the existing Deep Learning models and their merits, demerits and limitations. A brief overview of computer vision-based methods such as object localization, detection, feature extractions, segmentations and its based algorithms with hyperparameters and its way of optimizations. Based on the study we have found the problem statements discussed above and to address and resolving the above problems statements, we have given and proposed the appropriate deep learning models, transfer learning methodology, features used for our deep learning model and datasets used for training and testing. We also have provided with the comparative analysis of the deep learning models and network pipeline which helps us in deploying the appropriate model for our threat detection pipeline discussed in Chapter 3.

CHAPTER:3

CLASSIFICATION OF THREAT-DETECTION IMAGES: TRAINING FROM SCRATCH OR TRANSFER LEARNING

CHAPTER 3

3. CLASSIFICATION OF THREAT-DETECTION IMAGES: TRAINING FROM SCRATCH OR TRANSFER LEARNING?

3.1. Introduction

In Chapter 2, we have discussed about the complete overview and comparative studies of deep learning based algorithms, networks and computer vision based task in literature survey. We have found that the traditional old methods used by researchers in carrying out the detection and classification of objects and targets in different areas, such as HOG, SIFT, and SURF which are not the state-of-the art methods and out dated methodology on large datasets and struggling to classify threat images from input photographs and time consuming. Based on the literature survey discussed in Chapter 2, we have found the following gaps and proposed a new methodology to overcome the present issues:

- To improve the efficiency of obtaining image features.
- To increase picture feature classification performance.
- To handle complicated photos with variable attributes efficiently.
- To build a Deep learning model that scales well to high-dimensional data.
- To reduce the computational time, Training time and power and huge dataset.
- Improving the accuracy of the pre-trained CNN model's as threat detection requires optimizing the model's hyper-parameters such as learning rate.
- To avoid having to train several deep learning models from scratch to fulfill similar tasks, saving time and resources.
- To initialize the weights of the new model, use the weights from the pre-trained models.

To address the above mentioned gaps, we have provided proposed methodologies and techniques, which helps in increasing the model performances and detection accuracy.

A decent deep learning model is built with care. To obtain good performance, massive training data, effective hardware, knowledgeable developers, and a significant amount of time are required to train and hyper-tune the model. As a result, constructing a deep learning model from scratch and training is nearly impossible for every deep learning activity. Here comes the strength of Transfer Learning. Transfer Learning is the process of using an already learned model for a related task. Each deep learning task is unique in some way. As a result, it is difficult to reuse a previously trained model in new problems. It may require some task-specific changes. As a result, we use Transfer Learning to complete our work. A pre-trained model is one that has already been trained and will be used via Transfer Learning. A pre-trained model may be cutting-edge in the domain. However, our challenge must be in the same domain as the pre-trained model. For example, a pre-trained model designed for image segmentation cannot be used for image classification.

3.1.1. Transfer Learning over training from scratch.

The major issues we found during literature survey that the deep learning based algorithms take weeks to train a neural network on massive datasets. To overcome and addressing this issue we have decreased the training time and computational process time by leveraging model weights from previously trained models - in other words, by applying transfer learning. Transfer learning is the act of avoiding having to train a new model from start by leveraging feature representations from a previously trained model.

Typically, pre-trained models are trained on massive datasets that serve as a common benchmark in the field of computer vision. Model-derived weights can be used in additional computer vision applications. These models can be used to predict new tasks directly or as part of the training process for a new model. By incorporating pre-trained models into a new model, training time and generalization error are reduced. When the training dataset is tiny, transfer learning is extremely useful. In this case, we utilize the weights of the pre-trained models to initialize the weights of the new model depicted in Figure 3.1.

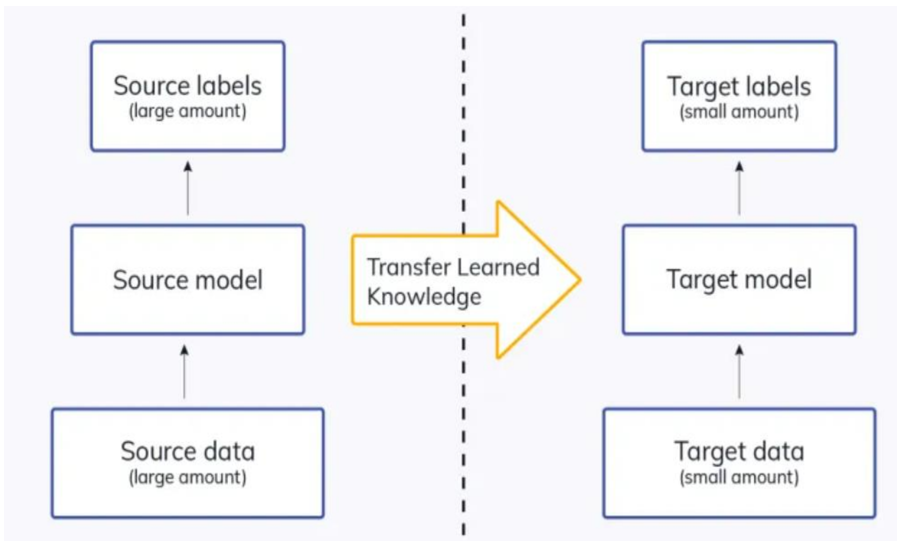


Figure 3.1 Transfer Learning Flow

3.1.2. Fine-tuning of the Hyperparameters

Transfer learning has an extra step for fine-tuning the model accuracy because we are retraining the whole model to adjust the weight based on our target labels, which can cause the overfitting problem to our trained model. To avoid the model to over fit, we retraining the model and update the learning rate. This is critical since it prohibits large modifications to the gradient decent that may cause poor performance in the model. To overcome and re-training the model, It is also beneficial for calling back the training process to terminate the model at early state for training. This method will improve the accuracy of the pre-trained CNN models by optimizing the hyperparameters.

Now, let's take a look at how we can put what we learned in Figure 3.2 into practice.

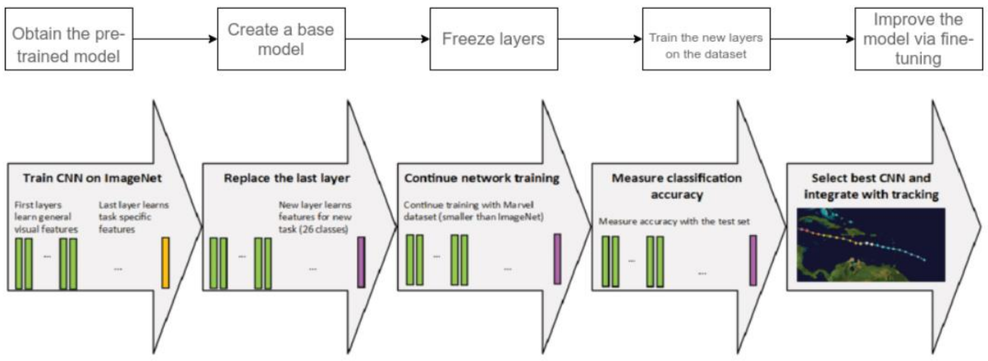

Obtain the pre-trained model
Create a base model
Freeze layers
Train the new layers on the dataset
Improve the model via fine-tuning
Train CNN on ImageNet
First layers learn general visual features
Last layer learns task specific features
Replace the last layer
New layer learns features for new task (26 classes)
Continue network training
Continue training with Marvel dataset (smaller than ImageNet)
Measure classification accuracy
Measure accuracy with the test set
Select best CNN and integrate with tracking

Figure 3.2 Steps for using Transfer Learning

3.2. Implementation of transfer learning

Transfer learning can be implemented in six broad steps.

3.2.1. Obtain the pre-trained model

The initial step is to get our hands upon this pre-trained model that it will be utilized to address our target problem. A pre-trained model is a stored network that has already been trained on a large dataset, typically on a large-scale image-classification problem. We can use the pre-trained model as is or modify it with transfer learning to fit a specific job and weights can be used as per our application area.

The principle behind transfer learning for image classification is that if a model is trained on a large and diverse dataset, it can efficiently serve as a generic model of the visual world. We have employed these learned feature maps instead of beginning from beginning by training a large model on a massive dataset.

Following two approaches adopted to modify a pretrained model:

- **Feature Extraction:** To extract useful characteristics from new samples, use existing network representations. To reuse the feature mappings learnt previously for the dataset, just layer a new classifier trained from scratch on top of the pretrained model. The complete model does not need to be (re)trained. The basic convolutional network already provides properties that are beneficial for image categorization in general. The pretrained model's final classification element, on the other hand, is specific to the original classification task and thus to the collection of classes on which the model was trained.
- **Fine-tuning:** Unfreeze a few of a frozen model base's top layers and train both the newly added classifier layers and the base model's final layers at the same time. This allows us to "fine-tune" the higher-order feature representations in the underlying model such that they are more relevant for the specific job.

Transfer Learning Framework consist of

- Analyze and understand the data.

- Create an input pipeline, in this example with the Keras
- ImageDataGenerator.
- Create the model
- Pre-train the underlying model and then load the data into it (and pretrained weights)
- The classification layers should be stacked on top of one another
- Form the model
- Model evaluation

3.3. Datasets, System Specification, Tools & Libraries

After identifying the pre-trained model and weights used for our target labels, we have now to take a datasets which consist of the increased in crime rates in congested areas and suspected secluded areas which have made security a major source of concern in almost any location, threats of violence are a source of concern for human rights activists. The right to life, which is our most fundamental human right, is under threat today because of gun violence. Violence with firearms, flames from a blazing fire, masked men, and bizarre human behaviour are all daily tragedies that affect people all over the world. Every day, at least 400 people die as a result of violence perpetrated with these weapons, according to estimates. The availability of gun violence, fire flames, masked men, and anomalous human behaviour has always played a significant role in the stakes of crime and mayhem, and this has not changed. Only the threat and non-threat classes are capable of demonstrating this. We have downloaded datasets from various different repositories and also created the dataset and labelled them between various classes such as threat and non threat, Fire and Smokes, Guns and Non Guns etc. To develop the following approach to detect threats using the SSD, Mask RCNN technique and the RCNN masking technique. As a result, we now have the necessary dataset, which will contain only one type of data. Using a variety of CNN methods, this dataset is then trained for threat categorization purposes. Closed-circuit television (CCTV) cameras, which provide real-time input footage, can be used to identify numerous forms of threats, including weapons, once the data has been properly educated. When a weapon is recognized, a threat alert is automatically sent, allowing for the required response time. After selection and creating the dataset, Now we have to select the best model which gives best results in detection and classification.

Table 3.1: Dataset for fire flames and Smoke

Repository	Format	Objects	Resources
Repository 1	Video	Fire Flames, Smokes, disturbance	Computer Vision-Based Fire Detection Software (bilkent.edu.tr)[1]
Repository 2	Video dataset	Fire Flames, Smokes and disturbance	Computer Vision and Pattern Recognition Laboratory Homepage (km.ac.kr)[2]
Repository 3	Image Dataset	Fire Flames	http://cfdb.univ-corse.fr/index.php?menu=1[3]
Repository 4	Image & Video dataset	Smoke	Welcome to the Wildfire Observers and Smoke Recognition Homepage (Feb.hr)[4]
Repository 5	Image dataset, Video	Smoke	Fire Detection Research Group (ustc.edu.cn)[5]
Repository 6	Video	Fire	Fire | NIST[6]

3.3.1. System Details and Specifications

For carrying out the research work, following hardware and software we have used and proposed for training and testing the model.

Hardware

1. Laptop with Intel i7 processor
2. Minimum of 8GB RAM
3. Minimum of 1 GB Hard disk
4. Windows 10 or Linux
5. Webcam

Software

6. Anaconda,Jupyter Notebook
7. Open CV Library used for Computer Vision (CV), DL and Images
8. MobileNet, InceptionNet
9. Google Colab

[1] http://signal.ee.bilkent.edu.tr/VisiFire/
[2] https://cvpr.kmu.ac.kr/
[3] http://cfdb.univ-corse.fr/index.php?menu=1
[4] http://wildfire.fesb.hr/
[5] http://smoke.ustc.edu.cn/datasets.htm
[6] https://www.nist.gov/fire

10. SSD, Fast RCNN, Mask RCNN

Raw photographs are unsuitable for analysis and must be converted to a processed format, such as jpeg or jpg, before being used for further investigation. The image size is reconstructed into a square image, and the photos are saved in the RGB/BGR colour format. To create the dataset, threat-related photos such as guns, fire flames, and abnormal actions are collected and prepared for inclusion in the dataset. It is possible that we obtained this data set from the internet, and that the photographs contained within it are in the jpeg format. An additional Labels folder contains text-format files in which the first line lists various items in the matching image and where the following line lists the coordinates of the object. This is solely for the purpose of training.

3.3.2. Tensorflow Deep Learning Libraries and Program Elements.

In order to conduct machine learning and deep learning programming, a number of widely available libraries are available. These libraries include: The following are a few of the most widely used library systems:

1. Keras
- Francois Chollet was the one who came up with the idea.
- Python is the programming language used to create the open-source library.
2. Theano
- The University of Montreal came up with this idea.
- Python was used to create this programme.
3. TensorFlow
- Google Brain Team came up with this idea.
- C++, Python, and CUDA were used to create the programme.
4. DL4J
- This product was created by the Skymind team and the DeepLearning4J community.
- C++ and Java were used to produce this document..
5. Torch
- Ronan Collobert, Koray Kavukcuoglu, and Clement Farabet collaborated to create this design.

- Python-based

  Access to a wide range of libraries is available to the user. This reserach, carried out and focuses solely on the open-source TensorFlow from Google. TensorFlow and Keras, another popular choice, recently merged. Python is by far the most appropriate and widely used of the many supported languages.

3.3.3. Training and Testing using TensorFlow Google Colab

Due to huge computation time, we have proposed and used Open-source Library of Google as its name implies, TensorFlow is primarily intended for use in deep learning applications. Traditional machine learning can also be used with this platform. Instead of focusing on deep learning, it was originally built to perform large numerical computations. In any case, Google decided to make it open source because it turned out to be extremely beneficial for the development of deep learning.

More than one dimension can be represented by a tensor, which is a multidimensional array in TensorFlow. Multi-dimensional arrays come in handy when dealing with large amounts of data. TensorFlow uses nodes and edges to represent data flow. It is simple to deploy TensorFlow code across a cluster of GPUs because it is written as graphs.

3.3.4. TensorFlow's purpose

Prior to the advent of libraries, machine learning and deep learning coding was considerably more difficult. Creating a neural network, configuring a neuron, or programming neurons is simplified thanks to this library's high-level API. All of these duties fall under the purview of the library. TensorFlow can also be used with Java and R, two other programming languages.

3.3.5. TensorFlow is compatible with both CPUs and Graphics Processing Units (GPUs).

To train deep learning models, a lot of computing power is required, which makes deep learning applications complicated. As a result of the large amount of data, the process is lengthy and requires a variety of steps including iterative procedures, mathematical computation and matrix multiplication. On a standard

Central Processing Unit, these tasks would take a long time to finish (CPU). A high-resolution screen and picture are necessary in the gaming industry, where graphics processing units (GPUs) are popular. There were graphics processing units developed in order to accomplish this However, deep learning applications are becoming more and more commonplace.

Additionally, TensorFlow can be used on both GPUs and CPUs. For machine learning, it is also faster to compile than other deep learning libraries like Keras and Torch.

3.4. Model Selection

After selecting the pre-trained and finetuned model, we have created our own dataset with our target label to detect by the model. The model is now we have to select which will work on top of the pre-trained model as only annotated images, objects, and pixels may be used to train a comprehensive object separation model. A comparable dataset does not exist for X-ray, thermal scanning, or any other type of security imaging method. For end-to-end training on a limited dataset, data that does not have adequate or no annotated must be deleted. In its place, we developed a threat detection pipeline that separates the task of discovering object instances from anticipating segmentation masks. The segmentation model is trained using all of the samples except for a small subset that is utilized for object identification. With the integration of these models, it is possible to construct an object separation model that can find and clearly define each instance of an object while also defining all of the pixels associated to that instance. The following sections of study are used to determine which models are most suited for the two mentioned tasks. Object recognition and segmentation models are trained using pixel-level data tagged with images and object-level annotations. These models result in a separation model that locates and delineates the pixels associated with an object instance by making use of any annotations that may be present. Because the tests below are designed to establish which models are most effective at achieving these objectives, the results are meaningful.

3.4.1. Detection Models

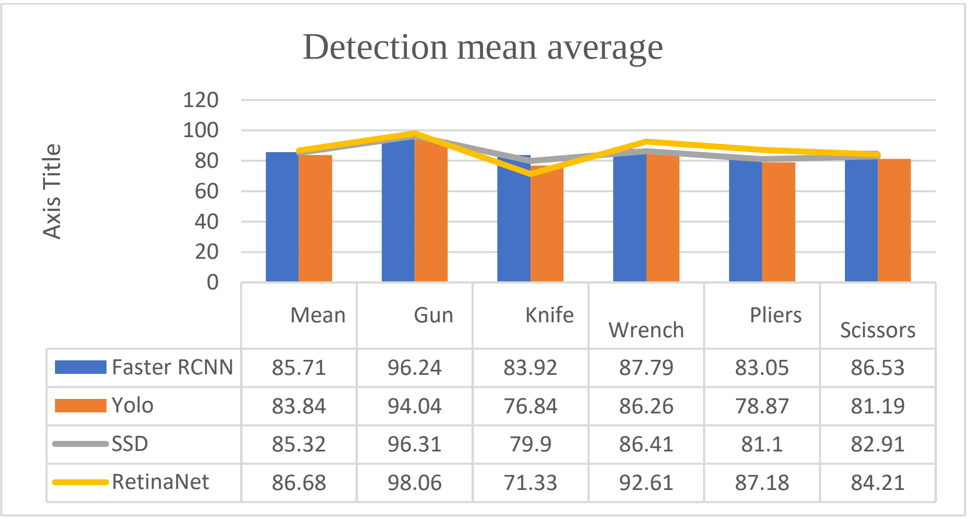
Now as per our proposed methodology we have to discover which of the four most popular current detection models is the best for our proposed model.

- **Faster R-CNN:** When it comes to recognizing objects, this is a CNN-based two-stage paradigm. An area that appears to include prospective targets for the algorithm's second stage is extracted in the first stage, and the objects identified in those extracted areas are then classified. The region proposal network is the first step in the method (RPN). Regression is used to improve localization when predicting bounding boxes based on differences in coordinates between the actual bounding boxes and the proposed areas.
- **You Only Look Once (YOLOv3):** A dense sample of the most likely places is used instead of region suggestions in order to detect an object in the first stage. The YOLO method uses a grid of cells to determine the number of bounding boxes (sometimes referred to as "priors") and the associated confidence ratings for each photo it receives as input. A single CNN can make all of these predictions at once, making it one of the quickest real-time algorithms. This study made use of a third-generation algorithm, which uses shortcuts to better detect small objects.
- **The Single-shot Multibox Detector (SSD):** At each pyramidal layer of the CNN, this one-stage technique may be able to detect objects of varying sizes. SSD predicts the anchor box offset for each feature map point rather than dividing data into grids. As a result, grid segmentation is no longer necessary, saving both effort and time. The receptive fields of feature maps fluctuate depending on where you are in the CNN pyramid. When comparing feature maps, the later layers have coarser maps whereas the older layers have finer maps. An object in a picture's size can be predicted by comparing its anchor boxes to the cell sizes in which they are situated.
- **RetinaNet:** Like SSD, which uses the same principle of detection at each pyramidal layer in a CNN and is also a one-stage method, this methodology does not require any additional steps. To create more robust representations, the feature pyramid network (FPN) concatenates later feature maps with earlier feature maps. The Researcher propose a new loss function called

"targeted loss" for samples that are particularly challenging to classify (such as those with a high rate of misclassification).

Only positive samples were utilized in the assessment of a model's ability to identify dangers in an image. The data was partitioned into two sets using training and validation. The backbone network underwent 60,000 iterations in order to train the models. To see how each category's detectors fared, look at the data in Table 3.1. Regardless of how many categories you look at, R-CNN is the best detector. RetinaNet has an edge over the other possibilities because its backbone network is already linked to an FPN. When compared to other one-stage detectors, such as the Faster R-CNN technique, it has the best overall performance (see below). Because of this, we decided to adopt faster R-CNN as our object identification model.

Table 3.2: Detection mean Average Precision(%) on the validation set



	Mean	Gun	Knife	Wrench	Pliers	Scissors
🟦 Faster RCNN	85.71	96.24	83.92	87.79	83.05	86.53
🟧 Yolo	83.84	94.04	76.84	86.26	78.87	81.19
— SSD	85.32	96.31	79.9	86.41	81.1	82.91
— RetinaNet	86.68	98.06	71.33	92.61	87.18	84.21

3.5. Segmentation Models

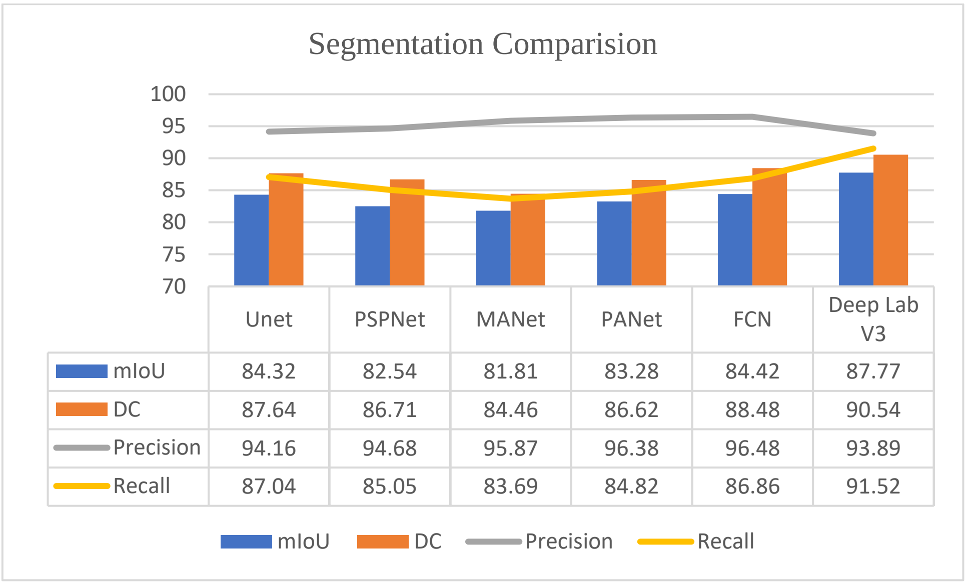
After selecting the detection model above now we have analyzed six cutting-edge approaches to find the optimum segmentation model for the task at hand for bounding box and masking of threat based detection.

- **FCN:** It is recognized as one of the earliest techniques for classifying pixels at the pixel level. The dense layers are said to be executing one-to-one convolutions due to the substitution of a convolutional layer for the fully linked layers in the standard CNN architecture. As demonstrated in the following picture, the final convolution layer is then up-sampled using deconvolution to learn non-linear upsampling and produce a feature map of equivalent size to the input.

- **U-Net:** The FCN is improved by adding U-shaped symmetric encoding and decoding fully convolutional layers. To emphasize data that was omitted during downsampling, it builds skip links between relevant encoder and decoder levels.
- **PSPNet:** An other method for determining the distribution of segmentation classes in an image is called "pyramid pooling," which mixes feature maps from various backbone model layers. As part of the training process, a monitoring system is installed at the pooling module's input to measure an additional loss.
- **DeepLabV3:** Atrous convolutions and spatial pyramid pooling are two techniques for tracing the contours of small objects (SPP). SPP use a technique known as layer pooling to capture data at several scales. Before being delivered to the next fully-connected layer in a convolutional network, output vectors from each pooling layer are concatenated. Dilated convolutions, also known as atrous convolutions, are widely utilized to enlarge the receptive fields of filters. It is possible to incorporate a broader background without increasing the employed parameters.
- **PAN:** Two additional modules, FPA and GAUS, have been added to the segmentation framework as complementary additions (GAU). Applying global average pooling to high-level data generates a global context, which is then used to direct the low-level characteristics. The output of the backbone model can be more accurately represented when using FPA, global pooling, and a spatial pyramid.
- **MA-Net:** The point-wise attention block and the multiscale fusion attention block are two additional blocks that the system uses to implement an attention mechanism (MFAB). For finding channel linkages between any two feature maps, PAB uses multi-scale semantic feature fusion in the global view, while MFAB relies on the local view's multi-scale semantic feature fusion.

The segmentation models were evaluated using only annotated pixel-level data. As part of a previous exercise in selection, we were able to extract patches or areas of interest from samples that would be utilized to train segmentation models. By running negative samples through a detection model, background regions or patches that are free of hazards can also be introduced. A training set and a validation set were created once the data was gathered. All 60,000 model iterations were trained on the same backbone network (ResNet-50). Table 3.2 shows the results of the validation set, which included a variety of criteria for evaluation.

Table 3.3: Performance of Segmentation on the validation set using several assessment



	Unet	PSPNet	MANet	PANet	FCN	Deep Lab V3
mIoU	84.32	82.54	81.81	83.28	84.42	87.77
DC	87.64	86.71	84.46	86.62	88.48	90.54
Precision	94.16	94.68	95.87	96.38	96.48	93.89
Recall	87.04	85.05	83.69	84.82	86.86	91.52

3.6. Threat Pipeline proposed model

After analysis from the base models, The initial step is usually to instantiate the underlying model using one of the architectures, such as ResNet or Xception. We also used the pre-trained weights that best fitted for our application using transfer learning. If we are not using the pre-trained weights, we have to train our model from scratch using the architecture, which is a time taking task. Remember that the final output layer of the base model will frequently contain more units than we require. As a result, when generating the base model, we must remove the final output layer. We have later added a final output layer compatible with our problem as discussed in figure 3.3.

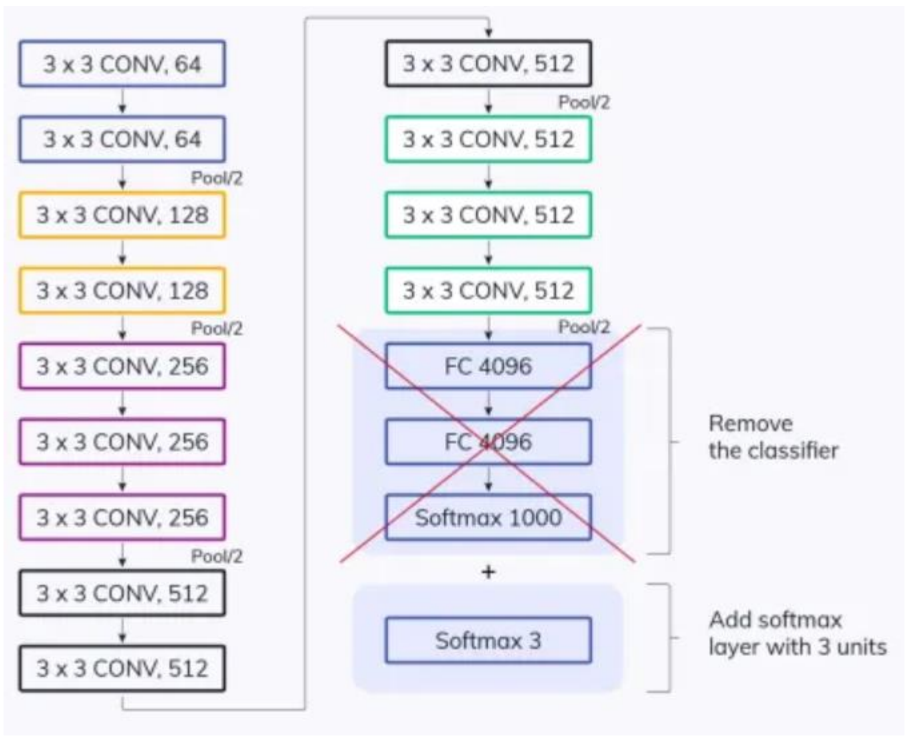


Figure 3.3 Pre trained weights

3.6.1. Freeze layers so they don’t change during training

In order to avoid overfitting concerns, which were covered in the subjects that came before this one, we need to freeze the layers of the pre-trained model during the first stages, which is very important. This is because we do not want the weights in those layers to be re-initialized, which is the reason why this is happening. If that is the case, we are going to lose all of the information that we have gained in the past. This will be the same as starting the model's training from the very beginning.

3.6.2. Train the new layers on the dataset

In order to train the new layers on the dataset, it is highly likely that the final output of the pre-trained model will be different from the output of our model. For example, models that have been pre-trained using the ImageNet dataset will be able to produce one thousand classifications. On the other hand, our model might only contain two different classes. In this particular instance, the model was trained with an additional output layer already in place.

As a result, we have built multiple more dense layers as needed, but most importantly, a final thick layer with units equal to the number of outputs required by our model. After selecting the base model and transfer learning method, now we have to train the model on our own dataset for classification between threat and nonthreat classes. Classification between Fire Flames and Smokes.

3.6.3. Improve the model via fine-tuning

Following the completion of the preceding procedures, We have proposed a model capable of making predictions on our own dataset. We fine-tuned it to increase its performance. Fine-tuning is accomplished by unfreezing the underlying model or a piece of it and retraining the entire model on the entire dataset at a very low learning rate. The slow learning rate will improve the model's performance on the new dataset while limiting overfitting.

Since the dataset is small and the model is big, the learning rate must be low. This is an example of overfitting, that is why the rate of learning is so low. After we've made these changes, we'll need to recompile the model to make sure they work. This is because when the compile method is called, the behaviour of a model is locked. This means that we have to recompile the model every time we want to change how it works. The model will be trained again while callbacks are used to make sure it doesn't overfit, as shown in figure 3.4. Threat detection as it appears in the final stages of the pipeline. When we first started photographing, we used to pre-process all of our images, deleting extraneous white space and keeping only the most important data from the raw file. The detection model projected the position of the alleged harmful objects using the clipped image.

Final pipeline included the introduction of Faster R-CNN model. Segmentation models were used to anticipate how each pixel in the projected areas would be classified after the projected areas were cut out of the image. DeepLabV3+ was applied as a final step in the pipeline to improve the model's segmentation discussed in 3.5.

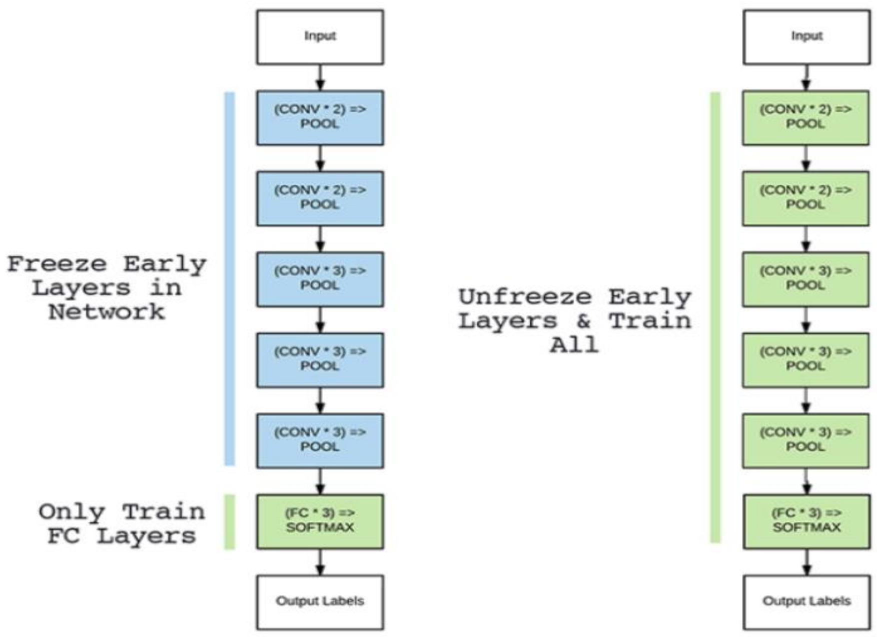


Figure 3.4 Performance and Freezing through fine-tuning

Arouse depthwise convolution and depthwise separable convolution concepts are coupled for the most recent iteration of the technology, which considerably reduces network parameters while preserving or even boosting network performance. Convolution in depth is arousing. Squeeze-and-excitation (SE) blocks from the ResNext-50 architecture were also added to enhance the encoder model. To ensure that only pixels with strong detections were considered for further examination, a 95 percent prediction threshold was applied. Sections that lacked pixel-level annotation were skipped. The segmentation strategy serves as a verification model, ensuring that all forecasted zones include potentially hazardous items. The below images discussed in figure 3.5 and figure 3.6 shows the complete threat detection/weapon detection framework clubbed with pre-trained model using transfer learning approach using COCO dataset. The SIXray dataset is used and pre-processed such as tagging and annotating the images as per the labels required for our threat pipeline using annotation.ai and open-source software. The images after annotations and labellings are then fed into model and Faster RCNN is implied on it for training. After no. of epochs and loss found in training time the model classifies between the weapons and non-weapons. After that we have merged and used the weights and pre-defined weights using transfer learning for complete the Human threat detection model and figure 3.6 shows the detection and labelling done at the time of training and deploying of the models.

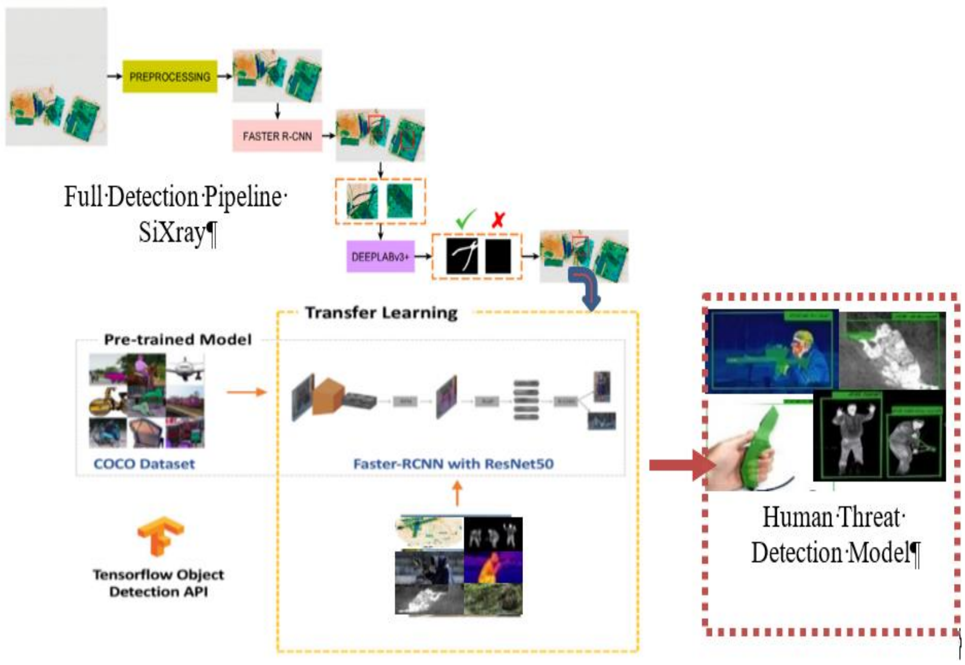
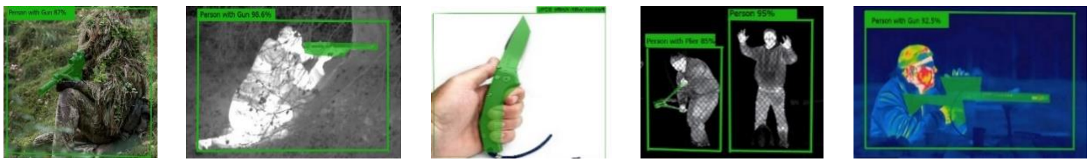

PREPROCESSING
FASTER R-CNN
Full Detection Pipeline SiXray
DEEPLABv3+
Transfer Learning
Pre-trained Model
COCO Dataset
Faster-RCNN with ResNet50
Tensorflow Object Detection API
Human Threat Detection Model

Figure 3.5 Threat detection pipeline using transfer learning approach



(a) (b) (c) (d) (e)

Figure 3.6 Exemplar images of verified detections (a) Person with gun (b) Person with knife (c) Person with gun (d) Thermal imaging person with gun (e) Person with plier

Similarly, we have used the same approach for the FireNet model. The transfer learning approach is used to merge the FCN+ Faster RCNN to create the Mask RCNN for detection of fire flames and classification between Smoke and Fire. After comparing all the base models and segmentation models we have came up with the conclusion that the Faster RCNN and Mask RCNN model will address all the gaps with less computation power for our proposed methodology and areas in which we proposed to work upon.

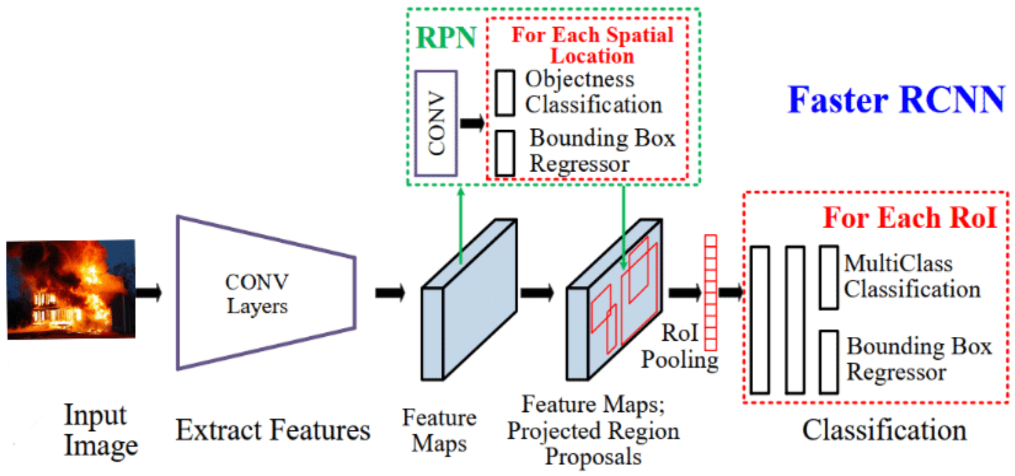


Figure 3.7 Faster RCNN + FCN(Fully Convolutional Network) -> Mask RCNN

The essential idea behind mask RCNN is that faster RCNN gives two outputs for each candidate item: a class label and a bounding-box offset. We extend this by adding a third branch that provides the object mask. This is a binary mask that determines the pixels in the bounding box to which the item belongs. The additional mask output, on the other hand, is not connected to the class or box outputs in any way. Instead, it requires the extraction of a significantly more precise spatial arrangement of an object. This is achieved by the Mask RCNN through the utilization of a Fully Convolutional Network (FCN). In conclusion, Mask RCNN is a huge architecture that combines Faster RCNN and FCN into a single unit. The loss function of the model is a representation of the total loss that occurs during the classification process, the production of bounding boxes, and the development of the mask. In the following chapter, we will discuss the experimental setup, the process of selecting the learning rate, the approaches for augmenting data, and the process of optimizing the hyperparameters along with the training and testing functions that are utilized in our suggested models.

3.7. Summary

In this chapter, we have covered the problem that we have found in the detection of threats, as well as the problems that are associated with training the model from scratch. After that, we talked about the transfer learning approach, which is the most important component of our model. After talking about transfer learning, we next covered the implementation of transfer learning on the data set as well as the fine tweaking of the pre-trained model. After that, we did a comparison analysis of the network and the models, found the model that works the best for our application areas, and suggested a new model for the threat detection pipeline. In the following chapter, we are going to talk about evaluating and analyzing the performance of the given model, and then we are going to provide the conclusion part in the sections where the results are presented.

CHAPTER:4

PERFORMANCE & RESULT ANALYSIS

CHAPTER 4

4. PERFORMANCE AND RESULT ANALYSIS

4.1. Introduction

In chapter 4, we have addressed the execution and performance analysis on two approaches, namely mask and non-mask, fire and smoke, and weaponry, based on the proposed techniques presented in chapter 3. Our original contribution consists of training the deep learning model using a one-shot training strategy on photos collected from current video input resources. These images include identification of crowds and mass gatherings in addition to mask and non-mask faces. Researchers have yet to make use of the trained model, despite its potential to significantly delay the spread of an epidemic like corona covid-19. Using an inductive transfer learning method, we have also enhanced and fine-tuned the pipeline's hyperparameters for this covid-19 application. Through the utilization of this method, the overfitting issue that was present in our trained model can be mitigated using the pre-trained model as a base job. We have finished improving the most essential hyperparameter of the model, which is the dropout value, as part of the process of perfecting the model. In certain quarters, this method is referred to as the regularization technique. The first step involved training a pre-trained model with TensorFlow and OpenCV, as well as implementing an object detection algorithm called single shot multi-box detector and Mobilenet as a classification model over the MSCOCO dataset, which shares features across 90 classes and detects and performs multiple bounding boxes around objects. Faster Regional Convolutional Neural Network model training took much longer than a single-shot multi-box detector model. We used the data augmentation method in addition to the transfer learning technique to help our model overcome the necessity for a large dataset as a prerequisite for the deep learning model. We used less data, but this method meets our needs for converting it to a large dataset with a variety of viewpoints, such as rotating images, flipping them, padding them, cropping them, and transforming them with color changes such as brightness, saturation, hue, and contrast to train our deep learning model. We changed the code that drives our deep learning model and integrated the required data augmentation algorithms to accommodate data augmentation.

4.1.1. Overview Mask and Non Mask detection

The deep learning model's initial training phase consisted of using a single set of still images acquired by real-time video cameras. This collection of photographs displayed individuals with and without masks, as well as large groups of individuals gathered together. If a Corona Covid-19-style outbreak occurs in society, researchers will need to employ the trained algorithm to detect potential hazards early on. Weapon detection using X-Ray images and Fire flame detection is also discussed as different application areas. At this time, no one is using the model in their study. An inductive transfer learning strategy was used to fine-tune the hyperparameters of the pipeline that was constructed for all different kinds of application domains. This was done to mitigate the overfitting issue produced by a pre-trained model.

The dropout value, commonly known as the regularization process or procedure, was optimized to improve the model. A pre-trained model was built with TensorFlow and OpenCV, and then used to recognize objects as they moved across a real-time surveillance system, while another model, Mobilenet, was used to classify the MSCOCO dataset using common features from 90 classes. To compare these results, a more advanced neural network, the FRCNN, was used. This model took longer to train than a single-shot multi-box detector model because it required more computational time.

It was possible to reduce the classification system to just two categories by using RMS prop for gradient descent on a dataset of 450 training examples of people wearing cloth over their faces to simulate a mask and not wearing a mask, and by using RELU as the activation function in a CNN layer with a minimum depth of 16 and a batch size of 100. The dataset consisted of people wearing cloth over their faces to simulate a mask and not wearing a mask. In the realm of machine learning algorithms, hyperparameters are an essential component. It is possible to enhance the precision of the deep learning-based object identification model by adjusting the hyperparameters that are detailed below.

Table 4.1: Hyperparameters & Values for training a Model

Parameter Name	Parameter Type	Required Range
Learning Rate	Continuous parameter	min:1e-6, max:0.5
Batch size	Integer parameter	min:0.8, max:64
Momentum	Continuous parameter	min:0.0 ,max :0.1
Optimizer	Categorical parameter	[SGD, ADAM, RMSPROP]
Weight Decay	Continuous parameter	min:0.0, max;0.999

The hyperparameters and the values that should be used for them are listed in Table 4.1. These are decided upon before the training process begins so that the model can be optimized to its full potential. In order to achieve a higher level of precision with the model, in addition to the parameters that have already been discussed, such as the number of epochs, dropouts, hidden layers, and units, activation functions are also incorporated as hyperparameters. This was done in order to improve the precision of the model. Because it has the appropriate learning rate, our model is able to train rapidly and accurately while simultaneously reducing the amount of loss function it sustains. This is made possible by the fact that it regulates the weights after each batch size. When we optimize batch size, we train our model on less sample data points, so we need less memory than when we train it on a bigger data set. Our memory simply isn't big enough to hold all of that information.

We used data augmentation in conjunction with the transfer learning method to assist our model in overcoming the requirement of a huge dataset, which is an essential requirement of the deep learning model. This requirement is a fundamental necessity of the deep learning model. For the purpose of training our deep learning model, we made use of a smaller amount of data, and this approach satisfied our requirements for expanding it into a large dataset with a variety of viewpoints by, for example, rotating photographs, flipping them, padding them, cropping them, and altering their color in various ways, such as by adjusting brightness, saturation, hue, and contrast. For the purpose of data augmentation, the code for the deep learning model was modified to include the appropriate data augmentation algorithms.

A direct correlation exists between the amount of time lost and the number of steps taken (see Figure 4.1). Learning rate in terms of localization loss is the hyperparameter that governs the weight values in deep CNNs and is adjusted in our network in terms of loss gradient. Changing the learning rate affects the model's accuracy, and the value of the learning rate dictates the rate at which we descend the slope of the function: a low value means slow descent, while a high value means rapid descent. When figuring out the rate of learning, it is essential to be aware of any local minima that may be present. As a result, a low rate of learning is preferable in order to ensure that local minima are covered.

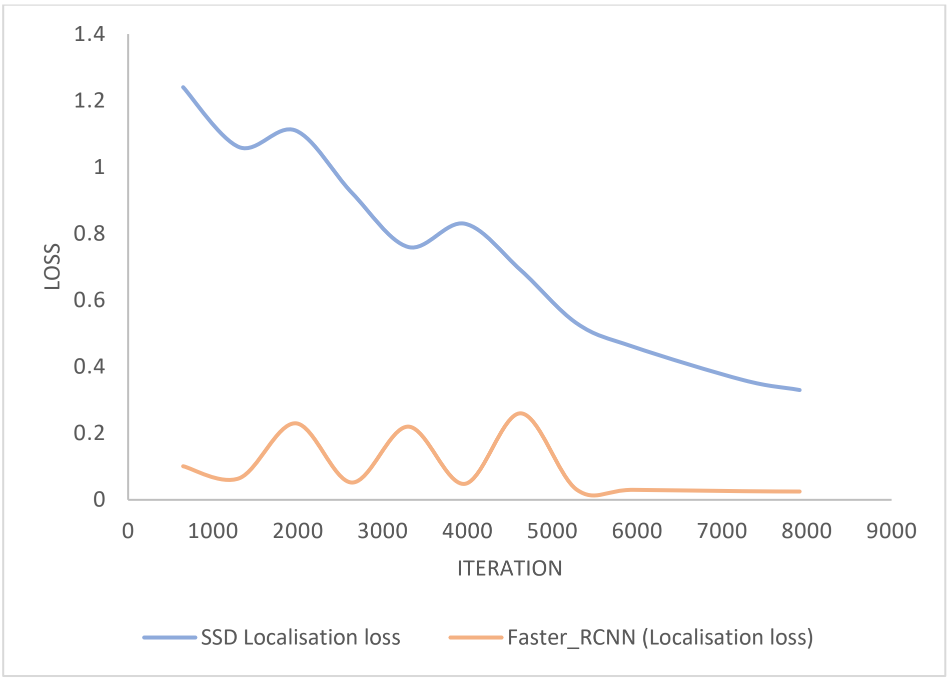


Figure 4.1 Localization Loss for training a model

In order to compensate for all of these different aspects, our initial training is set at a relatively fast pace. There is a compelling justification in favor of utilizing large learning due to the fact that the initial random weights assigned by the network are not ideal. When compared favourably to a SSD & FRCNN that takes use of loss functions, as shown in Figure 4.2, the model has been adequately trained if it can be considered superior to both of these methods.

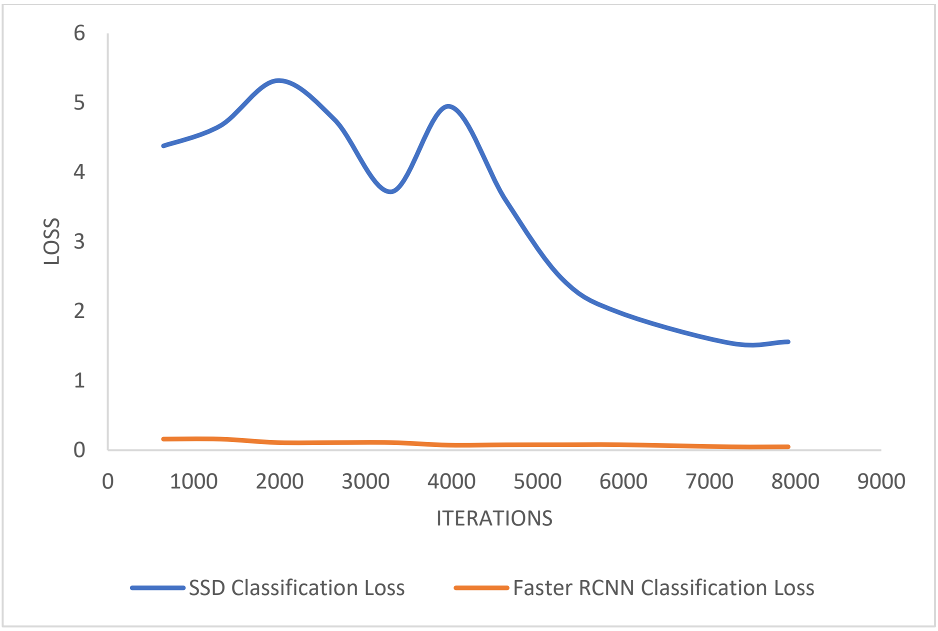


Figure 4.2 Classification loss for training the model

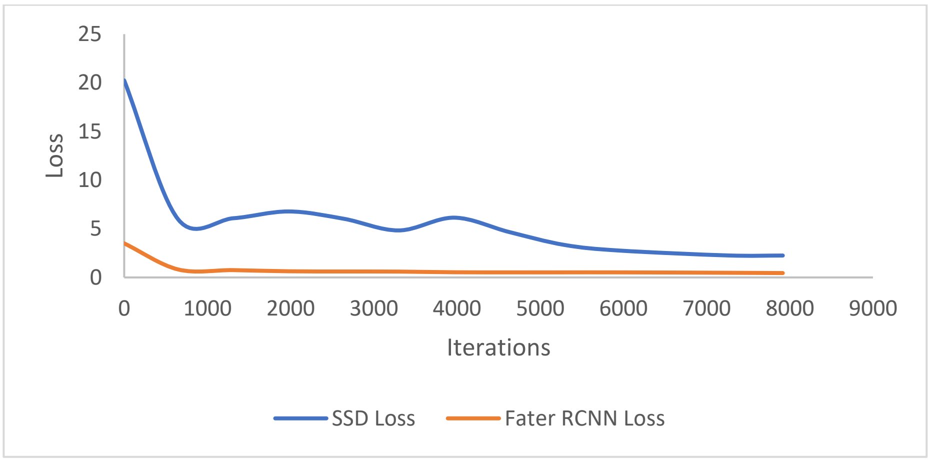


Figure 4.3 Total loss for training the model

The training loss is calculated using the loss function, which is a mechanism for describing a loss function to determine how much an algorithm earns (17) (18). It is a means of determining how effectively a certain algorithm mimics the training data. We can see in figure 4.3 that the initial loss value is fairly significant, but it steadily declines and approaches 1 after a given step. It is an indication that a reduction in loss results in an improvement in the classification accuracy.

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{conf}(x, l, g) \qquad (17)$$

$$L_{conf}(x,c) = -\sum_{i\in Pos}^{N} x_{ij}^{p} \log(c_i^p) - \sum_{i\in Neg} \log(c_i^0) \text{ where } c_i^p = \frac{exp(c_i^p)}{\sum_p (c_i^p)} \quad (18)$$

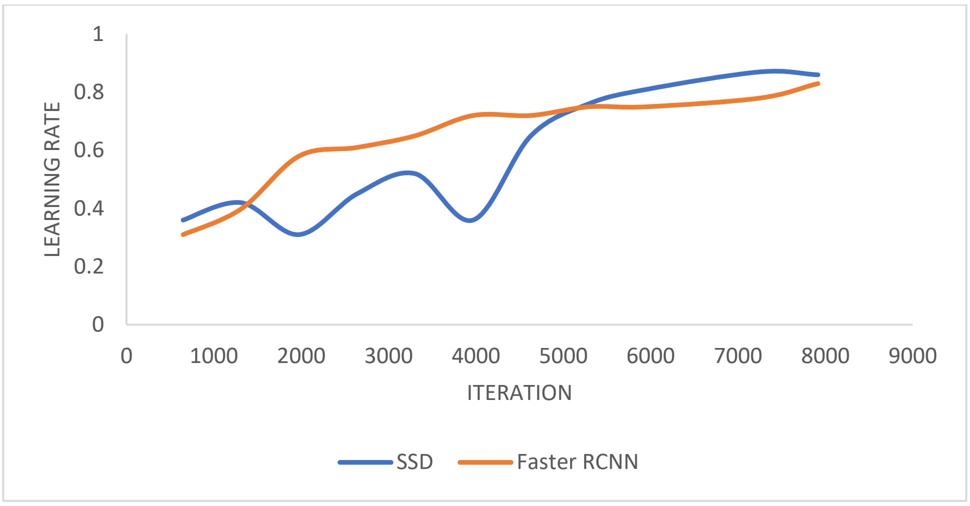


Figure 4.4 Comparative accuracy between both the models

As depicted in Figure 4.4 this system's ability to detect and identify a covid-19-avoidant face mask was demonstrated during validation testing. To train and achieve accuracy, as well as to detect the mask, an RCNN with a lower cycle time takes too long. Training the model from scratch or employing the Faster RCNN are both suboptimal choices compared to the recommended SSD technique in terms of accuracy, learning rate, classification loss and time spent in training.

This allows the model to conduct automatic mask identification on our own dataset with an IOU value of 0.75, and it also allows the model to annotate any type of threat. The SSD mobilenet v1 model presented in figure 4.5 benefits from the application of transfer learning techniques and hyper parameter optimization and tuning. The training processes that we have chosen are superior to those that were used in the past in terms of picking acceptable bounding boxes, sampling from a variety of different places, scaling, and aspect ratio.

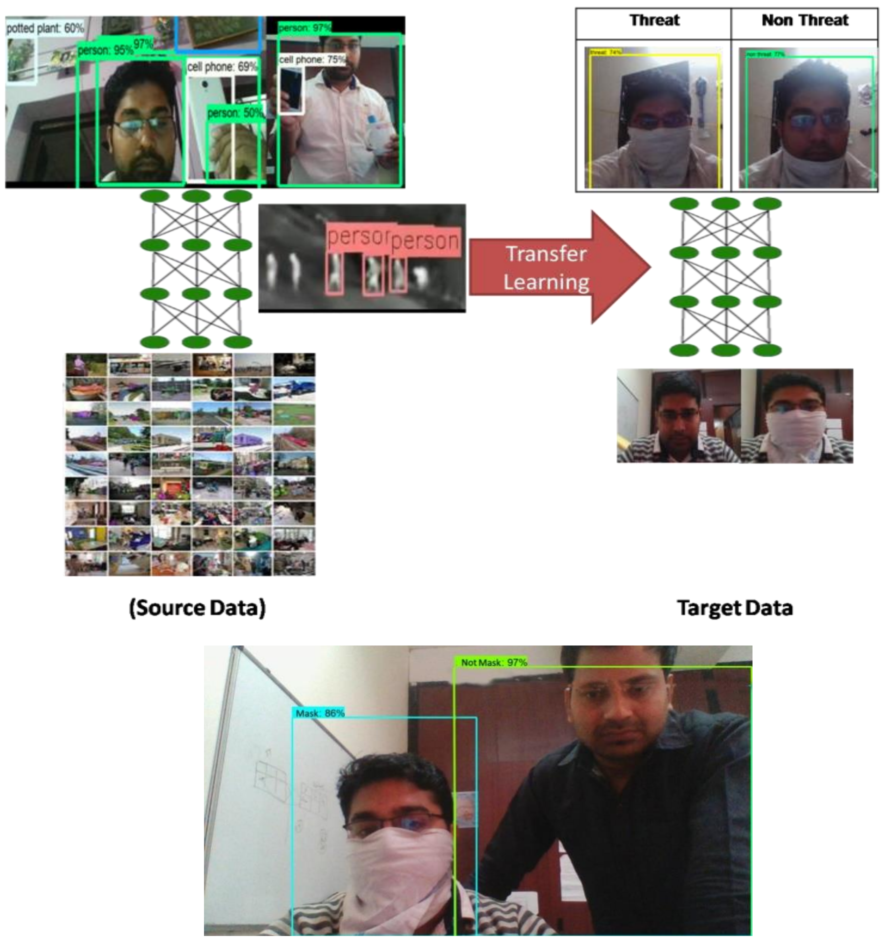


Figure 4.5 Flow diagram of experimentally implemented transfer learning

4.1.2. Overview of Fire Flames detection

Following to the fruitful results of the classification between person with masks and those not wearing masks. We trained our models using predefined subsets of both our own dataset and some of the dataset used in the Kaggle competition. It was recommended that the segmentation model should be trained using only the pixel-level annotated photographs from the training subsets, meanwhile the detection model should be trained using all of the photographs that are included in the training subsets. As noted earlier in the part preceding this one, we created the patches that were used in the training of the segmentation model detailed in the next section. All of the patch's spatial dimensions were brought to the same level, which is 192 pixels on each side.

The training of Faster-RCNN required 80,000 iterations of stochastic gradient descent (SGD), with a batch size of 2. The learning rate was set at 0.001, and after forty-odd iterations, the learning rate was decreased linearly by 0.10 between the third and fifth decimals. The Adam optimizer was used for 250 epochs and 32 batches with a batch size of 2, and the stochastic gradient descent (SGD) algorithm was utilized for 80,000 iterations with a base learning rate of. These were the parameters that were used to train DeepLabV3+. The learning rate decreased by a factor of 0.1 after each of the 75th, 150th, and 200th epochs, respectively. Take, for instance, the fact that the DeepLabV3+ is capable of learning at an epoch rate of 0.001. In order to evaluate the quality of our findings in relation to those obtained by the baseline model, we trained Mask R-CNN using the entire dataset, each and every case, and all of the annotations included. The Mask R-CNN was trained using parameters and CNNs with an R-backbone.

4.1.3. Transfer Learning Method

A huge amount of data must be collected and stored in order to train CNN-based algorithms. Small-scale picture and video fire databases, on the other hand, are now unable to meet demand. Table 4.2 contains a few small-scale fire image/video databases that were created for research purposes. As a result, this study acquired 30,180 photos from local public fire image/video databases, huge public image/video data sets, historical experimental data from academic organizations. Images which were obtained from a variety of sources and are of varying sizes, are included in the data collection. The horizontal images are approximately 500375 pixels in size, while the longitudinal images are approximately 375500 pixels in size. The object classes "fire" and "smoke" in each picture are separated by a separation of two disturbance classes ("fire" and "smoke") ("fire-like" and "smoke-like").

Table 4.2 Statistics for Fire and Smoke Images

Locations	**Targets**		**Disorders**		**Images**
	Smoke	Fire	Similar to smoke	Similar to Fire flames	
Indoor	4375	5640	1240	3275	7640
Outdoor	3065	4055	2750	3950	5760
Total	7440	9705	3990	7225	13400

Table 4.3 Images/video repositories

Repository	Format	Objects	Resources
Repository 1	Video	Fire Flames, Smokes, disturbance	Computer Vision-Based Fire Detection Software (bilkent.edu.tr)
Repository 2	Video dataset	Fire Flames, Smokes and disturbance	Computer Vision and Pattern Recognition Laboratory Homepage (km.ac.kr)
Repository 3	Image & Video dataset	Smoke	Welcome to the Wildfire Observers and Smoke Recognition Homepage (Feb.hr)
Repository 4	Image dataset, Video	Smoke	Fire Detection Research Group (ustc.edu.cn)
Repository 5	Video	Fire	Fire | NIST

Table 4.3 displays the statistics for the fire photos in the collection : Statistics for Fire Images A total of 13,400 fire photographs were included in the data collection. Total image data collection was 9705 "fire" elements & 7440 "smoke". Apart from that, there are 15800 photos in the collection that do not depict any fire at all. A total of sixteen different types of programme settings were used to create these images, which contain 49,600 disturbances altogether. There is at least one interruption in each image. It was determined that approximately half of all images in the dataset would be used in this study, and approximately half of all images would be used in this study.

4.2. Experiments

Due to the fact that the four CNNs had been trained on a massive picture dataset (COCO), they performed exceptionally well when it came to identifying visual objects. In this research, the pre-trained networks are exported to COCO for further analysis. The front end of the feature extraction network is immobilized when the transfer learning technique is utilized, and the networks were only fine-tuned using the training and validation data.

1. A workstation with a 3.6 GHz Intel Core I7-7700 processor, 16 GB of DDR4 RAM, and an NVIDIA Titan X Pascal GPU that has 3840 compute units (CUs).
2. Ubuntu 16.0.4 is the operating system.
3. The momentum of the asynchronous SGD is set at 0.9.
4. The maximum quantity of IOU is 0.9.
5. The starting rate of learning is 0.001.
6. The learning rate decreases by a factor of ten after 120 thousand iterations, and by a factor of ten more after 160 thousand iterations. There have been roughly 200 thousand iterations in total.

When annotation results from numerous annotation sessions were stored in different JSON files, the same picture identification numbers appeared in various JSON files. To create a one-to-one link between used photos and identification numbers, the annotation results were integrated by assigning unique image identification numbers. One JSON file was prepared for training and the other was created for validation. The polygon vertices' coordinates were then imported and recorded in the AxisFireDataset object, which was subsequently linked to the picture identification internally.

In order to calculate the training loss, you must first specify the loss function for the method in question (21) (22). This method can be used to evaluate the performance of an algorithm. As a result, the loss amount was overly high at the beginning. As opposed to other loss function hinges, the cross-entropy loss function has a higher convergence rate. Classification can be used to predict the output from a limited collection of categorical values based on photographs of fire and smoke if you have a large database.

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{conf}(x, l, g) \qquad (21)$$

$$L_{conf}(x,c) = -\sum_{i\in Pos}^{N} x_{ij}^{p} \, log\left(c_i^p\right) - \sum_{i\in Neg} log\;(c_i^0)\; where c_i^p = \frac{exp(c_i^p)}{\sum_p (c_i^p)} \quad (22)$$

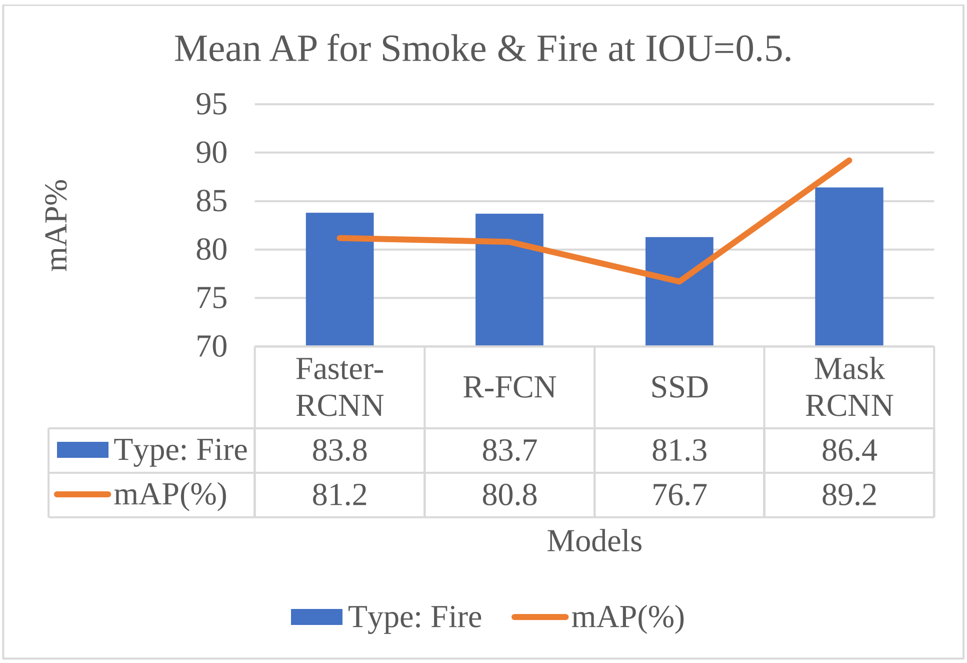
4.3. Evaluation Procedure & Techniques

By manually reducing the learning rate and supplementing the training data, the Mask R-CNN model was improved. Decreased learning rate during training is predicated on the premise that a high learning rate leads to a faster approach to the loss function's minimum; nevertheless, reduced parameter changes may result in a faster settling into the approaching minimum after a few epochs. A scheduler for learning rate was developed, which began at a learning rate of 0.01 and decreased linearly with each training session. Keras callbacks can now be scheduled. In the data augmentation stage, the current batch of images was transformed using a pipeline of transformations. As a result, the model is trained with slightly different data at each level. Four augmentation techniques were utilised with the Python package 'imaging.'

All augmenters were applied to a random half of the images in a batch in the following order:

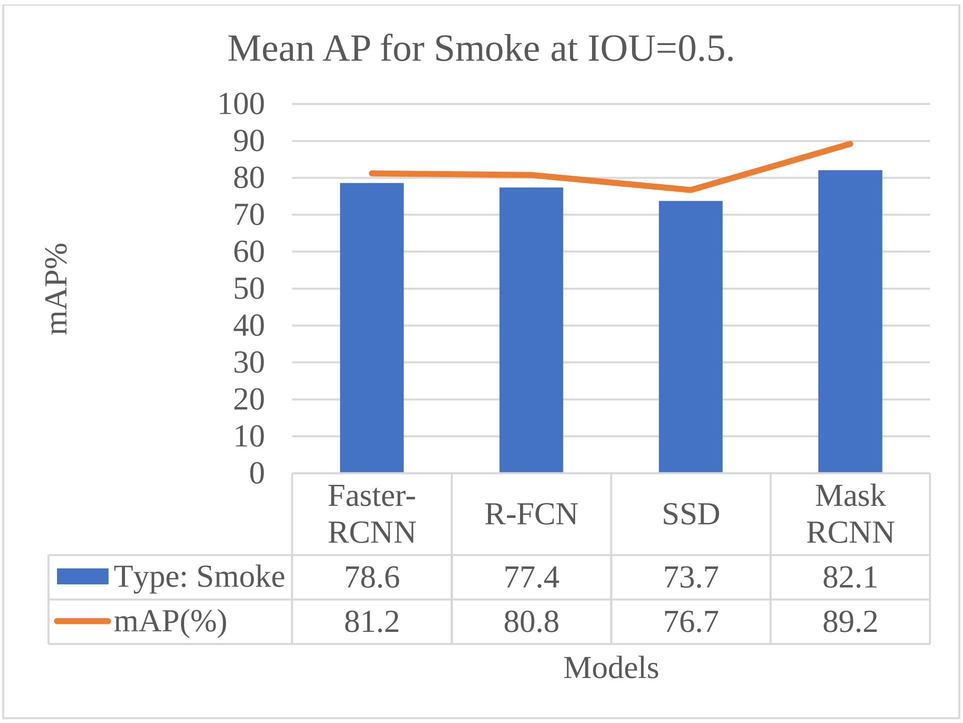
- Horizontal flip
- Vertical flip
- Cropping.
- Gaussian noise or blur is used.
- A change in contrast or brightness
- The affine transformation is used.

4.4. Results



	Faster-RCNN	R-FCN	SSD	Mask RCNN
Type: Fire	83.8	83.7	81.3	86.4
mAP(%)	81.2	80.8	76.7	89.2

Figure 4.6 Mean AP for Smoke & Fire at IOU=0.5.



	Faster-RCNN	R-FCN	SSD	Mask RCNN
Type: Smoke	78.6	77.4	73.7	82.1
mAP(%)	81.2	80.8	76.7	89.2

Figure 4.7 Mean AP for Smoke at IOU=0.5.

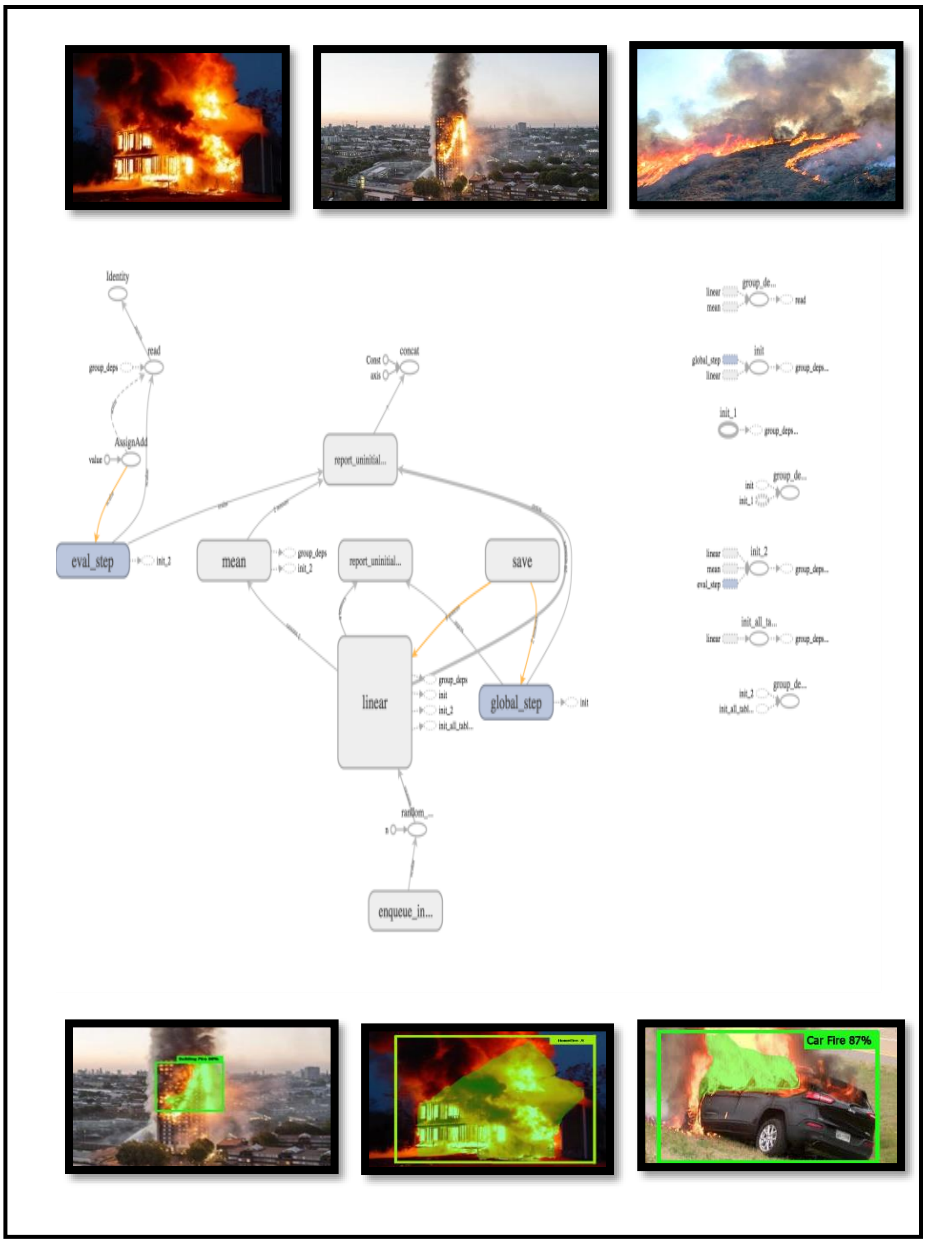


Figure 4.8 Examples of the results of fires with bounding box masking

In terms of quantitative indicators, our suggested strategy increased performance for fire extraction and segmentation tasks, particularly fire detection. However, we feel the performance might be enhanced further by addressing the following issues.

- There are more and more different types of fire examples. Deep neural networks require a large number of training examples to function properly, which other algorithms do not. Even though we classified hundreds of fire flames in order to train our network, more examples can significantly improve performance. Aside from that, the size and shape of the fire flames can be customised. When it comes to things like fire and smoke, the size, aspect ratio, and shape can all vary widely. An exhaustive collection should include as many samples as possible. Distracting objects, such as vehicles, ships, highways, and other similar structures, can cause the detector to lose focus when faced with a busy background. Figure 4.7 shows an illustration of this. It is preferable to designate a specific number of fire flames when dealing with complex backgrounds.

- The bounding box can be rotated in any direction you prefer. The value of a rotated angle is regressed against a fixed value using RPN. As there is only one type of fire to identify, the RPN network ROIs should be as close as possible to the detection branch ROIs. As a result, the final rotation angles are determined by the RPN network. But we thought the second stage, which is similar to bounding box regression, might be able to further enhance them. We plan to focus on two possible solutions in the future.
- Network compression is commonly used to compress networks. A huge number of parameters in the Mask R-CNN architecture necessitates a large quantity of computational resources, which will reduce inference time. Because of the fast growth of mobile devices and the necessity for real-time processing, researchers have been focusing on lowering the size of deep models while retaining their performance. Light network design, network pruning, and kernel sparseness are all taken into account by these methods. Both the Mask R-CNN and the suggested technique rely on residual networks for their backbones, which can be pruned to lighten their overall architectures.

4.5. Summary

As in this chapter we have discussed about our proposed methodology and implementations. We have addressed and filled up the gaps which were the objectives for us in carrying out the research work. We have discussed the result comparative analysis along with the optimization technique in both the applications we have discussed as a test case for our proposed methodology i.e weapon detection, mask detection and fire, smoke detection. In next chapter we will provide with brief summary and conclusion about of research work.

CHAPTER:5

SUMMARY & CONCLUSION

CHAPTER 5
SUMMARY AND CONCLUSION

5.1. Conclusion

In this work, We reached a conclusion regarding our work after implementing the experiment and identifying the accuracy and best models for our application. In this study, we explored the impact of incorporating pixel-level analysis throughout the threat detection workflow of a large-scale and unbalanced security picture dataset. We re-introduced object separation as an unique task area for analyzing security images generated from x-ray machine, answering the first half of the challenge, which was to accurately differentiate target object instances from X-ray images, including pixels shared by overlapping objects. We developed a simple and effective item separation technique that comprises of a detection model for detecting prospective risk zones and a segmentation model for confirming the presence of threats in the projected regions. We chose the top detection and segmentation models after closely studying existing state-of-the-art deep learning algorithms.

The application of a deep learning-based Mask R-CNN model and transfer learning to our own trained data set obtained from online sources such as Google Research and Kaggle results in 100% success in identifying known dangers. Because the model did not need to be built from scratch, the transfer learning strategy was used to train it with fewer data sets, and weights were taken from a pre-trained model on the MS COCO dataset. With an average accuracy score of 0.89 and segmentation masking of 0.85 for a set threshold of 0.9, the deep learning model efficiently segmented and detected the fire. We provide an autonomous fire extraction strategy based on an upgraded Mask R-CNN framework that identifies the rotating bounding boxes of flames while also segmenting them from exceedingly complex backgrounds in this study. The rotation anchor with an inclined angle is used in the RPN stage to regress the rotation bounding box of buildings. Following anticlockwise rotation and ROI Alignment with Data Augmentation, the feature maps are given to the multi-branch prediction network. RFB modules are also introduced to the segmentation branch to manage multi-scale variability, while other branches produce classification scores and horizontal rectangle coordinates. Finally, the inclined angle is used to rotate the mask and rectangle bounding box clockwise in the final instance segmentation result. Experiment results on a new dataset with different flames and backgrounds show that our technique may yield good results. As a consequence of our training and testing, the fire detection model will protect society from all fire-related hazards by providing early warnings and essentially eliminating the loss of living or non-living assets. Our findings on a deep learning-based segmentation model, namely Mask RCNN and transfer learning, demonstrated state-of-the-art accuracy for recognizing fire flames and smokes in any scenario. The proposed approach and technique do not need to be developed from scratch because the weights were taken and transferred from a pre-trained model on the MS COCO and were used for training a model with datasets having less data with the help techniques known as data augmentation as it will reduce the computational time with an average accuracy score of 82.6 percent for detecting smoke and 89.4 percent for detecting fire flame on a set of threshold IoU 0.9, the trained deep learning mode. Furthermore, the suggested autonomous fire extraction approach, which is based on an improved Mask RCNN framework, can identify the spinning bounding boxes of flames while also segmenting them from exceedingly complex backgrounds. The rotation anchor with an inclined angle was used in the RPN stage to regress the rotation bounding box of buildings, and the feature maps were then passed to the multi-branch prediction network after anticlockwise rotation and ROI Align. In addition, to control multi-scale variability, RFB modules were added to the segmentation branch, while other branches produced classification scores and horizontal rectangle coordinates. Finally, based on the inclined angle, the final instance segmentation result rotated the mask and rectangle bounding box clockwise. Experiment results on a new dataset with different flames against different backdrops show that our strategy could deliver good results for detecting fire, which is critical for protecting society from all fire-related threats by providing early warnings and virtually eliminating the loss of lives and non-living objects. To improve the accuracy of our technique, we propose that future studies study sample annotation, network structure optimization, and compression by merging the model with IoT-based devices. Future research could focus on sample annotation, network structure optimization, and compression by merging the model with IoT-based devices for instant utilization with more accuracy, which will aid in overcoming any loss of life.

5.2. Major Contributions of the Work

The most significant contribution of the research mentioned in this thesis is that it presents an automatic threat recognition and detection system based on Transfer learning methods, X-Ray image detection, and data augmentation techniques. The following are the thesis's significant contributions:

- Our proposed system has high detection rate with less training set required.
- Optimization of Learning rate for best results.
- Integration of Mask RCNN with Faster RCNN and SSD for transfer the predefined weights.
- Data Augmentation technique for boosting up the limited data set.

6. References

[1]. Fares Jalled and Ilia Voronkov. Object Detection Using Image Processing. arXiv preprint arXiv:1611.07791, 2016.

[2]. Krizhevsky, A., Sutskever, I. and Hinton, G.E. (2012) Imagenet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems, 25, 1097-1105.

[3]. Tom M. Mitchell. Machine Learning. McGraw-Hill Book Company, 1997

[4]. Michael A. Nielsen. Neural Networks and Deep Learning. Determination Press, 2015.

[5]. Robert D. Hof. Deep Learning: With massive amounts of computational power, machines can now recognize objects and translate speech in real time. Artificial intelligence is finally getting smart. MIT Technology Review, 2018.

[6]. Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep Learning. Nature, 521(7553):436, 2015.

[7]. Srinivas TK and Ramakrishnan Viswanathan. Cognitive Computing: The Next Stage in Human/Machine Coevolution. Technical report, Cognizant White Paper, 2017.

[8]. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In Proceedings IEEE Conference on 90 Computer Vision and Pattern Recognition, pages 1–9. IEEE, 2015.

[9]. Simonyan, K. and Zisserman, A. (2015) Very Deep Convolutional Networks for Large-Scale Image Recognition. The 3rd International Conference on Learning Representations (ICLR2015). https://arxiv.org/abs/1409.1556

[10]. Andrew Ng. What data scientists should know about deep learning. Extract Data Conference, 2015.

[11]. Yoshua Bengio et al. Learning deep architectures for AI. Foundations and trends R in Machine Learning, 2(1):1–127, 2009.

[12]. Robert D. Hof. Deep Learning: With massive amounts of computational power, machines can now recognize objects and translate speech in real

time. Artificial intelligence is finally getting smart. MIT Technology Review, 2018.

[13]. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), 2005, pp. 886-893 vol. 1, doi: 10.1109/CVPR.2005.177.

[14]. Cicero dos Santos and Maira Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In Proceedings 25th International Conference on Computational Linguistics, COLING 2014: Technical Papers, pages 69–78, 2014.

[15]. Khan, A., Sohail, A., Zahoora, U. et al. A survey of the recent architectures of deep convolutional neural networks. Artif Intell Rev 53, 5455–5516 (2020). https://doi.org/10.1007/s10462-020-09825-6

[16]. A. Shrestha and A. Mahmood, "Review of Deep Learning Algorithms and Architectures," in IEEE Access, vol. 7, pp. 53040-53065, 2019, doi: 10.1109/ACCESS.2019.2912200.

[17]. Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting.. Journal of Machine Learning Research, 15, 1929-1958.

[18]. Matthew D Zeiler. ADADELTA: an adaptive learning rate method. arXiv preprint arXiv:1212.5701, 2012.

[19]. LeCun Y, Jackel LD, Bottou L, Cortes C, Denker JS, Drucker H, Guyon I, Muller UA, Sackinger E, Simard P, et al. Learning algorithms for classification: a comparison on handwritten digit recognition. Neural Netw Stat Mech Perspect. 1995;261:276.

[20]. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Commun ACM. 2017;60(6):84–90

[21]. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural net- works from overfitting. J Mach Learn Res. 2014;15(1):1929–58.

[22]. Dahl GE, Sainath TN, Hinton GE. Improving deep neural networks for LVCSR using rectified linear units and drop- out. In: 2013 IEEE international conference on acoustics, speech and signal processing. IEEE; 2013. p. 8609–13.

[23]. Xu B, Wang N, Chen T, Li M. Empirical evaluation of rectified activations in convolutional network; 2015. arXiv preprint arXiv:1505.00853.

[24]. Hochreiter S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. Int J Uncertain Fuzziness Knowledge Based Syst. 1998;6(02):107–16.

[25]. Lin M, Chen Q, Yan S. Network in network; 2013. arXiv preprint arXiv:1312.4400

[26]. Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. In: European conference on computer vision. Springer; 2014. p. 818–33.

[27]. Erhan D, Bengio Y, Courville A, Vincent P. Visualizing higher-layer features of a deep network. Univ Montreal. 2009;1341(3):1.

[28]. Le QV. Building high-level features using large scale unsupervised learning. In: 2013 IEEE international conference on acoustics, speech and signal processing. IEEE; 2013. p. 8595–8.

[29]. Grün F, Rupprecht C, Navab N, Tombari F. A taxonomy and library for visualizing learned features in convolutional neural networks; 2016. arXiv preprint arXiv:1606.07757.

[30]. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition; 2014. arXiv pre- print arXiv:1409.1556.

[31]. Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. In: European conference on computer vision. Springer; 2014. p. 818–33

[32]. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Communication ACM. 2017;60(6):84–90.

[33]. Ranzato M, Huang FJ, Boureau YL, LeCun Y. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In: 2007 IEEE conference on computer vision and pattern recognition. IEEE; 2007. p. 1–8.

[34]. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2015. p. 1–9.

[35]. Lin M, Chen Q, Yan S. Network in network; 2013. arXiv preprint arXiv:1312.4400.

[36]. Bengio Y, et al. Rmsprop and equilibrated adaptive learning rates for nonconvex optimization; 2015. arXiv:1502. 04390corr abs/1502.04390

[37]. Srivastava RK, Gref K, Schmidhuber J. Highway networks; 2015. arXiv preprint arXiv:1505.00387.

[38]. Kong W, Dong ZY, Jia Y, Hill DJ, Xu Y, Zhang Y. Short-term residential load forecasting based on LSTM recurrent neural network. IEEE Trans Smart Grid. 2017;10(1):841–51.

[39]. Ordóñez FJ, Roggen D. Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. Sensors. 2016;16(1):115.

[40]. CireşAn D, Meier U, Masci J, Schmidhuber J. Multi-column deep neural network for traffic sign classification. Neural Network. 2012;32:333–8.

[41]. Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. *In: Proceedings of the thirteenth international conference on artificial intelligence and statistics;* 2010. p. 249–56.

[42]. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. *In: Proceedings of the IEEE conference on computer vision and pattern recognition;* 2016. p. 770–8.

[43]. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. Going deeper with convolutions. *In: Proceedings of the IEEE conference on computer vision and pattern recognition;* 2015. p. 1–9.

[44]. Szegedy C, Iofe S, Vanhoucke V, Alemi A. Inception-v4, inception-resnet and the impact of residual connections on learning; 2016. arXiv preprint arXiv:1602.07261.

[45]. Szegedy C, Vanhoucke V, Iofe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 2818–26.

[46]. Wu S, Zhong S, Liu Y. Deep residual learning for image steganalysis. Multimed Tools Appl. 2018;77(9):10437–53.

[47]. Srivastava RK, Gref K, Schmidhuber J. Highway networks; 2015. arXiv preprint arXiv:1505.00387

[48]. Wu S, Zhong S, Liu Y. Deep residual learning for image steganalysis. Multimed Tools Appl. 2018;77(9):10437–53.

[49]. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ. Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2017. p. 4700–08.

[50]. Kuang P, Ma T, Chen Z, Li F. Image super-resolution with densely connected convolutional networks. Appl Intell. 2019;49(1):125–36.

[51]. Sabour S, Frosst N, Hinton GE. Dynamic routing between capsules. In: Advances in neural information processing systems. San Mateo: Morgan Kaufmann Publishers; 2017. p. 3856–66.

[52]. Arun P, Buddhiraju KM, Porwal A. Capsulenet-based spatial-spectral classifer for hyperspectral images. IEEE J Sel Topics Appl Earth Obs Remote Sens. 2019;12(6):1849–65.

[53]. Ma B, Li X, Xia Y, Zhang Y. Autonomous deep learning: a genetic DCNN designer for image classifcation. Neuro- computing. 2020;379:152–61.

[54]. Wang J, Sun K, Cheng T, Jiang B, Deng C, Zhao Y, Liu D, Mu Y, Tan M, Wang X, et al. Deep high-resolution repre- sentation learning for visual recognition. IEEE Trans Pattern Anal Mach Intell. 2020. https://doi.org/10.1109/TPAMI. 2020.2983686.

[55]. Cheng B, Xiao B, Wang J, Shi H, Huang TS, Zhang L. Higherhrnet: scale-aware representation learning for bottomup human pose estimation. In: CVPR 2020; 2020.

[56]. Adams, A. A., & Ferryman, J. M. (2015). The future of video analytics for surveillance and its ethical implications. Security Journal, 28(3), 272-289.

[57]. Nixon, M., & Aguado, A. (2019). Feature extraction and image processing for computer vision. Academic Press.

[58]. Schmiedel, J. M., Klemm, S. L., Zheng, Y., Sahay, A., Blüthgen, N., Marks, D. S., & van Oudenaarden, A. (2015). MicroRNA control of protein expression noise. Science, 348(6230), 128-132.

[59]. Pflüger, H., & Ertl, T. (2016). Sifting through visual arts collections. Computers & Graphics, 57, 127-138.

[60]. DiCarlo, J. J., & Cox, D. D. (2007). Untangling invariant object recognition. Trends in cognitive sciences, 11(8), 333-341.

[61]. Felzenszwalb, P. F., Girshick, R. B., & McAllester, D. (2010, June). Cascade object detection with deformable part models. In 2010 IEEE Computer society conference on computer vision and pattern recognition (pp. 2241-2248). IEEE.

[62]. Calonder, M., Lepetit, V., Ozuysal, M., Trzcinski, T., Strecha, C., and Fua, P.(2012c). BRIEF: Computing a local binary descriptor very fast. IEEE Transactions on Pattern Analysis and Machine Intelligence. 34(7), 1281-1298.

[63]. Nixon, M., & Aguado, A. (2019). Feature extraction and image processing for computer vision. Academic Press.

[64]. Miller, I., Campbell, M., Huttenlocher, D., Kline, F. R., Nathan, A., Lupashin, S., et al. (2008). Team Cornell's Skynet: Robust perception and planning in an urban environment. Journal of Field Robotics. 25(8), 493-527.

[65]. K. He, G. Gkioxari, P. Dollar, and R. Girshick. Mask r-cnn. ´ In Computer Vision (ICCV), 2017 IEEE International Conference on, pages 2980–2988. IEEE, 2017.

[66]. T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. Focal ´ loss for dense object detection. IEEE transactions on pattern analysis and machine intelligence, 2018.

[67]. S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia. Path aggregation network for instance segmentation. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8759–8768, 2018.

[68]. Karimi H, Derr T, Tang J. Characterizing the decision boundary of deep neural networks; 2019. arXiv preprint arXiv: 1912.11460

[69]. Li Y, Ding L, Gao X. On the decision boundary of deep neural networks; 2018. arXiv preprint arXiv:1808.05385.

[70]. Alzubaidi L, Fadhel MA, Al-Shamma O, Zhang J, Santamaría J, Duan Y, Oleiwi SR. Towards a better understanding of transfer learning for medical imaging: a case study. Appl Sci. 2020;10(13):4523.

[71]. Yosinski J, Clune J, Bengio Y, Lipson H. How transferable are features in deep neural networks? In: Advances in neural information processing systems. San Mateo: Morgan Kaufmann Publishers; 2014. p. 3320–8.

[72]. Tan C, Sun F, Kong T, Zhang W, Yang C, Liu C. A survey on deep transfer learning. *In: International conference on artificial neural networks.* Springer; 2018. p. 270–9.

[73]. Weiss K, Khoshgoftaar TM, Wang D. A survey of transfer learning. J Big Data. 2016;3(1):9

[74]. Shorten C, Khoshgoftaar TM. A survey on image data augmentation for deep learning. J Big Data. 2019;6(1):60

[75]. Wang F, Wang H, Wang H, Li G, Situ G. Learning from the simulation: an end-to-end deep-learning approach for computational ghost imaging. Opt Express. 2019;27(18):25560–72.

[76]. Pan W. A survey of transfer learning for collaborative recommendation with auxiliary data. Neurocomputing. 2016;177:447–53

[77]. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Commun ACM. 2017;60(6):84–90

[78]. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2015. p. 1–9.

[79]. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 770–8.

[80]. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L. Imagenet: a large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. IEEE; 2009. p. 248–55.

[81]. Cook D, Feuz KD, Krishnan NC. Transfer learning for activity recognition: a survey. Knowl Inf Syst. 2013;36(3):537–56.

[82]. Cao X, Wang Z, Yan P, Li X. Transfer learning for pedestrian detection. Neurocomputing. 2013;100:51–7.

[83]. Raghu M, Zhang C, Kleinberg J, Bengio S. Transfusion: understanding transfer learning for medical imaging. In: Advances in neural information

processing systems. San Mateo: Morgan Kaufmann Publishers; 2019. p. 3347–57

[84]. Alzubaidi L, Fadhel MA, Al-Shamma O, Zhang J, Duan Y. Deep learning models for classifcation of red blood cells in microscopy images to aid in sickle cell anemia diagnosis. Electronics. 2020;9(3):427.

[85]. Alzubaidi L, Fadhel MA, Al-Shamma O, Zhang J, Santamaría J, Duan Y, Oleiwi SR. Towards a better understanding of transfer learning for medical imaging: a case study. Appl Sci. 2020;10(13):4523.

[86]. Alzubaidi L, Al-Shamma O, Fadhel MA, Farhan L, Zhang J, Duan Y. Optimizing the performance of breast cancer classification by employing the same domain transfer learning from hybrid deep convolutional neural network model. Electronics. 2020;9(3):445.

[87]. Shorten C, Khoshgoftaar TM. A survey on image data augmentation for deep learning. J Big Data. 2019;6(1):60

[88]. Saleh AM, Hamoud T. Analysis and best parameters selection for person recognition based on gait model using CNN algorithm and image augmentation. J Big Data. 2021;8(1):1–20.

[89]. Hirahara D, Takaya E, Takahara T, Ueda T. Efects of data count and image scaling on deep learning training. PeerJ Comput Sci. 2020;6:312.

[90]. Moreno-Barea FJ, Strazzera F, Jerez JM, Urda D, Franco L. Forward noise adjustment scheme for data augmentation. *In: 2018 IEEE symposium series on computational intelligence (SSCI). IEEE;* 2018. p. 728–34.

[91]. Dua D, Karra Taniskidou E. Uci machine learning repository. Irvine: University of california. School of Information and Computer Science; 2017. http://archive.ics.uci.edu/ml.

[92]. Johnson JM, Khoshgoftaar TM. Survey on deep learning with class imbalance. J Big Data. 2019;6(1):27.

[93]. Yang P, Zhang Z, Zhou BB, Zomaya AY. Sample subset optimization for classifying imbalanced biological data. In: Pacifc-Asia conference on knowledge discovery and data mining. Springer; 2011. p. 333–44.

[94]. Yang P, Yoo PD, Fernando J, Zhou BB, Zhang Z, Zomaya AY. Sample subset optimization techniques for imbalanced and ensemble learning problems in bioinformatics applications. IEEE Trans Cybernetics. 2013;44(3):445–55.

[95]. Wang S, Sun S, Xu J. Auc-maximized deep convolutional neural fields for sequence labeling 2015. arXiv preprint arXiv:1511.05265.

[96]. Li Y, Wang S, Umarov R, Xie B, Fan M, Li L, Gao X. Deeper: sequence-based enzyme EC number prediction by deep learning. Bioinformatics. 2018;34(5):760–9.

[97]. Li Y, Huang C, Ding L, Li Z, Pan Y, Gao X. Deep learning in bioinformatics: introduction, application, and perspective in the big data era. Methods. 2019;166:4–21.

[98]. Choi E, Bahadori MT, Sun J, Kulas J, Schuetz A, Stewart W. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In: Advances in neural information processing systems. San Mateo: Morgan Kaufmann Publishers; 2016. p. 3504–12

[99]. Zhou J, Troyanskaya OG. Predicting effects of noncoding variants with deep learning-based sequence model. Nat Methods. 2015;12(10):931–4.

[100].Pokuri BSS, Ghosal S, Kokate A, Sarkar S, Ganapathysubramanian B. Interpretable deep learning for guided microstructure-property explorations in photovoltaics. NPJ Computation Mater. 2019;5(1):1–11.

[101].Ribeiro MT, Singh S, Guestrin C. “Why should I trust you?” explaining the predictions of any classifier. *In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining;* 2016. p. 1135–44.

[102].Wang L, Nie R, Yu Z, Xin R, Zheng C, Zhang Z, Zhang J, Cai J. An interpretable deep-learning architecture of capsule networks for identifying cell-type gene expression programs from single-cell RNA-sequencing data. Nat Mach Intell. 2020;2(11):1–11.

[103].Platt J, et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Adv Large Margin Classification. 1999;10(3):61–74

[104].Nair T, Precup D, Arnold DL, Arbel T. Exploring uncertainty measures in deep networks for multiple sclerosis lesion detection and segmentation. Med Image Anal. 2020;59:101557.

[105].Herzog L, Murina E, Dürr O, Wegener S, Sick B. Integrating uncertainty in deep neural networks for MRI based stroke analysis. Med Image Anal. 2020;65:101790.

[106].Pereyra G, Tucker G, Chorowski J, Kaiser Ł, Hinton G. Regularizing neural networks by penalizing confident output distributions; 2017. arXiv preprint arXiv:1701.06548.

[107].Naeini MP, Cooper GF, Hauskrecht M. Obtaining well-calibrated probabilities using Bayesian binning. *In: Proceedings of the... AAAI conference on artificial intelligence. AAAI conference on artificial intelligence, vol. 2015.* NIH Public Access; 2015. p. 2901.

[108].Li M, Sethi IK. Confidence-based classifier design. Pattern Recognition. 2006;39(7):1230–40.

[109].Zadrozny B, Elkan C. Obtaining calibrated probability estimates from decision trees and Naive Bayesian classifiers. *In: ICML*, vol. 1, Citeseer; 2001. p. 609–16.

[110].Steinwart I. Consistency of support vector machines and other regularized kernel classifers. *IEEE Trans Inf Theory*. 2005;51(1):128–42.

[111].Lee K, Lee K, Shin J, Lee H. Overcoming catastrophic forgetting with unlabeled data in the wild. *In: Proceedings of the IEEE international conference on computer vision; 2019.* p. 312–21.

[112].Shmelkov K, Schmid C, Alahari K. Incremental learning of object detectors without catastrophic forgetting. *In: Proceedings of the IEEE international conference on computer vision;* 2017. p. 3400–09.

[113].Zenke F, Gerstner W, Ganguli S. The temporal paradox of Hebbian learning and homeostatic plasticity. Curr Opin Neurobiol. 2017;43:166–76.

[114].Andersen N, Krauth N, Nabavi S. Hebbian plasticity in vivo: relevance and induction. Curr Opin Neurobiol. 2017;45:188–92.

[115].Cheng Y, Wang D, Zhou P, Zhang T. Model compression and acceleration for deep neural networks: the principles, progress, and challenges. IEEE Signal Process Mag. 2018;35(1):126–36.

[116].Wiedemann S, Kirchhofer H, Matlage S, Haase P, Marban A, Marinč T, Neumann D, Nguyen T, Schwarz H, Wiegand T, et al. Deepcabac: a universal compression algorithm for deep neural networks. IEEE J Sel Topics Signal Process. 2020;14(4):700–14.

[117].Mehta N, Pandit A. Concurrence of big data analytics and healthcare: a systematic review. Int J Med Inform. 2018;114:57–65.

[118]. Shawahna A, Sait SM, El-Maleh A. Fpga-based accelerators of deep learning networks for learning and classification: a review. IEEE Access. 2018;7:7823–59.

[119]. Min Z. Public welfare organization management system based on FPGA and deep learning. Microprocess Microsyst. 2020;80:103333.

[120]. Al-Shamma O, Fadhel MA, Hameed RA, Alzubaidi L, Zhang J. Boosting convolutional neural networks performance based on fpga accelerator. *In: International conference on intelligent systems design and applications.* Springer; 2018. p. 509–17.

[121]. Han S, Mao H, Dally WJ. Deep compression: compressing deep neural networks with pruning, trained quantization and hufman coding; 2015. arXiv preprint arXiv:1510.00149.

[122]. Chen Z, Zhang L, Cao Z, Guo J. Distilling the knowledge from handcrafted features for human activity recognition. *IEEE Trans Ind Inform.* 2018;14(10):4334–42. 202. Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network; 2015. arXiv preprint arXiv:1503.02531.

[123]. Lenssen JE, Fey M, Libuschewski P. Group equivariant capsule networks. *In: Advances in neural information pro- cessing systems.* San Mateo: Morgan Kaufmann Publishers; 2018. p. 8844–53.

[124]. Denton EL, Zaremba W, Bruna J, LeCun Y, Fergus R. Exploiting linear structure within convolutional networks for efficient evaluation. *In: Advances in neural information processing systems.* San Mateo: Morgan Kaufmann Publishers; 2014. p. 1269–77.

[125]. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural net- works from overfitting. J Mach Learn Res. 2014;15(1):1929–58.

[126]. Xu Q, Zhang M, Gu Z, Pan G. Overfitting remedy by sparsifying regularization on fully-connected layers of CNNs. Neurocomputing. 2019;328:69–74.

[127]. Zhang C, Bengio S, Hardt M, Recht B, Vinyals O. Understanding deep learning requires rethinking generalization. Communication ACM. 2018;64(3):107–15.

[128].Xu X, Jiang X, Ma C, Du P, Li X, Lv S, Yu L, Ni Q, Chen Y, Su J, et al. A deep learning system to screen novel coronavirus disease 2019 pneumonia. Engineering. 2020;6(10):1122–9.

[129].Sharma K, Alsadoon A, Prasad P, Al-Dala'in T, Nguyen TQV, Pham DTH. A novel solution of using deep learning for left ventricle detection: enhanced feature extraction. Comput Methods Programs Biomed. 2020;197:105751.

[130].Zhang G, Wang C, Xu B, Grosse R. Three mechanisms of weight decay regularization; 2018. arXiv preprint arXiv: 1810.12281.

[131].Laurent C, Pereyra G, Brakel P, Zhang Y, Bengio Y. Batch normalized recurrent neural networks. *In: 2016 IEEE international conference on acoustics, speech and signal processing (ICASSP), IEEE; 2016.* p. 2657–61.

[132].Salamon J, Bello JP. Deep convolutional neural networks and data augmentation for environmental sound classification. IEEE Signal Process Lett. 2017;24(3):279–83.

[133].Pereyra G, Tucker G, Chorowski J, Kaiser Ł, Hinton G. Regularizing neural networks by penalizing confdent output distributions; 2017. arXiv preprint arXiv:1701.06548.

[134].Xin Wang, Yi Qin, Yi Wang, Sheng Xiang, Haizhou Chen,ReLTanh: An activation function with vanishing gradient resistance for SAE-based DNNs and its application to rotating machinery fault diagnosis,Neurocomputing,Volume 363,2019,Pages 88-98,ISSN 0925-2312,https://doi.org/10.1016/j.neucom.2019.07.017.

[135].Tan HH, Lim KH. Vanishing gradient mitigation with deep learning neural network optimization. In: *2019 7th international conference on smart computing & communications (ICSCC). IEEE;* 2019. p. 1–4.

[136].MacDonald G, Godbout A, Gillcash B, Cairns S. Volume-preserving neural networks: a solution to the vanishing gradient problem; 2019. arXiv preprint arXiv:1911.09576.

[137].Dahl GE, Sainath TN, Hinton GE. Improving deep neural networks for LVCSR using rectifed linear units and drop- out. In: *2013 IEEE international conference on acoustics, speech and signal processing.* IEEE; 2013. p. 8609–13.

[138].Iofe S, Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift; 2015. arXiv preprint arXiv:1502.03167.

[139].Sparsh Mittal, Shraiysh Vaishay,A survey of techniques for optimizing deep learning on GPUs,Journal of Systems Architecture,Volume 99,2019,101635,ISSN 1383-7621, https://doi.org/10.1016/j.sysarc.2019.101635.

[140].Information processing systems. San Mateo: Morgan Kaufmann Publishers; 2017. p. 435–44. 217. Hanin B. Which neural net architectures give rise to exploding and vanishing gradients? *In: Advances in neural information processing systems.* San Mateo: Morgan Kaufmann Publishers; 2018. p. 582–91.

[141].Ribeiro AH, Tiels K, Aguirre LA, Schön T. Beyond exploding and vanishing gradients: analysing RNN training using attractors and smoothness. In: *International conference on artificial intelligence and statistics, PMLR;* 2020. p. 2370–80.

[142].D'Amour A, Heller K, Moldovan D, Adlam B, Alipanahi B, Beutel A, Chen C, Deaton J, Eisenstein J, Hofman MD, et al. Underspecifcation presents challenges for credibility in modern machine learning; 2020. arXiv preprint arXiv: 2011.03395.

[143].Hossin M, Sulaiman M. A review on evaluation metrics for data classifcation evaluations. *International Journal of Data Mining & Knowledge Management Process (IJDKP)* Vol.5, No.2, March 2015. DOI : 10.5121/ijdkp.2015.5201

Biography

Anurag Singh was born on 09-Jan-1992. He has done his studies from Kendriya Vidyalaya, B.Tech CSE from Chitkara University, M.Tech CSE from Maharishi Dayanand University. He is having more than 9.0 years of experience in teaching, industry and consulting work for universities and colleges.

He is working as an Assistant Professor and Associate Academic Coordinator in School of computing science & Engineering, Galgotias University, Uttar Pradesh, India. His area of research is in computer vision and deep learning. He has published 6 Scopus publications with 11 Citations and H-index-2 which include conferences and Journals. Along with research and teaching he also play an important role in deploying end to end system establishments along with training to users on how to use moodle & erp with complete SOPs. Reporting to higher management and providing MIS Reports.

He can be contacted on: Anurag.singh485@gmail.com

ORCID: https://orcid.org/0000-0003-0601-8001

Google Scholar: https://scholar.google.com/citations?user=DOwVhI8AAAAJ&hl=en

Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57193580446

Publons ID: https://publons.com/wos-op/researcher/4430371/anurag-singh/