DARKNET/YOLO BASED ANIMAL SURVEILLANCE AND ACCIDENT PREVENTION USING AI

PROJECT REPORT

Submitted by

RANJITH.R (812117104021)

SARMA.S (812117104023)

SHARMILA.M (812117104024)

in partial fullfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

M.A.M. SCHOOL OF ENGINEERING

SIRUGANUR, TRICHY

ANNA UNIVERSITY::CHENNAI 600 025

APRIL-2021

BONAFIDE CERTIFICATE

Certified that this project report **"DARKNET/YOLO BASED ANIMAL SURVEILLANCE AND ACCIDENT PREVENTION USING AI"** is the bonafide work of **RANJITH.R (812117104021) and SARMA.S (812117104023) and SHARMILA.M (812117104024)** studying in final year Computer Science and Engineering at M.A.M. SCHOOL OF ENGINEERING, TRICHY. who carried out the project work under my supervision, during the academic year 2021.

SIGNATURE

HEAD OF THE DEPARTMENT

Ms.S.MURUGAVALLI ,BE.,ME

Department of Computer Science and Engineering,
M.A.M. School of Engineering,
Siruganur
Trichy-621 105.

SIGNATURE

ASSISTANT PROFESSOR

Mrs.P.SIVAMALAR ,BE.,ME

Department of Computer Science and Engineering,
M.A.M. School of Engineering,
Siruganur,
Trichy- 621 105.

Submitted for the project viva voce held on ……

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

First and foremost, we wish to acknowledge our gratitude to the almighty god and our parents, who had made better opportunities in our life, knowledge and good health so far. We would like to express thanks to the Correspondent of our College **Alhaj.Er.M.A.Peer Mohamed., B.E., M.B.A.,** for providing better working environments and educational facilities in a well sophisticated manner.

We convey our heartiest thanks to the respected principal of our college, **Dr.P.Ranjithkumar., M.E., Ph.D.,** for providing us, the necessary infrastructure for successful completion of our project.

We are much grateful to **Ms.S.Murugavalli. B.E.,M.E.,** Head of the Department, Computer Science & Engineering, for her encouragement, valuable comments and many innovative ideas in carrying out this project. Without his timely help, it would have been impossible for us to complete this work.

We acknowledge in no less term the qualified and excellent Assistance with many innovative ideas rendered by our guide **Mrs.P.SIVAMALAR ,B.E.,M.E.,** we owe a debt of gratitude for his/her valuable suggestions, kind inspiration and encouragement

We most sincerely acknowledge the staff members and Technical Assistant of Department of Computer Science & Engineering for their constant inspiration and suggestions. We owe a deep gratitude to our friends for their advice which keeps our spirits high to complete my project.

DECLARATION

I hereby declare that the work entitled "**DARKNET/YOLO BASED ANIMAL SURVEILLANCE AND ACCIDENT PREVENTION USING AI"** is submitted in partial fulfilment of the requirement of the award of the degree in B.E., Anna University, Chennai, is a record of our own work carried out by me during the academic year 2020-2021 under the supervision and guidance of **Mrs.P.SIVAMALAR,B.E.,M.E.,** department of computer science and engineering. The extent and source of information are derived from the existing literature and have indicated through the dissertation at the appropriate places. The matter embodied in this work is original and not been submitted for the award or diploma, either in this work or any other university.

RANJITH.R (812117104021)

S.SARMA (81211104023)

M.SHARMILA (812117104024)

I certify that the declaration made above by the candidate is true.

Mrs.P.SIVAMALAR ,B.E., M.E

(Assistant professor)

ABSTRACT

Now a days animal death increased based on Transportation. This project will decrease death rate from vehicle and train accident. In this project AI will monitor wild animals via surveillance system using DarkNet/yolo. In this system we will analyze and implement Yolo weights and Sound generator we will introduce animal detection and making crackers sound by using Speakers. This paper deals with the practical implementation of deep learning methods for increasing safety and security in a specific ITS scenario: railway crossings. This research work presents our proposed system called Artificial Intelligence-based Surveillance System for Railway Crossing Traffic (AISS4RCT) that is based on a combination of detection and classification methods focusing on various image processing inputs: vehicle presence, pedestrian presence, vehicle trajectory tracking, and railway barriers at railway crossings, railway warnings, and light signaling systems.

CHAPTER 1

INTRODUCTION

More than 32,000 animals, including cattle, lions and leopards, have been killed on railway tracks in the past three years, according to data provided by the railways. As many as 27 wild animals were killed along the walayar (Palakkad district)-Coimbatore railway line by trains during the past 19 years A 12-year-old male elephant died after being hit by an express train in Odisha in the wee hours of December 21, 2020. The animal was crushed to death under the wheels of the Puri-Surat train between Hatibari-Maneswarapur stations near Bhabanipali village within the Sambalpur forest division range of Sambalpur district.

The world has experienced a growing boom in deep learning (DL) and Artificial Intelligence (AI) technologies applied in various use cases and digital services. AI methods have become popular in many industrial fields such as medicine, smart cities, modern cyber physical systems, autonomous vehicles, security systems, and many more. Intelligent Transportation Systems (ITS) and Internet of Vehicles (IoV) services are not exceptions and there is potential for how AI can improve the quality, security, and safety of current ITS applications. Moreover, mature AI techniques allow humans to enjoy new features such as autonomous driving as well as autonomous detection of risky situations. One such AIbased method is deep learning (DL) which is a very complex technique that may replace classical algorithms in order to improve visual object detection and recognition. DL models and its intricate details have been studied in many recent works.

1.1 RAILWAY CROSSING BARRIER

A level crossing is an intersection where a railway line crosses a road or path, or in rare situations an airport runway, at the same level, as opposed to the railway line crossing over or under using an overpass or tunnel. The term also applies when a light rail line with separate right-of-way or reserved track crosses a road in the same fashion. Other names include railway level crossing, railway crossing (chiefly international), railroad crossing (chiefly North American), and grade crossing, road through railroad, crisscross, train crossing, and RXR (abbreviated).

1.2 SAFETY

Safety is the state of being "safe", the condition of being protected from harm or other non-desirable outcomes. Safety can also refer to the control of recognized hazards in order to achieve an acceptable level of risk.

1.3 SECURITY

Security is freedom from, or resilience against, potential harm (or other unwanted coercive change) caused by others. Beneficiaries (technically referents) of security may be of persons and social groups, objects and institutions, ecosystems or any other entity or phenomenon vulnerable to unwanted change. Refugees fleeing war and insecurity in Iraq and Syria arrive at Lesbos Island, supported by Spanish volunteers, 2015 Security mostly refers to protection from hostile forces, but it has a wide range of other senses: for example, as the absence of harm (e.g. freedom from want); as the presence of an essential good (e.g. food security); as resilience against potential damage or harm (e.g. secure foundations); as secrecy (e.g. a secure telephone line); as containment (e.g. a secure room or cell); and as a state of mind (e.g. emotional security).

1.3 TRAFFIC LIGHT

Security is freedom from, or resilience against, potential harm (or other unwanted coercive change) caused by others. Beneficiaries (technically referents) of security may be of persons and social groups, objects and institutions, ecosystems or any other entity or phenomenon vulnerable to unwanted change. Refugees fleeing war and insecurity in Iraq and Syria arrive at Lesbos Island, supported by Spanish volunteers, 2015 Security mostly refers to protection from hostile forces, but it has a wide range of other senses: for example, as the absence of harm (e.g. freedom from want); as the presence of an essential good (e.g. food security); as resilience against potential damage or harm (e.g. secure foundations); as secrecy (e.g. a secure telephone line); as containment (e.g. a secure room or cell); and as a state of mind (e.g. emotional security).

Green light: Allows traffic to proceed in the direction denoted, if it is safe to do so and there is room on the other side of the intersection.

Red light: Prohibits any traffic from proceeding. A flashing red indication requires traffic to stop and then proceed when safe (equivalent to a stop sign).

Amber light: (also known as 'orange light' or 'yellow light') Warns that the signal is about to change to red, with some jurisdictions requiring drivers to stop if it is safe to do so, and others allowing drivers to go through the intersection if safe to do so. In some European countries (such as the UK), red and amber is displayed together, indicating that the signal is about to change to green.

1.4 INTELLIGENT TRANSPORTATION SYSTEM

An intelligent transportation system (ITS) is an advanced application which aims to provide innovative services relating to different modes of transport and traffic management and enable users to be better informed and make safer, more

coordinated, and 'smarter' use of transport networks. Some of these technologies include calling for emergency services when an accident occurs, using cameras to enforce traffic laws or signs that mark speed limit changes depending on conditions.

1.5 OBJECTIVE DETECTION

Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos.Well-researched domains of object detection include face detection and pedestrian detection. Object detection has applications in many areas of computer vision, including image retrieval and video surveillance.

CHAPTER 2

LITERATURE SURVEY

2.1 Title: Invariant Feature based DarkNet Architecture for moving object classification

Author: S.Vasavi Sr, N.Kanthi Priyadarshini ,K.Harsha Vardhan

Year: 2020

Description:

Object detection and classification is important for video surveillance applications. Counting vehicles like cars, truck and vans is useful for intelligent transportation systems to identify dense and sparse roads, track loaded vehicles at the country borders. Even though many solutions such as appearance-based (Multi-block Local Binary Pattern) and model-based((DATMO) algorithm) are proposed to classify the moving objects within the satellite images using machine learning and deep learning techniques, they either have over fitting problems or low performance. Hence these challenges have to be addressed during detecting and classifying the objects. Instead of training the classifiers with hand-crafted features, this paper uses neural network based object detection and classification to achieve promising accuracy better than the humans. Invariant feature concept is added to the existing Darknet Architecture of You Only Look Once (YOLO) and is combined with Faster Region-Based Convolutional Neural Networks (Faster R-CNN) to count the number of vehicles with different spatial locations. This combined model improves feature extraction step and vehicle classification process. The proposed system is tested on two benchmark datasets Cars Overhead with Context (COWC) and Vehicle Detection in Aerial Imagery (VEDAI) for counting the cars and trucks.

2.2 Title: Ubiquitous and Low Power Vehicles Speed Monitoring for Intelligent Transport Systems

Author: Jos´e Luis Calder´on Choy, Graduate

Year: 2020

Description:

This paper presents a high scalability real-time intelligent traffic monitoring system, based on Radio Frequency Identification (RFID). The main features of this system are low cost, low power consumption, traffic monitoring, and connectivity. The systems architecture includes an RFID reader, a passive tag, and a Raspberry Pi. Our solution collects vehicle information from the labels and stores the data into a database by employing only one antenna. The main challenge is that the RFID module is not robust enough to recognize the information in the vehicle's RFID tag on each information query, while the label is in the reading zone. This instability does not allow us to know precisely when and where a vehicle enters or leaves the sensing zone. What is more, the high random error in the power signal and the complexity of its characteristic curve pattern add difficulty to the speed calculation, when we reduce the number of antennas to one. For this reason, an innovative approach has been designed, using customized modular neural network (MNN). This method fits the collected data (power signal vs time) affected by acute random noise, to the characteristic correspondence function among the signal power and the position of the terminal, which domains are dimensionally different. As a result, we can estimate the vehicle speeds and obtain the whole vehicle information. Under this novel method, we are able to reduce the hardware, in comparison with previous approaches, making it cheaper and decreasing power consumption.

2.3 Title: measurement and configuration of DSRC radio for vehicle to railroad (V2R) safety critical communications

Author: Junsung Choi1, Vuk Marojevic2, Aakanksha Sharma3, Biniyam Zewede4, Randall Nealy5, Christopher R.

Year: 2017

Description:

Despite the rapid development of wireless technology, there has been little application of the technology to improve railroad crossing safety. We present vehicle-to-railroad (V2R) channel characterization and Dedicated Short Range Communi-cations (DSRC) performance results at railroad crossings in rural and suburban environments. Our results show that an omnidirectional antenna provides slightly better performance in rural conditions. However, a bi-directional antenna increased warning range by more than 200 m in suburban conditions. A proper configuration of the DSRC radio provides reliable warning of an approaching train for cars near a railroad crossing.

2.4 Title: Intelligent Transportation System in Macao based on Deep Self Coding Learning

Author: Daming Li, Lianbing Deng, Zhiming Cai, Bill Franks, Xiang Yao

Year: 2018

Description:

As the basic content of the construction of the intelligent city, the application of intelligent transportation system plays an important role in improving the quality and safety of modern urban traffic operation. Intelligent transportation system combines information technology, data communication technology, electronic sensing technology, global positioning technology, and geographic information system technology, computer processing technology and system engineering technology together reasonably. The intelligent transportation system is a kind of real-time, accurate, efficient and intelligent transportation management system. In this paper, we introduce the deep code learning technique and apply it to the Macao intelligent system. At the same time, we combine the deep belief network model (DBN) and support vector regression classifier (SVR) as the prediction model, and use the deep belief network model to learn traffic flow characteristics.

2.5 Title: Preserving Privacy in the Internet of Connected Vehicles

Author: Soheila Ghane, Alireza Jolfaei , Lars Kulik, Kotagiri Ramamohanarao, and Deepak Puthal

Year: 2020

Description:

Today's vehicles are advancing from stand-alone transportation means to vehicle-to-vehicle, and vehicle-toinfrastructure communications enabled devices which are able to exchange data through the transportation communication infrastructure. As the IoT and data remain intrinsically linked together, the fast-changing mobility landscape of intent-based networking for the Internet of connected vehicles comes with a great risk of data security and privacy violations. This paper considers the privacy issues in the distributed edge computing, in which the data is communicated between a number of vehicles in the IoT layer and potentially untrusted edge controllers at the edge of the network. The sensory data communicated by the vehicles contain sensitive information, such as location and speed, which could violate the users' privacy if they are leaked with no perturbation. Recent studies suggest mechanisms for randomizing the stream of data to ensure individuals' privacy. Although the past works on differential privacy provide a strong privacy guarantee, they are limited to applications where communication parties are trusted and/or there is no correlation between the users or the featured of sensory data. In this paper, we address this gap by proposing a differentially private data streaming system that adds a correlated noise in the vehicle's side (IoT layer) rather than the transportation infrastructure. Also, our system is able to ensure a strong privacy level over time.

2.6 Title: Privacy Techniques for Edge Computing Systems

Author: FANG-YU RAO AND ELISA BERTINO

Year: 2019

Description:

In an edge-enabled data management and computing environment, it is critical to ensure the privacy of the information acquired, processed, and exchanged among the different parties. The problem is complex because of the large scale, mobility, device, and protocol heterogeneity. Also, unlike in conventional environments, communication may be fragmented and portions of the environment can be physically unprotected. To date, there are several privacy-enhancing techniques, such as secure multiparty computation techniques, private information retrieval (PIR), and data sanitization techniques. However, there is not a single technique that works for all possible uses of the data in edge systems. In addition, these techniques are computationally expensive and thus may not be suitable for edge devices. In this paper, we first cover basic privacy building blocks, including differential privacy and homomorphic encryption. We then discuss privacy solutions specific to three different types of data use that are relevant for edge-based applications: data aggregation techniques, point-of-interest (POI) services and traffic information services, and crowdsourcing. These applications have been selected as they provide a broad spectrum of edge computing applications. Throughout this paper, we outline open research directions.

2.7 Title: Secure Data Streaming to Untrusted Road Side Units in Intelligent Transportation System

Author: Alireza Jolfaei, Krishna Kant, Hassan Shafei

Year: 2019

Description:

The paper considers data security issues in vehicleto- infrastructure communications, where vehicles stream data to a road side unit. We assume aggregated data in road side units can be stored or used for data analytics. In this environment, there are issues in regards to the scalability of key management and computation limitations at the edge of the network. To address these issues, we suggest the formation of groups in the vehicle layer, where a group leader is assigned to communicate with group devices and the road side unit. We propose a lightweight permutation mechanism for preserving the confidentiality of sensory data.

2.8 Title: MobileNetV2: Inverted Residuals and Linear Bottlenecks

Author: Mark Sandler Andrew Howard Menglong Zhu Andrey Zhmoginov Liang-Chieh Chen

Year: 2018

Description:

In this paper we describe a new mobile architecture, MobileNetV2, that improves the state of the art performs of mobile models on multiple tasks and bench- marks as well as across a spectrum of different model sizes. We also describe efficient ways of applying these mobile models to object detection in a novel framework we call SSDLite. Additionally, we demonstrate how to build mobile semantic segmentation models through a reduced form of DeepLabv3 which we call Mobile DeepLabv3. Is based on an inverted residual structure where the shortcut connections are between the thin bottle- neck layers. The intermediate expansion layer uses lightweight depthwise convolutions to filter features as a source of non-linearity. Additionally, we find that it is important to remove non-linearities in the narrow layers in order to maintain representational power. We demon- strate that this improves performance and provide an in- tuition that led to this design. Finally, our approach allows decoupling of the in- put/output domains from the expressiveness of the trans- formation, which provides a convenient framework for further analysis. We measure our performance on ImageNet [1] classification, COCO object detection [2], VOC image segmentation [3]. We evaluate the trade-offs between accuracy, and number of operations measured by multiply-adds (MAdd), as well as actual latency, and the number of parameters.

2.9 Title: Real-time Pedestrian Traffic Light Detection

Author: Roni Ash+, Dolev Ofri+, Jonathan Brokman, Idan Friedman and Yair Moshe

Year: 2018

Description:

Crossing a road is a dangerous activity for pedestrians and therefore pedestrian crossings and intersections often include pedestrian-directed traffic lights. These traffic lights may be accompanied by audio signals to aid the visually impaired. In many cases, when such an audio signal is not available, a visually impaired pedestrian cannot cross the road without help. In this paper, we propose a technique that may help visually impaired people by detecting pedestrian traffic lights and their state (walk/don't walk) from video taken with a mobile phone camera. The proposed technique consists of two main modules - an object detector that uses a deep convolutional network, and a decision module. We investigate two variants for object detection (Faster R-CNN combined with a KCF tracker, or Tiny YOLOv2) and compare them. For better robustness, we exploit the fact that abrupt switching from red to green or vice versa is unique to traffic lights. The proposed technique aims to operate on a mobile phone in a client-server architecture. It proves to be fast and accurate with running time of 6 ms per frame on a desktop computer with GeForce GTX 1080 GPU and detection accuracy of more than 99%.

2.10 Title: Traffic Light Detection and Recognition for Self Driving Cars using Deep Learning

Author: Ruturaj Kulkarni, Shruti Dhavalikar, Sonal Bangar

Year: 2018

Description:

Self-driving cars has the potential to revolutionize urban mobility by providing sustainable, safe, convenient and congestion free transportability. This vehicle autonomy as an application of AI has several challenges like infallibly recognizing traffic lights, signs, unclear lane markings, pedestrians, etc. These problems can be overcome by using the technological development in the fields of Deep Learning, Computer Vision due to availability of Graphical Processing Units (GPU) and cloud platform. In this paper, we propose a deep neural network based model for reliable detection and recognition of traffic lights using transfer learning. The method incorporates use of faster region based convolutional network (R-CNN) Inception V2 model in TensorFlow for transfer learning. The model was trained on dataset containing different images of traffic signals in accordance with Indian Traffic Signals which are distinguished in five types of classes. The model accomplishes its objective by detecting the traffic light with its correct class type.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

- In this existing system Surveillance camera only fixed inside of the forest. It will record video every day. If any problem ,railway and transportation officers dispose the elephant body after death
- Before that sensor based demo projects student finished. They connected ultrasonic sensor in train. It will detect object in front of train. But it is not possible to real time.

3.2 DISADVANTAGE

- Surveillance camera working not properly.
- Still Surveillance camera monitoring animals in forest.
- If any problem, railway and transportation officers dispose the elephant body after death.

3.3 PROPOSED SYSTEM

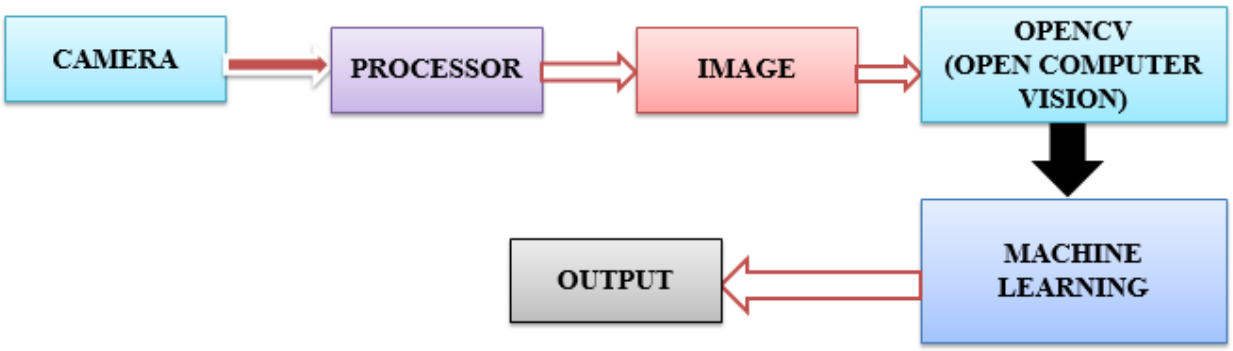
- In this proposed system animals Images trained to our processor .after this implementation most of the animals saved by our project.
- Processor record the video continuously. Animal detect in railway track video send the information to nearby railway station. They will take care everything.
- In this project we using YOLO file for animal detection and recognition.

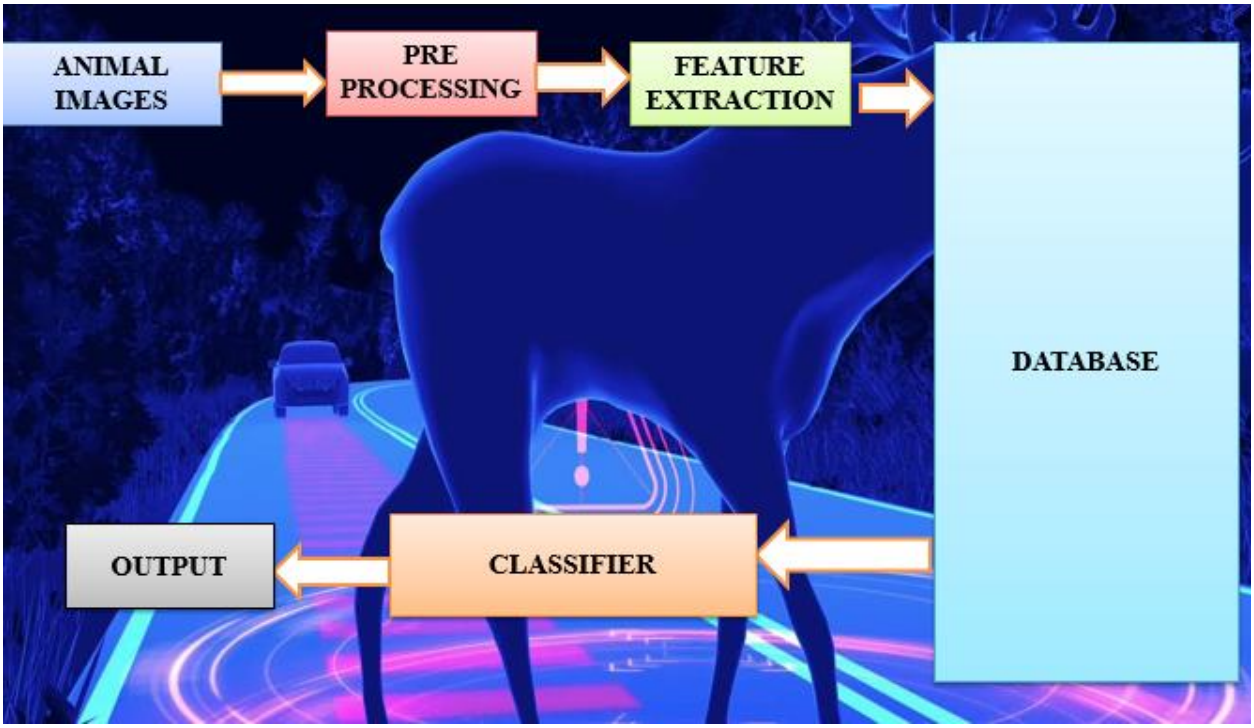
3.4 ADVANTAGE

- YOLO is extremely fast
- Accuracy high
- implement to any processor with its dependencies
- Low cost

3.5 SYSTEM ARCHITECTURE

A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. A system architecture can consist of system components and the sub-systems developed, that will work together to implement the overall system**.**




ANIMAL IMAGES
PRE PROCESSING
FEATURE EXTRACTION
DATABASE
OUTPUT
CLASSIFIER

CHAPTER 4

MODULE IMPLEMENTATION

4.1 MODULE LIST

- CAMERA
- PROCESSOR
- OPENCV
- MACHINE LEARNING

4.2 MODULE DESCRIPTION

4.2.1 CAMERA

A camera is an optical instrument used to capture an image. At their most basic, cameras are sealed boxes (the camera body) with a small hole (the aperture) that allow light in to capture an image on a light-sensitive surface (usually photographic film or a digital sensor). Cameras have various mechanisms to control how the light falls onto the light-sensitive surface. Lenses focus the light entering the camera, the size of the aperture can be widened or narrowed to let more or less light into the camera, and a shutter mechanism determines the amount of time the photo-sensitive surface is exposed to the light.

4.2.2 PROCESSOR

The processor is a chip or a logical circuit that responds and processes the basic instructions to drive a particular computer. The main functions of the processor are fetching, decoding, executing, and write back the operations of an instruction. The processor is also called the brain

of any system which incorporates computers, laptops, smart phones, embedded systems, etc. The ALU (Arithmetic Logic Unit) and CU (Control Unit) are the two parts of the processors. The Arithmetic Logic Unit performs all mathematical operations such as additions, multiplications, subtractions, divisions, etc and the control unit works like traffic police, it manages the command or the operation of the instructions. The processor communicates with the other components also they are input/output devices and memory/storage devices.

GENERAL PURPOSE PROCESSOR

There are five types of general-purpose processors they are, Microcontroller, Microprocessor, Embedded Processor, DSP and Media Processor.

MICROPROCESSOR

The general-purpose processors are represented by the microprocessor in embedded systems. There are different varieties of microprocessors available in the market from different companies. The microprocessor is also a general-purpose processor that consists of a control unit, ALU, a bunch of registers also called scratchpad registers, control registers and status registers. There may be an on-chip memory and some interfaces for communicating with the external world like interrupt lines, other lines for the memory and ports for communicating with the external world. The ports often called the programmable ports

that mean, we can program these ports either to be acting as an input or as an output. The general-purpose processors are shown in the below table.

Basic Components of Processor

- ALU stands for arithmetic logic unit, which help out to execute all arithmetic and logic operations.
- FPU (Floating Point Unit) is also called the “Math coprocessor” that helps to manipulate mathematically calculations.
- Registers store all instructions and data, and it fires operands to ALU and save the output of all operations.
- Cache memory helps to save more time in travelling data from main memory.

Primary CPU Processor Operations are

- **Fetch** – In which, to obtain all instructions from **main memory** unit (**RAM**).
- **Decode** – In this operation, to convert all instructions into understandable ways then other components of CPU are ready to continue further operations, and this entire operations ar performed by decoder.
- **Execute** – Here, to perform all operations and every components of **CPU** which are needed to activate to carry out instructions.

- **Write-Back** – After executing all operations, then its result is moved to write back.

TYPES OF PROCESSOR

Here, we will discuss about different types of CPU (Processors), which are used in computers. If you know how many types of CPU (Processors) are there, then short answer is 5 types of processor.

SINGLE CORE PROCESSOR

Single Core CPUs were used in the traditional **type of computers**. Those CPUs were able to perform one operation at once, so they were not comfortable to multi-tasking system. These CPUs got degrade the entire performance of computer system while running multiple programs at same time duration.

In Single Core CPU, FIFO (First Come First Serve) model is used, it means that couple of operations goes to CPU for processing according to priority base, and left operations get wait until first operation completed.

DUAL CORE PROCESSOR

Dual Core processor contains two processors, and they are linked with each other like as single IC (Integrated circuit). Every processor consist its own local cache and controller, so they are able to perform different difficult operations in quickly than single core **CPU**.

There are some examples which are used as **dual core processors** such as Intel Core Duo, the AMD X2, and the dual-core PowerPC G5.

MULTI CORE PROCESSOR

Multi core processor is designed with using of various processing units' means "Cores" on one chip, and every core of processor is able to perform their all tasks. For example, if you are doing multiple activities at a same time like as using WhatsApp and playing **games** then one core handles WhatsApp activities and other core manage to another works such as game.

QUAD CORE PROCESSOR

Quad core processor is high power CPU, in which four different processors cores are combined into one processor. Every processor is capable to execute and process all instructions own level without taking support to other left processor cores. Quad core processors are able to execute massive instructions at a time without getting waiting pools. Quad core CPU help to enhance the processing power of computer system, but it performance depend on their using computing components.

OCTA CORE PROCESSOR

Octa core processor is designed with using of multiprocessor architecture, and its design produces the higher processing speed. Octa core processor has best ability to perform multi-tasking and to boost up efficiency of your CPU. These **types of processors** are mostly used in your smart phones.

WEB CAM

A webcam is a video camera that feeds or streams its image in real time to or through a computer to computer network. When "captured" by the computer, the video stream may be saved, viewed or sent on to other networks via systems such as the internet, and email as an attachment. When sent to a remote location, the video stream may be saved, viewed or on sent there.

Unlike an IP camera (which connects using Ethernet or Wi-Fi), a webcam is generally connected by a USB cable, or similar cable, or built into computer hardware, such as laptops. The term "webcam" (a clipped compound) may also be used in its original sense of a video camera connected to the Web continuously for an indefinite time, rather than for a particular session, generally supplying a view for anyone who visits its web page over the Internet.

Video calling and videoconferencing

Videophone, Videoconferencing, and Video telephony Webcam can be added to instant messaging, text chat services such as AOL Instant Messenger, and VoIP services such as Skype, one-to-one live video communication over the Internet has now reached millions of mainstream PC users worldwide. Improved video quality has helped webcams encroach on traditional video conferencing systems. New features such as automatic lighting controls, real-time enhancements (retouching, wrinkle smoothing and vertical stretch), automatic face tracking and autofocus, assist users by providing substantial ease-of-use, further increasing the popularity of webcams. Webcam features and performance can vary by program, computer operating system, and also by the computer's processor capabilities. Video calling support has also been added to several popular instant messaging programs.

Video security

Webcams can be used as security cameras. Software is available to allow PC-connected cameras to watch for movement and sound, recording both when they are detected. These recordings can then be saved to the computer, e-mailed, or uploaded to the Internet. In one well-publicised case, a computer e-mailed images of the burglar during the theft of the computer, enabling the owner to give police a clear picture of the burglar's face even after the computer had been stolen. Unauthorized

access of webcams can present significant privacy issues (see "Privacy" section below).

Video clips and stills

Webcams can be used to take video clips and still pictures. Various software tools in wide use can be employed for this, such as Pic Master (for use with Windows operating systems), Photo Booth (Mac), or Cheese (with Unix systems). For a more complete list see Comparison of webcam software.

Input control devices

Special software can use the video stream from a webcam to assist or enhance a user's control of applications and games. Video features, including faces, shapes, models and colors can be observed and tracked to produce a corresponding form of control. For example, the position of a single light source can be tracked and used to emulate a mouse pointer, a head-mounted light would enable hands-free computing and would greatly improve computer accessibility.

This can be applied to games, providing additional control, improved interactivity and im mersiveness. Free Track is a free webcam motion-tracking application for Microsoft Windows that can track a special head-mounted model in up to six degrees of freedom and output data to mouse, keyboard, joystick and Free Track-supported games. By removing the IR filter of the webcam, IR LEDs can be used, which has

the advantage of being invisible to the naked eye, removing a distraction from the user. Track IR is a commercial version of this technology. The Eye Toy for the PlayStation 2, PlayStation Eye for the PlayStation 3, and the Xbox Live Vision camera and Kinect motion sensor for the Xbox 360 and are color digital cameras that have been used as control input devices by some games. Small webcam-based PC games are available as either standalone executables or inside web browser windows using Adobe Flash.

Astro photography

With very-low-light capability, a few specific models of webcams are very popular to photograph the night sky by astronomers and astro photographers. Mostly, these are manual-focus cameras and contain an old CCD array instead of comparatively newer CMOS array. The lenses of the cameras are removed and then these are attached to telescopes to record images, video, still, or both. In newer techniques, videos of very faint objects are taken for a couple of seconds and then all the frames of the video are "stacked" together to obtain a still image of respectable contrast.

4.2.3 OPENCV

Open Source Computer Vision Library' initiated by some enthusiast coders in '1999' to incorporate Image Processing into a wide

variety of coding languages. It has C++, C, and Python interfaces running on Windows, Linux, Android and Mac.

Officially launched in 1999 the Open CV project was initially an Intel Research initiative to advance CPU-intensive applications, part of a series of projects including real-time ray tracing and 3D display walls. The main contributors to the project included a number of optimization experts in Intel Russia, as well as Intel's Performance Library Team. Disseminate vision knowledge by providing a common infrastructure that developers could build on, so that code would be more readily readable and transferable. Advance vision-based commercial applications by making portable, performance-optimized code available for free – with a license that did not require code to be open or free itself. The first alpha version of Open CV was released to the public at the IEEE Conference on Computer Vision and Pattern Recognition in 2000, and five betas were released between 2001 and 2005. The first 1.0 version was released in 2006. A version 1.1 "pre-release" was released in October 2008. The second major release of the Open CV was in October 2009. Open CV 2 includes major changes to the C++ interface, aiming at easier, more type-safe patterns, new functions, and better implementations for existing ones in terms of performance (especially on multi-core systems). Official releases now occur every six months and development is now done by an independent Russian team supported by commercial corporations. In August 2012, support for Open CV was taken over by a non-profit

foundation OpenCV.org, which maintains a developer and user site. On May 2016, Intel signed an agreement to acquire Itseez, a leading developer of Open CV.[10] In July 2020, Open CV announced and began a Kick starter campaign for the Open CV AI Kit, a series of hardware modules and additions to Open CV supporting Spatial AI.

4.2.4 YOLO

You Only Look Once (YOLO) is a network that uses Deep Learning (DL) algorithms for object detection. YOLO performs object detection by classifying certain objects within the image and determining where they are located on it. For example, if you input an image of a herd of sheep into a YOLO network, it will generate an output of a vector of bounding boxes for each individual sheep and classify it as such.

HOW YOLO IMPROVES OVER PREVIOUS OBJECT DETECTION METHODS-

Previous object detection methods like Region-Convolution Neural Networks (R-CNN), including other variations of it like fast R-CNN, performed object detection tasks in a pipeline of multi-step series. R-CNN focuses on a specific region within the image and trains each individual component separately.

This process requires the R-CNN to classify 2000 regions per image, which makes it very time-consuming (47 seconds per individual test

image). Thus it, cannot be implemented in real-time. Additionally, R-CNN uses a fixed selective algorithm, which means no learning process occurs during this stage so the network might generate an inferior region proposal.

This makes object detection networks such as R-CNN harder to optimize and slower compared to YOLO. YOLO is much faster (45 frames per second) and easier to optimize than previous algorithms, as it is based on an algorithm that uses only one neural network to run all components of the task.

To gain a better understanding of what YOLO is, we first have to explore its architecture and algorithm.

YOLO Architecture—Structure Design and Algorithm Operation

A YOLO network consists of three main parts. First, the algorithm, also known as the predictions vector. Second, the network. Third, the loss functions.

The YOLO Algorithm

Once you insert input an image into a YOLO algorithm, it splits the images into an SxS grid that it uses to predict whether the specific bounding box contains the object (or parts of it) and then uses this information to predict a class for the object.

Before we can go into details and explain how the algorithm functions, we need to understand how the algorithm builds and specifies each bounding box. The YOLO algorithm uses four components and additional value to predict an output.

1. The center of a bounding box ($\mathbf{b_x}$ $\mathbf{b_y}$)
2. Width ($\mathbf{b_w}$)
3. Height ($\mathbf{b_h}$)
4. The Class of the object (**c**)

The final predicted value is confidence (p_c). It represents the probability of the existence of an object within the bounding box.The (x,y) coordinates represent the center of the bounding box.Typically, most of the bounding boxes will not contain an object, so we need to use the p_c prediction. We can use a process called non-max suppression to remove unnecessary boxes with low probability to contain objects and those who share big areas with other boxes.

THE NETWORK

A YOLO network is structured like a regular CNN; it contains convolution and max-pooling layers and then two fully connected CNN layers. The Loss Function We only want one of the bounding boxes to be responsible for the object within the image since the YOLO algorithm predicts multiple bounding boxes for each grid cell. To achieve this, we

use the loss function to compute the loss for each true positive. To make the loss function more efficient, we need to select the bounding box with the highest Intersection over Union (IoU) with the ground truth. This method improves predictions by making specialized bounding boxes which improves the predictions for some aspect ratios and sizes.

YOLO V3

YOLO V3 is an incremental upgrade over YOLO V2, which uses another variant of Dark net. This YOLO V3 architecture consists of 53 layers trained on Image net and another 53 tasked with object detection which amounts to 106 layers. While this has dramatically improved the accuracy of the network, it has also reduced the speed from 45 fps to 30 fps.

DARKNET

Dark net is a little awesome open source neural network written in C. Is fast, slim and friendly to use. In particular, if you are interested in a fast and small classifier you should try Tiny Dark net. I'm using it in conjunction with a Bayesian network in order to classify a huge amount of images (using my slow pc) in a fast and reliable way (maybe I'll explain more in another post). You should also use Nightmare the counterpart of Deep Dream. Like I did in a previous post, I wanted to analyze the community of projects surrounding Dark net. In this exploration I'm using the fork action in order to obtain a network.

CHAPTER 5

SOFTWARE REQUIREMENT

H/W SYSTEM CONFIGURATION:-

- processor - Pentium – IV
- RAM - 4 GB (min)
- Hard Disk - 20 GB

S/W SYSTEM CONFIGURATION:-

- Operating System : Windows 7 or 8
- Software : python Idle

SOFTWARE ENVIRONMENT

Python Technology:

Python is an interpreted, high-level, general-purpose programming language. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. **Python** is often described as a "batteries included" language due to its comprehensive standard library.

Python Programing Language:

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and met objects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming.

Python packages with a wide range of functionality, including:

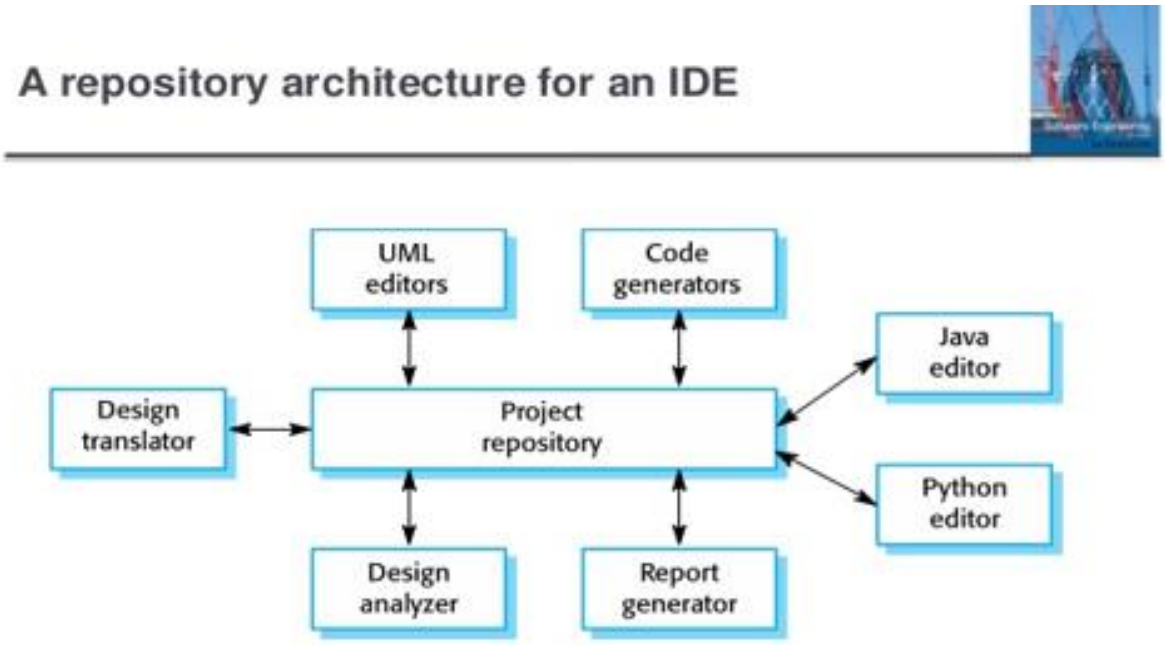
- Easy to Learn and Use
- Expressive Language
- Interpreted Language
- Cross-platform Language
- Free and Open Source

- Object-Oriented Language
- Extensible
- Large Standard Library
- GUI Programming Support
- Integrated

Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution.

Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach.

Python is meant to be an easily readable language. Its formatting is visually uncluttered, and it often uses English keywords where other languages use punctuation. Unlike many other languages, it does not use curly brackets to delimit blocks, and semicolons after statements are optional. It has fewer syntactic exceptions and special cases than C or Pascal.



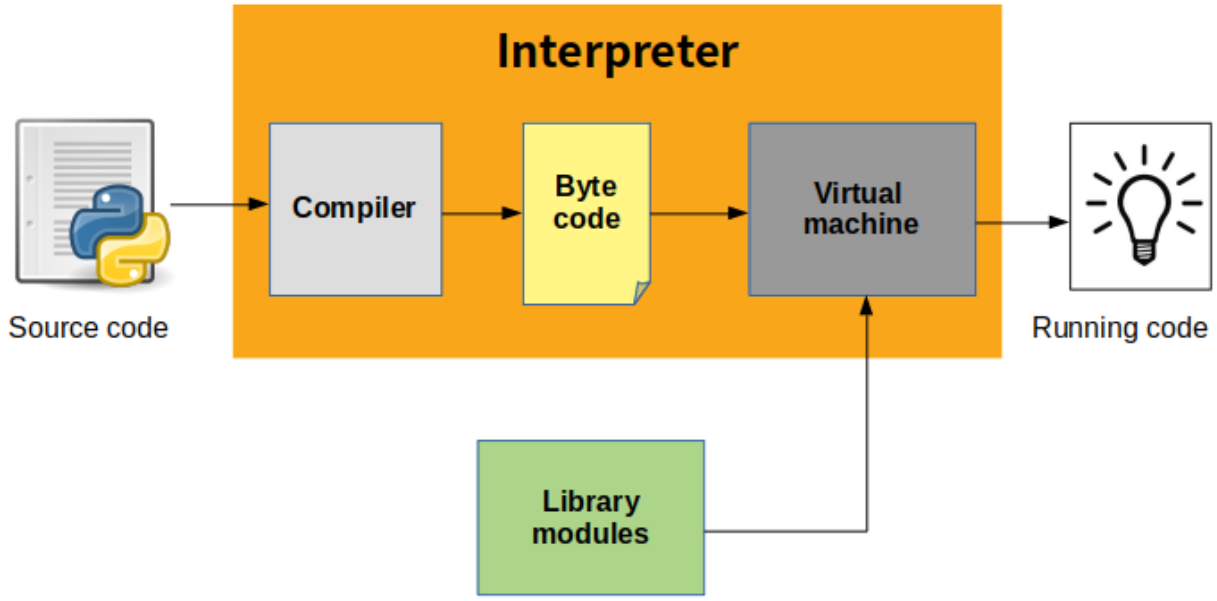
Python strives for a simpler, less-cluttered syntax and grammar while giving developers a choice in their coding methodology. In contrast to Perl's "there is more than one way to do it" motto, Python embraces a "there should be one and preferably only one obvious way to do it" design philosophy. Alex Martelli, a Fellow at the Python Software Foundation and Python book author, writes that "To describe something as 'clever' is not considered a compliment in the Python culture."

Python's developers strive to avoid premature optimization, and reject patches to non-critical parts of the Python reference implementation that would offer marginal increases in speed at the cost of clarity. When speed is important, a Python programmer can move time-critical functions

to extension modules written in languages such as C, or use PyPy, a just-in-time compiler. Python is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter.

An important goal of Python's developers is keeping it fun to use. This is reflected in the language's name a tribute to the British comedy group Monty Python and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (from a famous Monty Python sketch) instead of the standard foo and bar.

Python uses duck typing and has typed objects but untyped variable names. Type constraints are not checked at compile time; rather, operations on an object may fail, signifying that the given object is not of a suitable type. Despite being dynamically typed, Python is strongly typed, forbidding operations that are not well-defined (for example, adding a number to a string) rather than silently attempting to make sense of them.



The Python Platform:

The platform module in Python is used to access the underlying platform's data, such as, hardware, operating system, and interpreter version information. The platform module includes tools to see the platform's hardware, operating system, and interpreter version information where the program is running.

There are four functions for getting information about the current Python interpreter. python_version() and python_version_tuple() return different forms of the interpreter version with major, minor, and patch level components. python_compiler() reports on the compiler used to build the interpreter. And python_build() gives a version string for the build of the interpreter.

Platform() returns string containing a general purpose platform identifier. The function accepts two optional Boolean arguments. If aliased is true, the names in the return value are converted from a formal name to their more common form. When terse is true, returns a minimal value with some parts dropped.

What does python technology do?

Python is quite popular among programmers, but the practice shows that business owners are also Python development believers and for good reason. Software developers love it for its straightforward syntax and reputation as one of the easiest programming languages to learn. Business owners or CTOs appreciate the fact that there's a framework for pretty much anything – from web apps to machine learning.

Moreover, it is not just a language but more a technology platform that has come together through a gigantic collaboration from thousands of individual professional developers forming a huge and peculiar community of aficionados.

So what are the tangible benefits the language brings to those who decided to use it as a core technology? Below you will find just some of those reasons.

PRODUCTIVITY AND SPEED

It is a widespread theory within development circles that developing Python applications is approximately up to 10 times faster than

developing the same application in Java or C/C++. The impressive benefit in terms of time saving can be explained by the clean object-oriented design, enhanced process control capabilities, and strong integration and text processing capacities. Moreover, its own unit testing framework contributes substantially to its speed and productivity.

PYTHON IS POPULAR FOR WEB APPS

Web development shows no signs of slowing down, so technologies for rapid and productive web development still prevail within the market. Along with JavaScript and Ruby, Python, with its most popular web framework Django, has great support for building web apps and is rather popular within the web development community.

OPEN-SOURCE AND FRIENDLY COMMUNITY

As stated on the official website, it is developed under an OSI-approved open source license, making it freely usable and distributable. Additionally, the development is driven by the community, actively participating and organizing conference, meet-ups, hackathons, etc. fostering friendliness and knowledge-sharing.

PYTHON IS QUICK TO LEARN

It is said that the language is relatively simple so you can get pretty quick results without actually wasting too much time on constant improvements and digging into the complex engineering insights of the technology. Even though Python programmers are really in high demand these days, its

friendliness and attractiveness only help to increase number of those eager to master this programming language.

BROAD APPLICATION

It is used for the broadest spectrum of activities and applications for nearly all possible industries. It ranges from simple automation tasks to gaming, web development, and even complex enterprise systems. These are the areas where this technology is still the king with no or little competence:

- Machine learning as it has a plethora of libraries implementing machine learning algorithms.
- Web development as it provides back end for a website or an app.
- Cloud computing as Python is also known to be among one of the most popular cloud-enabled languages even used by Google in numerous enterprise-level software apps.
- Scripting.
- Desktop GUI applications.

Python compiler

The Python compiler package is a tool for analyzing Python source code and generating Python bytecode. The compiler contains libraries to generate an abstract syntax tree from Python source code and to generate Python bytecode from the tree.

The compiler package is a Python source to bytecode translator written in Python. It uses the built-in parser and standard parser module

to generate a concrete syntax tree. This tree is used to generate an abstract syntax tree (AST) and then Python bytecode

The full functionality of the package duplicates the built-in compiler provided with the Python interpreter. It is intended to match its behavior almost exactly. Why implement another compiler that does the same thing? The package is useful for a variety of purposes. It can be modified more easily than the built-in compiler. The AST it generates is useful for analyzing Python source code.

The basic interface

The top-level of the package defines four functions. If you import compiler, you will get these functions and a collection of modules contained in the package.

compiler.parse(buf)

Returns an abstract syntax tree for the Python source code in buf. The function raises Syntax Error if there is an error in the source code. The return value is a compiler.ast. Module instance that contains the tree.

compiler.parseFile(path)

Return an abstract syntax tree for the Python source code in the file specified by path. It is equivalent to parse(open(path).read()).

LIMITATIONS

There are some problems with the error checking of the compiler package. The interpreter detects syntax errors in two distinct phases. One set of errors is detected by the interpreter's parser, the other set by the compiler. The compiler package relies on the interpreter's parser, so it get the first phases of error checking for free. It implements the second phase itself, and that implementation is incomplete. For example, the compiler package does not raise an error if a name appears more than once in an argument list: def f(x, x): ...

A future version of the compiler should fix these problems.

PYTHON ABSTRACT SYNTAX

The compiler.ast module defines an abstract syntax for Python. In the abstract syntax tree, each node represents a syntactic construct. The root of the tree is Module object.

The abstract syntax offers a higher level interface to parsed Python source code. The parser module and the compiler written in C for the Python interpreter use a concrete syntax tree. The concrete syntax is tied closely to the grammar description used for the Python parser. Instead of a single node for a construct, there are often several levels of nested nodes that are introduced by Python's precedence rules.

The abstract syntax tree is created by the compiler. transformer module. The transformer relies on the built-in Python parser to generate a

concrete syntax tree. It generates an abstract syntax tree from the concrete tree.

The transformer module was created by Greg Stein and Bill Tutt for an experimental Python-to-C compiler. The current version contains a number of modifications and improvements, but the basic form of the abstract syntax and of the transformer are due to Stein and Tutt.

AST NODES

The compiler.ast module is generated from a text file that describes each node type and its elements. Each node type is represented as a class that inherits from the abstract base class compiler.ast.Node and defines a set of named attributes for child nodes.

class compiler.ast.Node

The Node instances are created automatically by the parser generator. The recommended interface for specific Node instances is to use the public attributes to access child nodes. A public attribute may be bound to a single node or to a sequence of nodes, depending on the Node type. For example, the bases attribute of the Class node, is bound to a list of base class nodes, and the doc attribute is bound to a single node.

Each Node instance has a lineno attribute which may be None. XXX Not sure what the rules are for which nodes will have a useful lineno.

All Node objects offer the following methods:

getChildren()

Returns a flattened list of the child nodes and objects in the order they occur. Specifically, the order of the nodes is the order in which they appear in the Python grammar. Not all of the children are Node instances. The names of functions and classes, for example, are plain strings.

getChildNodes()

Returns a flattened list of the child nodes in the order they occur. This method is like getChildren(), except that it only returns those children that are Node instances.

The While node has three attributes: test, body, and else_. (If the natural name for an attribute is also a Python reserved word, it can't be used as an attribute name. An underscore is appended to the word to make it a legal identifier, hence else_ instead of else.)

The if statement is more complicated because it can include several tests.

The If node only defines two attributes: tests and else_. The tests attribute is a sequence of test expression, consequent body pairs. There is one pair for each if/elif clause. The first element of the pair is the test expression. The second elements is a Stmt node that contains the code to execute if the test is true.

The getChildren() method of If returns a flat list of child nodes. If there are three if/elif clauses and no else clause, then getChildren() will

return a list of six elements: the first test expression, the first Stmt, the second text expression, etc.

The following table lists each of the Node subclasses defined in compiler.ast and each of the public attributes available on their instances. The values of most of the attributes are themselves Node instances or sequences of instances. When the value is something other than an instance, the type is noted in the comment. The attributes are listed in the order in which they are returned by getChildren() and getChildNodes().

DEVELOPMENT ENVIRONMENTS:

Most Python implementations (including CPython) include a read–eval–print loop (REPL), permitting them to function as a command line interpreter for which the user enters statements sequentially and receives results immediately.

Other shells, including IDLE and IPython, add further abilities such as auto-completion, session state retention and syntax highlighting.

IMPLEMENTATIONS

Reference implementation

CPython is the reference implementation of Python. It is written in C, meeting the C89 standard with several select C99 features. It compiles Python programs into an intermediate bytecode which is then executed by its virtual machine. CPython is distributed with a large standard library

written in a mixture of C and native Python. It is available for many platforms, including Windows and most modern Unix-like systems. Platform portability was one of its earliest priorities.

Other implementations

PyPy is a fast, compliant interpreter of Python 2.7 and 3.5. Its just-in-time compiler brings a significant speed improvement over CPython but several libraries written in C cannot be used with it.

Stackless Python is a significant fork of CPython that implements microthreads; it does not use the C memory stack, thus allowing massively concurrent programs. PyPy also has a stackless version.

MicroPython and CircuitPython are Python 3 variants optimized for microcontrollers. This includes Lego Mindstorms EV3.

RustPython is a Python 3 interpreter written in Rust.

Unsupported implementations

Other just-in-time Python compilers have been developed, but are now unsupported:

Google began a project named Unladen Swallow in 2009, with the aim of speeding up the Python interpreter five-fold by using the LLVM, and of improving its multithreading ability to scale to thousands of cores, while ordinary implementations suffer from the global interpreter lock.

Psyco is a just-in-time specialising compiler that integrates with CPython and transforms bytecode to machine code at runtime. The emitted code is specialized for certain data types and is faster than standard Python code.

In 2005, Nokia released a Python interpreter for the Series 60 mobile phones named PyS60. It includes many of the modules from the CPython implementations and some additional modules to integrate with the Symbian operating system. The project has been kept up-to-date to run on all variants of the S60 platform, and several third-party modules are available. The Nokia N900 also supports Python with GTK widget libraries, enabling programs to be written and run on the target device.

Cross-compilers to other languages

There are several compilers to high-level object languages, with either unrestricted Python, a restricted subset of Python, or a language similar to Python as the source language:

- Jython enables the use of the Java class library from a Python program.
- IronPython follows a similar approach in order to run Python programs on the .NET Common Language Runtime.
- The RPython language can be compiled to C, and is used to build the PyPy interpreter of Python.
- Pyjs compiles Python to JavaScript.

- Cython compiles Python to C and C++.
- Numba uses LLVM to compile Python to machine code.
- Pythran compiles Python to C++.
- Somewhat dated Pyrex (latest release in 2010) and Shed Skin (latest release in 2013) compile to C and C++ respectively.
- Google's Grumpy compiles Python to Go.
- MyHDL compiles Python to VHDL.
- Nuitka compiles Python into C++.

PERFORMANCE

A performance comparison of various Python implementations on a non-numerical (combinatorial) workload was presented at EuroSciPy '13.

API DOCUMENTATION GENERATORS

Python API documentation generators include:

- Sphinx
- Epydoc
- HeaderDoc
- Pydoc

USES

Python has been successfully embedded in many software products as a scripting language, including in finite element method software such as Abaqus, 3D parametric modeler like FreeCAD, 3D animation packages

such as 3ds Max, Blender, Cinema 4D, Lightwave, Houdini, Maya, modo, MotionBuilder, Softimage, the visual effects compositor Nuke, 2D imaging programs like GIMP, Inkscape, Scribus and Paint Shop Pro, and musical notation programs like scorewriter and capella. GNU Debugger uses Python as a pretty printer to show complex structures such as C++ containers. Esri promotes Python as the best choice for writing scripts in ArcGIS. It has also been used in several video games, and has been adopted as first of the three available programming languages in Google App Engine, the other two being Java and Go.

Python is commonly used in artificial intelligence projects with the help of libraries like TensorFlow, Keras and Scikit-learn. As a scripting language with modular architecture, simple syntax and rich text processing tools, Python is often used for natural language processing.

Many operating systems include Python as a standard component. It ships with most Linux distributions, AmigaOS 4, FreeBSD (as a package), NetBSD, OpenBSD (as a package) and macOS and can be used from the command line (terminal). Many Linux distributions use installers written in Python: Ubuntu uses the Ubiquity installer, while Red Hat Linux and Fedora use the Anaconda installer. Gentoo Linux uses Python in its package management system, Portage.

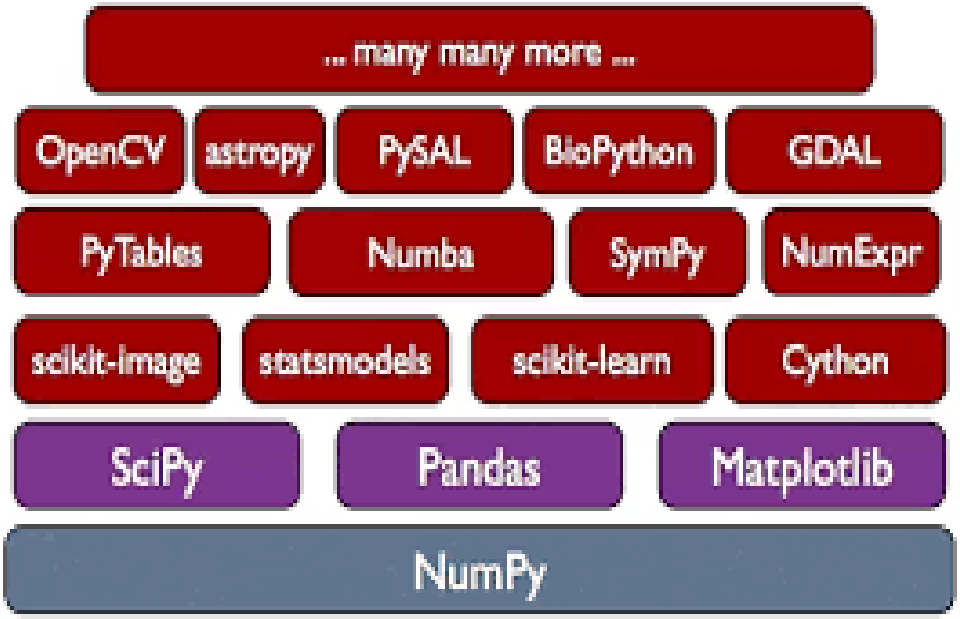
Python is used extensively in the information security industry, including in exploit development.

Most of the Sugar software for the One Laptop per Child XO, now developed at Sugar Labs, is written in Python. The Raspberry Pi single-board computer project has adopted Python as its main user-programming language.

LibreOffice includes Python, and intends to replace Java with Python. Its Python Scripting Provider is a core feature since Version 4.0 from 7 February 2013.

PANDAS

In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals.



Library features

- Data Frame object for data manipulation with integrated indexing.
- Tools for reading and writing data between in-memory data structures and different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and sub setting of large data sets.
- Data structure column insertion and deletion.
- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.

- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
- Time series-functionality: Date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging.
- Provides data filtration.

CSV READER

CSV (Comma Separated Values) is a simple file format used to store tabular data, such as a spreadsheet or database. A CSV file stores tabular data (numbers and text) in plain text. Each line of the file is a data record. Each record consists of one or more fields, separated by commas. The use of the comma as a field separator is the source of the name for this file format. For working CSV files in python, there is an inbuilt module called csv.

CHAPTER 6

SYSTEM DESIGN

6.1 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

6.2 USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

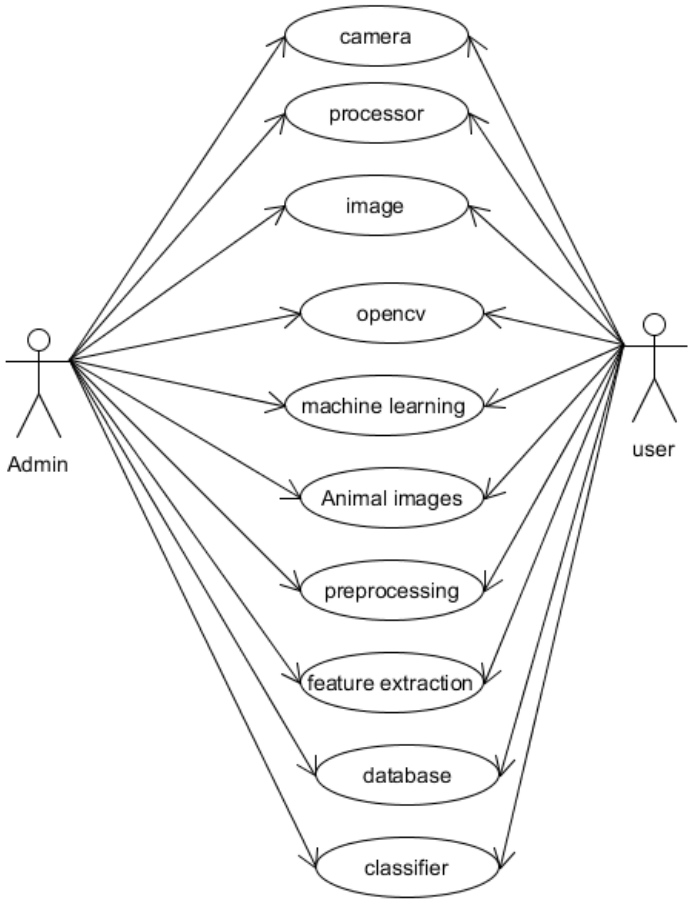


Fig. 6.2 use case diagram

6.3 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

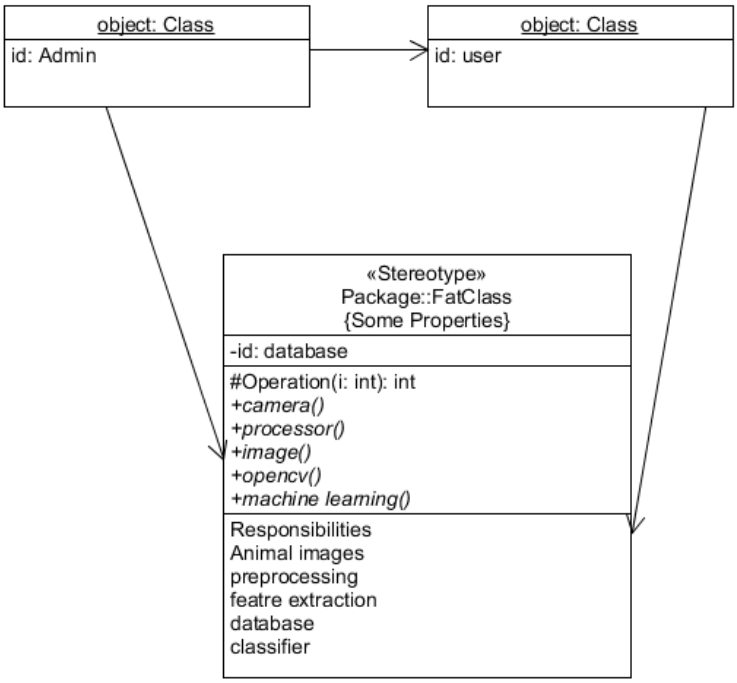


Fig. 6.3 class diagram

6.4 SEQUENCE:

Sequence diagrams, commonly used by developers, model the interactions between objects in a single use case. They illustrate how the different parts of a system interact with each other to carry out a function, and the order in which the interactions occur when a particular use case is executed.

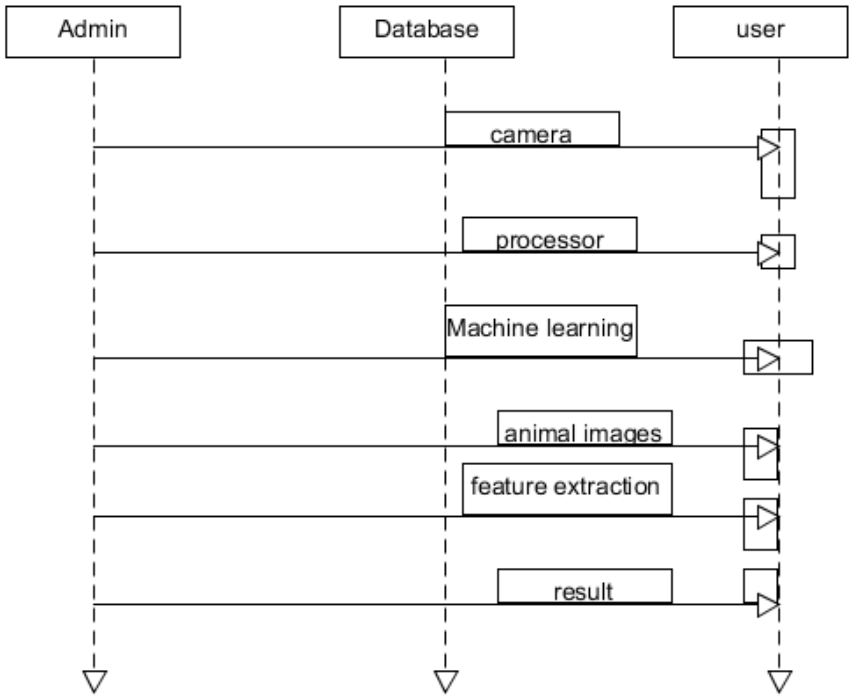


Fig. 6.4 sequence diagram

6.5 DEPLOYMENT:

Component diagrams are used to describe the components and deployment diagrams shows how they are deployed in hardware. UML is mainly designed to focus on the software artifacts of a system. However, these two diagrams are special diagrams used to focus on software and hardware components.

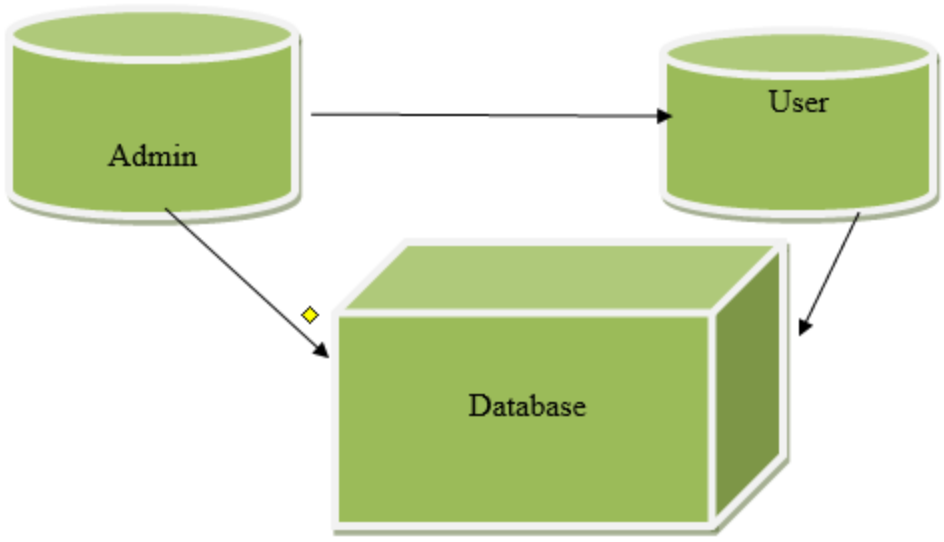


Fig. 6.4deployment diagram

6.4 DATA FLOW DIAGRAM:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

Level 0

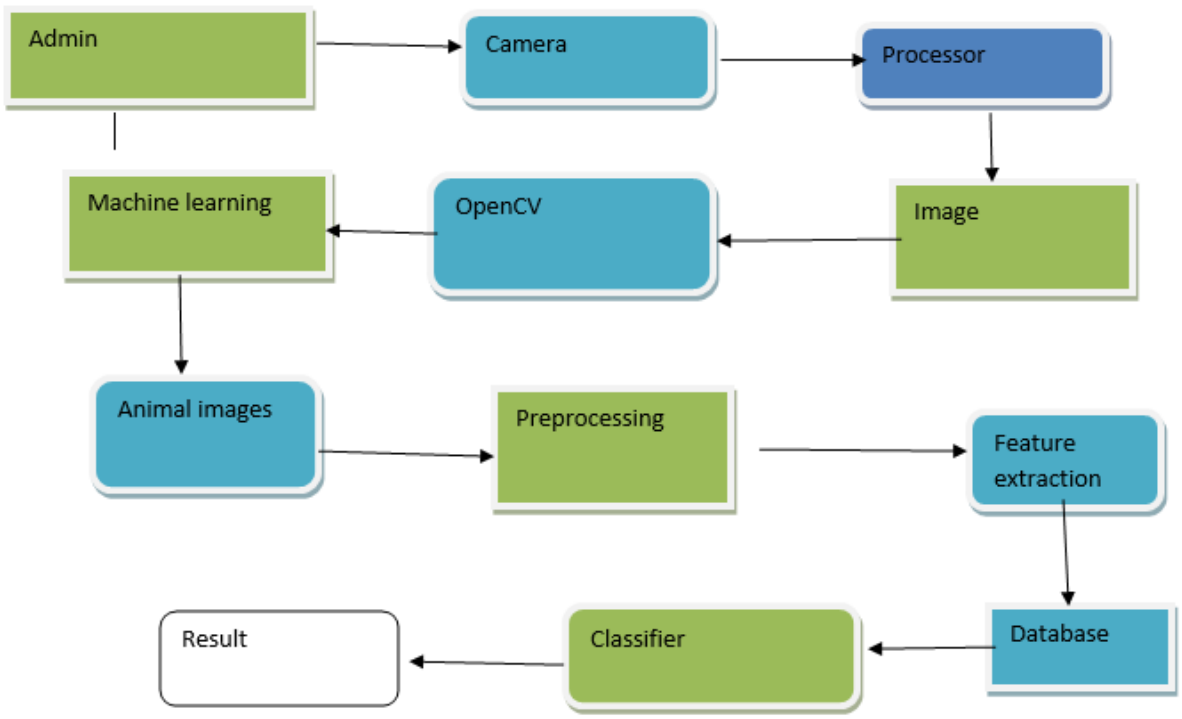


Fig 6.4 Level 0

CHAPTER 7
SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.1 TYPES OF TESTS

7.1.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs

accurately to the documented specifications and contains clearly defined inputs and expected results.

7.1.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

7.1.3 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

7.2 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

7.2.1 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

7.2.2 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

7.3 Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

7.4 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

7.5 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

CHAPTER 8

8.1 CONCLUSION

The project “DARKNET/YOLO BASED ANIMAL SURVEILLANCE AND ACCIDENT PREVENTION USING AI” has been successfully designed and tested. It has been developed by integrating features of all the processor and software used and tested. Presence of every module has been reasoned out and placed carefully thus contributing to the best working of the unit.

8.2 FUTURE WORK

The possible future extension of the system include the creation of an autonomous notification subsystem that provides direct connection with nearby vehicles and trains that are heading to railway crossing.

8.3 REFERENCE

[1] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” nature, vol. 521, no. 7553, pp. 436–444, 2015.

[2] I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. MIT press, 2016.

[3] S. Vasavi, N. K. Priyadarshini, and K. H. Vardhan, “Invariant feature based darknet architecture for moving object classification,” IEEE Sensors Journal, pp. 1–1, 2020.

[4] J. L. C. Choy, J. Wu, C. Long, and Y. Lin, “Ubiquitous and low power vehicles speed monitoring for intelligent transport systems,” IEEE Sensors Journal, vol. 20, no. 11, pp. 5656–5665, 2020.

[5] J. Singh, A. Desai, F. Acker, S. Ding, S. Prakasamul, A. Rachide, and P. Nelson-Furnell, “Cooperative intelligent transport systems to improve safety at level crossing,” in The 12th Global Level Crossing and Trespass Symp., London, 2012.

[6] G. S. Larue, A. Rakotonirainy, N. L. Haworth, and M. Darvell, “Assessing driver acceptance of intelligent transport systems in the context of railway level crossings,” Transportation Research Part F: Traffic Psychology and Behaviour, vol. 30, pp. 1–13, 2015.

[7] J. Choi, V. Marojevic, A. Sharma, B. Zewede, R. Nealy, C. R. Anderson, J. Withers, and C. B. Dietrich, “Measurement and configuration of dsrc radios for vehicle-to-train (v2t) safety-critical communications,” IEEE Wireless Communications Letters, vol. 7, no. 3, pp. 428–431, 2017.

[8] D. Li, L. Deng, Z. Cai, B. Franks, and X. Yao, “Intelligent transportation system in macao based on deep self-coding learning,” IEEE Transactions on Industrial Informatics, vol. 14, no. 7, pp. 3253–3260, 2018.

[9] Z. Zhao, W. Chen, X. Wu, P. C. Chen, and J. Liu, “Lstm network: a deep learning approach for short-term traffic forecast,” IET Intelligent Transport Systems, vol. 11, no. 2, pp. 68–75, 2017.

[10] D. Zang, Z. Chai, J. Zhang, D. Zhang, and J. Cheng, “Vehicle license plate recognition using visual attention model and deep learning,” Journal of Electronic Imaging, vol. 24, no. 3, p. 033001, 2015.

[11] H.-H. Kim, J.-K. Park, J.-H. Oh, and D.-J. Kang, “Multi-task convolutional neural network system for license plate recognition,” International Journal of Control, Automation and Systems, vol. 15, no. 6, pp. 2942– 2949, 2017.

[12] F. C. Soon, H. Y. Khaw, J. H. Chuah, and J. Kanesan, “Hyper-parameters optimisation of deep cnn architecture for vehicle logo recognition,” IET Intelligent Transport Systems, vol. 12, no. 8, pp. 939–946, 2018.

[13] V. D. Nguyen, H. Van Nguyen, D. T. Tran, S. J. Lee, and J. W. Jeon, “Learning framework for robust obstacle detection, recognition, and tracking,” IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 6, pp. 1633–1646, 2016.

[14] G. Prabhakar, B. Kailath, S. Natarajan, and R. Kumar, “Obstacle detection and classification using deep learning for tracking in high-speed autonomous driving,” in 2017 IEEE Region 10 Symposium (TENSYMP). IEEE, 2017, pp. 1–6.

[15] C. Hu, Y. Wang, G. Yu, Z. Wang, A. Lei, and Z. Hu, “Embedding cnnbased fast obstacles detection for autonomous vehicles,” SAE Technical Paper, Tech. Rep., 2018.

IEEE

SRM TRP ENGINEERING COLLEGE

CERTIFICATE OF PARTICIPATION

This is to certify that **R.Ranjith** of **M.A.M School Of Engineering** has Presented a Paper titled **Darknet/ Yolo Based Animal Surveillance And Accident Prevention Using Ai** presented a Paper titled in the International Conference on Artificial Intelligence for IoT and Sustainable Electrical Networks (ICAIISEN'21) conducted by the SRM TRP Engineering College in association with IEEE SB, IIC, IETE and CSI on 29.04.2021.

Dr.B.RAMASUBRAMANIAN	Dr.P.SUDHAKARAN	Dr.P.ELANGOVAN	Dr.B.GANESH BABU
VICE PRINCIPAL &HOD/ECE	HOD/CSE	HOD/EEE	PRINCIPAL

IEEE

SRM TRP ENGINEERING COLLEGE

CERTIFICATE OF PARTICIPATION

This is to certify that **S.Sarma** of **M.A.M School Of Engineering** has Presented a Paper titled **Darknet/ Yolo Based Animal Surveillance And Accident Prevention Using Ai** presented a Paper titled in the International Conference on Artificial Intelligence for IoT and Sustainable Electrical Networks (ICAIISEN'21) conducted by the SRM TRP Engineering College in association with IEEE SB, IIC, IETE and CSI on 29.04.2021.

Dr.B.RAMASUBRAMANIAN	Dr.P.SUDHAKARAN	Dr.P.ELANGOVAN	Dr.B.GANESH BABU
VICE PRINCIPAL &HOD/ECE	HOD/CSE	HOD/EEE	PRINCIPAL

IEEE

SRM TRP ENGINEERING COLLEGE

CERTIFICATE OF PARTICIPATION

This is to certify that **M.Sharmila** of **M.A.M School Of Engineering** has Presented a Paper titled **Darknet/ Yolo Based Animal Surveillance And Accident Prevention Using Ai** presented a Paper titled in the International Conference on Artificial Intelligence for IoT and Sustainable Electrical Networks (ICAIISEN'21) conducted by the SRM TRP Engineering College in association with IEEE SB, IIC, IETE and CSI on 29.04.2021.

Dr.B.RAMASUBRAMANIAN	Dr.P.SUDHAKARAN	Dr.P.ELANGOVAN	Dr.B.GANESH BABU
VICE PRINCIPAL &HOD/ECE	HOD/CSE	HOD/EEE	PRINCIPAL