JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Security issues and defensive approaches in deep learning framework

Dr. Bhaskara Rao B
Associate Professor
Dept of Computer Science and Engineering
GITAM University
Visakhapatnam, India

Salma Sulthana Shaik
M.Tech Scholar
Dept of Computer Science and Engineering
GITAM University
Visakhapatnam, India

Abstract— ***Deep learning frameworks promote the development of artificial intelligence and demonstrate considerable potential in numerous applications. However, the security issues of deep learning frameworks are among the main risks preventing its wide application of it. Attacks on deep learning frameworks by malicious internal or external attackers would exert substantial effects on society and life. We start with a description of the framework of deep learning algorithms and a detailed analysis of their attacks and vulnerabilities in them. We propose a highly comprehensive classification approach for security issues and defensive approaches in deep learning frameworks and connect different attacks to corresponding defensive approaches. Moreover, we analyze a case of the physical-world use of deep learning security issues. In addition, we discuss future directions and open issues in deep learning frameworks. We hope that our research will inspire future developments and draw attention from academic and industrial domains to the security of deep learning frameworks***

I. Introduction

1.1 MOTIVATION

We propose a highly comprehensive classification approach for security issues and defensive approaches in deep learning frameworks and connect different attacks to corresponding defensive approaches. Moreover, we analyze a case of the physical-world use of deep learning security issues.

1.2 PROBLEM DEFINITION

Although deep learning can bring certain conveniences, it is prone to numerous vulnerabilities. recent research has found that deep learning is vulnerable to well-designed adversarial samples, which can easily fool a well-behaved deep learning model.

1.3 OBJECTIVE OF THE PROJECT

Deep learning security problems can occur when an attacker either uses the DNN implementation principle to reversely generate adversarial samples or exploits the vulnerabilities of third-party libraries that are dependent on the underlying DNN.

II. LITERATURE SURVEY

2.1 Geospatial data to images: A deep-learning framework for traffic forecasting

AUTHORS: W. W. Jiang and L. Zhang,

ABSTRACT: Traffic forecasting has been an active research field in recent decades, and with the development of deep-learning technologies, researchers are trying to utilize deep learning to achieve tremendous improvements in traffic forecasting, as has been seen in other research areas, such as speech recognition and image classification. In this study, we summarize recent works in which deep-learning methods were applied for geospatial data-based traffic forecasting problems. Based on the insights from previous works, we further propose a deep-learning framework, which transforms geospatial data into images and then utilizes state-of-the-art deep-learning methodologies such as Convolutional Neural networks (CNN) and residual networks. To demonstrate the simplicity and effectiveness of our framework, we present a formulation of the New York taxi pick-up/drop-off forecasting problem and show that our framework significantly outperforms traditional methods, including Historical Average (HA) and AutoRegressive Integrated Moving Average (ARIMA).

2.2 Improved Dota2 lineup recommendation model based on a bidirectional LSTM

AUTHORS: L. Zhang, C. B. Xu, Y. H. Gao, Y. Han, X. J. Du, and Z. H. Tian,

ABSTRACT: In recent years, e-sports has rapidly developed, and the industry has produced large amounts of data with specifications, and these data are easy to be obtained. Due to the above characteristics, data mining and deep learning methods can be used to guide players and develop appropriate strategies to win games. As one of the world's most famous e-sports events, Dota2 has a large audience base and a good game system. A victory in a game is often associated with a hero's match, and players are often unable to pick the best lineup to compete. To solve this problem, in this paper, we present an improved bidirectional Long Short-Term Memory (LSTM) neural network model for Dota2 lineup recommendations. The model uses the Continuous Bag Of Words (CBOW) model in the Word2vec model to generate hero vectors. The CBOW model can predict the context of a word in a sentence. Accordingly, a word is transformed into a hero, a sentence into a lineup, and a word vector into a hero vector, the model applied in this article recommends the last hero according to the first four heroes selected first, thereby solving a series of recommendation problems.

2.3 Adversarial Examples: Attacks and Defenses for Deep Learning

AUTHORS: X. Y. Yuan, P. He, Q. L. Zhu, and X. L. Li

ABSTRACT: With rapid progress and significant successes in a wide spectrum of applications, deep learning is being applied in many safety-critical environments. However, deep neural networks have been recently found vulnerable to well-designed input samples, called adversarial examples. Adversarial examples are imperceptible to humans but can easily fool deep neural networks in the testing/deploying stage. The vulnerability to adversarial examples becomes one of the major risks for applying deep neural networks in safety-critical environments. Therefore, attacks and defenses on adversarial examples draw great attention. In this paper, we review recent findings on adversarial examples for deep neural networks, summarize the methods for generating adversarial examples, and propose a taxonomy of these methods. Under the taxonomy, applications for adversarial examples are investigated. We further elaborate on countermeasures for adversarial examples and explore the challenges and the potential solutions.

2.4 A MEMORY-RELATED VULNERABILITY DETECTION APPROACH BASED ON VULNERABILITY FEATURES

AUTHORS: J. C. Hu, J. F. Chen, L. Zhang, Y. S. Liu, Q. H. Bao, H. Ackah-Arthur, and C. Zhang

ABSTRACT: Developing secure software systems is a major challenge in the software industry due to errors or weaknesses that bring vulnerabilities to the software system. To address this challenge, researchers often use the source code features of vulnerabilities to improve vulnerability detection. Notwithstanding the success achieved by these techniques, the existing studies mainly focus on the conceptual description without an accurate definition of vulnerability features. In this study, we introduce a novel and efficient Memory-Related Vulnerability Detection Approach using Vulnerability Features (MRVDAVF). Our framework uses three distinct strategies to improve vulnerability detection. In the first stage, we introduce an improved Control Flow Graph (CFG) and Pointer-related Control Flow Graph (PCFG) to describe the features of some common vulnerabilities, including memory leak, double-free, and use-after-free. Afterward, two algorithms, namely the Vulnerability Judging algorithm based on Vulnerability Feature (VJVF) and Feature Judging (FJ) algorithm, are employed to detect memory-related vulnerabilities. Finally, the proposed model is validated using three test cases obtained from Juliet Test Suite. The experimental results show that the proposed approach is feasible and effective.

2.5 Intriguing properties of neural networks

AUTHORS: C. Szegedy, W. Zaremba, I. Sutskever I, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus,

ABSTRACT: DEEP NEURAL NETWORKS ARE HIGHLY EXPRESSIVE MODELS THAT HAVE RECENTLY ACHIEVED STATE-OF-THE-ART PERFORMANCE ON SPEECH AND VISUAL RECOGNITION TASKS. WHILE THEIR EXPRESSIVENESS IS THE REASON THEY SUCCEED, IT ALSO CAUSES THEM TO LEARN UNINTERPRETABLE SOLUTIONS THAT COULD HAVE COUNTERINTUITIVE PROPERTIES. IN THIS PAPER, WE REPORT TWO SUCH PROPERTIES. FIRST, WE FIND THAT THERE IS NO DISTINCTION BETWEEN INDIVIDUAL HIGH-LEVEL UNITS AND RANDOM LINEAR COMBINATIONS OF HIGH-LEVEL UNITS, ACCORDING TO VARIOUS METHODS OF UNIT ANALYSIS. IT SUGGESTS THAT IT IS THE SPACE, RATHER THAN THE INDIVIDUAL UNITS, THAT CONTAINS THE SEMANTIC INFORMATION IN THE HIGH LAYERS OF NEURAL NETWORKS. SECOND, WE FIND THAT DEEP NEURAL NETWORKS LEARN INPUT-OUTPUT MAPPINGS THAT ARE FAIRLY DISCONTINUOUS TO A SIGNIFICANT EXTENT. SPECIFICALLY, WE FIND THAT WE CAN CAUSE THE NETWORK TO MISCLASSIFY AN IMAGE BY APPLYING A CERTAIN IMPERCEPTIBLE PERTURBATION, WHICH IS FOUND BY MAXIMIZING THE NETWORK'S PREDICTION ERROR. IN ADDITION, THE SPECIFIC NATURE OF THESE PERTURBATIONS IS NOT A RANDOM ARTIFACT OF LEARNING: THE SAME PERTURBATION CAN CAUSE A DIFFERENT NETWORK, THAT WAS TRAINED ON A DIFFERENT SUBSET OF THE DATASET, TO MISCLASSIFY THE SAME INPUT.

III. SYSTEM ANALYSIS

3.1 EXISTING SYSTEM:

Although deep learning has several advantages, it is vulnerable to several flaws. Deep learning is vulnerable to well-designed adversarial samples, according to recent research, which can readily trick a well-behaved deep learning model. Szegedy et al. tricked the most advanced Deep Neural Network (DNN) with high probability by generating minor perturbations in an image categorization issue. As a result, DNN-misclassified samples are referred to as adversarial samples.

Deep learning frameworks' security vulnerabilities, on the other hand, are one of the key hazards impeding their widespread use. Malicious internal or external attackers attacking deep learning frameworks would have a significant impact on society and life.

3.1.1 DISADVANTAGES OF THE EXISTING SYSTEM:

- ❖ Attacks against deep learning frameworks by hostile internal or external attackers would have significant societal and life-changing consequences.

3.2 PROPOSED SYSTEM:

- ❖ We start with a description of the framework of deep learning algorithms and a detailed analysis of their attacks and vulnerabilities in them. We propose a highly comprehensive classification approach for security issues and defensive approaches in deep learning frameworks and connect different attacks to corresponding defensive approaches.

3.2.1 ADVANTAGES OF THE PROPOSED SYSTEM:

We analyze a case of the physical-world use of deep learning security issues.

3.3 SYSTEM REQUIREMENTS:

SOFTWARE REQUIREMENTS

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints, and user documentation.

The appropriation of requirements and implementation constraints gives the general overview of the project in regards to what the areas of strength and deficit are and how to tackle them.

- **Python ideal 3.7 version (or)**
- **Anaconda 3.7 (or)**
- **Jupiter (or)**
- **Google collab**

HARDWARE REQUIREMENTS

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

- **Operating system: Windows, Linux**
- **Processor : minimum intel i3**
- **Ram : minimum 4 gb**
- **Hard disk : minimum 250gb**

3.4 FUNCTIONAL REQUIREMENTS

1. Data Collection

2. Data Preprocessing

3. Training And Testing

4. Modeling

5. Predicting

3.5 NON FUNCTIONAL REQUIREMENTS

NON-FUNCTIONAL REQUIREMENT (NFR) is a software system's quality attribute. They assess the software system based on its responsiveness, usability, security, portability, and other non-functional criteria that are crucial to its success. "How quickly does the website load?" is an example of a nonfunctional demand. Non-functional requirements that aren't met can lead to systems that don't meet user needs. Non-functional Requirements allow you to place constraints or limitations on the system's architecture across several agile backlogs. When there are more than 10000 simultaneous users, for example, the site should load in 3 seconds. Non-functional requirements must be described just as carefully as functional requirements.

Usability requirement

- Serviceability requirement
- Manageability requirement
- Recoverability requirement
- Security requirement
- Data Integrity requirement
- Capacity requirement
- Availability requirement
- Scalability requirement
- Interoperability requirement
- Reliability requirement
- Maintainability requirement
- Regulatory requirement
- Environmental requirement

3.6 SYSTEM STUDY

FEASIBILITY STUDY

In this phase, the project's feasibility is assessed, and a business proposal is presented, along with a very general project design and some cost estimates. A feasibility study of the proposed system is to be carried out during system analysis. This is to ensure that the planned system will not cause the organization any problems. A basic understanding of the system's primary requirements is required for feasibility analysis. Three key considerations involved in the feasibility analysis are

- ♦ ECONOMICAL FEASIBILITY
- ♦ TECHNICAL FEASIBILITY
- ♦ SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of funds that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system is well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

IV. TECHNICAL FEASIBILITY

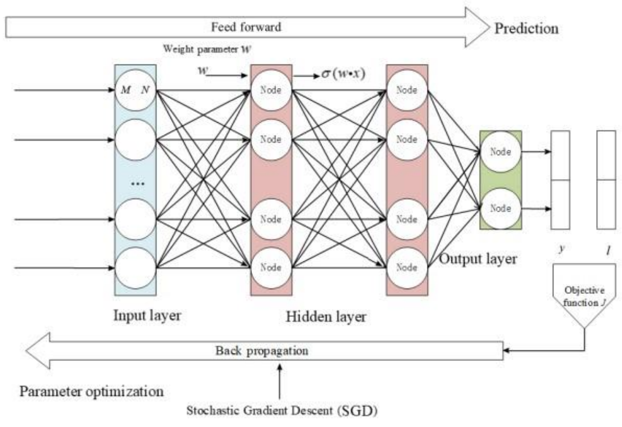
This research is being carried out to determine the system's economic impact on the organization. The amount of money the corporation has to invest in the system's research and development is limited. It is necessary to justify the spending. As a result, the developed system came in under budget, which was made possible by the fact that the majority of the technologies used were freely available. The customized products were the only ones that needed to be acquired.

SOCIAL FEASIBILITY

The aspect of the study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

V. SYSTEM DESIGN

5.1 SYSTEM ARCHITECTURE:



5.2 DATA FLOW DIAGRAM:

1. A bubble chart is another name for a DFD. It is a basic graphical formalism that can be used to depict a system in terms of the data it receives, the processing it performs on that data, and the data it generates as output.
2. One of the most essential modeling tools is the data flow diagram (DFD). It's used to represent the system's many components. The system process, the data used by the process, an external entity that interacts with the system, and the information flows in the system are all examples of these components.

3. DFD depicts how information flows through the system and is transformed through a sequence of transformations. It's a graphical representation of data flow and the transformations that occur when data goes from input to output.
4. DFD is sometimes referred to as a bubble chart. At any level of abstraction, a DFD can be used to depict a system. DFD can be divided into levels, each representing a different level of information flow and functional detail.

5.3 UML DIAGRAMS

Unified Modeling Language (UML) is an acronym for "Unified Modeling Language." In the realm of object-oriented software engineering, UML is a standardized general-purpose modeling language. The Object Management Group is in charge of the standard, and it was produced by them.
The goal is for UML to become a standard language for developing object-oriented software models. UML has two primary components in its current form: a meta-model and a notation. Some type of method or process may be added to or related to, UML in the future.
The Unified Modeling Language (UML) is a standard language for describing, visualizing, constructing, and documenting software system artifacts, as well as business modeling and other non-software 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.

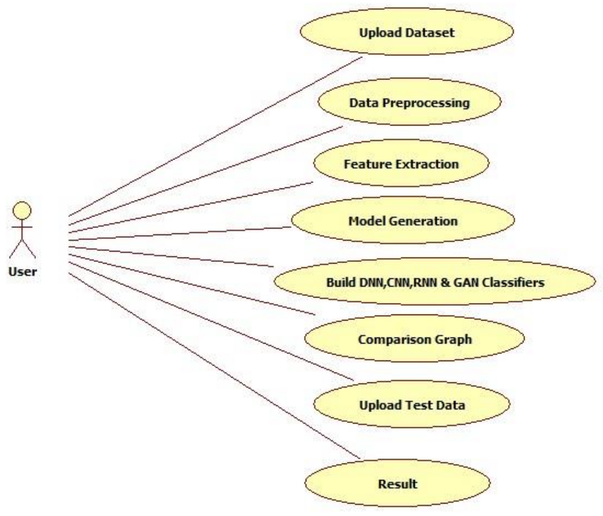
GOALS:

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 extensibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development processes.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of the OO tools market.
6. Support higher-level development concepts such as collaborations, frameworks, patterns, and components.
7. Integrate best practices.

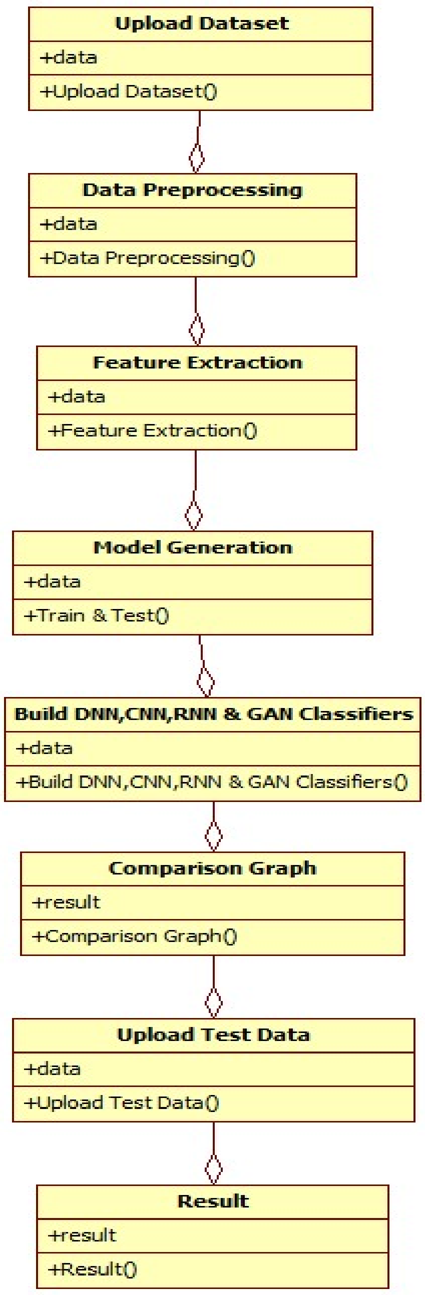
Use case diagram:

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. The roles of the actors in the system can be depicted.



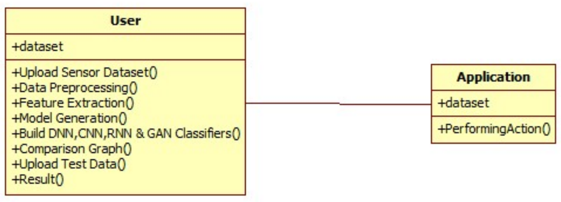
Class diagram:

The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an "is-a" or "has-a" relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed "methods" of the class. Apart from this, each class may have certain "attributes" that uniquely identify the class.



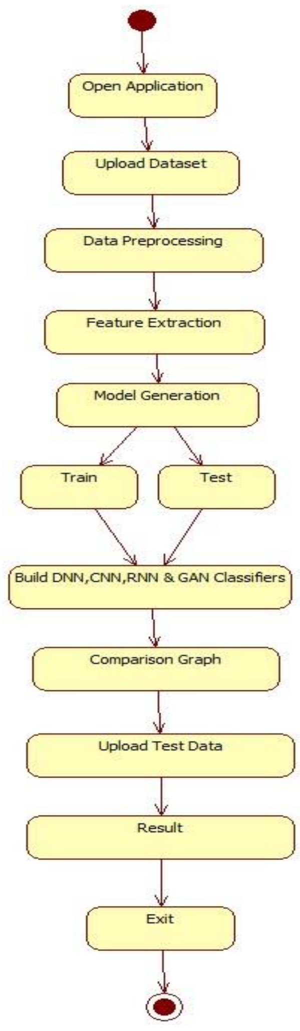
Object diagram:

The object diagram is a special kind of class diagram. An object is an instance of a class. This essentially means that an object represents the state of a class at a given point of time while the system is running. The object diagram captures the state of different classes in the system and their relationships or associations at a given point in time.



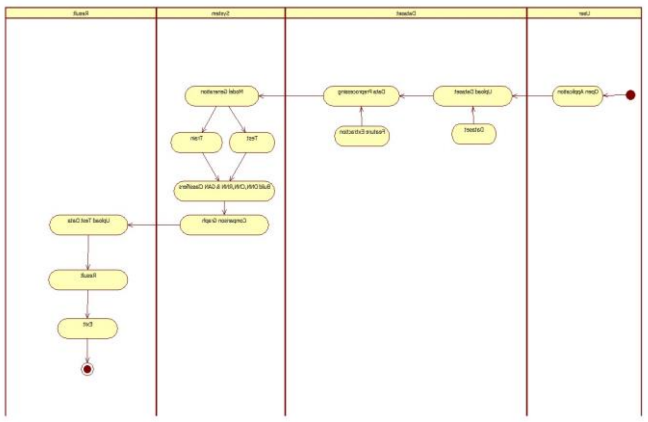
State diagram:

A state diagram, as the name suggests, represents the different states that objects in the system undergo during their life cycle. Objects in the system change state in response to events. In addition to this, a state diagram also captures the transition of the object's state from an initial state to a final state in response to events affecting the system.



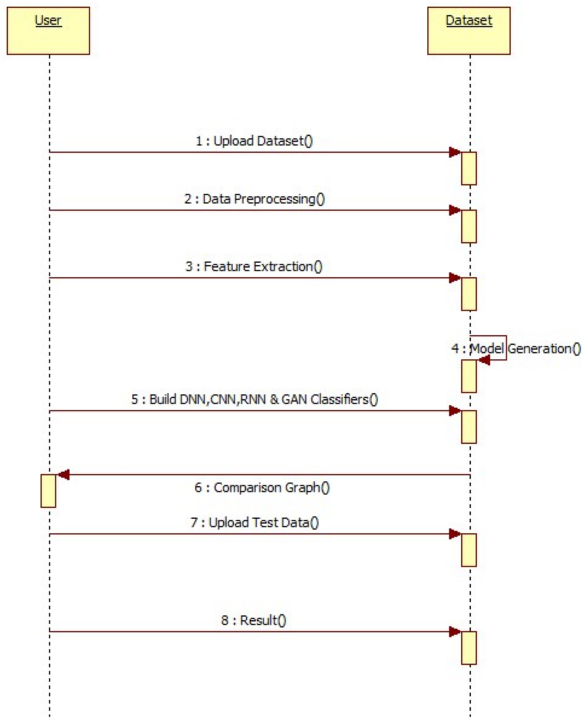
Activity diagram:

The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and final states, and guard conditions.



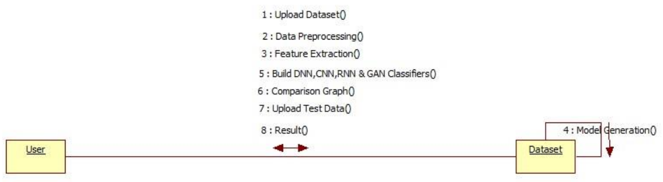
Sequence diagram:

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing "messages".



Collaboration diagram:

A collaboration diagram groups together the interactions between different objects. The interactions are listed as numbered interactions that help to trace the sequence of the interactions. The collaboration diagram helps to identify all the possible interactions that each object has with other objects.



5.4 IMPLEMENTATION:

MODULES:

1. Data Collection

The composition of the dataset**.** understand the relationship among different features. A plot of the core features and the entire dataset. The dataset is further split into 2/3 for training and 1/3 for testing the algorithms. Furthermore, to obtain a representative sample, each class in the full dataset is represented in about the right proportion in both the training and testing datasets. The various proportions of the training and testing datasets used in the paper.

2. Data Preprocessing

The data which was collected might contain missing values that may lead to inconsistency. To gain better results data need to be preprocessed to improve the efficiency o the algorithm. The outliers have to be removed and also variable conversion need to be done. To overcome these issues we use the map function.

3. Model Selection

Machine learning is about predicting and recognizing patterns and generating suitable results after understanding them. ML algorithms study patterns in data and learn from them. An ML model will learn and improve on each attempt. To gauge the effectiveness of a model, it's vital to split the data into training and test sets first. So before training our models, we split the data into the Training set which was 70% of the whole dataset, and Test set which was the remaining 30%. Then it was important to implement a selection of performance metrics to the predictions made by our model.

4. Predict the results

The designed system is tested with a test set and the performance is assured. Evolution analysis refers to the description and model of regularities or trends for objects whose behavior changes over time.

5. SOFTWARE ENVIRONMENT

WHAT IS PYTHON:-

Below are some facts about Python.

Python is currently the most widely used multi-purpose, high-level programming language.

Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.

Programmers have to type relatively less and the indentation requirement of the language makes them readable all the time.

Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is its huge collection of standard libraries which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt, etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

Advantages of Python:-

Let's see how Python dominates over other languages.

1. Extensive Libraries

Python downloads with an extensive library and it *contains code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more.* So, we don't have to write the complete code for that manually.

2. Extensible

As we have seen earlier, Python can be **extended to other languages**. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add **scripting capabilities** to our code in the other language.

4. Improved Productivity

The language's simplicity and extensive libraries render programmers **more productive** than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

6. Simple and Easy

When working with Java, you may have to create a class to print **'Hello World'**. But in Python, just a print statement will do. It is also quite **easy to learn, understand,** and **code.** This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and **indentation is mandatory.** This further aids the readability of the code.

8. Object-Oriented

This language supports both the **procedural and object-oriented** programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the **encapsulation of data** and functions into one.

9. Free and Open-Source

Like we said earlier, Python is **freely available.** But not only can you **download Python** for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to **code only once**, and you can run it anywhere. This is called **Write Once Run Anywhere (WORA)**. However, you need to be careful enough not to include any system-dependent features.

11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, **debugging is easier** than in compiled languages.
Any doubts till now about the advantages of Python? Mention in the comment section.

Advantages of Python Over Other Languages

1. Less Coding

Almost all of the tasks done in Python require less coding than the same task done in other languages. Python also has awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac, or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and **machine learning**, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

Disadvantages of Python

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in **slow execution**. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is rarely seen on the **client side**. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called **Carbonnelle**.
The reason it is not so famous despite the existence of Brython is that it isn't that secure.

3. Design Restrictions

As you know, Python is **dynamically-typed**. This means that you don't need to declare the type of variable while writing the code. It uses **duck-typing**. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can **raise run-time errors**.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like **JDBC (Java DataBase Connectivity)** and **ODBC (Open DataBase Connectivity)**, Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

No, we're not kidding. Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

History of Python: -

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde &Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time on a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners[1], Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum Voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it."Later on, in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

What is Machine Learning: -

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of *building models of data*.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models *tunable parameters* that can be adapted to observed data; in this way, the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

Categories Of Machine Leaning:-

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

Supervised learning involves somehow modeling the relationship between measured features of data and some labels associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into *classification* tasks and *regression* tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

Unsupervised learning involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself." These models include tasks such as *clustering* and *dimensionality reduction.* Clustering algorithms identify distinct groups of data, while dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and hasn't surpassed human intelligence in many aspects. Then the question is that what is the need to make a machine learn? The most suitable reason for doing this is, "to make decisions, based on data, with efficiency and scale".

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning, and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programing logic, in problems that cannot be programmed inherently. The fact is that we can't do without human intelligence, but another aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

Challenges in Machines Learning:-

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as a whole still has a long way to go. The reason behind this is that ML has not been able to overcome several challenges. The challenges that ML is facing currently are −

Quality of data − Having good-quality data for ML algorithms is one of the biggest challenges. The use of low-quality data leads to problems related to data preprocessing and feature extraction.

Time-Consuming task − Another challenge faced by ML models is the consumption of time, especially for data acquisition, feature extraction, and retrieval.

Lack of specialist persons − As ML technology is still in its infancy stage, availability of expert resources is a tough job.

No clear objective for formulating business problems − Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

Issue of overfitting & underfitting − If the model is overfitting or underfitting, it cannot be represented well for the problem.

Curse of dimensionality − Another challenge ML model faces is too many features of data points. This can be a real hindrance.

Deployment difficulty − The complexity of the ML model makes it quite difficult to be deployed in real life.

Applications of Machines Learning:-

Machine Learning is the most rapidly growing technology and according to researchers, we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with the traditional approach. Following are some real-world applications of ML −

- Emotion analysis
- Sentiment analysis
- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customers in online shopping

HOW TO START LEARNING MACHINE LEARNING?

Arthur Samuel coined the term **"Machine Learning"** in 1959 and defined it as a **"Field of study that gives computers the capability to learn without being explicitly programmed".**

And that was the beginning of Machine Learning! In modern times, Machine Learning is one of the most popular (if not the most!) career choices. According to Indeed, Machine Learning Engineer Is The Best Job of 2019 with a *344%* growth and an average base salary of **$146,085** per year.

But there is still a lot of doubt about what exactly is Machine Learning and how to start learning it? So this article deals with the Basics of Machine Learning and also the path you can follow to eventually become a full-fledged Machine Learning Engineer. Now let's get started!!!

How to start learning ML?

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer. Of course, you can always modify the steps according to your needs to reach your desired end goal!

Step 1 – Understand the Prerequisites

In case you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don't know these, never fear! You don't need a Ph.D. degree in these topics to get started but you do need a basic understanding.

(a) Learn Linear Algebra and Multivariate Calculus

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you are more focused on application-heavy machine learning, then you will not be that heavily focused on maths as there are many common libraries available. But if you want to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

(b) Learn Statistics

Data plays a huge role in Machine Learning. Around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data. So it is no surprise that you need to learn it!!!

Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking is a very important part of ML which deals with various concepts like Conditional Probability, Priors, Posteriors, Maximum Likelihood, etc.

(c) Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus, and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is Python! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. Many Python libraries are specifically useful for Artificial Intelligence and Machine Learning such as Keras, TensorFlow, Scikit-learn, etc.

So if you want to learn ML, it's best if you learn Python! You can do that using various online resources and courses such as **Fork Python** available Free on GeeksforGeeks.

Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to learning ML (Which is the fun part!!!) It's best to start with the basics and then move on to the more complicated stuff. Some of the basic concepts in ML are:

(a) Terminologies of Machine Learning

- **Model –** A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.
- **Feature –** A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, to predict a fruit, there may be features like color, smell, taste, etc.
- **Target (Label) –** A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.
- **Training –** The idea is to give a set of inputs(features) and its expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.
- **Prediction –** Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

(b) Types of Machine Learning

- **Supervised Learning –** This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.
- **Unsupervised Learning –** This involves using unlabelled data and then finding the underlying structure in the data to learn more and more about the data itself using factor and cluster analysis models.
- **Semi-supervised Learning –** This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.
- **Reinforcement Learning –** This involves learning optimal actions through trial and error. So the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

Advantages of Machine learning:-

1. Easily identifies trends and patterns -

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, an e-commerce website like Amazon serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

2. No human intervention is needed (automation)

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus software; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

3. Continuous Improvement

As **ML algorithms** gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

4. Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

5. Wide Applications

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

Disadvantages of Machine Learning:-

1. Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times when they must wait for new data to be generated.

2. Time and Resources

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

3. Interpretation of Results

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

4. High error-susceptibility

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

Python Development Steps: -

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt. sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str, and others. It was also object-oriented and had a module system.

Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions and a full garbage collector and it was supported by Unicode. Python flourished for another 8 years in the versions 2. x before the next major release Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backward compatible with Python 2. x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it."Some changes in Python 7.3:

- Print is now a function
- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g. a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e. int. long is int as well.
- The division of two integers returns a float instead of an integer. "//" can be used to have the "old" behavior.
- Text Vs. Data Instead Of Unicode Vs. 8-bit

Purpose:-

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional, and procedural, and has a large and comprehensive standard library.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills, and the huge standard library are key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Modules Used in Project:-

Tensorflow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions

- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python library providing high-performance data manipulation and analysis tools using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library that produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting, the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object-oriented interface or a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. **Python**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional, and procedural, and has a large and comprehensive standard library.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills, and the huge standard library are key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Install Python Step-by-Step in Windows and Mac :

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.
The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

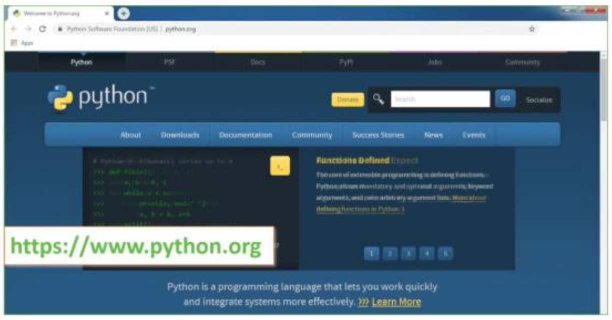
How to Install Python on Windows and Mac :

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.
Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your **System Requirements**. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a **Windows 64-bit operating system**. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheatsheet here. The steps on how to install Python on Windows 10, 8, and 7 are **divided into 4 parts** to help understand better.

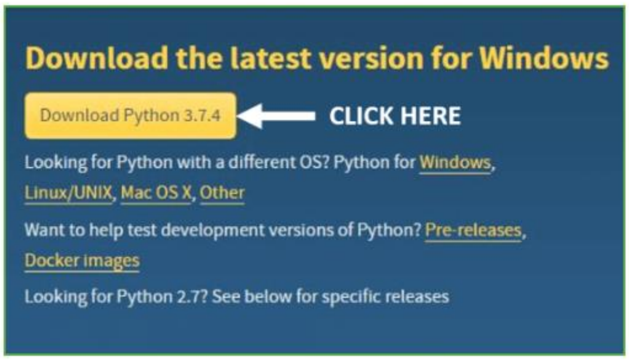
Download the Correct version into the system

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: **https://www.python.org**



Python is a programming language that lets you work quickly and integrate systems more effectively. >>> Learn More

Now, check for the latest and the correct version for your operating system.
Step 2: Click on the Download Tab.



Download the latest version for Windows

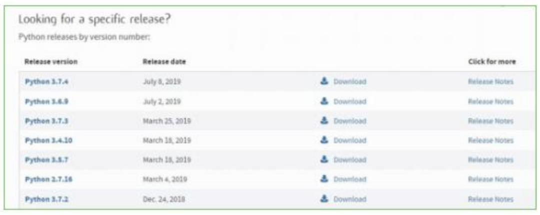
Download Python 3.7.4 CLICK HERE

Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Pre-releases, Docker images

Looking for Python 2.7? See below for specific releases

Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4



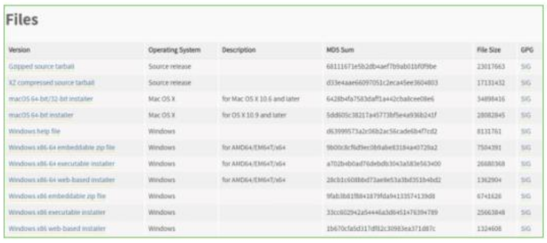
Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	Download	Release Notes
Python 3.6.9	July 2, 2019	Download	Release Notes
Python 3.7.3	March 25, 2019	Download	Release Notes
Python 3.4.10	March 18, 2019	Download	Release Notes
Python 3.5.7	March 18, 2019	Download	Release Notes
Python 2.7.16	March 4, 2019	Download	Release Notes
Python 3.7.2	Dec. 24, 2018	Download	Release Notes

Step 4: Scroll down the page until you find the Files option.

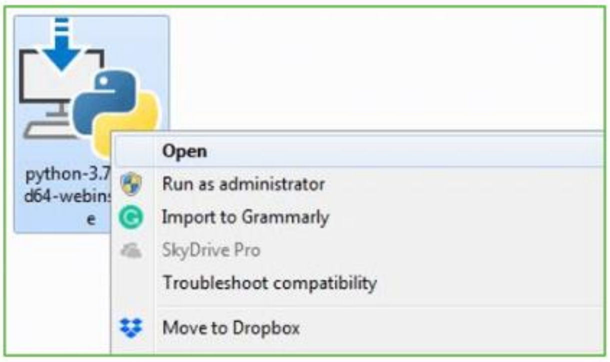
Step 5: Here you see a different version of python along with the operating system.



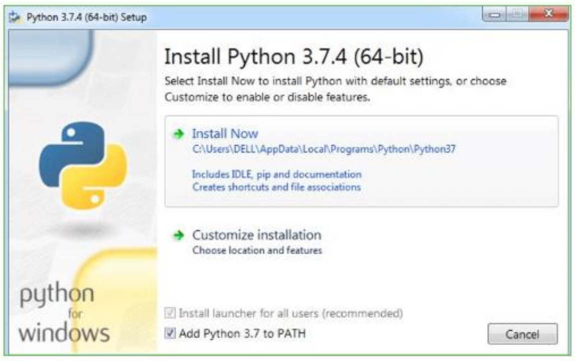
Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		68111671e5b2db4aef7b9ab01bf0f9be	23017663	SIG
XZ compressed source tarball	Source release		d33e4aae66097051c2eca45ee3604803	17131432	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6428b4fa7583daff1a442cba8cee08e6	34898416	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5dd605c38217a45773bf5e4a936b241f	28082845	SIG
Windows help file	Windows		d63999573a2c06b2ac56cade6b4f7cd2	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9b00c8cf6d9ec0b9abe83184a40729a2	7504391	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad76debdb3043a583e563400	26680368	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28cb1c608bbd73ae8e53a3bd351b4bd2	1362904	SIG
Windows x86 embeddable zip file	Windows		9fab3b81f8841879fda94133574139d8	6741626	SIG
Windows x86 executable installer	Windows		33cc602942a54446a3d6451476394789	25663848	SIG
Windows x86 web-based installer	Windows		1b670cfa5d317df82c30983ea371d87c	1324608	SIG

• To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer, or Windows x86 web-based installer.
•To download Windows 64-bit python, you can select any one of the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer, or Windows x86-64 web-based installer.
Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part of installing python i.e. Installation
Note: To know the changes or updates that are made in the version you can click on the Release Note Option.
Installation of Python
Step 1: Go to Download and Open the downloaded python version to carry out the installation process.



Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.



Python 3.7.4 (64-bit) Setup

Install Python 3.7.4 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

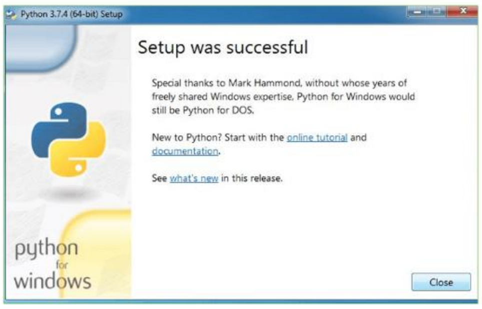
→ Install Now
C:\Users\DELL\AppData\Local\Programs\Python\Python37

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.7 to PATH

Step 3: Click on Install NOW After the installation is successful. Click on Close.



Python 3.7.4 (64-bit) Setup

Setup was successful

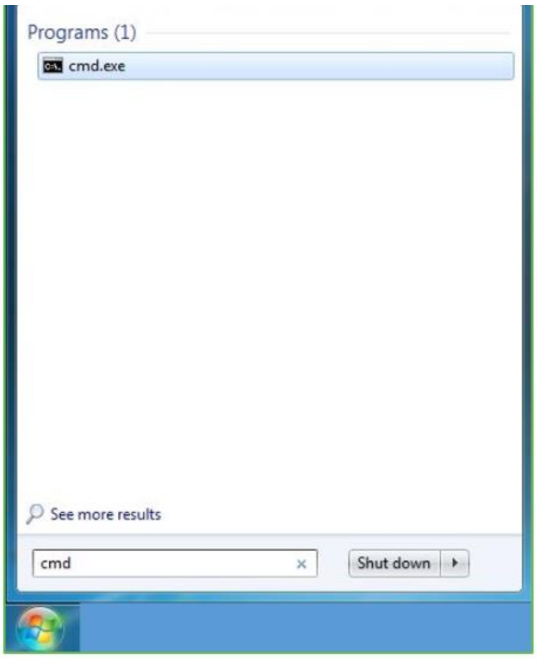
Special thanks to Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

New to Python? Start with the online tutorial and documentation.

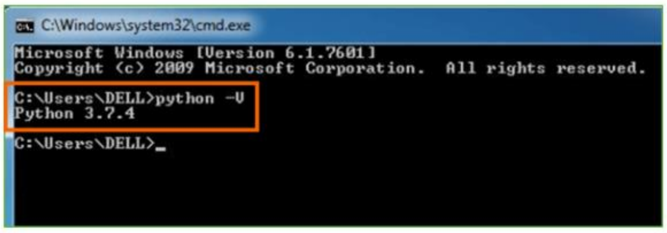
See what's new in this release.

With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.
Note: The installation process might take a couple of minutes.

Verify the Python Installation
Step 1: Click on Start
Step 2: In the Windows Run Command, type “cmd”.



Step 3: Open the Command prompt option.
Step 4: Let us test whether the python is correctly installed. Type **python –V** and press Enter.



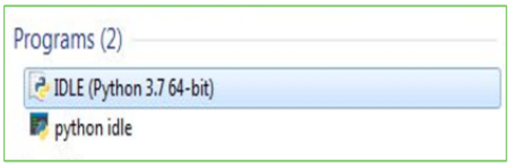
C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.
C:\Users\DELL>python -V
Python 3.7.4

C:\Users\DELL>_
```

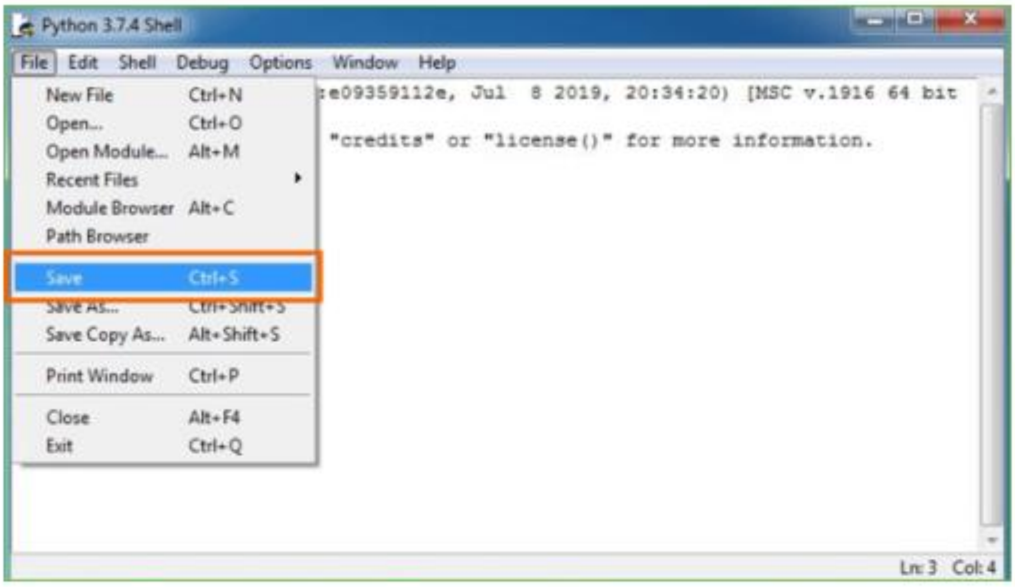
Step 5: You will get the answer as 3.7.4
Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works
Step 1: Click on Start
Step 2: In the Windows Run command, type “python idle”.



Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

Step 4: To go ahead with working in IDLE you must first save the file. **Click on File > Click on Save**



```
:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit

"credits" or "license()" for more information.
```

Step 5: Name the file and save it as a type that should be Python files. Click on SAVE. Here I have named the files Hey World.

Step 6: Now e.g. **enter print**

VI. SYSTEM TEST

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 unacceptably. There are various types of tests. Each test type addresses a specific testing requirement.

TYPES OF TESTS

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 structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at the 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.

Integration testing

Integration tests are designed to test integrated software components to determine if they 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 satisfied, as shown by successful 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.

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 are focused on requirements, key functions, or special test cases. In addition, systematic coverage of identified 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.

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.

White Box Testing

White Box Testing is a testing in which the software tester knows 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.

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, like 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.

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 in the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Use case ID	Security issues and defensive approaches in deep learning frameworks
Use case Name	Home button
Description	Display the home page of the application
Primary actor	User
Precondition	Users must open the application
Postcondition	Display the Home Page of an application
Frequency of Use case	Many times
Alternative use case	N/A
Use case diagrams	
Attachments	N/A

INTEGRATION TESTING

Software integration testing is the incremental integration testing of two or more integrated software components on a single

VII. CONCLUSION

Starting from the basic composition structure and principles of deep learning, this study describes security problems behind the application of deep learning and summarizes classic attack algorithms for deep learning technologies and development processes. Moreover, it also confirms that adversarial samples against deep learning are widespread. Studying confrontational algorithms can help us better understand and learn deep learning principles and their training and prediction processes. In this study, algorithm cases of deep learning attacks in recent years are summarized and analyzed; and defense techniques against countermeasure technologies are listed. Furthermore, examples of software flaws in specific implementations are provided. Deep learning prediction is susceptible to slight disturbances, thereby indicating that the deep learning structure has large defects, which is one of the factors hindering its further development. Deep learning can achieve very high prediction accuracy in fixed problems, such as image classification. However, in dynamic real-time scenes with complex interactions with the environment, making mistakes and misjudging emerging scenes are easy. This situation is also an AI technology bottleneck. Therefore, studying the security issues behind deep learning architecture algorithms has far-reaching significance.

VIII. FUTURE ENHANCEMENT

(1) Attacks and defensive approaches in deep learning are continuously being developed. The two elements involve a long-term development process, from the discovery of the poor robustness of deep learning to the emergence of various defensive approaches. Along with this process, both are constantly being developed and improved.

(2) The widespread existence of adversarial samples helps improve the robustness of deep learning algorithms. Deep learning prediction results have considerable deviations in slight disturbances, thereby indicating that deep learning algorithms require a long period. Progress is ongoing, and current developments remain incomplete and immature.

IX. REFERENCES

[1] W. W. Jiang and L. Zhang, Geospatial data to images: A deep-learning framework for traffic forecasting, Tsinghua Science and Technology, vol. 24, no. 1, pp. 52–64, 2019.

[2] L. Zhang, C. B. Xu, Y. H. Gao, Y. Han, X. J. Du, and Z. H. Tian, Improved Dota2 lineup recommendation model based on a bidirectional LSTM, Tsinghua Science and Technology, vol. 25, no. 6, pp. 712–720, 2020.

[3] H. M. Huang, J. H. Lin, L. Y. Wu, B. Fang, Z. K. Wen, and F. C. Sun, Machine learning-based multi-modal information perception for soft robotic hands, Tsinghua Science and Technology, vol. 25, no. 2, pp. 255–269, 2020.

[4] X. Y. Yuan, P. He, Q. L. Zhu, and X. L. Li, Adversarial Examples: Attacks and defenses for deep learning, IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 9, pp. 2805– 2824, 2019.

[5] J. C. Hu, J. F. Chen, L. Zhang, Y. S. Liu, Q. H. Bao, H. Ackah-Arthur, and C. Zhang, A memory-related vulnerability detection approach based on vulnerability features, Tsinghua Science and Technology, vol. 25, no. 5, pp. 604–613, 2020.

[6] C. Szegedy, W. Zaremba, I. Sutskever I, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, Intriguing properties of neural networks, arXiv preprint arXiv: 1312.6199, 2013.

[7] A. Athalye, N. Carlini, and D. Wagner, Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples, arXiv preprint arXiv: 1802.00420, 2018.

[8] Y. T. Xiao, C. M. Pun, and J. Z. Zhou Generating adversarial perturbation with root mean square gradient, arXiv preprint arXiv: 1901.03706, 2019.

[9] I. J. Goodfellow, J. Shlens, and C. Szegedy Explaining and harnessing adversarial examples, arXiv preprint arXiv: 1412.6572, 2014.

[10] J. W. Su, D. V. Vargas, and K. Sakurai, One-pixel attack for fooling deep neural networks, IEEE Trans. Evol. Comput., vol. 23, no. 5, pp. 828–841, 2019.

[11] W. He, J. Wei, X. Y. Chen, N. Carlini, and D. Song, Adversarial example defense: Ensembles of weak defenses are not strong, in Proc 11th USENIX Workshop on Offensive Technologies, Vancouver, Canada, 2017.

[12] G. W. Xu, H. W. Li, H. Ren, K. Yang, and R. H. Deng, Data security issues in deep learning: Attacks, countermeasures, and opportunities, IEEE Comm. Mag., vol. 57, no. 11, pp. 116–122, 2019.

[13] M. I. Tariq, N. A. Memon, S. Ahmed, S. Tayyaba, M. T. Mushtaq, N. A. Mian, M. Imran, and M. W. Ashraf, A review of deep learning security and privacy defensive techniques, Mobile Inf. Syst., vol. 2020, p. 6535834, 2020.

[14] H. Bae, J. Jang, D. Jung, H. Jang, H. Ha, and S. Yoon, Security, and privacy issues in deep learning, arXiv preprint arXiv: 1807.11655, 2018.

[15] S. L. Qiu, Q. H. Liu, S. J. Zhou, and C. J. Wu, Review of artificial intelligence adversarial attack and defense technologies. Appl. Sci., vol. 9, no. 5, p. 909.