

DETECTING MALICIOUS SOCIAL BOTS BASED ON CLICKSTREAM SEQUENCES

Submitted in partial fulfillment of the requirements for the award of Bachelor
of Science Degree in Computer Science

By

Haritha S (Reg.No:38290023)

Kamali K (Reg.No:38290031)



DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,

CHENNAI – 600119, TAMILNADU

MARCH 2021



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI- 600119
www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Haritha S (Reg.No:38290023)** and **Kamali K (Reg.No:38290031)** who carried out the project entitled "**Detecting Malicious Social Bots Based on Clickstream Sequences**" under our supervision from November 2020 to April 2021

Internal Guide

Dr. J. ALBERT MAYAN M.E., Ph.D.,

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

We, **Haritha S (Reg.No:38290023)** and **Kamali K (Reg.No:38290031)** hereby declare that the Project Report entitled “**Detecting Malicious Social Bots Based on Clickstream Sequences**” done by me under the guidance of **Dr.J.Albert Mayan** submitted in partial fulfillment of the requirements for the award of Bachelor of Science degree in Computer Science.

DATE:

PLACE:CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M. E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneshwari M. E., Ph. D.,** and **Dr. L. Lakshmanan, M. E., Ph. D., Heads of the Department** of Computer Science for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **DR J Albert Mayan** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **DEPARTMENT OF COMPUTER SCIENCE** who were helpful in many ways for the completion of the project.

ABSTRACT

The presence of bots has been felt in many aspects of social media. Twitter, one example of social media, has especially felt the impact, with bots accounting for a large portion of its users. These bots have been used for malicious tasks such as spreading false information about political candidates and inflating the perceived popularity of celebrities. Furthermore, these bots can change the results of common analyses performed on social media. With the significant increase in the volume, velocity, and variety of user data (e.g., user generated data) in online social networks, there have been attempts to design new ways of collecting and analyzing such big data. For example, social bots have been used to perform automated analytical services and provide users with improved quality of service. However, malicious social bots have also been used to disseminate false information (e.g., fake news), and this can result in real-world consequences. Therefore, detecting and removing malicious social bots in online social networks is crucial. The most existing detection methods of malicious social bots analyze the quantitative features of their behavior. These features are easily imitated by social bots; thereby resulting in low accuracy of the analysis. A novel method of detecting malicious socialbots, including both features selection based on the transition probability of clickstream sequences and semi-supervised clustering, is presented in this paper. This method not only analyzes transition probability of user behavior clickstreams but also considers the time feature of behavior. Findings from our experiments on real online social network platforms demonstrate that the detection accuracy for different types of malicious socialbots by the detection method of malicious social bots based on transition probability of user behavior clickstreams increases by an average of 12.8%, in comparison to the detection method based on quantitative analysis of user behavior.

TABLE OF CONTENTS	PAGE NOS
ABSTRACT	1
1.INTRODUCTION	4
1.1. Cloud Computing	5
1.2 Modules in the project	5
2.LITERATURE SURVEY	6
3.REQUIREMENTS SPECIFICATION	9
3.1 Introduction	9
3.2 Hardware Requirements	9
3.3 Software Requirements	10
4. SYSTEM ANALYSIS	11
4.1 Existing System	11
4.2 Proposed System	12
4.3 Feasibility Study	13
4.4 Software Specification	14
4.5 Software Description	16
5.SYSTEM FRAMEWORK	21
6.DESIGN	22
6.1 Data Flow Diagram	23
6.2 UML Diagram	26
7. SYSTEM IMPLEMENTATION	30
7.1 Introduction	30
8. TESTING	74
8.1 Introduction	74
8.2 Types of Testing	75
9. SAMPLE SCREENSHOTS	78

10. CONCLUSION	84
REFERENCE	85
APPENDIX	
A. SOURCE CODE	
B. PUBLICATION	

CHAPTER - 1

INTRODUCTION

In online social networks, social bots are social accounts controlled by automated programs that can perform corresponding operations based on a set of procedures. The increasing use of mobile devices (e.g., Android and iOS devices) also contributed to an increase in the frequency and nature of user interaction via social networks. It is evidenced by the significant volume, velocity and variety of data generated from the large online social network user base. Social bots have been widely deployed to enhance the quality and efficiency of collecting and analyzing data from social network services. For example, the social bot SF QuakeBot is designed to generate earthquake reports in the San Francisco Bay, and it can analyze earthquake related information in social networks in real-time. However, public opinion about social networks and massive user data can also be mined or disseminated for malicious or nefarious purpose. In online social networks, automatic social bots cannot represent the real desires and intentions of normal human beings, so they are usually looked upon malicious ones. For example, some fake social bots accounts created to imitate the profile of a normal user, steal user data and compromise their privacy, disseminate malicious or fake information, malicious comment, promote or advance certain political or ideology agenda and propaganda, and influence the stock market and other societal and economical markets. Such activities can adversely impact the security and stability of social networking platforms. In previous research, various methods were used to protect the security of online social network. User behavior is the most direct manifestation of user intent, as different users have different habits, preferences, and online behavior (e.g., the way one clicks or types, as well as the speed of typing). In other words, we may be able to mine and analyze information hidden in user's online behavior to profile and identify different users. However, we also need to be conscious of situational factors that may play a role in changing user's online behavior. In other words, user behavior is dynamic and its environment is

constantly changing i.e., external observable environment (e.g., environment and behavior) of application context and the hidden environment in user information. In order to distinguish social bots from normal users accurately, detect malicious social bots, and reduce the harm of malicious social bots, we need to acquire and analyze social situation of user behavior and compare and understand the differences of malicious social bots and normal users in dynamic behavior. Specifically, in this paper, we aim to detect malicious social bots on social network platforms in real-time, by (1) proposing the transition probability features between user clickstreams based on the social situation analytics; and (2) designing an algorithm for detecting malicious social bots based on spatiotemporal features.

1.1.Cloud computing:

Cloud computing is a general term for the delivery of hosted services over the internet. Cloud computing enables companies to consume a compute resource, such as a virtual machine (VM), storage or an application, as a utility just like electricity rather than having to build and maintain computing infrastructures in house.

1.2 .Modules in the project:

Security remains a primary concern for businesses contemplating cloud adoption -- especially public cloud adoption. Public cloud service providers share their underlying hardware infrastructure between numerous customers, as public cloud is a multi-tenant environment. This environment demands copious isolation between logical compute resources. At the same time, access to public cloud storage and compute resources is guarded by account login credentials.

Many organizations bound by complex regulatory obligations and governance standards are still hesitant to place data or workloads in the public cloud for fear of outages, loss or theft. However, this resistance is fading, as logical isolation has proven reliable, and the addition of data encryption and various identity and access management tools has improved security within the public cloud.

CHAPTER - 2

LITERATURE SURVEY

2.1 Behavior Enhanced Deep Bot Detection in Social Media:

Social bots are regarded as the most common kind of malwares in social platform. They can produce fake messages, spread rumors, and even manipulate public opinions. Recently, massive social bots are created and widely spread in social platform, they bring negative effects to public and netizen security. Bot detection aims to distinguish bots from human and it catches more and more attentions in recent years. In this paper, we propose a behavior enhanced deep model (BeDM) for bot detection. The proposed model regards user content as temporal text data instead of plain text to extract latent temporal patterns. Moreover, BeDM fuses content information and behavior information using deep learning method. To the best of our knowledge, this is the first trial that applies deep neural network in bot detection. Experiments on real world dataset collected from Twitter also demonstrate the effectiveness of our proposed model.

Author: Chiyu Cai, Linjing Li, and Daniel Zeng, Fellow.

Published In: 2017

2.2 An Analysis of Socware Cascades in Online Social Networks:

Online social networks (OSNs) have become a popular new vector for distributing malware and spam, which we refer to as socware. Unlike email spam, which is sent by spammers directly to intended victims, socware cascades through OSNs as compromised users spread it to their friends. In this paper, we analyze data from the walls of roughly 3 million Facebook users over five months, with the goal of developing a better understanding of socware cascades. We study socware cascades to understand: (a) their spatiotemporal properties, (b) the underlying motivations and mechanisms, and (c) the social engineering tricks used to con users. First, we identify an evolving trend in which cascades appear to be throttling their rate of growth to evade detection, and thus, lasting longer. Second, our forensic investigation into the infrastructure that supports these cascades shows that, surprisingly, Facebook seems to be inadvertently enabling most cascades; 44% of cascades are disseminated via Facebook applications. At the same time, we observe large groups of synergistic Facebook apps (more than 144 groups of size 5 or more) that collaborate to support

multiple cascades. Lastly, we find that hackers rely on two social engineering tricks in equal measure luring users with free products and appealing to users' social curiosity to enable socware cascades. Our findings present several promising avenues towards reducing socware on Facebook, but also highlight associated challenges.

Author: Ting-Kai Huang, Md Sazzadur Rahman, Harsha V. Madhyastha, Michalis Faloutsos, and Bruno Ribeiro

Published In: 2013.

2.3 A New Approach to Bot Detection: Striking the Balance Between Precision and Recall:

The presence of bots has been felt in many aspects of social media. Twitter, one example of social media, has especially felt the impact, with bots accounting for a large portion of its users. These bots have been used for malicious tasks such as spreading false information about political candidates and inflating the perceived popularity of celebrities. Furthermore, these bots can change the results of common analyses performed on social media. It is important that researchers and practitioners have tools in their arsenal to remove them. Approaches exist to remove bots, however they focus on precision to evaluate their model at the cost of recall. This means that while these approaches are almost always correct in the bots they delete, they ultimately delete very few, thus many bots remain. We propose a model which increases the recall in detecting bots, allowing a researcher to delete more bots. We evaluate our model on two real world social media datasets and show that our detection algorithm removes more bots from a dataset than current approaches.

Author: Fred Morstatter, Liang Wu, Tahora H.Tempe, Huan Liu

Published In: 2016.

2.4 Efficient Compressed Cipher text Length Scheme Using Multi Authority CP-ABE for Hierarchical Attributes:

In an attribute based encryption (ABE), user is identified with help of some attributes and their functions for encryption and decryption of the data. The current techniques based on attribute-based encryption have found that if user's access structure includes a considerable amount of attribute information labeled as Don't Care, then the encryption pairing operation has low calculation efficiency and cipher

text information redundancy. In this paper, we have proposed a hierarchical multi-authority attribute-based encryption on prime order groups to tackle these problems. Our encryption technique has a polycentric attribute authorization system based on an AND gate access structure, with a unified attribute index established by each attribute authority throughout the system, to form a binary tree, i.e., attribute access tree. The state value of the parent node can be determined by the state of its child node in an attribute access tree. The attribute-based encryption established in this manner is theoretically proven to effectively decrease the calculation amount for decryption and compress the redundant information in the cipher text as much as possible. Our encryption technique has a theoretical and practical significance in the system of “large universe” constructions.

Author:Zhiyong Zhang, Cheng Li¹, Brij B. Gupta²,Danmei Niu¹

Published In: 2018.

2.5 ProGuard: Detecting Malicious Accounts in Social-Network Based Online Promotions:

Online social networks gradually integrate financial capabilities by enabling the usage of real and virtual currency. They serve as new platforms to host a variety of business activities such as online promotion events, where users can possibly get virtual currency as rewards by participating such events. Both OSNs and business partners are significantly concerned when attackers instrument a set of accounts to collect virtual currency from these events, which make these events ineffective and result in significant financial loss. It becomes of great importance to proactively detect these malicious accounts before the online promotion activities and subsequently decrease their priority to be rewarded. We have performed extensive experiments based on data collected from Tencent QQ, a global leading OSN with built-in financial management activities. Experimental results have demonstrated that our system can accomplish a high detection rate of 96.67% at a very low false positive rate of 0.3%.

Author: Yadong Zhou, Dae Wook Kim, Junjie Zhang, Lili Liu, Huan Jin, Hongbo Jin, Ting Liu.

Published In: 2016.

CHAPTER – 3

REQUIREMENTS SPECIFICATION

3.1 INTRODUCTION:

To be used efficiently, all computer software needs certain hardware components or the other software resources to be present on a computer. These pre-requisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that this trend plays a bigger part in driving upgrades to existing computer systems than technological advancements.

3.2 HARDWARE REQUIREMENTS:

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatibility and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements

Hardware requirements for present project:

- System : Intel Pentium.
- Hard Disk : 120 GB.
- Monitor : 15" LED
- Input Devices : Keyboard, Mouse
- Ram : 2 GB

3.3 SOFTWARE REQUIREMENTS:

Software Requirements deal with defining software resource requirements and pre-requisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or pre-requisites are generally not included in the software installation package and need to be installed separately before the software is installed.

Software requirements for present project:

- Operating system : Windows 7
- Coding Language : PYTHON
- Tool : Netbeans
- Database : SQL

CHAPTER - 4

SYSTEM ANALYSIS

4.1 EXISTING SYSTEM:

Recent statistics show that more than 50% of Twitter accounts are not human users. Social network administrators are well aware of these harmful activities and try to delete these users using their suspension/removal systems. By one estimate 28% of accounts created in 2008 and half of the accounts created in 2014 have been suspended by Twitter. What is not well taken care of is the role of bots in facilitating these malicious activities. In one study, 145,000 accounts survived for months without detection. Today, 16% of spammers in Twitter are bots. Although social network spam detection approaches are still insufficient, bot detection in social networks has received wide attention from the research community in recent years. Botnets become widespread in wired and wireless networks. In particular, bots in a botnet are able to cooperate towards a common malicious purpose. In recent years, social bots have become very popular in social networks, and they can imitate human activities in social networks. They are also programmed to work together to fulfill the prescribed tasks. There are a wide range of methods (e.g., sophisticated techniques and tools that may be associated with nation states and state-sponsored actors) used by some users with malicious or nefarious intent as well as social bots. For example, in order to imitate the features of human users successfully, socialbots may 'crawl' for words and pictures from online social networks to complete fabricated user profiles and so on. Semi-social bots between humans and social bots have also reportedly emerged in social networks, which are highly complex social bots that bear the characteristics of human behavior and social bot behavior. The automated procedure for semi-social bot is generally activated by humans, and the subsequent actions are automatically performed by socialbots. This process further increases the uncertainty of the operation time of social bots. Social bots are generally more intelligent and they can more easily imitate human behavior, and they cannot be easily detected. In existing literature, social bots are generally detected using machine learning-based approaches, such as Bot or not released by the Twitter in 2014. In Bot or Not, the random forest model is used in both training and analysis by using historical social information of normal users and social bots accounts. Based on six features (i.e.

network, user, making friends, time, content and emotion), this model distinguished normal users from social bots. Morstatter *et al.* proposed a heuristic type supervised Boost OR model with increasing recall rate to detect malicious bots, which using the proportion of tweets forwarded to P. Shi *et al.*: Detecting Malicious Social Bots Based on Clickstream Sequences the published tweets on the Twitter, the mean length of tweets, URL, and forwarding interval. Wang *et al.* constructed a semi-supervised clickstream similarity graph model for user behavior to detect abnormal accounts in Renren. According to the social interactions between users of the Twitter user to identify the active, passive and inactive users, a supervised machine learning method was proposed to identify socialbots on the basis of age, location and other static features of active, passive, and inactive users in the Twitter, as well as interacting person, interaction content, interaction theme, and some dynamic characteristics. A time act model, namely, Act-M, was constructed focusing on the timing of user behavior activities, which can be used to accurately determine the interval between different behaviors of social media users to accurately detect malicious users. There have been focused on detecting semi-socialbots too. For example, a management framework relying on entropy component, spam detection component, account attribute component, and decision maker was proposed by Chu *et al.* In the approach, Naive Bayes is adopted to categorize automated Twitter accounts into human, social bots, or semi-social bots. Previous studies have also shown that quantitative features such as friends, fans, forwarders, and tweets can be used in feature selection. The supervised learning method can be effective in detecting social bots, however annotation and training for large amounts of data are required in supervised learning. Tagging data requires time, manpower, and is generally unsuitable for the big data social networking environment.

DISADVANTAGES OF EXISTING SYSTEM:

- Social bots are generally more intelligent and they can more easily imitate human behavior, and they cannot be easily detected.
- It is not possible to determine which cluster is normal and which cluster is abnormal.
- Social network spam detection approaches are still insufficient

4.2 PROPOSED SYSTEM:

In this project, we aim to detect malicious social bots on social network platforms in real-time, by (1) proposing the transition probability features between user clickstreams based on the social situation analytics; and(2) designing an algorithm for detecting malicious social bots based on spatiotemporal features. In order to better detect malicious social bots in online social networks, we analyze user behavior features and identify transition probability features between user clickstreams Based on the transition probability features and time interval features, a semi-supervised social bots detection method based on space-time features is proposed.

ADVANTAGES OF PROPOSED SYSTEM:

- We evaluate user behavior features and select the transition probability of user behavior on the basis of general behavior Characteristics.
- We then analyze and classify situation aware user behaviors in social networks using our proposed semi supervised clustering detection method.
- This allows us to promptly detect malicious social bots using only a small number of tagged users.

4.3 FEASIBILITY STUDY:

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. 'Three key considerations involved in the feasibility analysis are:

4.3.1 Economic Feasibility:

This study is carried out to check the economic impact will have on the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the

budget and this was achieved because most of the technologies used are freely available. Only the customized products have to be purchased.

4.3.2 Technical Feasibility:

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes for the implementing this system.

4.3.3 Operational Feasibility:

The aspect of 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.

4.4 SOFTWARE SPECIFICATION:

PYTHON:

Python is an interpreter, high-level, general-purpose programming language. Python is dynamically typed and garbage-collected. 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 interpreters are available for many operating systems. A global community of programmers develops and maintains C, Python, an open source reference implementation. A non-profit organization, the Python Software Foundation, manages and directs resources for Python and CPython development.

Microsoft SQL Server:

Microsoft SQL Server is a relational database management system developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications—which may run either on the same computer or on another computer across a network (including the Internet).

Microsoft markets at least a dozen different editions of Microsoft SQL Server, aimed at different audiences and for workloads ranging from small single-machine applications to large Internet-facing applications with many concurrent users.

SQLCMD:

SQLCMD is a command line application that comes with Microsoft SQL Server, and exposes the management features of SQL Server. It allows SQL queries to be written and executed from the command prompt. It can also act as a scripting language to create and run a set of SQL statements as a script. Such scripts are stored as a .sql file, and are used either for management of databases or to create the database schema during the deployment of a database.

SQLCMD was introduced with SQL Server 2005 and this continues with SQL Server 2012, 2014 and 2016. Its predecessor for earlier versions was OSQL and ISQL, which is functionally equivalent as it pertains to TSQL execution, and many of the command line parameters are identical, although SQLCMD adds extra versatility.

SQL Server Management Studio:

SQL Server Management Studio is a GUI tool included with SQL Server 2005 and later for configuring, managing, and administering all components within Microsoft SQL Server. The tool includes both script editors and graphical tools that work with objects and features of the server. SQL Server Management Studio replaces Enterprise Manager as the primary management interface for Microsoft SQL Server since SQL Server 2005. A version of SQL

Server Management Studio is also available for SQL Server Express Edition, for which it is known as SQL Server Management Studio Express (SSMSE).

A central feature of SQL Server Management Studio is the Object Explorer, which allows the user to browse, select, and act upon any of the objects within the server. It can be used to visually observe and analyze query plans and optimize the database performance, among others. SQL Server Management Studio can also be used to create a new database, alter any existing database schema by adding or modifying tables and indexes, or analyze performance. It includes the query windows which provide a GUI based interface to write and execute queries.

Currently:

As of October 2017 the following versions are supported by Microsoft:

- SQL Server 2008
- SQL Server 2008 R2
- SQL Server 2012
- SQL Server 2014
- SQL Server 2016
- SQL Server 2017

From SQL Server 2016 onward, the product is supported on x64 processors only.

4.5 Software Description:

PYTHON:

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. Many other paradigms are supported via extensions, including design by contract and logic programming.

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.

Python's design offers some support for functional programming in the Lisp tradition. It has `filter`, `map` and `reduce` functions; list comprehensions, dictionaries, sets and generator expressions. The standard library has two modules (`itertools` and `functools`) that implement functional tools borrowed from Haskell and Standard ML.

The language's core philosophy is summarized in the document The Zen of Python (PEP 20), which includes aphorisms such as:

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Readability counts.

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 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 CPython 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 Pythoon, a just-in-time compiler. Pythoon 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.

A common neologism in the Python community is pythonic, which can have a wide range of meanings related to program style. To say that code is pythonic is to say that it uses Python idioms well, that it is natural or shows fluency in the language, that it conforms with Python's minimalist philosophy and emphasis on readability. In contrast, code that is difficult to understand or reads like a rough transcription from another programming language is called unpythonic.

Users and admirers of Python, especially those considered knowledgeable or experienced, are often referred to as Pythonistas.

Libraries:

Python's large standard library, commonly cited as one of its greatest strengths, provides tools suited to many tasks. For Internet-facing applications, many standard formats and protocols such as MIME and HTTP are supported. It includes modules for creating graphical user interfaces, connecting to relational databases, generating pseudorandom numbers, arithmetic with arbitrary precision decimals, manipulating regular expressions, and unit testing.

Some parts of the standard library are covered by specifications, but most modules are not. They are specified by their code, internal

documentation, and test suites (if supplied). However, because most of the standard library is cross-platform Python code, only a few modules need altering or rewriting for variant implementations.

As of March 2018, the Python Package Index (PyPI), the official repository for third-party Python software, contains over 130,000 packages with a wide range of functionality, including:

- Graphical user interfaces
- Web frameworks
- Multimedia
- Databases
- Networking
- Test frameworks
- Automation
- Web scraping
- Documentation
- System administration
- Scientific computing
- Text processing
- Image processing

Development:

Python's development is conducted largely through the Python Enhancement Proposal (PEP) process, the primary mechanism for proposing major new features, collecting community input on issues and documenting Python design decisions. Python coding style is covered in PEP 8. Outstanding PEPs are reviewed and commented on by the Python community and the steering council.

Enhancement of the language corresponds with development of the CPython reference implementation. The mailing list python-dev is the primary forum for the language's development. Specific issues are discussed in the Roundup bug tracker maintained at python.org. Development originally took

place on a self-hosted source-code repository running Mercurial, until Python moved to GitHub in January 2017.

CPython's public releases come in three types, distinguished by which part of the version number is incremented:

- Backward-incompatible versions, where code is expected to break and need to be manually ported. The first part of the version number is incremented. These releases happen infrequently for example, version 3.0 was released 8 years after 2.0.
- Major or "feature" releases, about every 18 months, are largely compatible but introduce new features. The second part of the version number is incremented. Each major version is supported by bugfixes for several years after its release.
- Bugfix releases, which introduce no new features, occur about every 3 months and are made when a sufficient number of bugs have been fixed upstream since the last release. Security vulnerabilities are also patched in these releases. The third and final part of the version number is incremented.

Many alpha, beta, and release-candidates are also released as previews and for testing before final releases. Although there is a rough schedule for each release, they are often delayed if the code is not ready. Python's development team monitors the state of the code by running the large unit test suite during development, and using the BuildBot continuous integration system.

The community of Python developers has also contributed over 86,000 software modules (as of 20 August 2016) to the Python Package Index (PyPI), the official repository of third-party Python libraries.

The major academic conference on Python is Python. There are also special Python mentoring programs.

CHAPTER - 5

SYSTEM FRAMEWORK

A total of 1500 malicious social bots accounts on the CyVOD platform are assigned different tasks, including malicious social bots that perform a single task, malicious social bots that coordinate to perform tasks, and malicious social bots that perform mixed tasks. For example, a user can perform two or more actions in the actions of liking, comment, sharing and so on. The social bot for malicious likes, the value of the $P(\text{play}; \text{like})$ (the transition probability of "the current click event is and the next click event is liking") would be high and the value of other transition probability features would be small or zero. The mixed social bot, the values of six transition probability features maybe average, which looks like normal user. In this experiment, four malicious social bots that perform different specific purposes and two malicious social bots with mixed behavior are set up. We designed an Android application called "SocialBot" to simulate the behavior of social bots. According to the functional characteristics of the experimental platform, we designed such seven categories of social bot that perform different tasks. The social bot program can be activated automatically or manually by clicking these buttons.

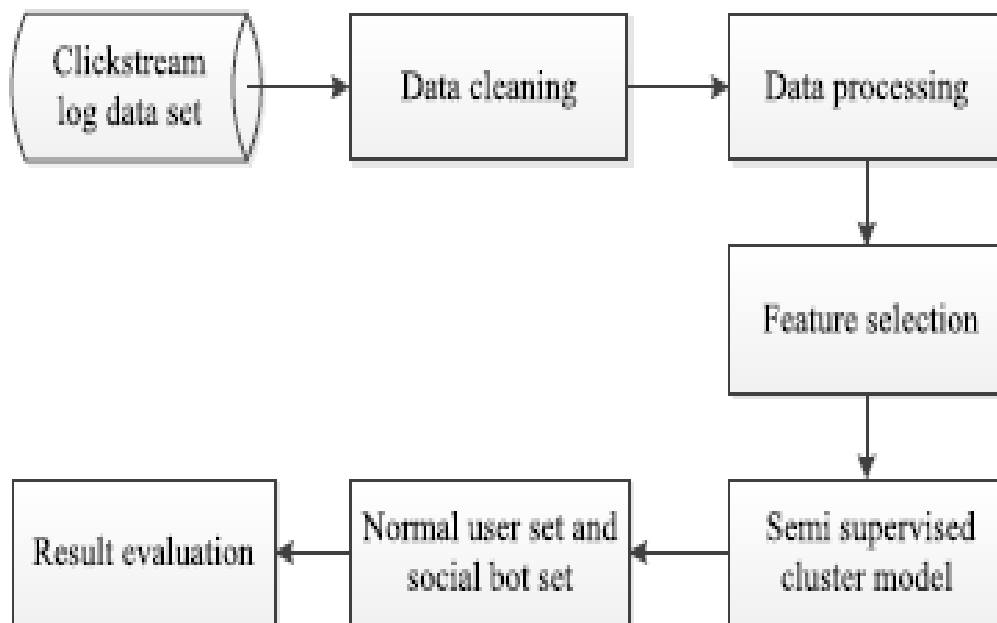


Fig 1 : SYSTEM FRAME WORK

CHAPTER - 6

SYSTEM DESIGN

6.1. 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.

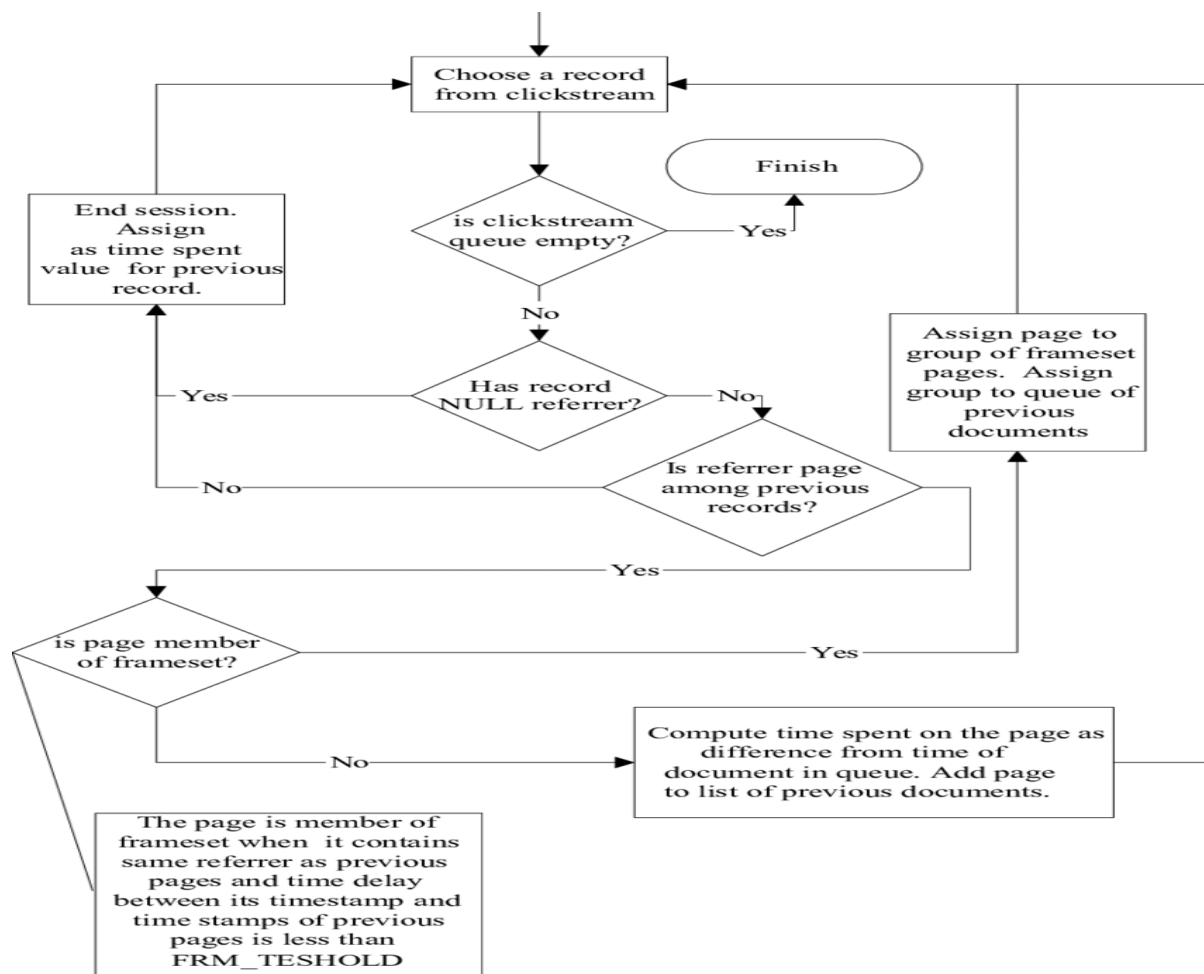


Fig 2 :DATA FLOW DIAGRAM

6.2 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.

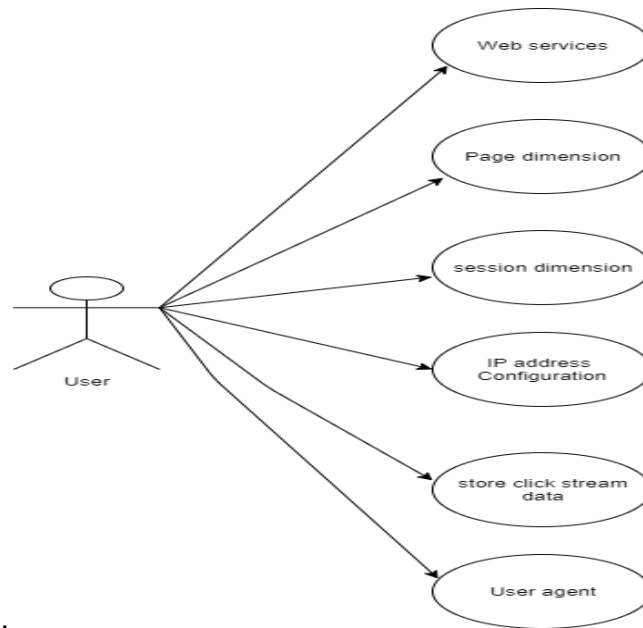
GOALS:

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.

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 3: USECASE DIAGRAM

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.

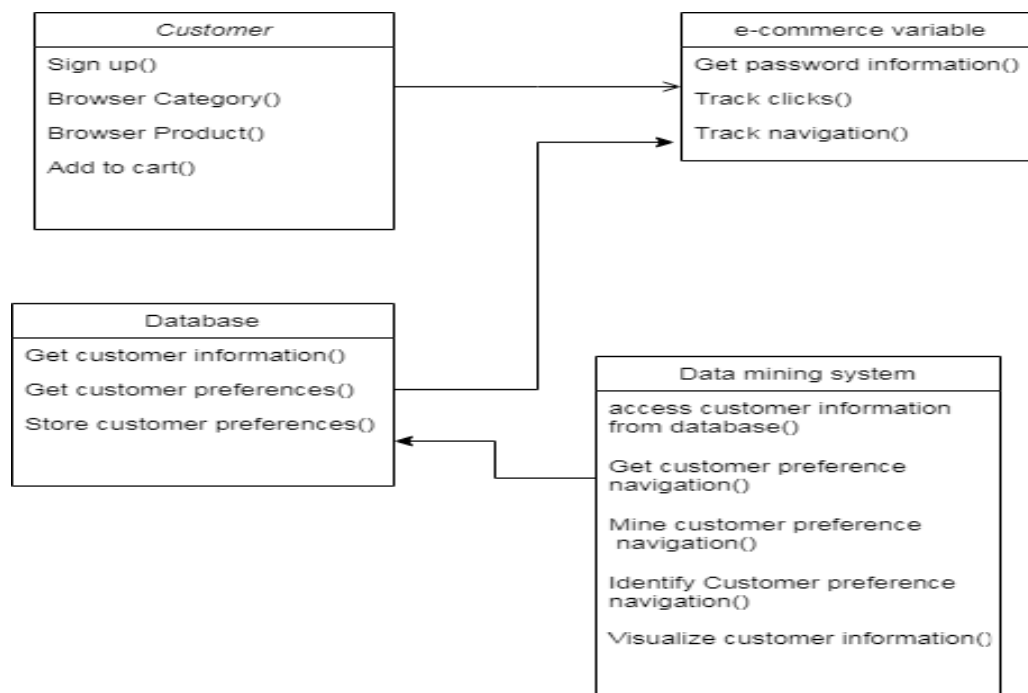


Fig 4: CLASS DIAGRAM

SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

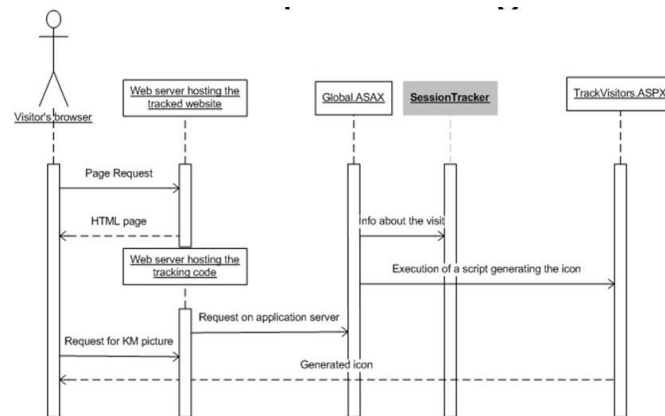


Fig 5: SEQUENCE DIAGRAM

ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

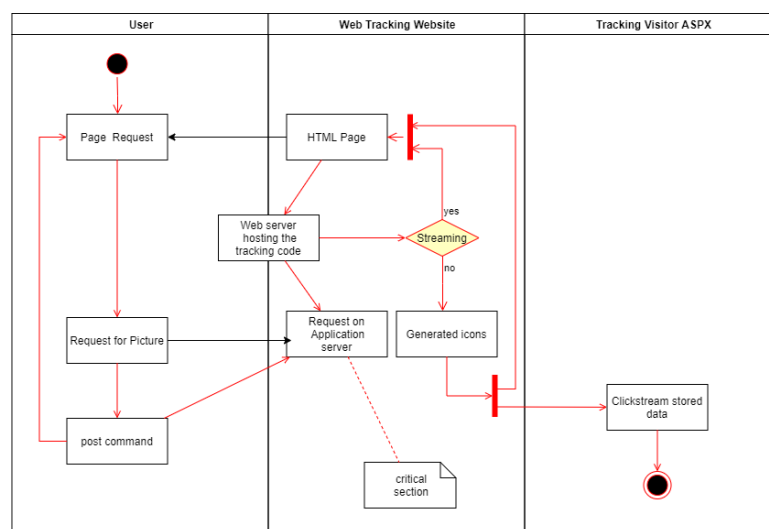


Fig 6: ACTIVITY DIAGRAM

CHAPTER - 7

SYSTEM IMPLEMENTATION

7.1 INTRODUCTION:

System Implementation is the important stage of project when the theoretical design is tuned into practical system. The main stages in the implementation are as follows:

- Planning
- Training
- System testing and
- Changeover Planning

Planning is the first task in the system implementation. Planning means deciding on the method and the time scale to be adopted. At the time of implementation of any system people from different departments and system analysis involve. They are confronted to practical problem of controlling various activities of people outside their own data processing departments. The line managers controlled through an implementation coordinating committee. The committee considers ideas, problems and complaints of user department, it must also consider:

- The implication of system environment;
- Self-selection and allocation for implementation tasks
- Consultation with unions and resources available
- Standby facilities and channels of communication

CHAPTER - 8

SYSTEM TESTING

8.1 INTRODUCTION:

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.

8.2 TYPES OF TESTING:

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.

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 satisfactory, 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.

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.

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 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.

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.

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.

Test Results:

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

Acceptance Testing:

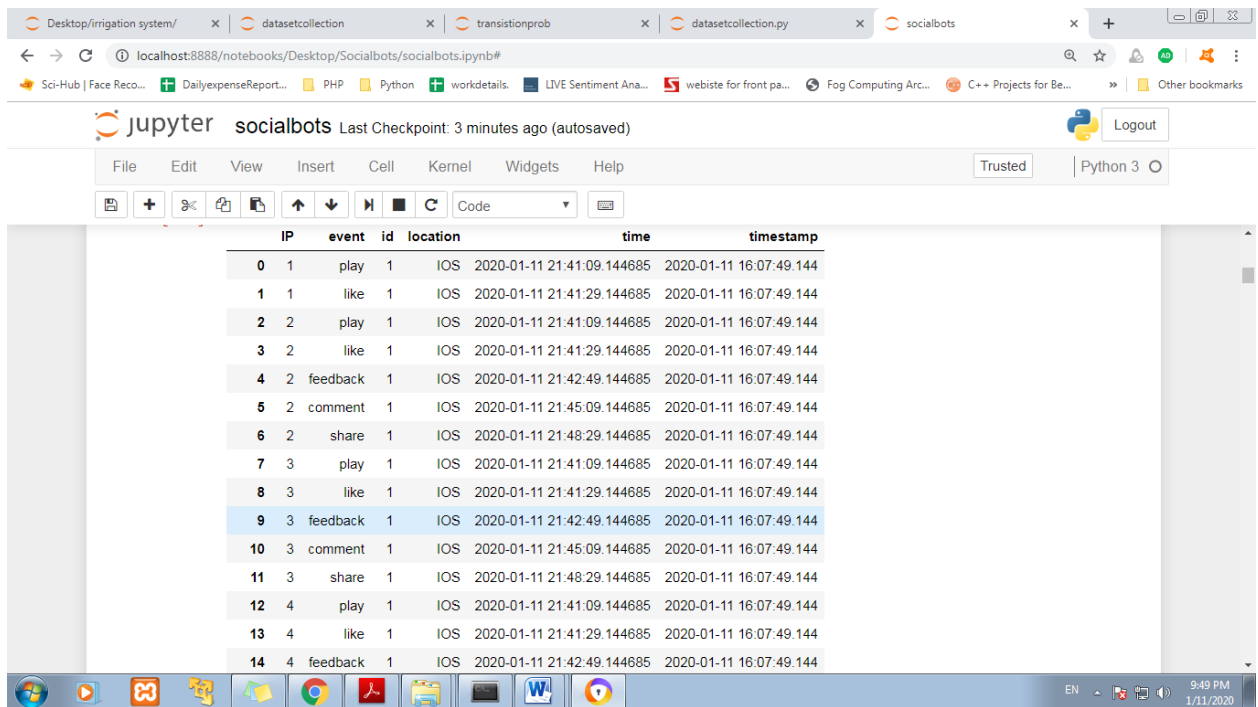
User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

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

CHAPTER - 9

SAMPLE SCREENSHOTS

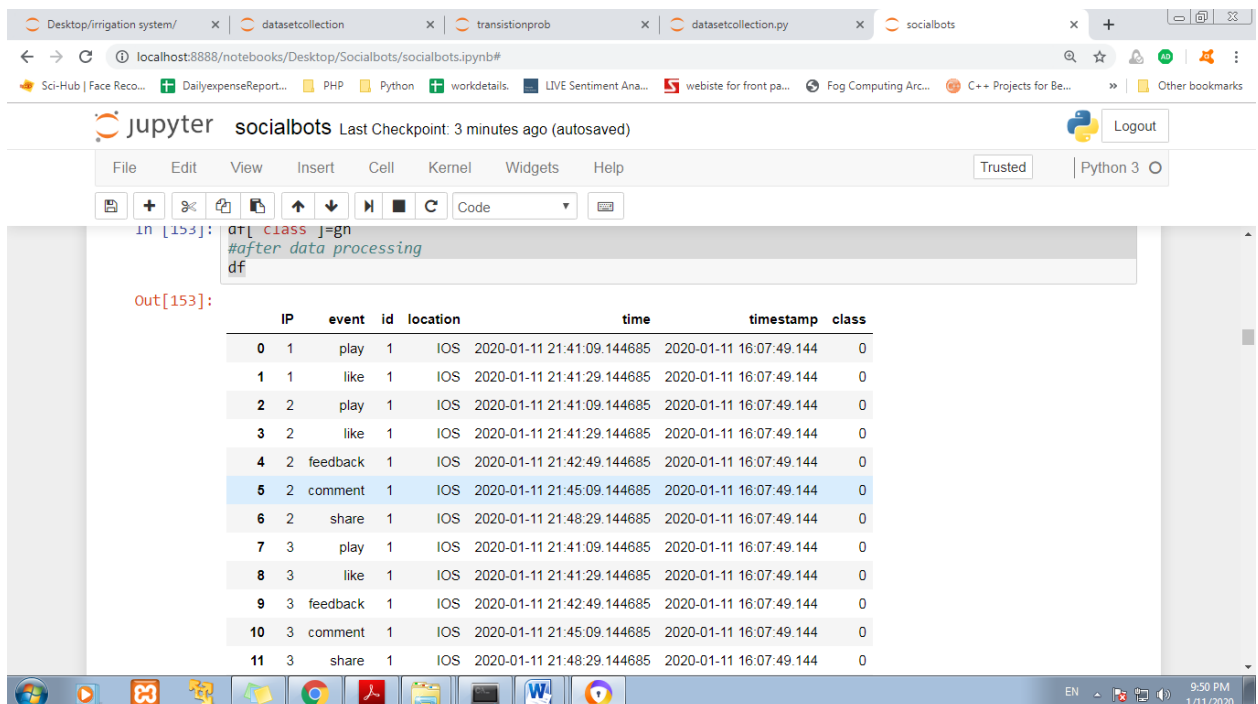
Dataset Collection:



The screenshot shows a Jupyter Notebook interface with a single code cell displaying a dataset. The dataset is a table with columns: IP, event, id, location, time, and timestamp. The data consists of 15 rows of social media-like interactions.

	IP	event	id	location	time	timestamp
0	1	play	1	IOS	2020-01-11 21:41:09.144685	2020-01-11 16:07:49.144
1	1	like	1	IOS	2020-01-11 21:41:29.144685	2020-01-11 16:07:49.144
2	2	play	1	IOS	2020-01-11 21:41:09.144685	2020-01-11 16:07:49.144
3	2	like	1	IOS	2020-01-11 21:41:29.144685	2020-01-11 16:07:49.144
4	2	feedback	1	IOS	2020-01-11 21:42:49.144685	2020-01-11 16:07:49.144
5	2	comment	1	IOS	2020-01-11 21:45:09.144685	2020-01-11 16:07:49.144
6	2	share	1	IOS	2020-01-11 21:48:29.144685	2020-01-11 16:07:49.144
7	3	play	1	IOS	2020-01-11 21:41:09.144685	2020-01-11 16:07:49.144
8	3	like	1	IOS	2020-01-11 21:41:29.144685	2020-01-11 16:07:49.144
9	3	feedback	1	IOS	2020-01-11 21:42:49.144685	2020-01-11 16:07:49.144
10	3	comment	1	IOS	2020-01-11 21:45:09.144685	2020-01-11 16:07:49.144
11	3	share	1	IOS	2020-01-11 21:48:29.144685	2020-01-11 16:07:49.144
12	4	play	1	IOS	2020-01-11 21:41:09.144685	2020-01-11 16:07:49.144
13	4	like	1	IOS	2020-01-11 21:41:29.144685	2020-01-11 16:07:49.144
14	4	feedback	1	IOS	2020-01-11 21:42:49.144685	2020-01-11 16:07:49.144

After Data Processing:



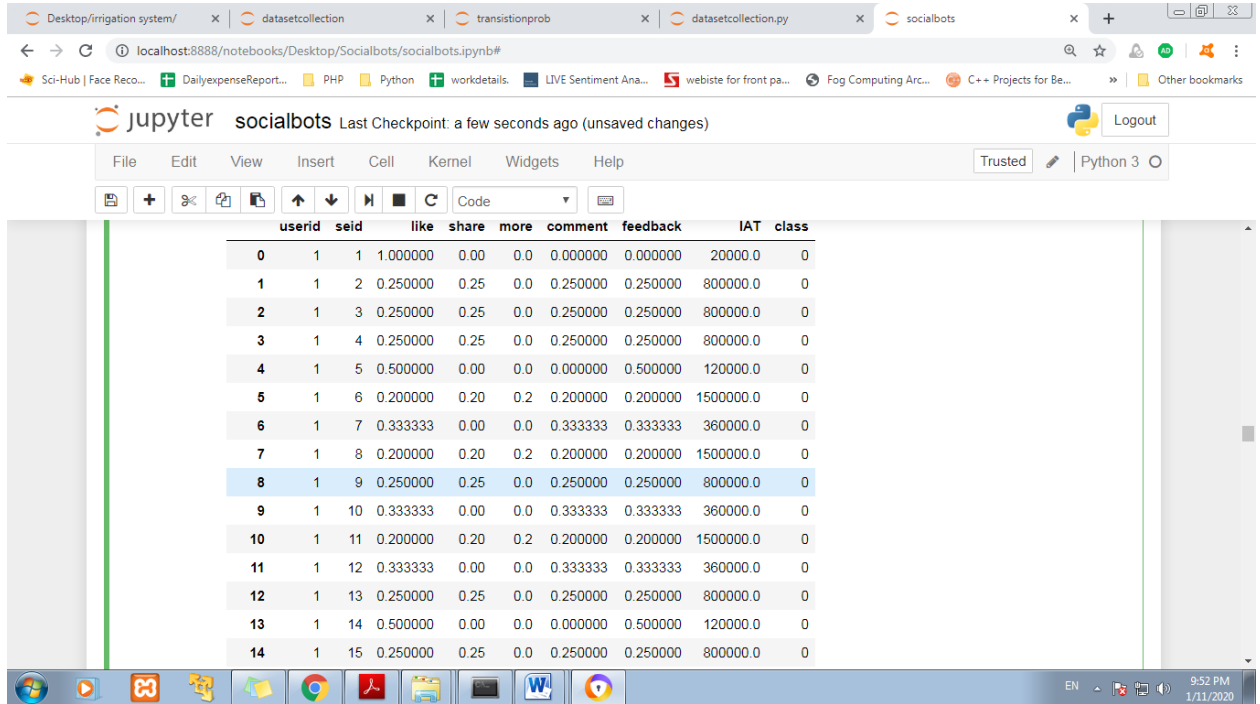
The screenshot shows a Jupyter Notebook interface with a code cell that has been executed. The code defines a class and processes the data. The output is a table with an additional 'class' column.

```
in [153]: or class = gn
          #after data processing
          df
```

Out[153]:

	IP	event	id	location	time	timestamp	class
0	1	play	1	IOS	2020-01-11 21:41:09.144685	2020-01-11 16:07:49.144	0
1	1	like	1	IOS	2020-01-11 21:41:29.144685	2020-01-11 16:07:49.144	0
2	2	play	1	IOS	2020-01-11 21:41:09.144685	2020-01-11 16:07:49.144	0
3	2	like	1	IOS	2020-01-11 21:41:29.144685	2020-01-11 16:07:49.144	0
4	2	feedback	1	IOS	2020-01-11 21:42:49.144685	2020-01-11 16:07:49.144	0
5	2	comment	1	IOS	2020-01-11 21:45:09.144685	2020-01-11 16:07:49.144	0
6	2	share	1	IOS	2020-01-11 21:48:29.144685	2020-01-11 16:07:49.144	0
7	3	play	1	IOS	2020-01-11 21:41:09.144685	2020-01-11 16:07:49.144	0
8	3	like	1	IOS	2020-01-11 21:41:29.144685	2020-01-11 16:07:49.144	0
9	3	feedback	1	IOS	2020-01-11 21:42:49.144685	2020-01-11 16:07:49.144	0
10	3	comment	1	IOS	2020-01-11 21:45:09.144685	2020-01-11 16:07:49.144	0
11	3	share	1	IOS	2020-01-11 21:48:29.144685	2020-01-11 16:07:49.144	0

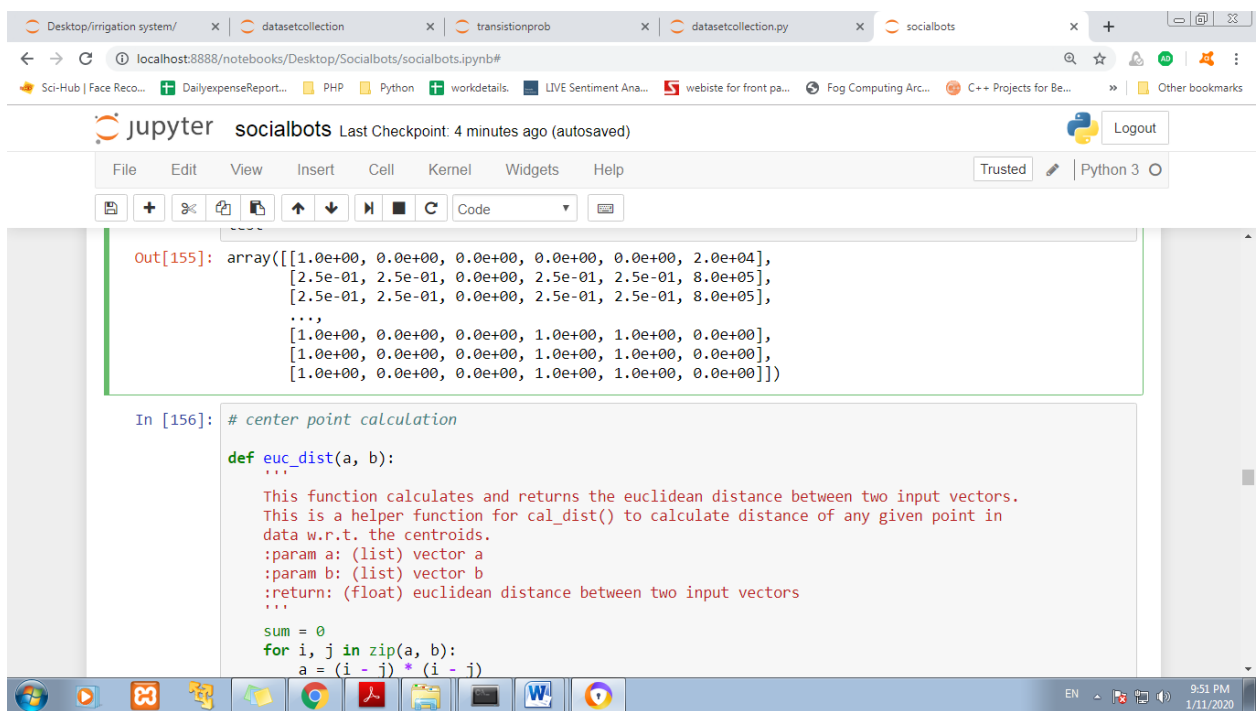
Transition Probability:



The screenshot shows a Jupyter Notebook titled 'socialbots' with a 'Last Checkpoint: a few seconds ago (unsaved changes)' status. The notebook is running on a local host (localhost:8888/notebooks/Desktop/Socialbots/socialbots.ipynb#). The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations and code execution. The main content area displays a table of data with the following columns: **userid**, **seid**, **like**, **share**, **more**, **comment**, **feedback**, **IAT**, and **class**. The table contains 15 rows of data, with the 8th row (index 8) highlighted in blue. The bottom status bar shows the system time as 9:52 PM on 1/11/2020.

	userid	seid	like	share	more	comment	feedback	IAT	class
0	1	1	1.000000	0.00	0.0	0.000000	0.000000	20000.0	0
1	1	2	0.250000	0.25	0.0	0.250000	0.250000	80000.0	0
2	1	3	0.250000	0.25	0.0	0.250000	0.250000	80000.0	0
3	1	4	0.250000	0.25	0.0	0.250000	0.250000	80000.0	0
4	1	5	0.500000	0.00	0.0	0.000000	0.500000	120000.0	0
5	1	6	0.200000	0.20	0.2	0.200000	0.200000	150000.0	0
6	1	7	0.333333	0.00	0.0	0.333333	0.333333	360000.0	0
7	1	8	0.200000	0.20	0.2	0.200000	0.200000	150000.0	0
8	1	9	0.250000	0.25	0.0	0.250000	0.250000	80000.0	0
9	1	10	0.333333	0.00	0.0	0.333333	0.333333	360000.0	0
10	1	11	0.200000	0.20	0.2	0.200000	0.200000	150000.0	0
11	1	12	0.333333	0.00	0.0	0.333333	0.333333	360000.0	0
12	1	13	0.250000	0.25	0.0	0.250000	0.250000	80000.0	0
13	1	14	0.500000	0.00	0.0	0.000000	0.500000	120000.0	0
14	1	15	0.250000	0.25	0.0	0.250000	0.250000	80000.0	0

Test data:



The screenshot shows a Jupyter Notebook titled 'socialbots' with a 'Last Checkpoint: 4 minutes ago (autosaved)' status. The notebook is running on a local host (localhost:8888/notebooks/Desktop/Socialbots/socialbots.ipynb#). The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations and code execution. The main content area displays two code cells. The first cell shows the output of a previous operation, displaying a 2D array of test data. The second cell shows a function definition for calculating the Euclidean distance between two input vectors.

```
Out[155]: array([[1.0e+00, 0.0e+00, 0.0e+00, 0.0e+00, 0.0e+00, 2.0e+04],
 [2.5e-01, 2.5e-01, 0.0e+00, 2.5e-01, 2.5e-01, 8.0e+05],
 [2.5e-01, 2.5e-01, 0.0e+00, 2.5e-01, 2.5e-01, 8.0e+05],
 ...,
 [1.0e+00, 0.0e+00, 0.0e+00, 1.0e+00, 1.0e+00, 0.0e+00],
 [1.0e+00, 0.0e+00, 0.0e+00, 1.0e+00, 1.0e+00, 0.0e+00],
 [1.0e+00, 0.0e+00, 0.0e+00, 1.0e+00, 1.0e+00, 0.0e+00]])

In [156]: # center point calculation

def euc_dist(a, b):
    """
    This function calculates and returns the euclidean distance between two input vectors.
    This is a helper function for cal_dist() to calculate distance of any given point in
    data w.r.t. the centroids.
    :param a: (list) vector a
    :param b: (list) vector b
    :return: (float) euclidean distance between two input vectors
    """
    sum = 0
    for i, j in zip(a, b):
        a = (i - j) * (i - j)
```

Prediction:

Desktop/irrigation system/ x datasetcollection x transitionprob x datasetcollection.py x socialbots

localhost:8888/notebooks/Desktop/Socialbots/socialbots.ipynb#

Sci-Hub | Face Reco... DailyexpenseReport... PHP Python workdetails. LIVE Sentiment Ana... website for front pa... Fog Computing Arc... C++ Projects for Be... Other bookmarks

jupyter socialbots Last Checkpoint: a few seconds ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

predict

```
Out[163]: [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1]
```

```
In [164]: from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score, classification_report
from sklearn.metrics import accuracy_score
predictions = predict
test_y = clabel
print("Accuracy :: ", accuracy_score(test_y, predictions))
print("precision :: ", precision_score(test_y, predictions, average="macro"))
print("Recall :: ", recall_score(test_y, predictions, average="macro"))
```

```
Accuracy :: 0.96
precision :: 0.9285714285714286
Recall :: 0.9736842105263157
```

```
In [165]: ccoun=0;
for jj in predictions:
```

EN 9:53 PM 1/11/2020

Desktop/irrigation system/ x datasetcollection x transitionprob x datasetcollection.py x socialbots

localhost:8888/notebooks/Desktop/Socialbots/socialbots.ipynb#

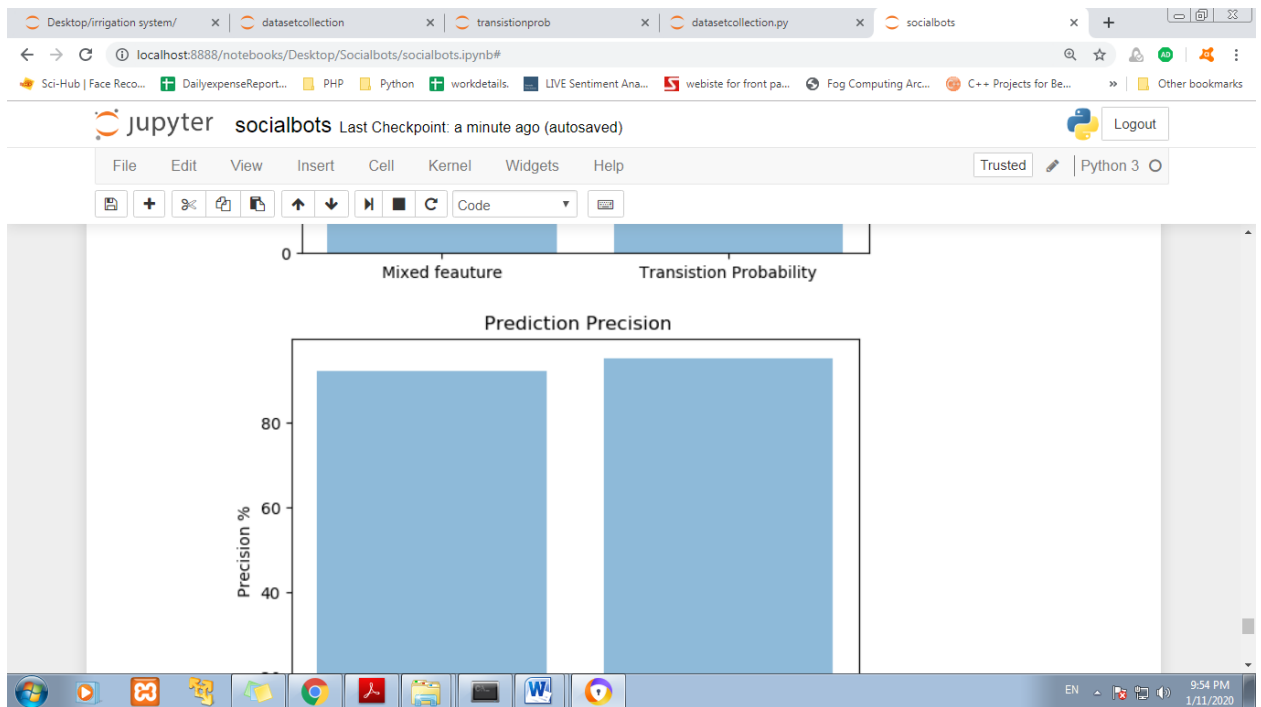
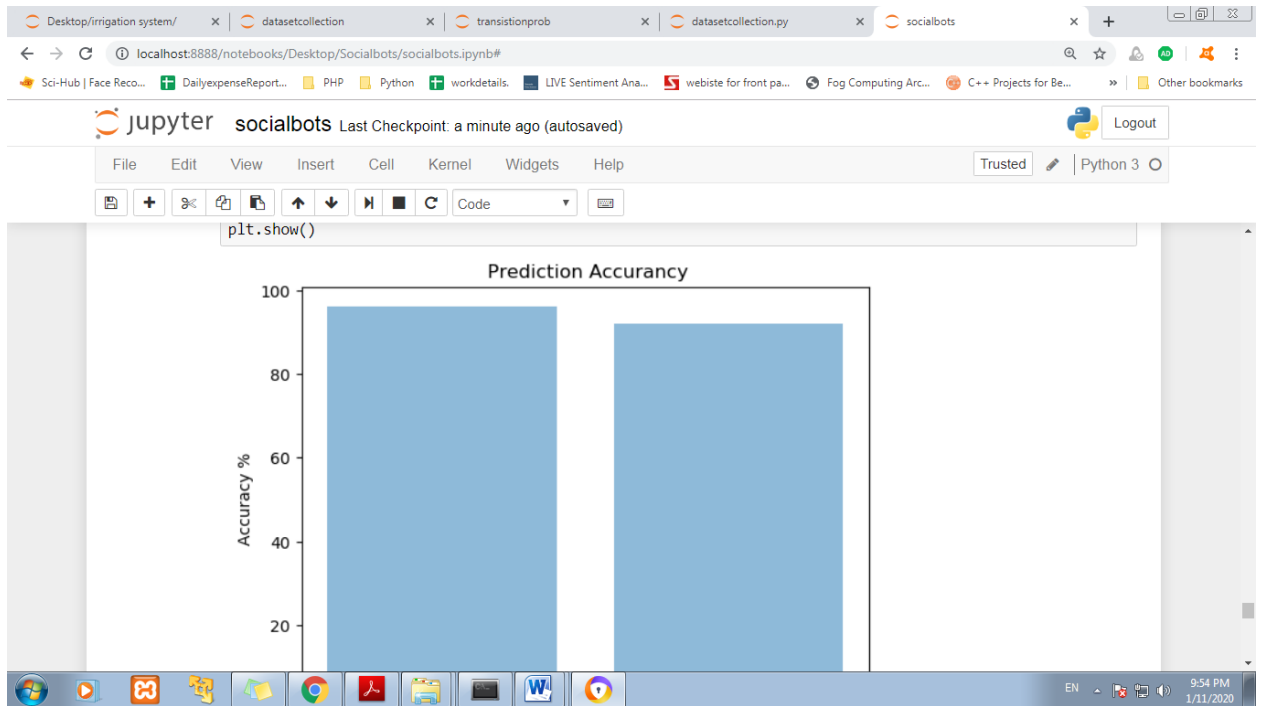
Sci-Hub | Face Reco... DailyexpenseReport... PHP Python workdetails. LIVE Sentiment Ana... website for front pa... Fog Computing Arc... C++ Projects for Be... Other bookmarks

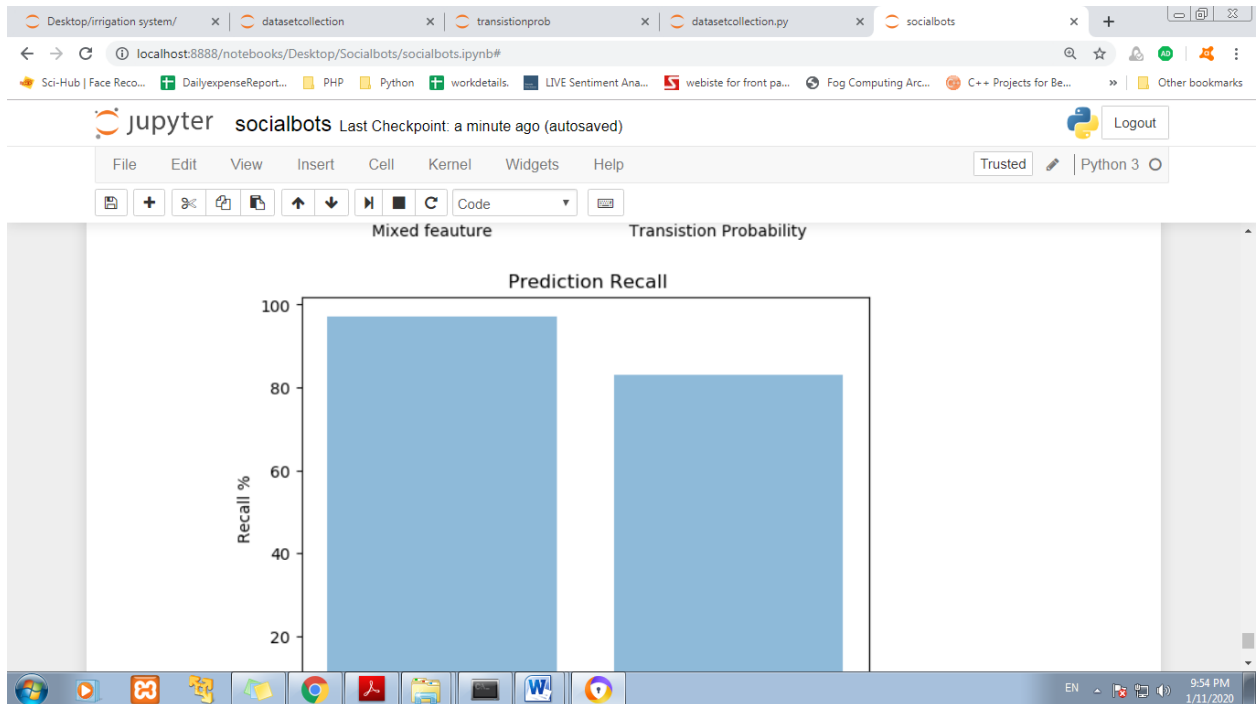
jupyter socialbots Last Checkpoint: a minute ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

```
User id 1 : Normal User
User id 2 : Normal User
User id 3 : Normal User
User id 4 : Normal User
User id 5 : Normal User
User id 6 : Normal User
User id 7 : Malicious Bot User
User id 8 : Normal User
User id 9 : Normal User
User id 10 : Normal User
User id 11 : Normal User
User id 12 : Normal User
User id 13 : Normal User
User id 14 : Normal User
User id 15 : Normal User
User id 16 : Normal User
User id 17 : Normal User
User id 18 : Normal User
User id 19 : Normal User
User id 20 : Malicious Bot User
User id 21 : Malicious Bot User
User id 22 : Malicious Bot User
User id 23 : Malicious Bot User
```

EN 9:54 PM 1/11/2020





Transition probability feature accuracy:

```
predictions = predict
test_y = clabel
print ("Accuracy :: ", accuracy_score(test_y, predictions))
print ("precision :: ", precision_score(test_y, predictions, average="macro"))
print ("Recall :: ", recall_score(test_y, predictions, average="macro"))

Accuracy :: 0.92
precision :: 0.9523809523809523
Recall :: 0.8333333333333333

In [60]: ccoun=0;
for jj in predictions:
    ccoun=ccoun+1;
    if(jj==0):
        print("User id "+str(ccoun)+ " : Normal User");
    else:
        print("User id "+str(ccoun)+ " : Malicious Bot User");

User id 1 : Normal User
User id 2 : Normal User
User id 3 : Normal User
User id 4 : Normal User
```

CHAPTER - 10

CONCLUSION

We proposed a novel method to accurately detect malicious social bots in online social networks. Experiments showed that transition probability between user clickstreams based on the social situation analytics can be used to detect malicious social bots in online social platforms accurately. In future research, additional behaviors of malicious social bots will be further considered and the proposed detection approach will be extended and optimized to identify specific intentions and purposes of a broader range of malicious social bots.

REFERENCES

- [1] Brito F, I. Petiz, P. Salvador, A. Nogueira, and E. Rocha, ``Detecting social network bots based on multiscale behavioral analysis," in *Proc.7th Int. Conf. Emerg. Secur. Inf., Syst. Technol. (SECURWARE)*, Barcelona, Spain, 2013, pp. 81_85
- [2] Barbon S, Jr., G. F. C. Campos, G. M. Tavares, R. A. Igawa, M. L. Proença, Jr., and R. C. Guido, ``Detection of human, legitimate bot, and malicious bot in online social networks based on wavelets," *ACM Trans. MultimediaComput., Commun., Appl.*, vol. 14, no. 1s, Feb. 2018, Art. no. 26
- [3] Cai C, L. Li, and D. Zengi, ``Behavior enhanced deep bot detection in social media," in *Proc. IEEE Int. Conf. Intell. Secur. Inform. (ISI)*, Beijing, China, Jul.2017, pp. 128_130.
- [4] Chang C K, ``Situation analytics: A foundation for a new software engineering paradigm," *Computer*, vol. 49, no. 1, pp. 24_33, Jan. 2016.
- [5] De Lima Salge C A and N. Berente, ``Is that social bot behaving unethically?" *Commun. ACM*, vol. 60, no. 9, pp. 29_31, Sep. 2017.
- [6] Ferrara E, O. Varol, C. Davis, F. Menczer, and A. Flammini, ``The rise of social bots," *Commun. ACM*, vol. 59, no. 7, pp. 96_104, Jul. 2016.
- [7] Gao H *et al.*, ``Spam ain't as diverse as it seems: Throttling OSN spam with templates underneath," in *Proc. 30th ACSAC*, New Orleans, LA, USA, 2014, pp. 76_85.
- [8] Huang T K, M. S. Rahman, H. V. Madhyastha, M. Faloutsos, and B. Ribeiro, ``An analysis of socware cascades in online social networks," in *Proc. 22nd Int. Conf. World Wide Web*, Rio de Janeiro, Brazil, 2013, pp. 619_630.
- [9] Hwang T, I. Pearce, and M. Nanis, ``Socialbots: Voices from the fronts," *Interactions*, vol. 19, no. 2, pp. 38_45, Mar. 2012.
- [10] Morstatter F, L. Wu, T. H. Nazer, K. M. Carley, and H. Liu, ``A new approach to bot detection: Striking the balance between precision and recall," in *Proc. IEEE/ACM*

Int. Conf. Adv. Social Netw. Anal. Mining, San Francisco, CA, USA, Aug. 2016, pp. 533_540.

[11] Park J Y, N. O'Hare, R. Schifanella, A. Jaimes, and C.-W. Chung, "A large scale study of user image search behavior on the Web," in *Proc.33rd Annu. ACM Conf. Hum. Factors Comput. Syst.*, Seoul, South Korea, 2015, pp. 985_994.

[12] Sahlabadi M, R. C. Muniyandi, and Z. Shukur, "Detecting abnormal behavior in social network Websites by using a process mining technique," *J. Comput. Sci.*, vol. 10, no. 3, pp. 393_402, 2014.

[13] Zhang Z, C. Li, B. B. Gupta, and D. Niu, "Efficient compressed ciphertext length scheme using multi-authority CP-ABE for hierarchical attributes," *IEEE Access*, vol. 6, pp. 38273_38284, 2018. doi: 10.1109/ACCESS.2018.2854600.

[14] Zhang Z, R. Sun, X. Wang, and C. Zhao, "A situational analytic method for user behavior pattern in multimedia social networks," *IEEE Trans. Big Data*, to be published. doi: 10.1109/TBDATA.2017.2657623.

[15] [13] Zhou Y *et al.*, "ProGuard: Detecting malicious accounts in socialnetwork-based online promotions," *IEEE Access*, vol. 5, pp. 1990_1999, 2017.

APPENDIX

A. SOURCE CODE

```
{
  "id": "25",
  "time": "2020-02-03 18:31:43.646518",
  "timestamp": "1580734803646",
  "event": "start",
  "IP": "1",
  "location": "IOS"
}

{
  "id": "25",
  "time": "2020-02-03 18:33:03.646518",
  "timestamp": "1580734803646",
  "event": "like",
  "IP": "1",
  "location": "IOS"
}

{
  "id": "25",
  "time": "2020-02-03 18:35:23.646518",
  "timestamp": "1580734803646",
  "event": "feedback",
  "IP": "1",
  "location": "IOS"
}

{
  "id": "25",
  "time": "2020-02-03 18:31:43.646518",
  "timestamp": "1580734803646",
  "event": "start",
  "IP": "2",
  "location": "IOS"
}

{
  "id": "25",
```

```
"time": "2020-02-03 18:33:03.646518",
"timestamp": "1580734803646",
"event": "like",
"IP": "2",
"location": "IOS"
}
{"id": "25",
"time": "2020-02-03 18:35:23.646518",
"timestamp": "1580734803646",
"event": "feedback",
"IP": "2",
"location": "IOS"
}
{"id": "25",
"time": "2020-02-03 18:31:43.646518",
"timestamp": "1580734803647",
"event": "start",
"IP": "3",
"location": "IOS"
}
{"id": "25",
"time": "2020-02-03 18:33:03.646518",
"timestamp": "1580734803647",
"event": "like",
"IP": "3",
"location": "IOS"
}
{"id": "25",
"time": "2020-02-03 18:35:23.646518",
"timestamp": "1580734803647",
"event": "feedback",
"IP": "3",
"location": "IOS"
}
```

```
}  
  {"id": "25",  
   "time": "2020-02-03 18:38:43.646518",  
   "timestamp": "1580734803647",  
   "event": "comment",  
   "IP": "3",  
   "location": "IOS".
```

B. PUBLICATION

MULTICLOUD COMPUTING ENVIRONMENTS : SECURITY ISSUES

Padmavathi G¹ , Albert Mayan J²

³P.G Student , Department of CSE , Sathyabama Institute of Science and Technology, Chennai.

²Assistant Professor , School of Computing, Sathyabama Institute of Science and Technology, Chennai.

keerjey@gmail.com , albertmayan@gmail.com

Abstract: Cloud computing have been trending in recent times because of the software , infrastructure and platform that is being offered without having a large expenditure being spent on it. Cloud computing involves the provision of services, provision and receiving of infrastructure and other services by the clients. On demand self service , broad network access , resource pooling, rapid elasticity , measured service are the essential features of cloud computing .In cloud computing large amount of data is being stored and accessed by multiple clients; computing can be auto released and provisioned and the data is metered to reduce the expenditure and restricted to usage only. The client obtains the required data based on requirement, clouds ensure high availability and the data is scalable based on user needs. The client requires maximum use of the available resources and this is easily done by the clouds, here the reconfigurations of the resources are being done and instant transfer of data is assured.. Hence cloud mashups provide a higher number of. All these provide a pooled data, instant sharing, storage and reconfiguration service for the mashups.

Keyword: multicloud, cloud computing, mashups, cloud service provider

1. INTRODUCTION

Cloud computing is a pay per utility model that enable shared configurable pool of computing resources such as network , servers , storage and applications[1]. Multiple cloud management provides a platform that would decrease the workload of the server helped in collaborative management and virtual server migration .Amazon simple storage service(S3) and Amazon elastic compute cloud(EC2) are the prominent cloud computing vendors. In India, company such as Ashok Leyland ,Tata , Asian paints and Maruti use cloud computing. Here, the client has the authorization to use multiple cloud service when using cloud mashups which helps in cost management and avoids vendor locking syndrome[2]. Motivated by attempting to find a solution, we will describe one of the most important modules, authentication and identity management and propose a solution to enhance security of the framework.A service in a certain CSP does not allow other CSP to use it without the required registration being done.

There are various risk factors that are involved in a multi cloud environment for instance privacy ,integrity and availability. The cloud computing is itself a very complex form of computing. When multiple cloud services are collaborated [3], the complexity increases. When using services provided by different clouds, there are varied models, techniques, rules, standards and regulations that are to be followed which increases the risk in collaboration of various services into a single application.

Agreements are necessary to establish willingness to collaborate and build trust among one another , failing to do so restricts the collaborative framework due to incompatible intentions , business rules and policies .Cloud service interactions, computing ,caching , routing are the various nodes that help in the work flow of multicloud.

II . RELATED WORK

The research community has started to develop infrastructure , technology and standard to support collaboration among multi-cloud systems. These research proposal have their limitations due to their provider-centric approach. For the cloud collaboration to be successful opportunistic collaboration among the services and neglecting the standards should be put into effect; allowing increased provisioning of the services to clients thereby increasing interoperability.

Security is a problem that is hard to tackle in a multi-cloud environment[2]. It still remains a major factor that limits the clients preferring multi-cloud environments over the rest.Replication of application helps in verifying data integrity once the data is executed ;logic is separated from the data ;data and applications are distributed by breaking application logic into parts and executing them over multi cloud platform.In the proposed system cloud ,collaboration is achieved by prior business agreements among cloud providers[5]. On behalf of the clients, the proxies are often found to access the data hence violating the multicloud environments. To this problem exhibiting secured data[6] to proxies and granting the authorization to access the data on behalf of clients is not an attractive solution.

III . PROPOSED SYSTEM

Our proposed framework for generic cloud collaboration allows clients and cloud applications to simultaneously use service from and route data among multiple clouds. This framework supports universal and dynamic collaboration in a multicloud system. It lets clients simultaneously use services from multiple clouds without prior business agreements among cloud providers, and without adopting common standards and specifications. As more organizations adopt cloud computing, cloud service providers (CSPs) are developing new technologies to enhance the cloud's capabilities. Cloud mashups are a recent trend; mashups combine services from multiple clouds into a single service or application, possibly with on-premises (client-side) data and services. This service composition lets CSPs offer new functionalities to clients at lower development costs.

3.1 System Architecture:

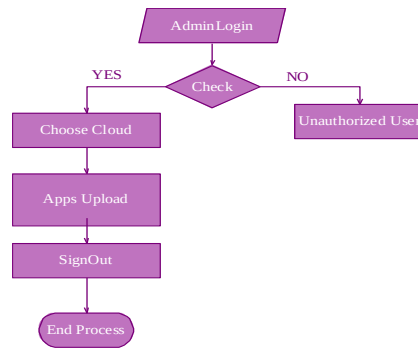


Fig1: Admin data flow diagram

3.2 Implementation

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective[8]. The implementation stage involves careful planning, investigation of the existing system and its constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

3.3 Working Procedure:

The proposed system is being divided into the various modules. The use of these various functionalities come at a lower expenditure than single service clouds as there is a decrease in expenditure for the development of these services when collaborated in a single cloud.

3.3.1. Collaboration Framework For Multicloud

System Module:

Here the clients have the access to the data across multiple clouds and have the authorization so as to use and execute the data as and where required for their requirements and needs.

3.3.2. Client/Users Module:

In the client or user module there exists a system where the client would need the access to multiple clouds[5]. The request is being provided to a single cloud and the routing is being done with the help of proxies being sent to multiple clouds and the required data could be exchanged based on the agreements[7] which were already carried out pre-request. The client has to individually collect the required information from the various clouds based on need but is exempted from seeking permission from individual clouds.

3.3.3. Cloud Service Provider Module:

Cloud service providers (CSPs) deploy proxies as an autonomous cloud system and offer it as a service to clients. A client employs two proxies to interact with CSPs C1 and C2. Alternatively, a client initiates a service request with C1, which then discovers the need for a service from C2. PSP: proxy service provider. Clients deploy proxies within the infrastructure of their organization. A client employs two proxies to interact with CSPs C1 and C2. A client initiates a service request with C1, which then discovers the need for a service from C2.

3.3.4. Proxy Service Provider Module:

It involves deploying proxies as an autonomous cloud that offers collaborative services to clients and CSPs. For inter-cloud collaboration the required proxies[9] that are being used for the interoperability of the clouds would be

deployed on the basis of their individual transfers across various clouds. Appropriate proxies are being used based on their operability.

4. Results and Discussions:

4.1 Screenings



Figure2: The collaboration of the multiclouds in Multicloud Environment

Clients directly subscribe to the proxy cloud service and employ them for intercloud collaboration[11]. To protect data at rest and data in transit, proxies must provide a trusted computing platform that prevents malicious software from taking control and compromising sensitive client and cloud application data.

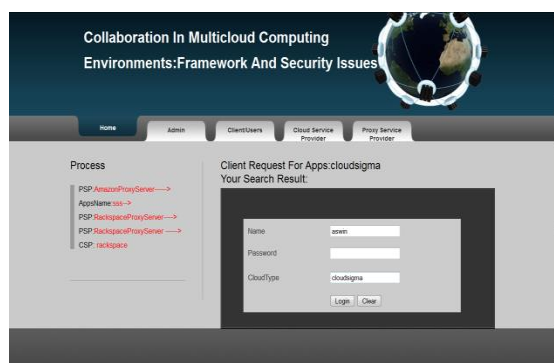


Figure3: Client request for apps in cloud sigma

The admin as in Fig 3 logs into multiple type of clouds such as cloudsigma, amazon and rackspace. The collaboration of the clouds are then being done. When the client requires the information from the other clouds when a request is being provided to the cloudsigma for instance, then the client has to collect data from s3 and rackspace individually in order to complete requirements but they could be void of separate permission seeking in each of them

V. Conclusion

In this paper, we have proposed an approach that identifies which part of intermediate datasets needs to be encrypted while the rest does not, in order to save the privacy-preserving cost. We have modelled the problem of saving privacy-preserving cost as a constrained optimization problem which is addressed by decomposing the privacy leakage constraints. A practical heuristic algorithm has been designed accordingly. Evaluation results on real-world datasets and larger extensive datasets have demonstrated the cost of preserving privacy in cloud can be reduced significantly with our approach over existing ones where all datasets are encrypted.

In accordance with various data and computation intensive applications on cloud, intermediate dataset management is becoming an important research area. Privacy pre-serving for intermediate datasets is one of important yet challenging research issues, and needs intensive investigation. With the contributions of this paper, we are planning to further investigate privacy-aware efficient scheduling of intermediate datasets in cloud by taking privacy preserving as a metric together with other metrics such as storage and computation. Optimized balanced scheduling strategies are expected to be developed towards overall highly efficient privacy aware dataset scheduling.

References

1. P. Mell and T. Grance, *The NIST Definition of Cloud Computing*, special publication 800-145, Nat'l Inst. Standards and Technology, 2011.

2. D. Bernstein and D. Vij, "Intercloud Security Considerations," *Proc. 2nd Int'l Conf. Cloud Computing*, IEEE Press, 2010, pp. 537-544.
3. Albert Mayan J, Dr. T. Ravi, "Optimized Regression Testing using Genetic Algorithm and Dependency Structure Matrix", *International Journal of Applied Engineering Research*, Vol:9, Issue:20, pp: 7679-7690, Nov 2014, ISSN: 1087-1090
4. Velmurugan. A, Ravi. T, "Optimal Symptom Diagnosis for Efficient Disease Identification Using Somars Approach", *Journal of Computational and Theoretical Nanoscience*, Vol.14, No.2, February 2017, pp. 1157-1162
5. B. Rochwerger et al., "Reservoir—When One Cloud Is Not Enough," *Computer*, 2011, pp. 44-51.
6. M.P. Papazoglou and W. van den Heuvel, "Blueprinting the Cloud," *IEEE Internet Computing*, 2011, pp. 74-79.
7. S. Dhamodaran, Albert Mayan J., N. Nagendra and M. Sundarajan, "Spatial interpolation of meteorological data and forecasting rainfall using ensemble techniques", *AIP Conference Proceedings* 2207, pp.050005, 2020.
8. S. Ortiz Jr., "The Problem with Cloud Computing Standardization," *Computer*, July 2011, pp. 13-16.
9. Ankayarkanni B, Albert Mayan J, Aruna J, "Support vector machine for effective robust visual tracking", *Journal of Computational and Theoretical Nanoscience*, Vol.18, No.8, pp.3571-3575.
10. Albert Mayan J, Kumar Akash Deep, Mukesh Kumar, Livingston Alvin, Siva Prasad Reddy (2016), "Number plate recognition using template comparison for various fonts in MATLAB", *IEEE International Conference on Computational Intelligence and Computing Research*, pp: 1-6.
11. Mary Posonia A, S. Vigneshwari, Albert Mayan J, "Service Direct : Platform that Incorporates Service Providers and Consumers Directly", *International Journal of Engineering and Advanced Technology*, Vol.8, No.6, August 2019.
12. W. Jansen and T. Grance, *Guidelines on Security and Privacy in Public Cloud Computing*, special publication 800-144, Nat'l Inst. Standards and Technology, 2011.
13. Sankari. A, Albert Mayan. J, "Retrieving call logs and SMS by messaging services", *International Journal of Pharmacy & Technology*, Vol. 8, Issue No.4, pp.22951-22958, Dec 2016.
14. C.M. Ellison et al., *SPKI Certificate Theory*, IETF RFC 2693, 1999, www.ietf.org/rfc/rfc2693.txt.
15. Albert Mayan J, Kuldeep Anand D.S, "Efficient and secure server migration on cloud storage with VSM and dropbox services", *International Conference on Information Communication and Embedded Systems*, pp. 1-5.
16. Eswar V.O.S, Vinil. B, Ankayarkanni B and Albert, "Integrated Collective Node Behavior Analysis with Onion Protocol for Best and Secured Data Transmission," 2018 *International Conference on Communication and Signal Processing (ICCSP)*, Chennai, 2018, pp. 0918-0921.
17. R. Buyya et al., "Market-Oriented Cloud Computing: Vision, Hype, and Reality of Delivering Computing as the 5th Utility," *Proc. 9th IEEE/ACM Int'l Symp. Cluster Computing and the Grid (CCGRID 09)*, IEEE CS, 2009, pp. 599-616.
18. Anish Hamlin M R, Albert Mayan J, "Blood donation and life saver-blood donation app", *International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, pp. 625 - 628, 2016.
19. Velmurugan A, Albert Mayan J, Mohana Prasad R, Yovan Felix A, "Implementing Health Care Center using Hadoop for Analysis and Prediction of Diseases", *International Journal of Engineering and Advanced Technology (IJEAT)*, Vol.8, No.6, pp.3456-3459, 2019.
20. Albert Mayan J, Lakshmi Priya K, Yovan Felix A, "Empirical study of GUI to reuse unusable test cases", *Journal of Theoretical and Applied Information Technology*, Vol.75, No.2, pp.186-192, 2015.