**DETECTION OF FAKE NEWS THROUGH IMPLEMENTATION OF DATA SCIENCE APPLICATION**

**1. INTRODUCTION:**

Online media deliver news stories to the public every day. These reports shape people’s perception of the ongoing social, political, and economical changes around them. Although numerous events are reported in the news every hour around the globe, only a few reported events attract enormous attention of the online media hundreds of news reports suddenly break out after the critical event happens. The burstiness of the reporting behaviour in the online media makes the prediction of which events will trigger viral news challenging. Many previous works model the information diffusion as epidemics in networks the acceptance of information is viewed as an infection of a node by infected the observed information diffusion messages can only spread along the predefined edges between nodes. Although the research shows that the news reports of an event are usually confined to the geographical and cultural boundaries between the global news sites, the explicit connections between pairs of individual news sites are typically unknown. Over defining the edges by assuming that the core porting relationship might exist between any two news sites would inevitably result in an extremely dense network and require the number of parameters in the order of square of the number of nodes. Therefore, we focus on modelling the news sites in the online media and propose a general probabilistic framework, which directly infer the affiliation of nodes with the communities. Although our model does not use the explicit network topology, the node clustering based on these affiliations matches the community structure detected by the traditional community detection algorithms in the explicit network topology. The early adopters of an information cascade, which are embedded in the so-inferred community structure, are used to predict the final cascade size. In the global online news media, news sites usually have a preference for the content of their reports due to the local regional reach. Although many media companies’ ambition is to have global market presence, most media sites have only a regional reach. This regional reach phenomenon is supported by the surveys of local newscasts that demonstrates the dominance of local news. According to the geographical and cultural boundaries among the global news sites, we model clustering of news sites as community structures, which are also widely observed in a variety of technological, biological, and social networks. In the context of information cascades, the community structures play an important role in facilitating the local spread of messages because the community members are more likely to accept inputs from each other than from the outsiders. On the other hand, the community structures slow the global diffusion by trapping the news in dense regions and thus preventing global penetration. The experimental results show that, compared with the machine learning models which extract point process-based features, our model exploiting the community-level signals improves the prediction accuracy by about 20%. We also parallelize the inference algorithm for distributed memory machines. The proposed parallelization scheme uses the standard message passing approach to exchange data between different processors. We design an asynchronous intercore communication paradigm so that the local computations occur simultaneously with the cores exchanging data with each other. We use MPI to implement and evaluate our parallelization scheme on the pet flops class IBM Blue Gene/Q supercomputer at the Rensselaer Polytechnic Institute (RPI). The algorithm gains orders of magnitude speedup inferring the parameters for the large networks, and it scales well with the number of nodes, the number of cascades, and the number of communities in a network. In , we introduced an initial work on the discovery of viral cascades based on their initial period observations. This paper was limited to sequential execution, and the algorithm for cascade likelihood evaluation had quadratic time complexity because the probability of infection between every pair of nodes in the same cascade was computed. In contrast, our new model assigns a set of parameters to each cascade, modelling the infections of nodes by the cascades instead of the infections between every pair of nodes. Therefore, evaluating the likelihood of a cascade in our new model has a linear time complexity and requires much less communication overhead than before when running the inference algorithm on distributed memory machines, making it scalable to large news media networks and a large number of cascades. To conclude, the major contributions of this paper are as follows: 1) a new modelling approach in which information diffusion is modelled at the community level instead of the level of the individual nodes, which reduces the computational complexity of the inference algorithm; 2) a parallelization scheme based on the message passing paradigm that infers the community structure without the explicit topology of the network, which makes it scalable, because it gains orders of magnitude speedup in computing the parameters for the large networks.

**1.1 Objective of the project:**

News reports shape the public perception of the critical social, political, and economical events around the world. Yet, the way in which emergent phenomena are reported in the news makes the early prediction of such phenomena a challenging task. We propose a scalable community-based probabilistic framework to model the spreading of news about events in online media. Our approach exploits the latent community structure in the global news media and uses the affiliation of the early adopters with a variety of communities to identify the events widely reported in the news at the early stage of their spread. The time complexity of our approach is linear in the number of news reports. It is also amenable to efficient parallelization. To demonstrate these features, the inference algorithm is parallelized for message passing paradigm and tested on the Rensselaer Polytechnic Institute Advanced Multiprocessing Optimized System, one of the fastest Blue Gene/Q supercomputers in the world. Thanks to the community-level features of the early adopters, the model gains an improvement of 20% in the early detection of the most massively reported events compared with the feature-based machine learning algorithm. Its parallelization scheme achieves orders of magnitude speedup.

**2. LITERETURE SURVEY:**

**“Optimal network modularity for information diffusion,”**

We investigate the impact of community structure on information diffusion with the linear threshold model. Our results demonstrate that modular structure may have counter-intuitive effects on information diffusion when social reinforcement is present. We show that strong communities can facilitate global diffusion by enhancing local, intra-community spreading. Using both analytic approaches and numerical simulations, we demonstrate the existence of an optimal network modularity, where global diffusion require the minimal number of early adopters.

**“The changing role of the local news media in enabling citizens to engage in local democracies,”**

Using Leeds City Council in the United Kingdom as a case study, we analyse comparatively the changing role of local journalism in the public communications and engagement strategies of local government. Drawing on over 20 semi-structured interviews with elected politicians, Council strategists, mainstream journalists, and citizen journalists, the article explores perceptions of the mainstream news media's role versus new modes of communication in engaging and communicating with citizens. We evaluate the Council's perceptions of its online and offline practices of engagement with different publics, and focus in particular on their interactions with journalists, the news media, and citizen journalists. The article considers how moves towards digital modes of engagement are changing perceptions of the professional role orientations of journalists in mediating between the Council and the general public.

**“Decline of a paradigm? Bias and objectivity in news media studies,”**

This essay outlines emerging empirical, methodological, and epistemolog‐ical challenges to several key assumptions associated with conventional research on news bias. These assumptions are: (1) the news can and ought to be objective, balanced and a reflection of social reality; (2) the political attitudes of journalists or editorial decision‐makers are a major determinant of news bias; (3) bias in news content can be detected with existing reading methods; (4) the most important form of bias is partisanship. It is concluded that the concepts of structured orientation and ideological effectiv‐ity are more fruitful than that of partisan bias, and that the concepts of bias and objectivity ought themselves to be objects of research rather than evaluative standards.

**“The Fox News effect: Media bias and voting,”**

Does media bias affect voting? We analyze the entry of Fox News in cable markets and its impact on voting. Between October 1996 and November 2000, the conservative Fox News Channel was introduced in the cable programming of 20 percent of U. S. towns. Fox News availability in 2000 appears to be largely idiosyncratic, conditional on a set of controls. Using a data set of voting data for 9,256 towns, we investigate if Republicans gained vote share in towns where Fox News entered the cable market by the year 2000. We find a significant effect of the introduction of Fox News on the vote share in Presidential elections between 1996 and 2000. Republicans gained 0.4 to 0.7 percentage points in the towns that broadcast Fox News. Fox News also affected voter turnout and the Republican vote share in the Senate. Our estimates imply that Fox News convinced 3 to 28 percent of its viewers to vote Republican, depending on the audience measure. The Fox News effect could be a temporary learning effect for rational voters, or a permanent effect for nonrational voters subject to persuasion.

**“Media bias and reputation,”**

A Bayesian consumer who is uncertain about the quality of an information source will infer that the source is of higher quality when its reports conform to the consumer’s prior expectations. We use this fact to build a model of media bias in which firms slant their reports toward the prior beliefs of their customers in order to build a reputation for quality. Bias emerges in our model even though it can make all market participants worse off. The model predicts that bias will be less severe when consumers receive independent evidence on the true state of the world and that competition between independently owned news outlets can reduce bias. We present a variety of empirical evidence consistent with these predictions. We are extremely grateful to an anonymous referee for thorough and insightful comments on an earlier draft of this paper. We also thank Alberto Alesina, Attila Ambrus, Nigel Ashford, Chris Avery, Heski Bar-Isaac, Gary Becker, Tyler Cowen, Jonathan Feinstein, Jeremy Fox, Drew Fudenberg, Ed Glaeser, Jerry Green, James Heckman, Tom Hubbard, Steve Levitt, Larry Katz, Kevin M. Murphy, Roger Myerson, Canice Prendergast, Matthew Rabin, Andrei Shleifer, Lars Stole, Richard Thaler, Richard Zeckhauser, and seminar participants at Harvard University, the University of Chicago, the Institute for Humane Studies, and the University of British Columbia for helpful comments. We thank Christopher Avery, Judith Chevalier, Matthew Hale, Martin Kaplan, Bryan Boulier, and H. O. Stekler for generously providing access to their data. Karen Bernhardt, Fuhito Kojima, Jennifer Paniza, and Tina Yang provided excellent research assistance. Gentzkow acknowledges financial assistance from the Social Science Research Council and the Centel Foundation/Robert P. Reuss Faculty Research Fund. Shapiro acknowledges financial assistance from the Institute for Humane Studies, the Center for Basic Research in the Social Sciences, the Chiles Foundation, and the National Science Foundation.

**“Small but slow world: How network topology and burstiness slow down spreading,”**

While communication networks show the small-world property of short paths, the spreading dynamics in them turns out slow. Here, the time evolution of information propagation is followed through communication networks by using empirical data on contact sequences and the susceptible-infected model. Introducing null models where event sequences are appropriately shuffled, we are able to distinguish between the contributions of different impeding effects. The slowing down of spreading is found to be caused mainly by weight-topology correlations and the bursty activity patterns of individuals.

**“Prime suspects: The influence of local television news on the viewing public,”**

Local television news is the public’s primary source of public affairs information. News stories about crime dominate local news programming because they meet the demand for “action news.”. The prevalence of this type of reporting has led to a crime narrative or “script” that includes two core elements: crime is violent and perpetrators of crime are non-white males. We show that this script has become an ingrained heuristic for understanding crime and race. Using a multi-method design, we assess the impact of the crime script on the viewing public. Our central finding is that exposure to the racial element of the crime script increases support for punitive approaches to crime and heightens negative attitudes about African-Americans among white, but not black, viewers. In closing, we consider the implications of our results for intergroup relations, electoral politics, and the practice of journalism.

**“Community structure in social and biological networks,”**

A number of recent studies have focused on the statistical properties of networked systems such as social networks and the Worldwide Web. Researchers have concentrated particularly on a few properties that seem to be common to many networks: the small-world property, power-law degree distributions, and network transitivity. In this article, we highlight another property that is found in many networks, the property of community structure, in which network nodes are joined together in tightly knit groups, between which there are only looser connections. We propose a method for detecting such communities, built around the idea of using centrality indices to find community boundaries. We test our method on computer-generated and real-world graphs whose community structure is already known and find that the method detects this known structure with high sensitivity and reliability. We also apply the method to two networks whose community structure is not well known—a collaboration network and a food web—and find that it detects significant and informative community divisions in both cases.

**“Overlapping community detection at scale: A nonnegative matrix factorization approach,”**

Network communities represent basic structures for understanding the organization of real-world networks. A community (also referred to as a module or a cluster) is typically thought of as a group of nodes with more connections amongst its members than between its members and the remainder of the network. Communities in networks also overlap as nodes belong to multiple clusters at once. Due to the difficulties in evaluating the detected communities and the lack of scalable algorithms, the task of overlapping community detection in large networks largely remains an open problem. In this paper we present BIGCLAM (Cluster Affiliation Model for Big Networks), an overlapping community detection method that scales to large networks of millions of nodes and edges. We build on a novel observation that overlaps between communities are densely connected. This is in sharp contrast with present community detection methods which implicitly assume that overlaps between communities are sparsely connected and thus cannot properly extract overlapping communities in networks. In this paper, we develop a model-based community detection algorithm that can detect densely overlapping, hierarchically nested as well as non-overlapping communities in massive networks. We evaluate our algorithm on 6 large social, collaboration and information networks with ground-truth community information. Experiments show state of the art performance both in terms of the quality of detected communities as well as in speed and scalability of our algorithm.

**“Parallelizing word2vec in shared and distributed memory.”**

Word2Vec is a widely used algorithm for extracting low-dimensional vector representations of words. It generated considerable excitement in the machine learning and natural language processing (NLP) communities recently due to its exceptional performance in many NLP applications such as named entity recognition, sentiment analysis, machine translation and question answering. State-of-the-art algorithms including those by Mikolov et al. have been parallelized for multi-core CPU architectures but are based on vector-vector operations that are memory-bandwidth intensive and do not efficiently use computational resources. In this paper, we improve reuse of various data structures in the algorithm through the use of minibatching, hence allowing us to express the problem using matrix multiply operations. We also explore different techniques to distribute word2vec computation across nodes in a compute cluster, and demonstrate good strong scalability up to 32 nodes. In combination, these techniques allow us to scale up the computation near linearly across cores and nodes, and process hundreds of millions of words per second, which is the fastest word2vec implementation to the best of our knowledge.

**3. SYSTEM ANALYSIS**

**3.1 Existing System:**

News reports shape the public perception of the critical social, political, and economical events around the world. Yet, the way in which emergent phenomena are reported in the news makes the early prediction of such phenomena a challenging task.

**Disadvantage:**

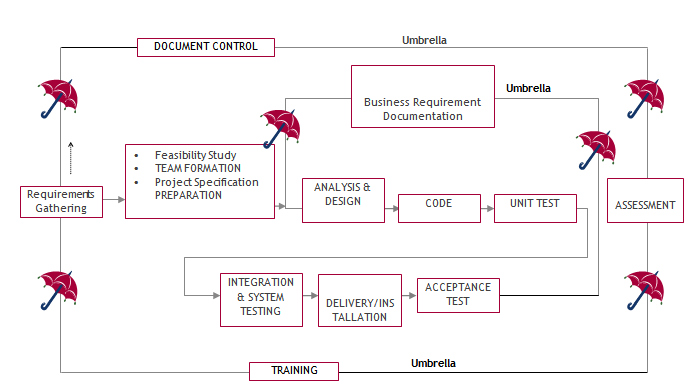
1. Prediction is less.

**3.2. Proposed System:**

We propose a scalable community-based probabilistic framework to model the spreading of news about events in online media. Our approach exploits the latent community structure in the global news media and uses the affiliation of the early adopters with a variety of communities to identify the events widely reported in the news at the early stage of their spread. The time complexity of our approach is linear in the number of news reports. It is also amenable to efficient parallelization. To demonstrate these features, the inference algorithm is parallelized for message passing paradigm and tested on the Rensselaer Polytechnic Institute Advanced Multiprocessing Optimized System, one of the fastest Blue Gene/Q supercomputers in the world. Thanks to the community-level features of the early adopters, the model gains an improvement of 20% in the early detection of the most massively reported events compared with the feature-based machine learning algorithm. Its parallelization scheme achieves orders of magnitude speedup.

**Advantage:**

1.Prediction is more.

**3.3. PROCESS MODEL USED WITH JUSTIFICATIONSDLC (Umbrella Model): **

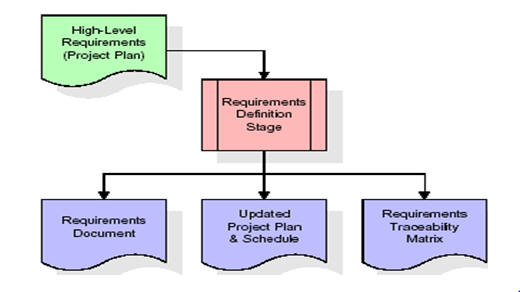
The requirements gathering process takes as its input SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

**Stages in SDLC:**

* Requirement Gathering
* Analysis
* Designing
* Coding
* Testing
* Maintenance

**Requirements Gathering** **stage:**

the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.



These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

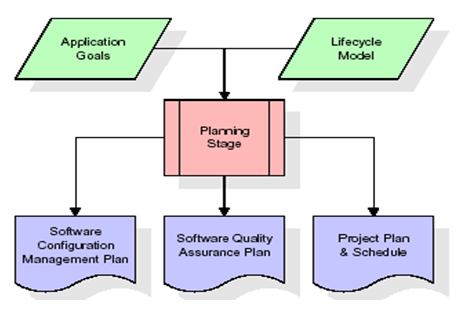
In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

* Feasibility study is all about identification of problems in a project.
* No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
* Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

**Analysis Stage:**

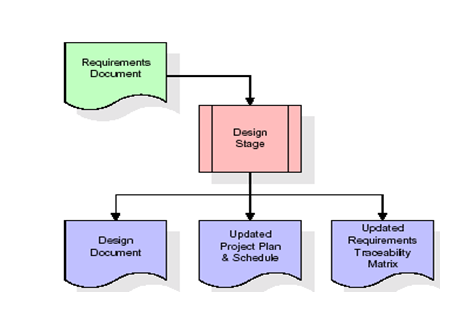
The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.



The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

**Designing Stage:**

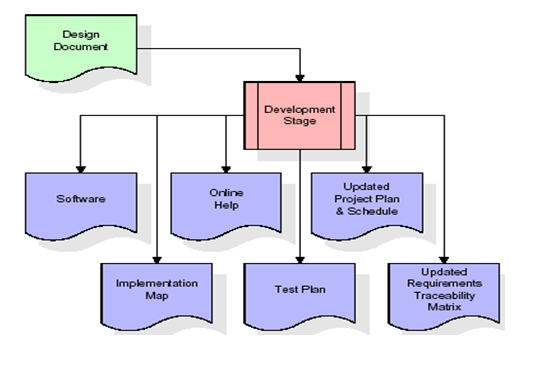
The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.



When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

**Development (Coding) Stage:**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.



The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

**Integration & Test Stage:**

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.



The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

* **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

**Maintenance:**

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

**3.4. Software Requirement Specification**

**3.4.1. Overall Description**

A Software Requirements Specification (SRS) – a [requirements specification](http://en.wikipedia.org/wiki/Requirements_specification) for a [software system](http://en.wikipedia.org/wiki/Software_system) is a complete description of the behavior of a system to be developed. It includes a set of [use cases](http://en.wikipedia.org/wiki/Use_case) that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. [Nonfunctional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements) are requirements which impose constraints on the design or implementation (such as [performance engineering](http://en.wikipedia.org/wiki/Performance_engineering) requirements, [quality](http://en.wikipedia.org/wiki/Quality_%28business%29) standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A [business analyst](http://en.wikipedia.org/wiki/Business_analyst), sometimes titled [system analyst](http://en.wikipedia.org/wiki/System_analyst), is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the [systems development lifecycle](http://en.wikipedia.org/wiki/Systems_development_life_cycle) domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

* [Business requirements](http://en.wikipedia.org/wiki/Business_requirements) describe in business terms *what* must be delivered or accomplished to provide value.
* Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
* Process requirements describe activities performed by the developing organization. For instance, process requirements could specify .Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:
* **ECONOMIC FEASIBILITY**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

* **Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization’s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would

ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

* **TECHNICAL FEASIBILITY**

Earlier no system existed to cater to the needs of ‘Secure Infrastructure Implementation System’. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to .the users. The database’s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

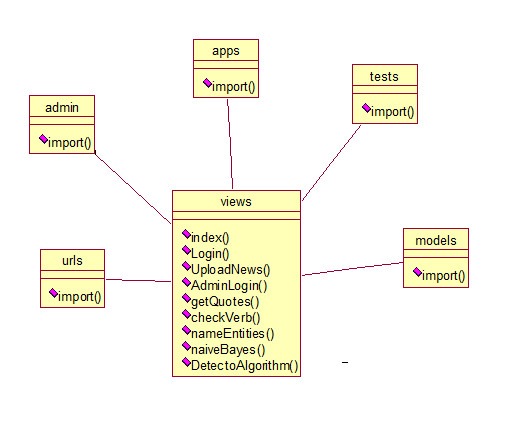
**4. SYSTEM DESIGN**

**UML Diagram:**

**Class Diagram:**

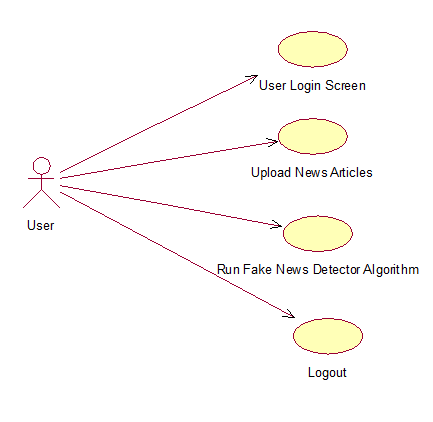
The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

* The upper part holds the name of the class
* The middle part contains the attributes of the class
* The bottom part gives the methods or operations the class can take or undertake.



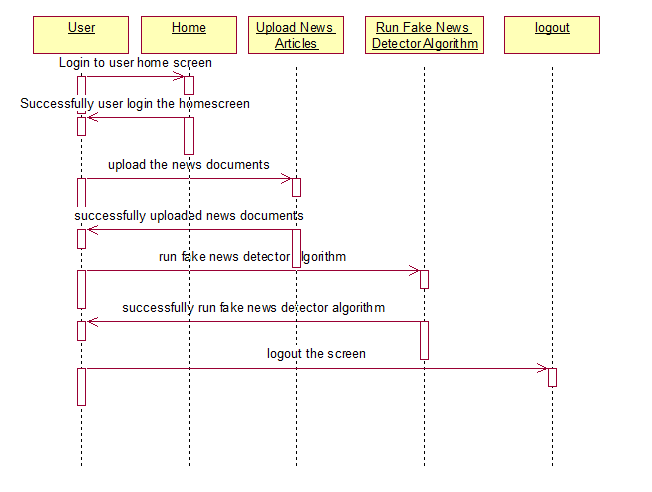
**Use case Diagram:**

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.



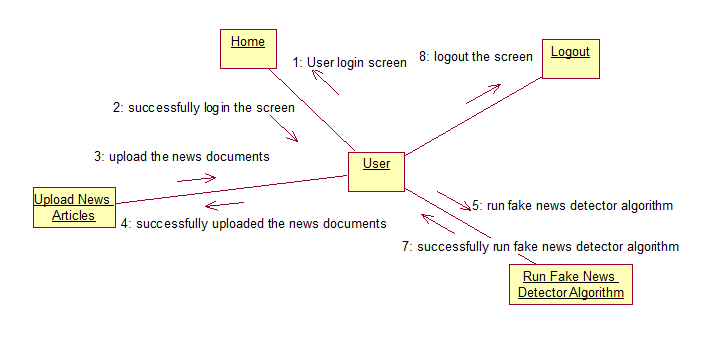
**Sequence Diagram:**

A sequence diagram 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. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



**Collaboration diagram:**

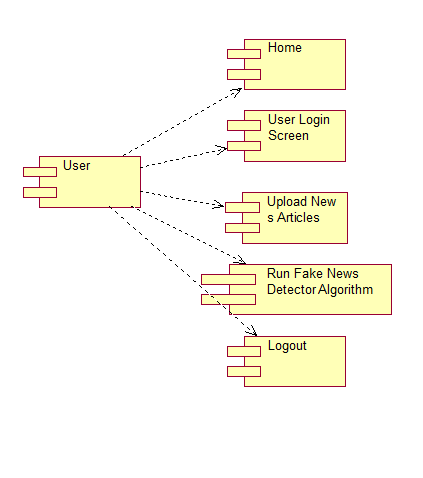
A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behavior of a system.



**Component Diagram:**

In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

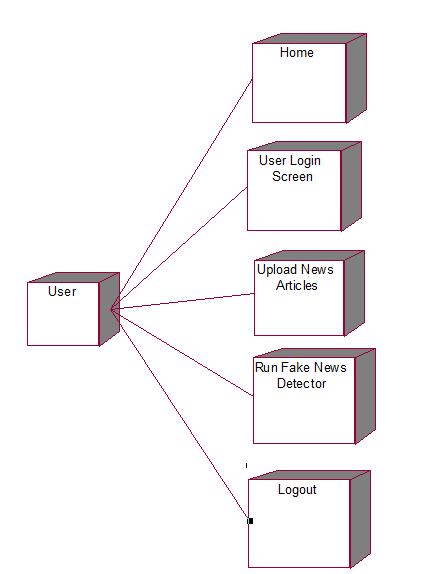
Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.



**Deployment Diagram:**

A deployment diagram in the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.



**5. IMPLEMETATION:**

**5.1 Python**

Python is a general-purpose language. It has wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean and length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

**History of Python:**

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

**Why Python was created?**

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to-understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

**Why the name Python?**

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

**Features of Python:**

**A simple language which is easier to learn**

Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like: C++, Java, C#. Python makes programming fun and allows you to focus on the solution rather than syntax.

If you are a newbie, it's a great choice to start your journey with Python.

**Free and open-source**

You can freely use and distribute Python, even for commercial use. Not only can you use and distribute softwares written in it, you can even make changes to the Python's source code.

Python has a large community constantly improving it in each iteration.

**Portability**

You can move Python programs from one platform to another, and run it without any changes.

It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.

**Extensible and Embeddable**

Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code.

This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

**A high-level, interpreted language**

Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on.

Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

**Large standard libraries to solve common tasks**

Python has a number of standard libraries which makes life of a programmer much easier since you don't have to write all the code yourself. For example: Need to connect MySQL database on a Web server? You can use MySQLdb library using import MySQLdb .

Standard libraries in Python are well tested and used by hundreds of people. So you can be sure that it won't break your application.

**Object-oriented**

Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively.

With OOP, you are able to divide these complex problems into smaller sets by creating objects.

**Applications of Python:**

**1. Simple Elegant Syntax**

Programming in Python is fun. It's easier to understand and write Python code. Why? The syntax feels natural. Take this source code for an example:

a = 2

b = 3

sum = a + b

print(sum)

**2. Not overly strict**

You don't need to define the type of a variable in Python. Also, it's not necessary to add semicolon at the end of the statement.

Python enforces you to follow good practices (like proper indentation). These small things can make learning much easier for beginners.

**3. Expressiveness of the language**

Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

**4. Great Community and Support**

Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck.

**5.2 Sample Code:**

**FakeNews.py**

from tkinter import messagebox

from tkinter import \*

from tkinter import simpledialog

import tkinter

import matplotlib.pyplot as plt

import numpy as np

from tkinter import ttk

from tkinter import filedialog

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from string import punctuation

from nltk.corpus import stopwords

import nltk

from nltk.stem import WordNetLemmatizer

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.preprocessing import LabelEncoder

from keras.models import Sequential

from keras.layers.core import Dense,Activation,Dropout

from sklearn.preprocessing import OneHotEncoder

import keras.layers

from keras.models import model\_from\_json

import pickle

import os

from sklearn.preprocessing import normalize

from keras.models import Sequential

from keras.layers import Dense, Dropout, Flatten, LSTM

main = Tk()

main.title("DETECTION OF FAKE NEWS THROUGH IMPLEMENTATION OF DATA SCIENCE APPLICATION")

main.geometry("1300x1200")

global filename

global X, Y

global tfidf\_X\_train, tfidf\_X\_test, tfidf\_y\_train, tfidf\_y\_test

global tfidf\_vectorizer

global accuracy,error

stop\_words = set(stopwords.words('english'))

lemmatizer = WordNetLemmatizer()

textdata = []

labels = []

global classifier

def cleanPost(doc):

tokens = doc.split()

table = str.maketrans('', '', punctuation)

tokens = [w.translate(table) for w in tokens]

tokens = [word for word in tokens if word.isalpha()]

tokens = [w for w in tokens if not w in stop\_words]

tokens = [word for word in tokens if len(word) > 1]

tokens = [lemmatizer.lemmatize(token) for token in tokens]

tokens = ' '.join(tokens)

return tokens

def uploadDataset():

global filename

text.delete('1.0', END)

filename = filedialog.askopenfilename(initialdir="TwitterNewsData")

textdata.clear()

labels.clear()

dataset = pd.read\_csv(filename)

dataset = dataset.fillna(' ')

for i in range(len(dataset)):

msg = dataset.get\_value(i, 'text')

label = dataset.get\_value(i, 'target')

msg = str(msg)

msg = msg.strip().lower()

labels.append(int(label))

clean = cleanPost(msg)

textdata.append(clean)

text.insert(END,clean+" ==== "+str(label)+"\n")

def preprocess():

text.delete('1.0', END)

global X, Y

global tfidf\_vectorizer

global tfidf\_X\_train, tfidf\_X\_test, tfidf\_y\_train, tfidf\_y\_test

stopwords=stopwords = nltk.corpus.stopwords.words("english")

tfidf\_vectorizer = TfidfVectorizer(stop\_words=stopwords, use\_idf=True, ngram\_range=(1,2),smooth\_idf=False, norm=None, decode\_error='replace', max\_features=200)

tfidf = tfidf\_vectorizer.fit\_transform(textdata).toarray()

df = pd.DataFrame(tfidf, columns=tfidf\_vectorizer.get\_feature\_names())

text.insert(END,str(df))

print(df.shape)

df = df.values

X = df[:, 0:df.shape[1]]

X = normalize(X)

Y = np.asarray(labels)

le = LabelEncoder()

Y = le.fit\_transform(Y)

indices = np.arange(X.shape[0])

np.random.shuffle(indices)

X = X[indices]

Y = Y[indices]

Y = Y.reshape(-1, 1)

print(X.shape)

encoder = OneHotEncoder(sparse=False)

#Y = encoder.fit\_transform(Y)

X = X.reshape((X.shape[0], X.shape[1], 1))

print(Y)

print(Y.shape)

print(X.shape)

tfidf\_X\_train, tfidf\_X\_test, tfidf\_y\_train, tfidf\_y\_test = train\_test\_split(X, Y, test\_size=0.2)

text.insert(END,"\n\nTotal News found in dataset : "+str(len(X))+"\n")

text.insert(END,"Total records used to train machine learning algorithms : "+str(len(tfidf\_X\_train))+"\n")

text.insert(END,"Total records used to test machine learning algorithms : "+str(len(tfidf\_X\_test))+"\n")

def runLSTM():

text.delete('1.0', END)

global classifier

if os.path.exists('model/model.json'):

with open('model/model.json', "r") as json\_file:

loaded\_model\_json = json\_file.read()

classifier = model\_from\_json(loaded\_model\_json)

classifier.load\_weights("model/model\_weights.h5")

classifier.\_make\_predict\_function()

print(classifier.summary())

f = open('model/history.pckl', 'rb')

data = pickle.load(f)

f.close()

acc = data['accuracy']

acc = acc[9] \* 100

text.insert(END,"LSTM Fake News Detection Accuracy : "+str(acc)+"\n\n")

text.insert(END,'LSTM Model Summary can be seen in black console for layer details\n')

with open('model/model.txt', 'rb') as file:

classifier = pickle.load(file)

file.close()

else:

lstm\_model = Sequential()

lstm\_model.add(LSTM(128, input\_shape=(X.shape[1:]), activation='relu', return\_sequences=True))

lstm\_model.add(Dropout(0.2))

lstm\_model.add(LSTM(128, activation='relu'))

lstm\_model.add(Dropout(0.2))

lstm\_model.add(Dense(32, activation='relu'))

lstm\_model.add(Dropout(0.2))

lstm\_model.add(Dense(2, activation='softmax'))

lstm\_model.compile(loss='sparse\_categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])

hist = lstm\_model.fit(X, Y, epochs=10, validation\_data=(tfidf\_X\_test, tfidf\_y\_test))

classifier = lstm\_model

classifier.save\_weights('model/model\_weights.h5')

model\_json = classifier.to\_json()

with open("model/model.json", "w") as json\_file:

json\_file.write(model\_json)

accuracy = hist.history

f = open('model/history.pckl', 'wb')

pickle.dump(accuracy, f)

f.close()

acc = accuracy['accuracy']

acc = acc[9] \* 100

text.insert(END,"LSTM Accuracy : "+str(acc)+"\n\n")

text.insert(END,'LSTM Model Summary can be seen in black console for layer details\n')

print(lstm\_model.summary())

def graph():

f = open('model/history.pckl', 'rb')

data = pickle.load(f)

f.close()

acc = data['accuracy']

loss = data['loss']

plt.figure(figsize=(10,6))

plt.grid(True)

plt.xlabel('Epcchs')

plt.ylabel('Accuracy/Loss')

plt.plot(acc, 'ro-', color = 'green')

plt.plot(loss, 'ro-', color = 'blue')

plt.legend(['Accuracy','Loss'], loc='upper left')

#plt.xticks(wordloss.index)

plt.title('LSTM Model Accuracy & Loss Graph')

plt.show()

def predict():

testfile = filedialog.askopenfilename(initialdir="TwitterNewsData")

testData = pd.read\_csv(testfile)

text.delete('1.0', END)

testData = testData.values

testData = testData[:,0]

print(testData)

for i in range(len(testData)):

msg = testData[i]

msg1 = testData[i]

print(msg)

review = msg.lower()

review = review.strip().lower()

review = cleanPost(review)

testReview = tfidf\_vectorizer.transform([review]).toarray()

predict = classifier.predict(testReview)

print(predict)

if predict == 0:

text.insert(END,msg1+" === Given news predicted as GENUINE\n\n")

else:

text.insert(END,msg1+" == Given news predicted as FAKE\n\n")

font = ('times', 15, 'bold')

title = Label(main, text='DETECTION OF FAKE NEWS THROUGH IMPLEMENTATION OF DATA SCIENCE APPLICATION')

title.config(bg='gold2', fg='thistle1')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 13, 'bold')

ff = ('times', 12, 'bold')

uploadButton = Button(main, text="Upload Fake News Dataset", command=uploadDataset)

uploadButton.place(x=20,y=100)

uploadButton.config(font=ff)

processButton = Button(main, text="Preprocess Dataset", command=preprocess)

processButton.place(x=20,y=150)

processButton.config(font=ff)

dtButton = Button(main, text="Run LSTM Algorithm", command=runLSTM)

dtButton.place(x=20,y=200)

dtButton.config(font=ff)

graphButton = Button(main, text="Accuracy & Loss Graph", command=graph)

graphButton.place(x=20,y=250)

graphButton.config(font=ff)

predictButton = Button(main, text="Test News Detection", command=predict)

predictButton.place(x=20,y=300)

predictButton.config(font=ff)

font1 = ('times', 12, 'bold')

text=Text(main,height=30,width=100)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=330,y=100)

text.config(font=font1)

main.config(bg='DarkSlateGray1')

main.mainloop()

**6. TESTING**

**Implementation and Testing:**

Implementation is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

## 

## Implementation

## The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifies as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

## Testing

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

**System Testing**

Testing has become an System integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to user the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

**Module Testing**

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

**Integration Testing**

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

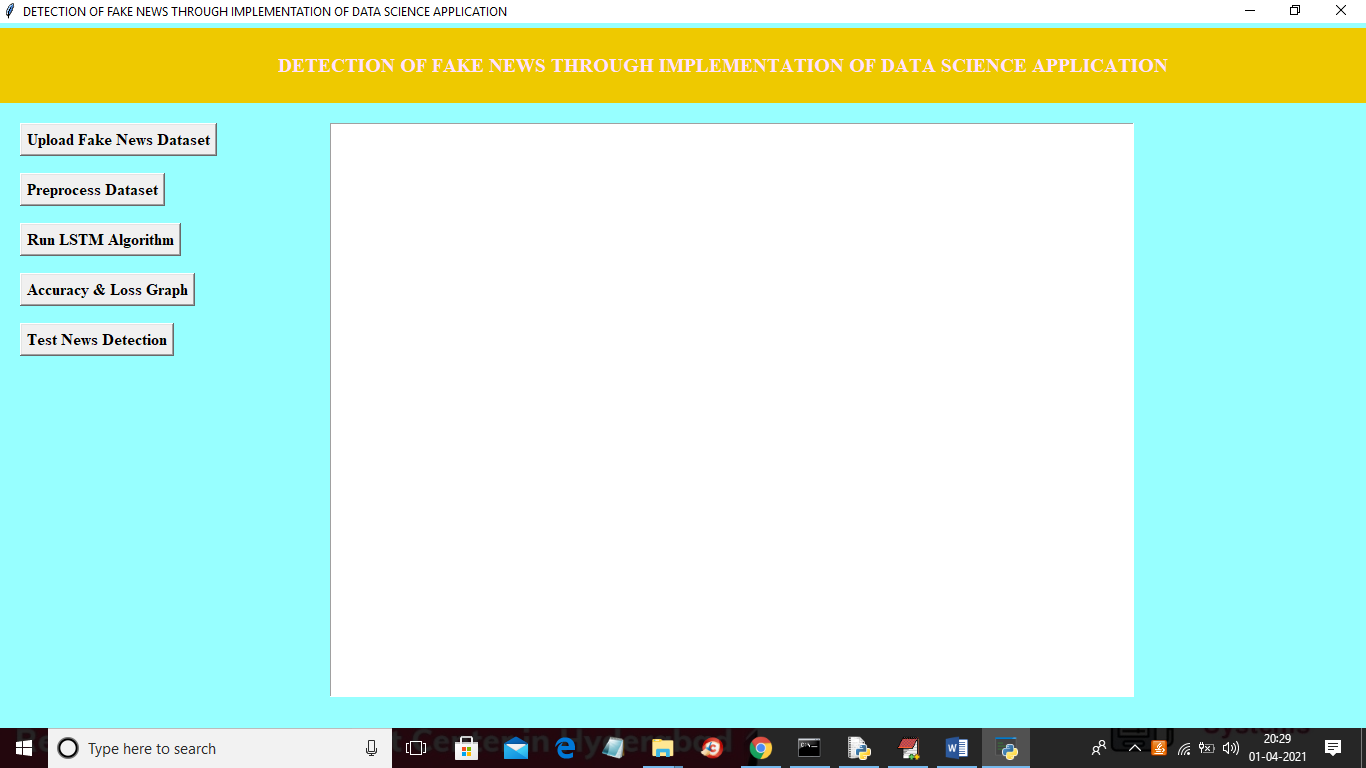
**Acceptance Testing**

When that user fined no major problems with its accuracy the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

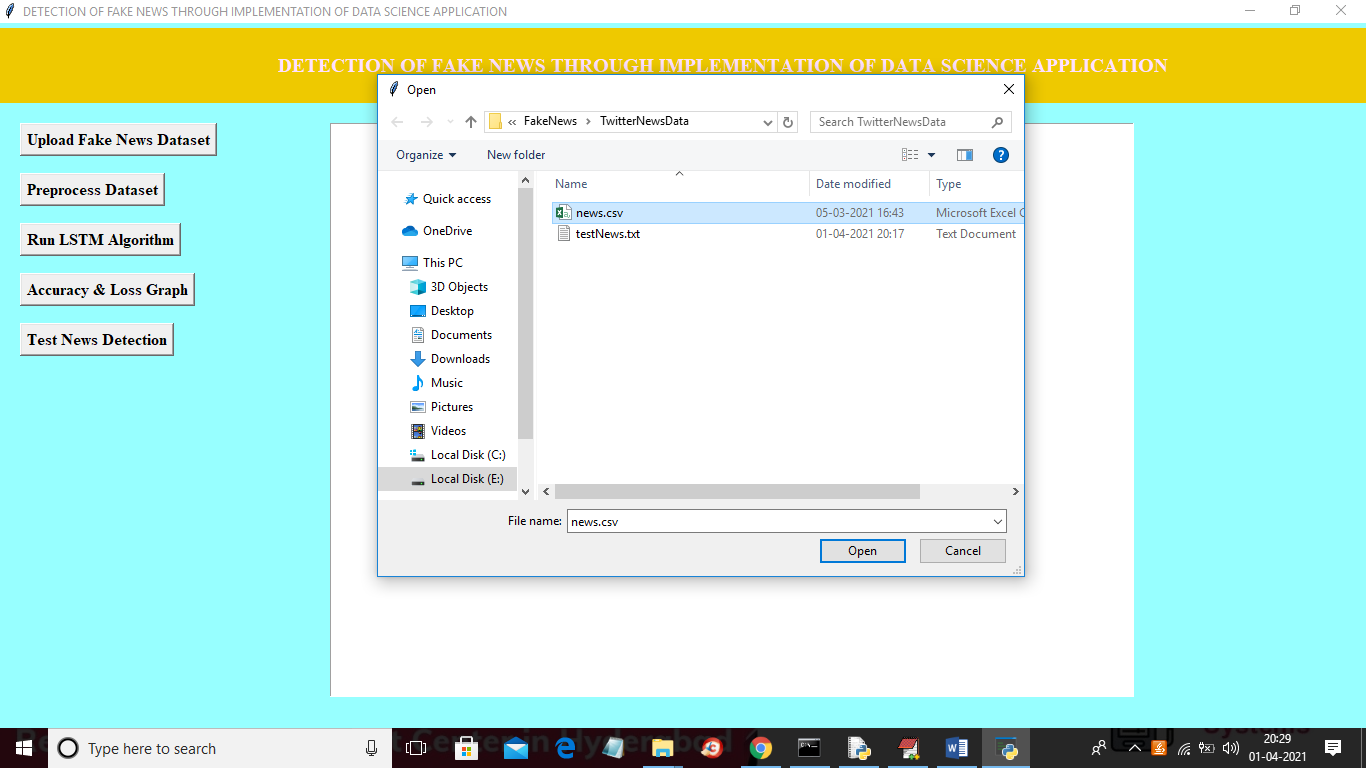
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case Id** | **Test Case Name** | **Test Case Desc.** | **Test Steps** | | | **Test Case Status** | **Test Priority** |
| **Step** | **Expected** | **Actual** |
| 01 | User  Login | Verify whether the  Username &  password is correct or not | If the username& password may not be corrected | we cannot do user login screen | we can do user login screen | High | High |
| 02 | Upload News  Articles | Verify the News Articles are uploaded or not | Without loading the dataset | we cannot  do further operations | We can do further operations | High | High |
| 03 | Run Fake News Detector Algorithm | Verify the Fake News Detector Algorithm  Run or not | Without loading the dataset | we cannot  run Fake News Detector Algorithm | The Fake News Detector Algorithm  run  Successfully | High | High |

**7.SCREEN SHOTS:**

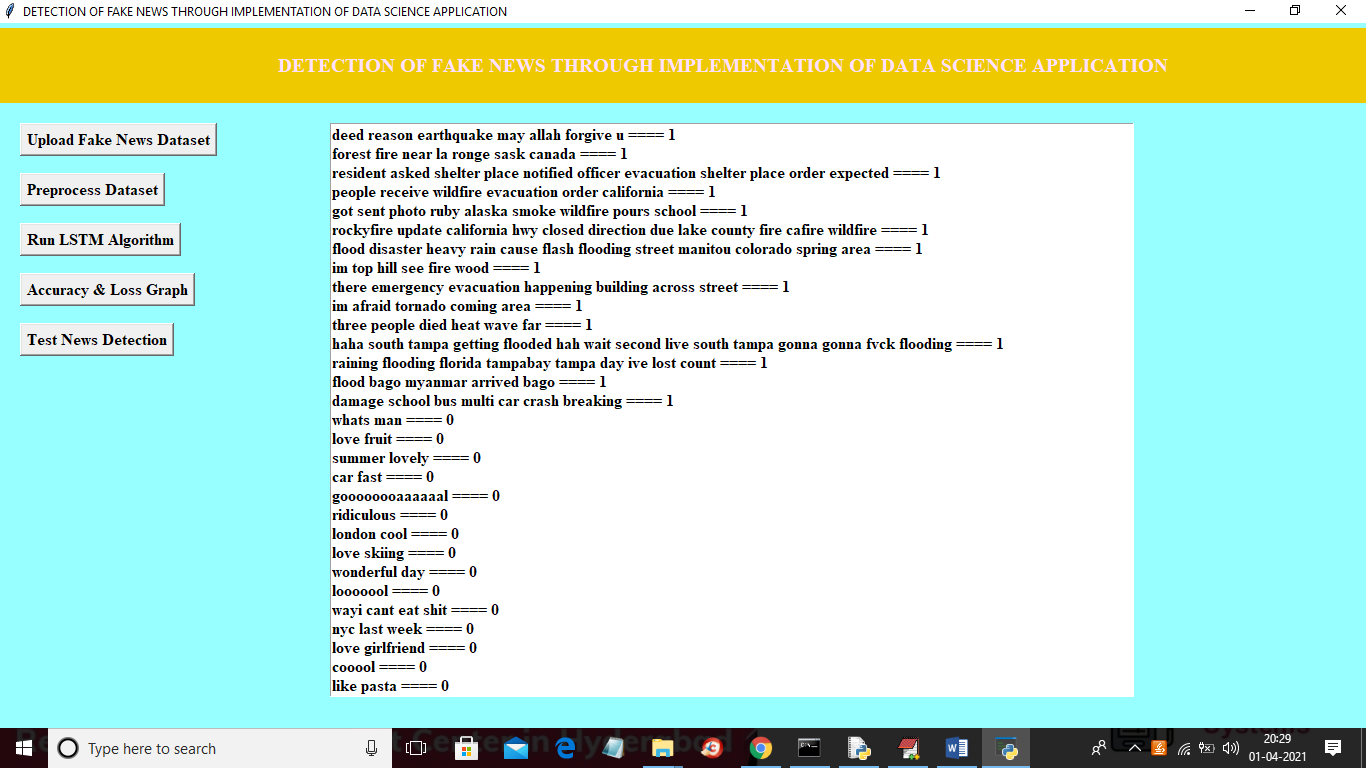
To run project double click on ‘run.bat’ file to get below screen



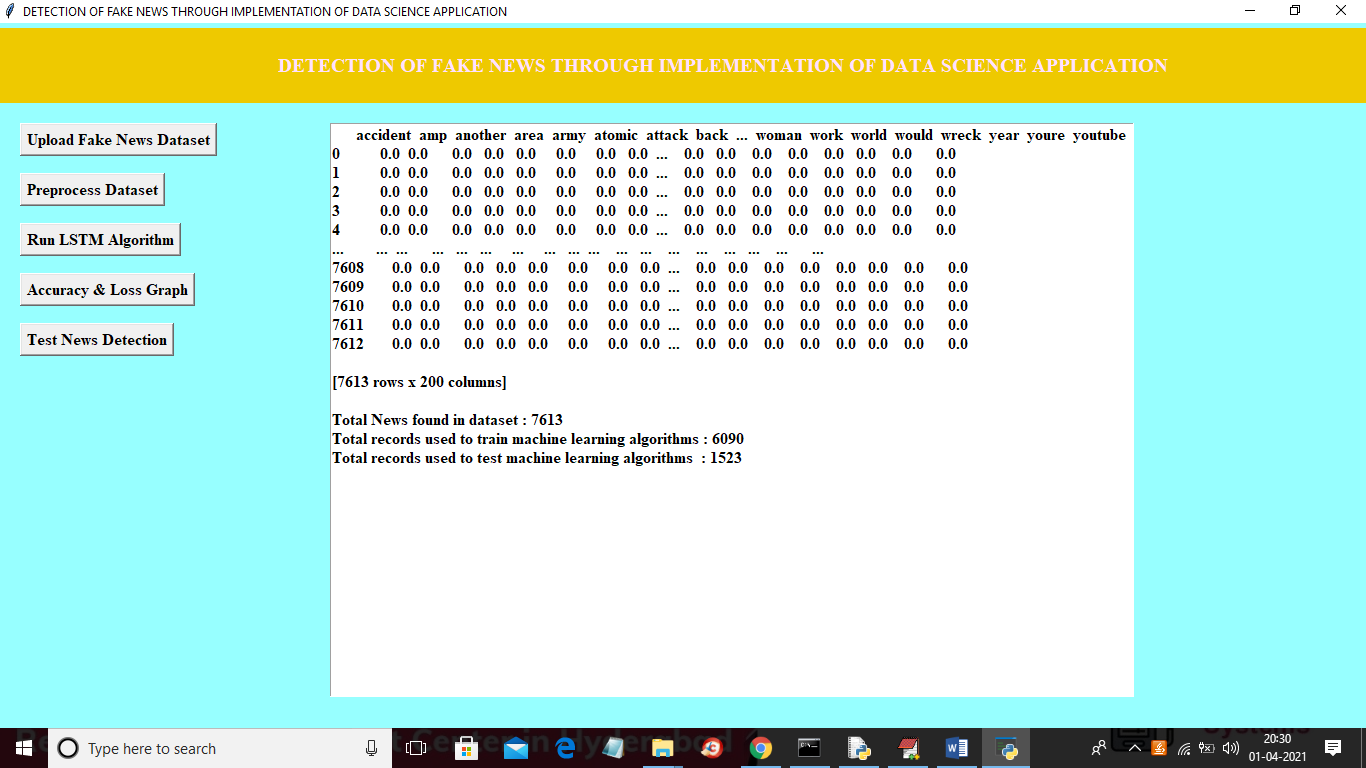
In above screen click on ‘Upload Fake News Dataset’ button to upload dataset



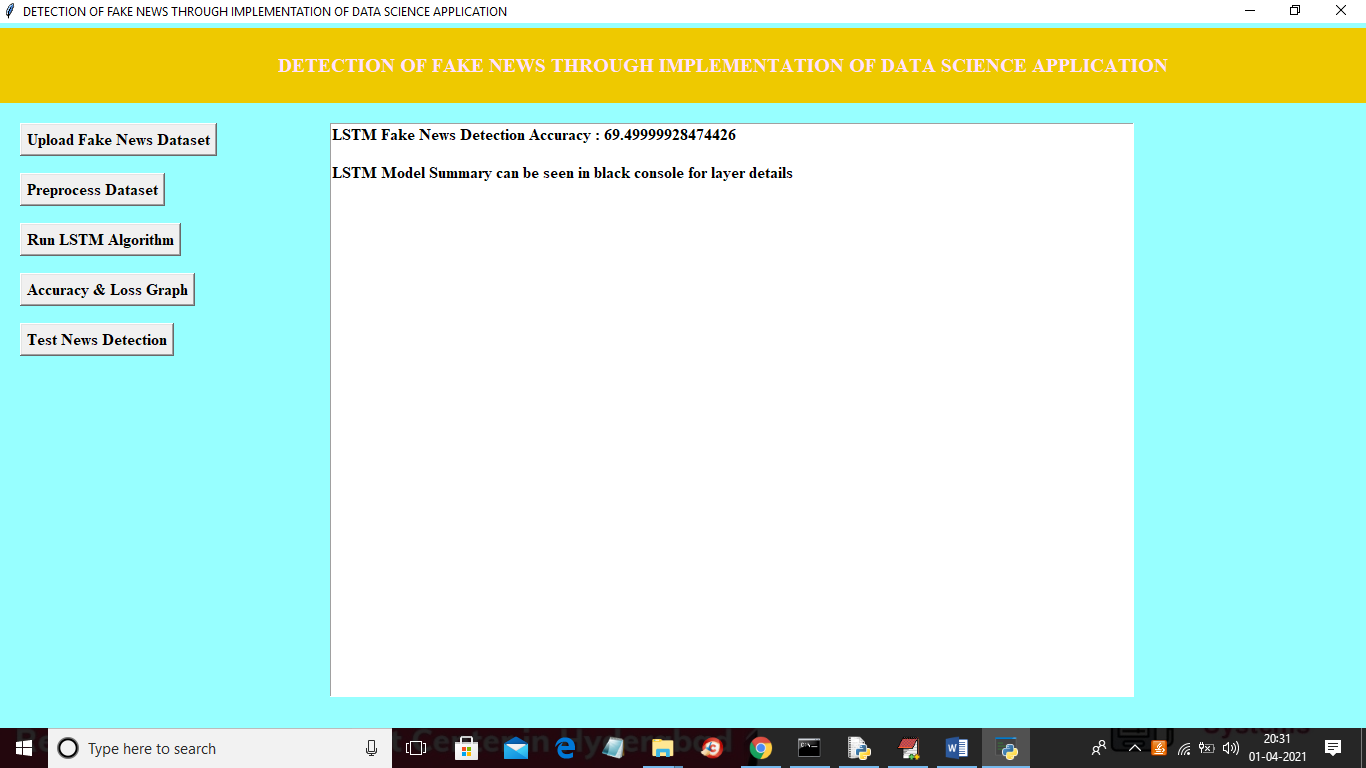
In above screen selecting and uploading ‘news.csv’ file and then click on ‘Open’ button to load dataset and to get below screen



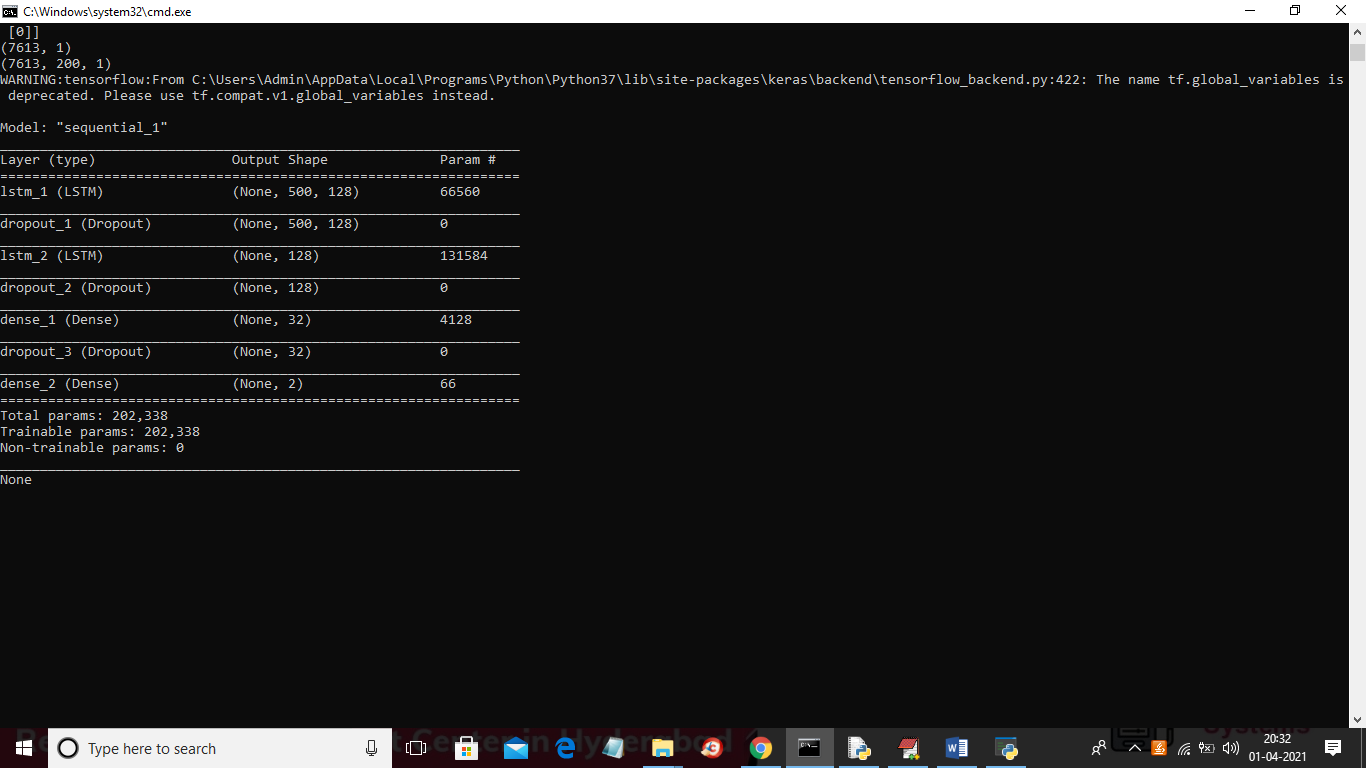
In above screen dataset loaded and then in text area we can see all news text with the class label as 0 or 1 and now click on ‘Preprocess Dataset & Apply NGram’ button to convert above string data to numeric vector and to get below screen



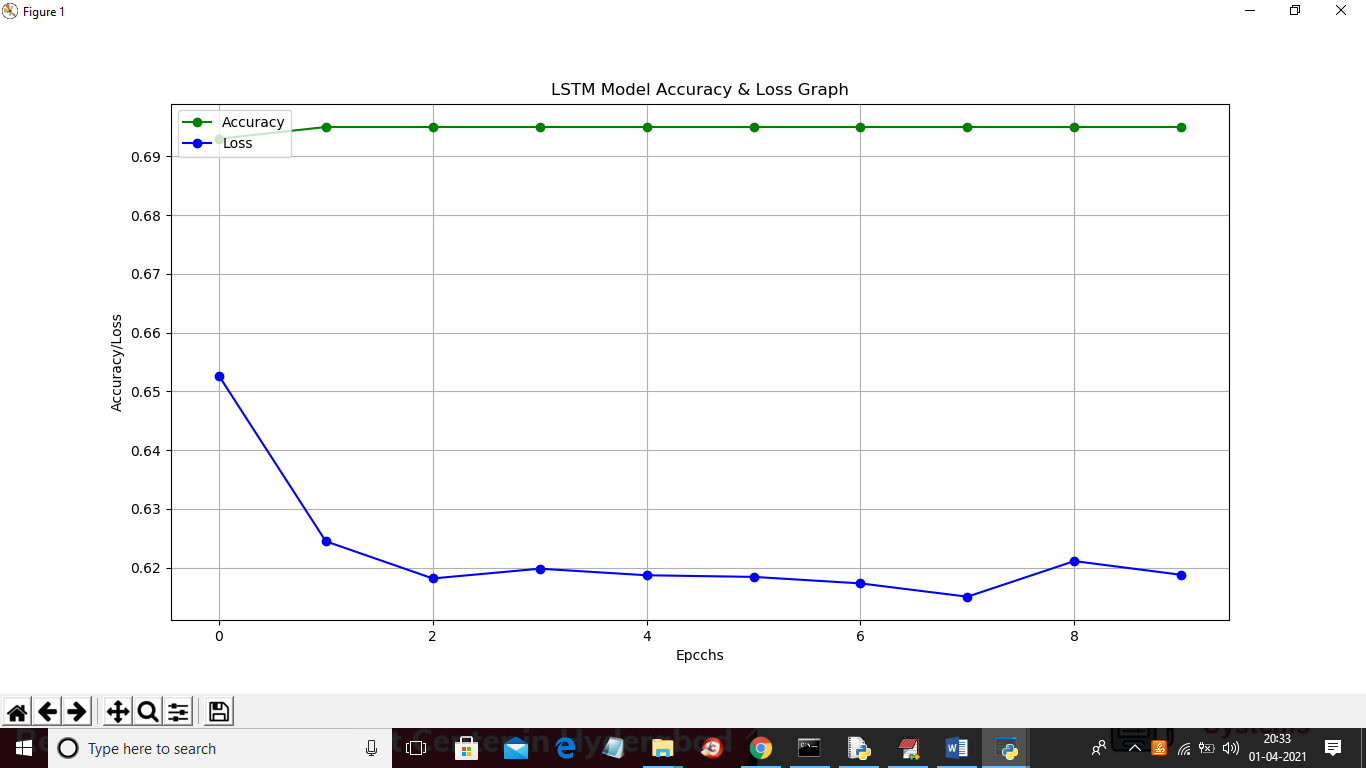
In above screen all news words put in column header and if that word appear in any row then that rows column will be change with word count and if not appear then 0 will be put in column. In above screen showing some records from total 7612 news records and in bottom lines we can see dataset contains total 7613 records and then application using 80% (6090 news records) for training and then using 20% (1523 news records) for testing and now dataset is ready with numeric record and now click on ‘Run LSTM Algorithm’ button to train above dataset with LSTM and then build LSTM model and then calculate accuracy and error rate



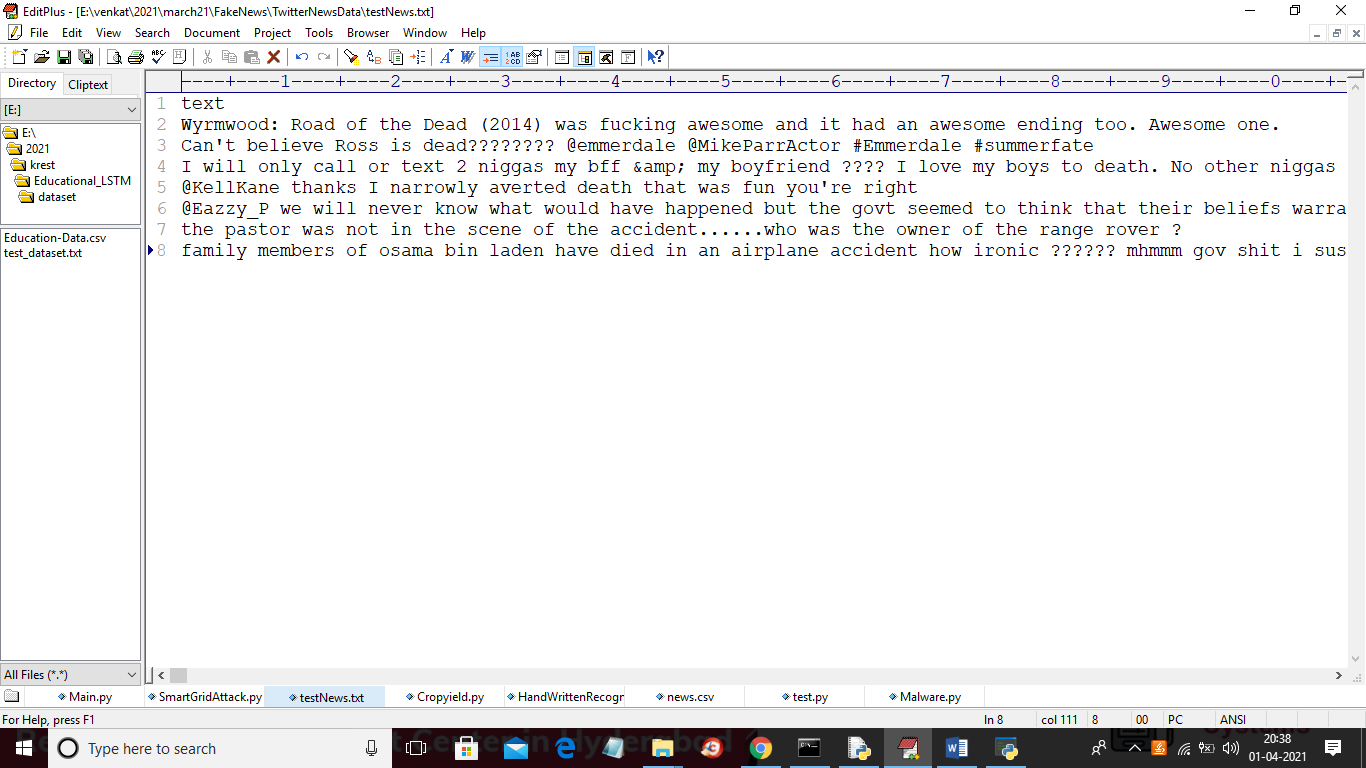
In above screen LSTM model is generated and we got its prediction accuracy as 69.49% and we can see below console to see LSTM layer details



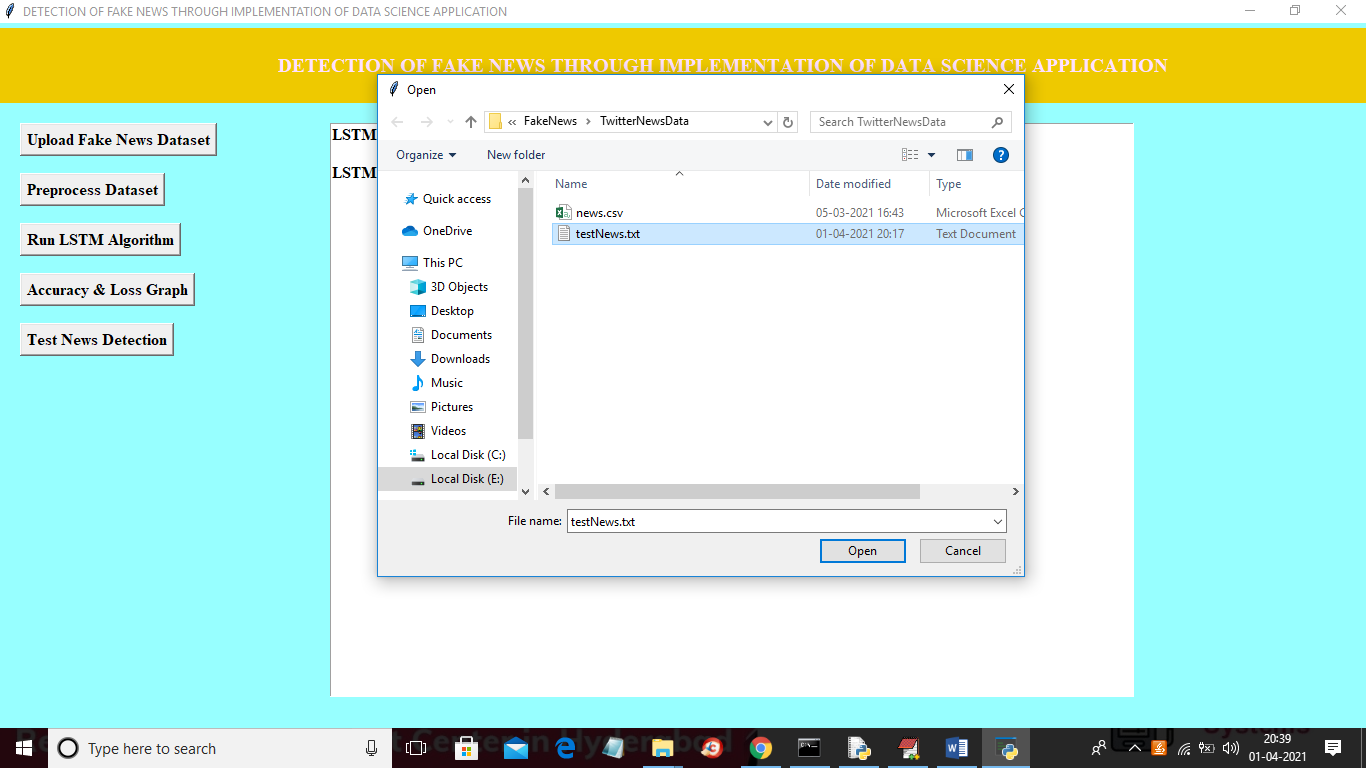
In above screen different LSTM layers are created to filter input data to get efficient features for prediction. Now click on ‘Accuracy & Loss Graph’ button to get LSTM graph



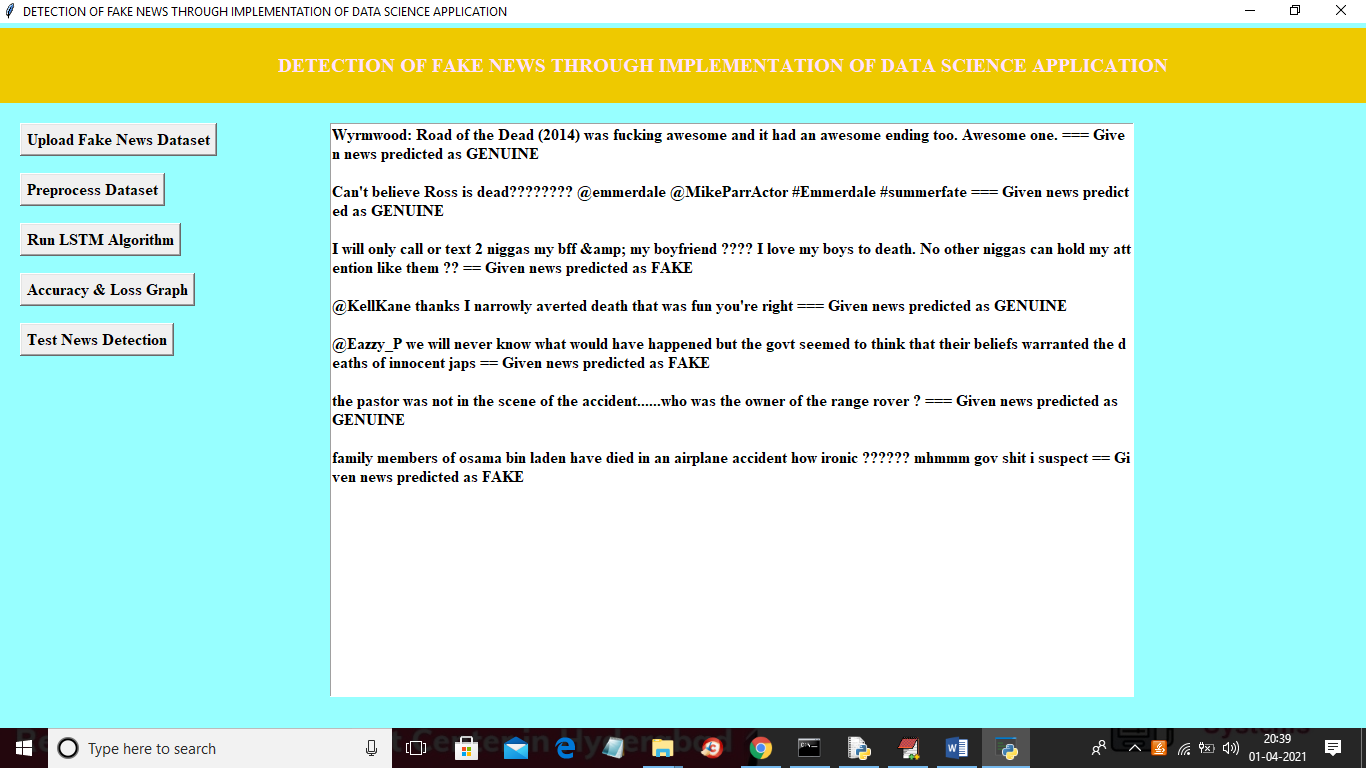
In above graph x-axis represents epoch/iterations and y-axis represents accuracy and loss value and green line represents accuracy and blue line represents loss value and at each increasing epoch loss values get decrease and accuracy reached to 70%. Now click on ‘Test News Detection’ button to upload some test news sentences and then application predict whether that news is genuine or fake. In below test news dataset we can see only TEXT data no class label and LSTM will predict class label for that test news



In above screen in test news we have only one column which contains only news ‘TEXT’ and after applying above test news we will get prediction result



In above screen selecting and uploading ‘testNews.txt’ file and then click on ‘Open’ button to load data and to get below prediction result



In above screen before dashed symbols we have news text and after dashed symbol application predict news as ‘FAKE or GENUINE’. After building model when we gave any news text then LSTM will check whether more words belongs to genuine or fake category and whatever category get more matching percentage then application

**8. CONCLUSION :**

We exploit the latent community structure in the global news network to improve the prediction of the viral cascades of news about events. The cascades which have early adopters in different communities have advantages in disseminating the contagion to these communities in parallel and therefore are more likely to result in the viral infections within a limited time period. Our model captures such property by inferring the community structure using the response times of nodes. Thus, we avoid using the explicit network topology which is often not known because the references to propagation sources are usually missing in the real data sets. Due to the size of the relevant data sets, we successfully parallelized the inference algorithm for distributed memory machines and tested this parallelization on the RPI AMOS achieving orders of magnitude speedup.

**9. REFERENCES :**

[1] A. Nematzadeh, E. Ferrara, A. Flammini, and Y. Y. Ahn, “Optimal network modularity for information diffusion,” Phys. Rev. Lett., vol. 113, no. 8, p. 088701, 2014.

[2] J. Firmstone and S. Coleman, “The changing role of the local news media in enabling citizens to engage in local democracies,” Journalism Pract., vol. 8, no. 5, pp. 596–606, 2014.

[3] R. A. Hackett, “Decline of a paradigm? Bias and objectivity in news media studies,” Crit. Stud. Media Commun., vol. 1, no. 3, pp. 229–259, 1984.

[4] S. Della Vigna and E. Kaplan, “The Fox News effect: Media bias and voting,” Quart. J. Econ., vol. 122, no. 3, pp. 1187–1234, 2007.

[5] M. Gentzkow and J. M. Shapiro, “Media bias and reputation,” J. Political Economy, vol. 114, no. 2, pp. 280–316, 2006.

[6] M. Karsai et al., “Small but slow world: How network topology and burstiness slow down spreading,” Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top., vol. 83, no. 2, p. 025102, 2011.

[7] F. D. Gilliam, Jr., and S. Iyengar, “Prime suspects: The influence of local television news on the viewing public,” Amer. J. Political Sci., vol. 44, no. 3, pp. 560–573, 2000.

[8] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” Proc. Nat. Acad. Sci. USA, vol. 99, no. 12, pp. 7821–7826, Apr. 2002.

[9] J. Yang and J. Leskovec, “Overlapping community detection at scale: A nonnegative matrix factorization approach,” in Proc. 6th ACM Int. Conf. Web Search Data Mining, 2013, pp. 587–596.

[10] S. Ji, N. Satish, S. Li, and P. Dubey. (2016). “Parallelizing word2vec in shared and distributed memory.” [Online]. Available: https://arxiv.org/abs/1604.04661?context=stat.ML [11] T. Mikolov, I. C. K. Sutskever, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in Proc. Adv. Neural Inf. Process. Syst., 2013, pp. 3111–3119.

[12] A. Mnih and Y. W. Teh. (2012). “A fast and simple algorithm for training neural probabilistic language models.” [Online]. Available: https://arxiv.org/abs/1206.6426

[13] B. Recht, C. Re, S. Wright, and F. Niu, “HOGWILD: A lock-free approach to parallelizing stochastic gradient descent,” in Proc. Adv. Neural Inf. Process. Syst., 2011, pp. 693–701.

[14] R. G. Miller, Jr., Survival Analysis, vol. 66, Hoboken, NJ, USA: Wiley, 2011.

[15] A.-L. Barabási. (2005). “The origin of bursts and heavy tails in human dynamics.” [Online]. Available: https://arxiv.org/abs/cond-mat/0505371

[16] B. Perozzi, R. Al-Rfou, and S. Skiena, “DeepWalk: Online learning of social representations,” in Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2014, pp. 701–710.

[17] D. Kempe, J. Kleinberg, and E. Tardos, “Maximizing the spread of influence through a social network,” in Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2003, pp. 137–146.

[18] E. Mossel, J. Neeman, and A. Sly. (2012). “Stochastic block models and reconstruction.” [Online]. Available: https://arxiv.org/abs/1202.1499

[19] R. M. Gomez, J. Leskovec, and B. Schölkopf, “Structure and dynamics of information pathways in online media,” in Proc. 6th ACM Int. Conf. Web Search Data Mining, 2013, pp. 23–32.

[20] N. X. Vinh, J. Epps, and J. Bailey, “Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance,” J. Mach. Learn. Res., vol. 11, pp. 2837–2854, Jan. 2010.