ISSN 2210-142X

International Journal of Computing and Digital Systems (IJCDS)

VOLUME NO. 12
ISSUE NO. 2
MAY - AUGUST 2023

EIS LEARNING
No - 198, Upper Hatia, Near Mahatma Gandhi
Smarak High School, Ranchi - 834003, Jharkhand
Ph : 919097302256 / Email : info@eislearning.com

ISSN 2210-142X

International Journal of Computing and Digital Systems (IJCDS)

Aims and Scope

International Journal of Computing and Digital Systems (IJCDS) is a peer-reviewed International Journal that currently publishes 6 issues annually. IJCDS journal publishes technical papers, as well as review articles and surveys, describing recent research and development work that covers all areas of computer science, information systems, and computer / electrical engineering.

The topics covered by IJCDS are including and not limited to the following research areas:

- Reconfigurable Computing & Embedded systems
- Computer Communications and Networking
- Internet of Things & Real Time Systems
- Cyber Security
- Cloud Computing
- Smart Systems
- Information Systems and Communication Service
- Innovation/Technology Management
- Business Information Systems
- Software Engineering
- Mobile & Web Applications
- Theory of Computation
- Data Structures, Cryptology and Information Theory
- Artificial Intelligence & Robotics
- Image Processing, Computer Vision, Pattern Recognition & Graphics
- Data Mining & Big Data
- Smart Grids & Renewable Energy
- Human Computer Interaction

Responsiveness Speed

Time from submission to first decision after peer review: 4-8 Weeks
Time to immediate reject: 2-4 Weeks

Publication Speed

Time from final acceptance to print: 3-6 Months
Total time to publication: 4-8 Months
Time from submission to acceptance: 4-12 Weeks
Time from acceptance to online Publishing: 4-8 Weeks

Publishing Ethics

Plagiarism in all its forms constitutes unethical publishing behavior and is unacceptable. The journal has no tolerance on plagiarism. All submitted manuscripts must go through cross checking using turnitin as an online plagiarism checker.

ISSN 2210-142X

International Journal of Computing and Digital Systems (IJCDS)

(Volume No. 12, Issue No. 2, May - August 2023)

Contents

ISSN 2210-142X

Characterizing Reverse Engineering Practices on Decayed Software Applications

Samir Obaid[1], Ibrar Arshad[2], and Muhammad Usman Abid [3]
[1,2] Department of Computer Science, Capital University of Science and Technology, Islamabad, Pakistan
[3]Department of Software Engineering, Riphah International University, Islamabad, Pakistan
E-mail address: [1]samir.obaid@cust.edu.pk, [2]ibrar.arshad@cust.edu.pk, [3]abid.musman@gmail.com

ABSTRACT

Architecture reverse engineering is an approach to reproduce architectural contents once an application has deviated from its planned architecture. Unassisted understandings of an application by an individual, interviewing a person knowing the subject system, and computer-aided tools are few approaches that produce architectural contents from a decayed software application. The former two approaches are helpful when there is an individual in the organization who can understand the software application. Worst comes when computer-aided tools remain the only way to produce architectural contents from an application's source code. This research aims to identify architectural contents that industrial practitioners reproduce through reverse engineering, finding out the users of identified architectural contents and how the existing tools help in meeting industrial practitioners' needs. To achieve the research objectives, a qualitative study was performed by choosing a homogenous sampling approach from the organizations where software applications were under gradual development for many years. Semi-structured interviews were conducted, and a coding approach was used to find out themes from transcribed data. We identified different architectural contents that practitioners produce from source code. Our findings show that practitioners use reverse engineering tools to produce architectural content from an application's source code. However, there are some architectural contents that practitioners need to reverse engineer, but no available tool produces those contents. The reverse engineering tools produce a wide range of architectural contents from source code but, contents visualization as required by practitioners is a challenge that needs to be addressed.

Keywords: software architecture, reverse engineering, code decay, architecture recovery

1. INTRODUCTION

Architecture reverse engineering is an approach for analyzing a system to identify its components, their interrelationship and create representations of the system in another simplified form or a higher level of abstraction [1, 2]. Architecture reverse engineering can also be explained as identifying the architectural contents of a system that are deviated from its planned architecture [3], or its architecture is never documented at all. There are several reasons for this deviation, for instance, the system was easy to develop at the early stages; therefore, no one was bothered to document architecture during further development and maintenance, the architecture documentation was considered a time-consuming activity, there exists an architecture document, but it was not updated with the upcoming changes of the system. Due to these reasons, the system source code becomes the only source to understand the system's major components and dependencies, design information, and quality aspects that different stakeholders require. The term code decay [3-5] is used if source code mismatches documented or planned architecture. The term architecture degeneration [6] is also used for systems in which actual architecture mismatches the planned architecture. Parnas [7] uses the term software aging for degraded system design and increased complexity. Once the system goes through many years of development activities, it tends to be complex, poorly structured and has little or no documentation. The system becomes hard to understand; any new decision becomes hard to undertake as it requires paying the extra cost of development and testing to ensure that new functionalities are added without placing any new problem. Architecture reverse engineering is helpful in this regard, as it recovers some valid architectural contents of the system.

There are some tools that can reproduce architecture contents from source code information. However, it is important to know that produced architectural contents and their representations are acceptable for industrial practitioners who use reverse engineering tools. To address this challenge, the high-level goal of this research is to:

- Find out architectural contents that industrial practitioners need to produce from source code.

To meet the high-level goal of the research, we have set the following objectives:

- To find architectural contents that industrial practitioners reproduce from source code
- To find users of reproduced architectural contents
- To know how existing tools help reproduce required architectural contents?

To meet the first two research objectives, a qualitative study was performed where practitioners are interviewed to know about: how and what architecture contents they produce from source code? What is their aspiration from reverse engineering tools? In order to meet our third research objective, existing vendor tools were studied, and then their features were compared with survey results. Our finding showed that existing reverse engineering tools produce various architecture contents from source code information; however, there were architecture contents that need to be produced, yet none of the existing tools produce these contents from source code.

The remaining of this paper is organized as follows: Section II discusses existing tools. The research methodology is described in section III. Section IV reports our findings from a qualitative survey, and section V concludes the work done and future direction.

2. EXISTING TOOLS

There are many standalone tools and plug-ins for major IDEs that recover architecture from application source code. These tools help improve comprehension of source code, check code quality against quality metrics, delta analysis to show the difference between two versions of the code, and code compliance to enforce architecture design decisions on code. Among earlier reverse engineering tools, Wong et al. [8] propose Rigi, a tool that produces flow graphs from application source code. The flow graph presents functions, functions calls, and data accesses. The produced flow graph can be much more detailed, depending upon several function calls in the application source code. To minimize the flow graph's noise or complexity, Rigi uses the cluster and filtering [3] technique. Through this technique, a user can group low-level entities (function calls) into high-level entities to obtain an abstract model that meets the user's reverse engineering goal. Rigi is a freeware with its source code, and pre-compiled downloads are available on their official website [9]. Rigi comes with extensive learning resources, user manuals, and sample programs of increasing complexity, demonstrating how Rigi can reverse-engineer software systems.

Systa et al. [10] propose Shimba, a prototype reverse engineering tool that is designed to understand java source code. Shimba analyzes java byte code and visualizes a subject system's static and dynamic components by customizing Rigi. These components include classes, interfaces, methods, constructors, variables, static initialization blocks, return types, and visibility. It also extracts relationship components from java byte code such as extension, implementation, containments, method call, and assignment. These extracted components can be visualized using Rigi dependency graphs. Shimba creates a sequence and state-chart diagrams from extracted components to visualize dynamic aspects of the subject system. A sequence diagram helps understand the relationships among different objects, whereas a state-chart diagram helps to understand the overall behavior of certain key objects. Shimba supports the model slicing technique, through which a user can filter out unnecessary details from the dynamic models to focus on particular parts of the target system.

Imagix4D [11] is a reverse engineering tool that helps software developers to understand, document, and improve complex, third-party, and legacy source code. Imagix4D analyzes source code and produces different abstract views of the system, such as subsystem architecture (well-segmented subsystems with cleanly specified interfaces), package diagram, class diagram, functions, and variables dependencies control flow graphs. The tool also assesses analyzed code over a hundred different quality metrics to ensure that the software meets planned development criteria and determines where to focus testing efforts. Its delta analysis feature helps in visually reviewing the structural differences between different versions of the code. Imagix4D is available with the latest stable release since 2019, and it also has a rich user manual and technical support document.

Sangal et al. [12] propose Lattix [13], a static analysis tool that reverse-engineers architecture from an application's source code. Architecture contents, reverse engineered by the tool, can be viewed in the form of graphs (lines and boxes) or dependency structure matrix (DSM). DSM is a simple, compact, and visual representation of system components (classes, packages) in the form of rows and columns. Like Rigi, the DSM feature of Lattix helps filter low-level architecture contents and create clusters to make a more abstract view of the system architecture. Lattix facilitates code compliance with existing architecture by providing rules-based architecture enforcement features that enable the architect to define design rules and restrict the development team from violating those rules during the development phase.

NDepend [14] is a static analysis tool that explores existing architecture from C# and Visual Basic code. Like Lattix, NDepend presents architecture details in the form of DSM that helps filter low-level components to visualize the high-level architecture of the system. In addition to DSM, the initially explored architecture can also be filtered with the help of a query language called CQLink. Through CQLink, a user can query for specific elements from the model by hiding other elements of the model, which helps a user to define his/her own way of presenting a lightweight architecture of the system. Like Lattix, NDepend provides rules-based architecture enforcement features that help its user define design rules and then restrict developers from violation during release and eventually, before committing to source control. NDepend comes with a compare build feature that can graphically visualize what has been changed between two versions of the code.

BOUML [15] recovers UML diagrams from Java, PHP, C++, and Python source code. Along with generating UML models from the source mentioned above code, it also generates XML Metadata Interchange (XMI) that can be used by any other model generator tool to extract a model from XMI. This XMI file can also help generate diagrams other than UML and thus make it convenient to develop their own models for the representation of architectural contents.

There are many plug-ins for eclipse IDE that visualize architectural details from Java source code. UML-Lab [16] supports round trip engineering, i.e., reverse engineering architecture from source code and generating code from architectural diagrams. The tool presents architecture by the UML class diagram, whereas it also facilitates creating customized models with the help of UML profiling. Another similar tool is ObjectAid [17] that creates a UML sequence and class diagram from Java source code. The tool helps automatic synchronization between code and diagrams. Any changes in the code are automatically reflected in the diagram without executing the model generation process again. AgileJ [18] creates a class diagram reverse-engineered from Java source code. Since generated diagrams can be cluttered; therefore, the tool also provides filtering ability to reduce the noise in the presented information. MaintainJ [19] visualizes a static view from java source code with the help of a class diagram. In order to visualize a dynamic view, the tool analyzes the program call stack and presents stack traces by a UML sequence diagram. The tool can also visualize a call stack between two applications when running on two different Java virtual machines. For example, when an application calls a service (running on different JVM), the call flow across JVMs can be shown by a single sequence diagram. ModelGoon [20] visualizes component dependency by reverse engineering Java source code into package and class diagram. The tool also produces a sequence and collaboration diagram to present a dynamic behavior of a selected codebase. Diver [21, 22] is a code analyzer that records traces of running programs and visualizes a sequence diagram of captured traces. It also helps the user to explore the code from a sequence diagram to gain the same perspective from the code that a user understands from a sequence diagram.

We found a large number of reverse engineering tools that produce architectural content from an application source code. Tools come with various static and dynamic models to facilitate users of the tool with a wide range of information. In the presence of diverse reverse engineering tools, it is important to know about practitioners' needs and challenges that they face while reverse engineering architecture contents from application source code

3. METHODS

We have chosen a qualitative research method and conducted interviews to gather focused, in-depth, and qualitative textual data from respondents. The research process started by formulating the research objective as described in Section I. Following the research objective, a vendor study was performed to understand existing reverse engineering tools and their features, as described in section II. We then selected the study sample, conducted interviews, and executed a qualitative coding process described in subsections A, B, and C.

A. Study Sampling

We have used a homogeneous sampling [23] approach, which can be described as "gathering people in a similar situation with a similar background". The rationale behind homogeneous sample was to select only those organizations where software product has undergone few years of development practices. During this time, the application has reached a certain level of complexity, and practitioners tried reverse engineering during the development process. While searching for the required sample, we found the following similar situations with our respondents:

1) Product time in the market: The first common aspect of all respondent companies was their long product life in the market. As described in Fig. 1, the minimum product life is seven years, whereas the maximum product life is seventeen years. Respondents told us that a significant amount of code was written years ago. Due to this, developers who have written that code hardly memorizes it. New developers who have joined the team find it difficult to understand the code without their colleagues' assistance. The worst comes when old developers have left the company, and new developers try to understand the code with unaided browsing.

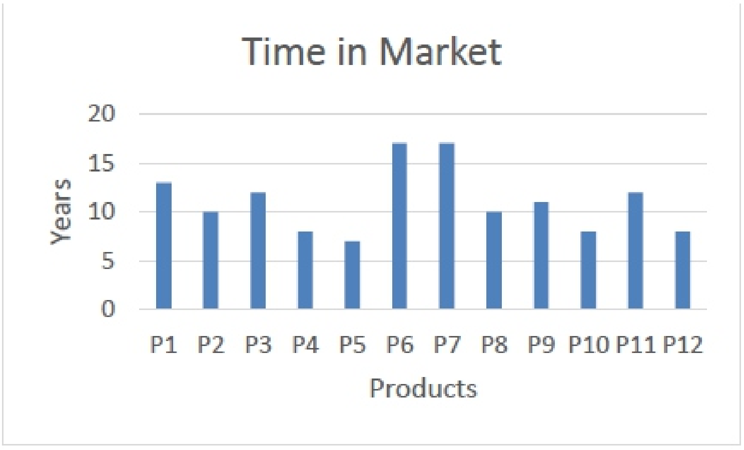


Fig. 1 Products time in the market

2) Team size: The second common aspect among respondent companies was their team size. Fig. 2 describes team size, which ranged from 65 members to 180 members, and it included: developers, quality assurance (QA) engineers, technical support team, architects, team leads, technical report writers, and company's high-ups who may also be non-technical personals. Respondents told us that there are varying interests of team members which depended upon their role. We have thoroughly discussed those interests in our findings. For instance, if two development groups worked concurrently on different tasks, they must know about execution flow, input and output of developed components, and code organization. It was not feasible to assist other groups with developed code, especially when there were rapid release cycles.

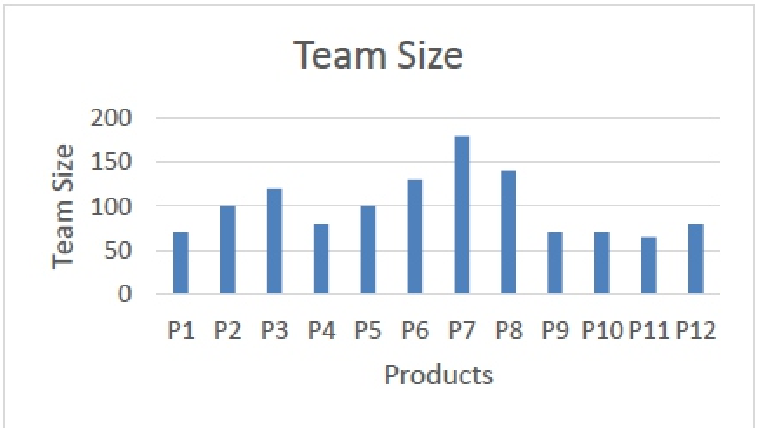


Fig. 2 Team Size with respect to the number of people

3) Product size: The third common aspect among respondent organizations was lines of code (LOC). As described in Fig. 3, the code size ranged from 0.8 million LOC to 4.0 million LOC. As mentioned by respondents, having a huge code has several challenges: it is difficult to understand, communicate, and properly document it.

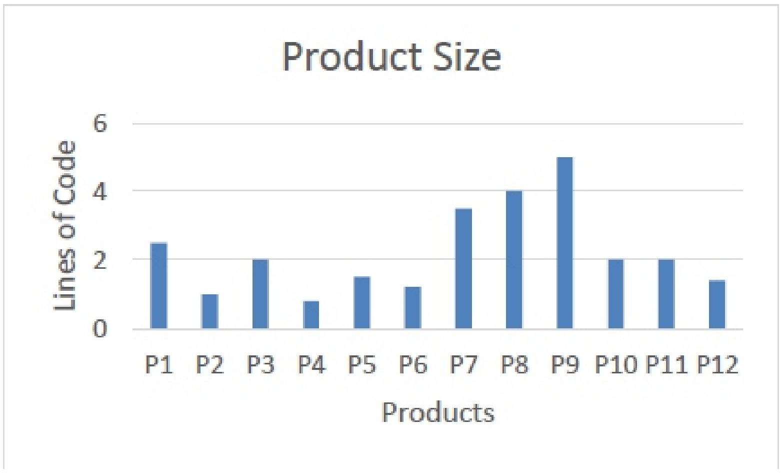


Fig. 3 Product size with respect to lines of code

B. Interview and Data Collection

The interviewer visited each respondent in person for an interview session, and data were recorded in audio and hand-written form. Follow-up interviews were conducted in order to confirm and clarify data where required. Our in-depth interview had many questions for each respondent; however, we planned some starting questions for each interview. For instance:

1) What are the architectural contents that you reproduce from the source code?
2) Who are users of reproduced architecture content?
3) How are existing tools helpful in reproducing required architectural contents?

Each interview lasted from 40 to 60 minutes, and data were transcribed on paper or audio form after having the respondent's consent. We have also conducted four follow-up interviews with four respondents after having confusion in transcribed data. Data were transcribed and then analyzed to find common patterns and to place forward general observation.

C. Quilatative Coding Process

1) Content Analysis: We have selected conceptual analysis as a content analysis approach, i.e., the focus was on the existence of concepts and frequency of occurrence of those concepts. According to Carley [1], several steps must be followed before conducting the content analysis. TABLE I briefly describes those steps, along with the decision taken by the author at each step. At the end of the analysis, reverse engineering practices were categorized, and they were used to find major architectural contents produced by the practitioners from source code and stakeholders of those architectural contents.

TABLE I STEPS FOR CONDUCTING CONTENT ANALYSIS [1]

Tool	Output artifacts
Decide the level of analysis	We have decided to code for both single and set of words that appear in the text (transcribed data from respondents)
How many concepts to code for	We have decided to code for relevant concepts as they appear in the text
Code for existence or frequency of concepts	Since we are not sure about the significance of frequency in this research topic, therefore, we have decided to code for the existence of the concept even if it appears multiple times.

Distinguish between concepts	We have decided to highlight the concepts exactly as they appear in the text. If two concepts with different phrases appeared, then there is a chance to distinguish them based on their relation to reverse engineering practices
Rules for coding the text	One researcher has highlighted the concepts and revised them by another researcher,
Code the text	We have chosen the manual coding technique by using paper and a highlighter. Computer-aided tools are not used due to lack of practice.
Analysis of result	We have analyzed the data to determine respondents' expectations from reverse engineering tools and architecture contents that they reproduce from source code. We also compared their expectations with available reverse engineering tools.

2) Identifying Codes: Coding as a method of organizing transcribed data is widely used in qualitative research [1, 2]. We have chosen a top-down coding approach (aka theoretical coding approach) through which we just code those concepts that were relevant to our concerns as expressed in research questions. The coding task was executed by two researchers where one researcher identified codes from transcribed data and the other reviewed the codes to omit chances of errors. While highlighting codes, we have adopted the Boyatzis[3] approach in which a researcher briefly describes coded concepts and how a researcher highlights codes from transcribed data and the corresponding excerpt of transcribed data. The coding process was stopped after few iterations, and identified concepts were: sequencing GUI snippets, configuration details, roles, and access control, static and dynamic execution flow, APIs with required and provided services, code annotation, modules re-organization, visualization of ERD, data tables, and data rows presentation. Once codes were identified, then they were organized into themes as described in Fig. 4.

3). Themes: Themes are outlines (also known as patterns) that group together common codes to address research questions [2]. Once gone through codes, we observed some obvious themes that emerged, and codes fit into them. For example, some frequent codes such as user stories, code snippets, and GUI snippets are grouped to set up a Scenarios theme. We identified a total of five themes at the end of this process and described them in Fig. 4. Themes are user stories, program execution flow, service APIs, code organization, and database view. Each theme further contained a sub-theme as described by rounded corner boxes in Fig. 4. For example, static and dynamic execution flow practices are categorized into a theme which is named program execution flow. Service APIs consolidate all practices where practitioners try to produce service details from the codebase. The database view contains practices: ERD generation from database schema, table's view and associations, and finding redundant tables and fields. User Stories consolidates sequencing GUI snippets, configuration details, and access control policies. Codebase organization practice consolidates code annotation and modules re-organization.

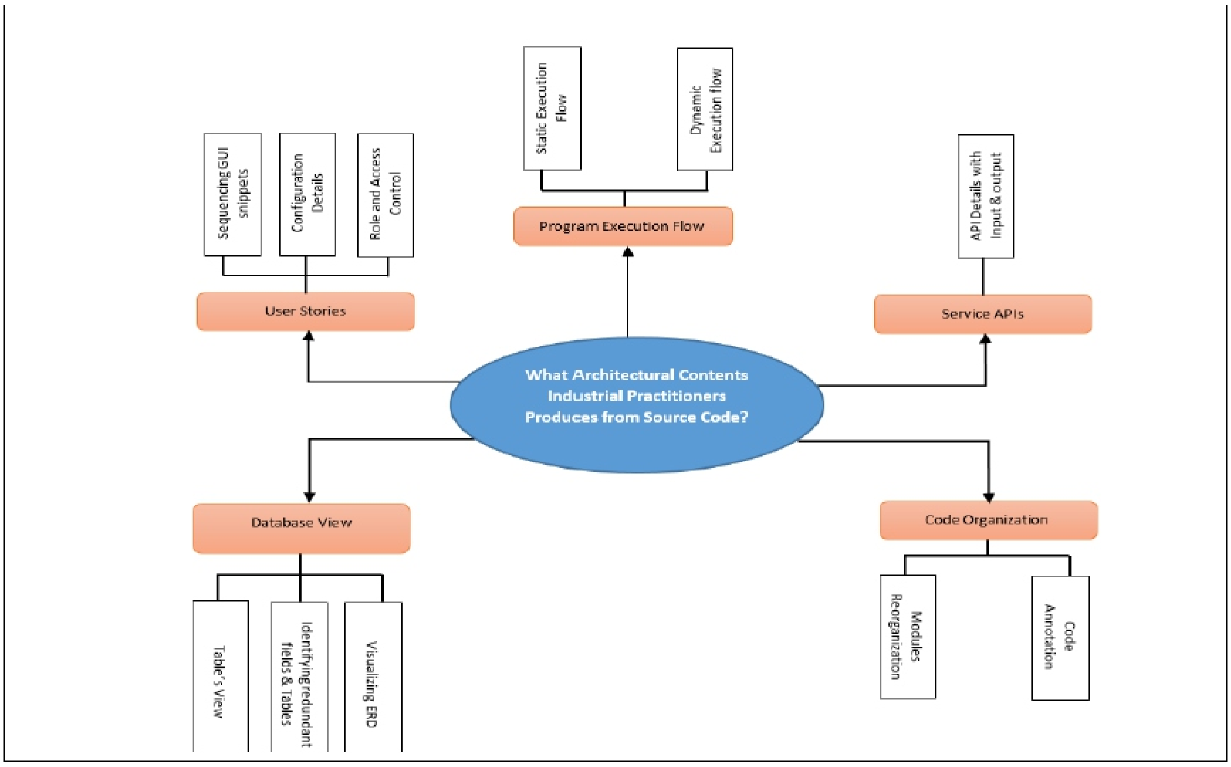


Fig. 4 Thematic Map to Present Architectural Contents that Practitioners Produce from Source Code

4. FINDINGS

To answer our first research question, TABLE II describes architectural contents, challenges faced by practitioners to produce those contents, and strategies adopted by industrial practitioners to generate these contents. The following section describes what respondents demanded under each architectural content.

1) Program Execution Flow: Program execution flow describes exaction sequence, either control flow or data flow, once the system receives input. We observed that all respondents were reproducing program execution flow details from the application source code.

For instance, one of our respondents mentioned that 'We document screenshot of GUI and back-end code because our fellow developers need them to understand program execution flow by GUI snippets with code behind'. After asking about this activity's practice, the respondent mentioned recording input events and execution flow (both code and GUI screens) against that input event. Team members used this execution sequence to understand the execution sequence with no or minimum assistance from other fellows. Respondent also mentioned that sometimes they had to demonstrate an application to high-ups (not from the IT domain) of the company. With the help of this document, the high-ups understood how their aspirations are realized in a developed system. Another respondent mentioned that; 'program execution flows are maintained with the help of boxes and arrows to assist team members' and 'execution flows in the business layer are important for us, and we document it from code'. This respondent showed the same intent as the previous one, i.e., helping developers understand program execution flow against the input event without any assistance and communicating work done with high-ups. From the content analysis, we observed that screenshots of GUI and code and lines and boxes diagrams were two common approaches by respondents to capture execution flow details. Since respondents were capturing system-wide execution sequence instead of a particular input event, therefore, we attributed this activity as static execution flow details of the program.

Respondents also showed their interest in extracting dynamic execution flow information from program source code. For instance, a respondent mentioned that 'We maintain a log of program execution traces to find the exact point where fault exists'. Another respondent mentioned that 'We record input data and execution sequence to observe where does fault exist'. When we asked about the purpose, respondents mentioned that they mostly failed to reproduce the client's reported bugs. Therefore, they have developed a tool that logs client-side configurations, input events, and failure occurrences. Once received a log, then another version of the tool is used on the developer end, which sets the same configuration, provides input data, and records code execution sequence to reach fault position. Another respondent followed the same practice by mentioning that 'we have an in-house developed tool that records input events and program execution trace against event'. It helps them to figure out specific input events that caused failure and program execution trace until failure occurs. Since respondents were recording execution sequences for a particular input(s), therefore, we attributed this activity as dynamic execution flow details of the program.

TABLE II ARCHITECTURAL CONTENTS REVERSE ENGINEERED BY PRACTITIONERS

Architectural content	Characteristics of Architectural Contents	Challenges Faced by Practitioners	Strategies Adopted by Industrial Practitioners to Generate these Contents
Program Execution Flow	Capturing input event to program, identification of program execution path from source code, finding dependencies among modules and business flow	*'Execution flows in the business layer are important for us, and we document it from code ...'* *'New developer uses to record input event and execution sequence on a paper...'*	Static Execution Flow *'program execution flows are maintained with the help of boxes and arrows to assist team members'* *'We keep execution sequence along with code screenshots ...'* Dynamic Execution Flow *'We maintain a log of program execution traces to find the exact point where fault exists.'* *'We maintain a log for execution traces of program to investigate the exact point where the system gets crashed.'* *'We observe input data, methods sequence, and the point where application crash.'* *'We have an in-house developed tool that records input events and program execution trace against an event.'*
Service APIs	Contents related to documenting: purpose, input, and output of reusable program libraries and web services	*'development team frequently modify already developed APIs to generate a new type of record....'*	API Details with Input & Output *'Frequently document the purpose of an API, with input and output.'* *'We have to frequently re-document the purpose of APIs because there are frequent changes by the development team ...'*
Database View	The understanding database schema, identification of redundancy (both tables and fields)	*'database schema evolves with time, and it became hard to work on report generation, adding new records and understand the entire schema, especially by new team members...'*	Visualizing ERD *'inspect schema to generate ERD from it.'* *'We revise our databases to find out common fields across tables'* Table's View *'We have developed an in-house tool named *** that takes a table name as input and displays all associated table'*
User Stories	Contents related to documenting and maintaining user stories from the source code information	*'QA team cannot manage their work due to a lack of knowledge about interfaces and APIs. We, therefore, work with the development team to reproduce user stories ...'* *'Sometimes we need to create models to share our user stories with HR and technical support team to let them use it ...'*	Capturing and Sequencing of GUI Snippets *'for each use case, we document a sequence of GUI snippets along with back-end code. It helps both non-technical and technical personals to understand the use case.'* Documenting Configuration Details *'Keep configuration details required to run an application.'* Documenting Role and Access Control *'We maintain role table to keep access and authorization on user stories'*
Code Organization	Re-modularization of existing code, the Understanding purpose of code, and identification of reusable code from the codebase	*'after gradual development, some code segments become complex and contain reusable chunks...'* *'It mostly happens that the developer didn't bother to annotate comments on code or comments are vague, such situation becomes very problematic to other developers'*	Code Annotation *'we often write the comments on the top of the store procedures and in the code so that we may get instant knowledge about it.'* *'write details about complex code alongside to guide other developers when they modify code.'* Modules Re-organization *'inspect our code to segregate code chunks into modules'* *'we are manually inspecting our code to find reusable code chunks and then reorganize the code.'*

2) Service APIs: Services APIs refer to services that an application provides in the form of web services. Analysis of respondent's data showed that respondents were documenting API details which purpose was manifold.

For example, one of our respondents stated that 'we maintain List of API, with their purpose, input and output'. After asking about this activity's practice, the respondent mentioned that they were maintaining API details in a spreadsheet to let new developers understand the purpose of an API and input data that APIs take and output data that it produces. Another respondent mentioned that they 'frequently document the purpose of an API, with input and output'. While asking about the purpose, the respondent mentioned that their testing team required APIs details to implement or modify unit tests to validate those APIs. Since the development team frequently changed those services, they were required to re-document the purpose of APIs, inputs taken, and output. Some other respondents were annotating APIs details in a codebase, and they were using integrated utilities with IDE to produce API details.

3) Database View: Database view refers to entity types, entity fields, and relation and type of relationships among entities.

We found many respondents were reproducing high-level models from the developed database schema. As described in TABLE II, one of our respondents mentioned that 'database schema evolves with time. It became hard to work on report generation, adding new records, and understanding the entire schema, especially by new team members. One of our respondents mentioned that 'We have developed an in-house tool named *** that takes a table name as input and displays all associated tables'. After asking for purpose, the respondent said they had planned to write a short and optimal query for report generation. For a particular table, the tool helped them list down table fields, a relation of a selected table with other tables based on its primary key, relation with other tables on a foreign key that it possessed. Another respondent mentioned that 'we revise our databases to find out common fields across tables'. The respondent told us that their database schema grew with time and, even after careful development, some data fields become redundant across multiple tables. Thus, they had to revise the entire schema to find redundant fields in the database schema. We also found many other respondents showing their aspiration to reproduce entity-relationship Diagram (ERD) from a database schema. For instance, a respondent mentioned that: 'our team like to have a tool that could be connected to the database, and it produces ERD', 'I would like to have a diagram which would help me in explaining the DB to my fellows' and 'ERD generator from schema would be helpful tool'. Instead of spending time on ERD design, practitioners spent time developing database schema and report generations. The schema started growing with gradual development activities, and then, a lightweight representation like ERD was required to understand and discuss the database with fellows.

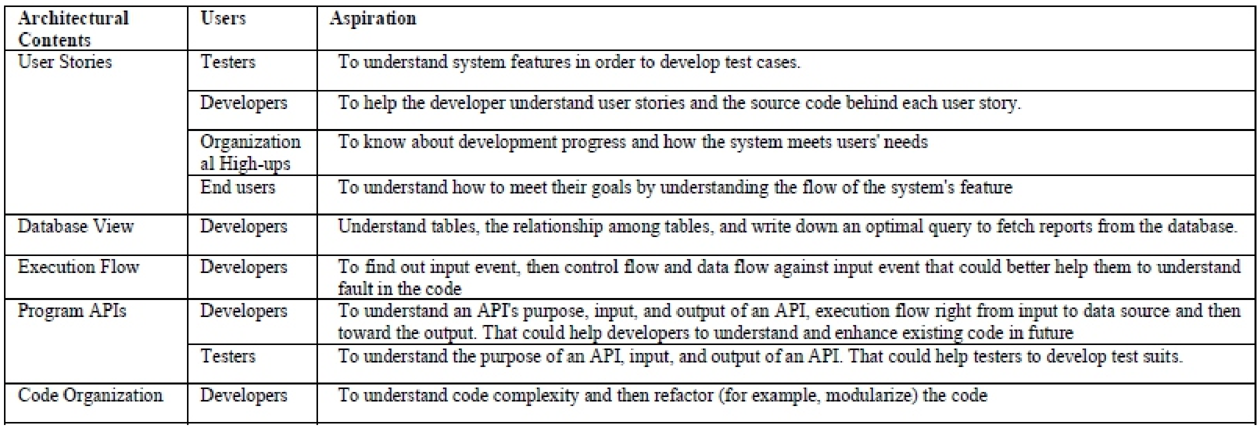
4) User Stories: We found eight out of twelve respondents tried to produce user stories from application source code.

One of our respondents expressed the challenge as the 'QA team is unable to manage their work due to a lack of knowledge about interfaces and APIs. We, therefore, work with the development team to reproduce user stories. Respondents were using different tactics to reproduce user stories. For instance, a respondent mentioned that: 'for each use case, we document sequence of GUI snippets and back-end code'. Another respondent mentioned that 'we communicate with the client on email and give them demos based on the system's GUI screenshots. Sequencing GUI snippets of an application was more convenient than any other way to communicate and document user stories. In addition to the GUI sequence, some respondents were found to document the configuration details (environment setup) required before starting the user story. Examples of configuration details, as mentioned by respondents, were: configuring endpoints in distributed systems, loading data to memory, services availability, etc. The third important aspect that respondents were documenting was roles and access control policies that helped them to reserve information about different users' roles and authorizations with each role.

5) Code Organization: Code organization refers to re-modularizing code, annotating existing code, and identifying reusable components from the codebase.

Once code undergoes a few years of development, it becomes complex and hinders new developers' ability to understand it easily. For instance, a respondent mentioned that 'after gradual development, some code segments become complex and contain reusable chunks…'. Therefore, they were inspecting existing code to find out redundant components and code complexity that raised due to continuous development. Another respondent mentioned that 'It mostly happens that the developer didn't bother to annotate comments on code or comments are vague, such situation becomes very problematic to other developers'. While explaining the statement, the respondent mentioned that sometimes it becomes difficult to understand and modify the code script written years ago. Therefore, they execute an activity to annotate, explain code scripts, and map code with specification documents. Doing so helps new developers to understand the code and further work on it. Another respondent mentioned that 'we were having a tangled code of different features. So, we identified tangled code, modularized the codebase, and defined a kind of sub-MVC project within a large project'. The respondent mentioned that they had violated design rules during gradual development for few years. Thus, code becomes complex, and re-modularization was the only solution. Now, any new development requires peer review by experienced developers to avoid design violations. The same situation was faced by another respondent who mentioned that 'we were reviewing 5 years old testing application because test cases about different features of an application were tangled among different test suites. Therefore, their team revised an application to segregate tangled test cases into their appropriate test suites.

TABLE III USERS OF PRODUCED ARCHITECTURE CONTENTS



Architectural Contents	Users	Aspiration
User Stories	Testers	To understand system features in order to develop test cases.
	Developers	To help the developer understand user stories and the source code behind each user story.
	Organization al High-ups	To know about development progress and how the system meets users' needs
	End users	To understand how to meet their goals by understanding the flow of the system's feature
Database View	Developers	Understand tables, the relationship among tables, and write down an optimal query to fetch reports from the database.
Execution Flow	Developers	To find out input event, then control flow and data flow against input event that could better help them to understand fault in the code
Program APIs	Developers	To understand an API's purpose, input, and output of an API, execution flow right from input to data source and then toward the output. That could help developers to understand and enhance existing code in future
	Testers	To understand the purpose of an API, input, and output of an API. That could help testers to develop test suits.
Code Organization	Developers	To understand code complexity and then refactor (for example, modularize) the code

B. Who are users of reproduced architectural contents?

Since users of architectural contents are actual stakeholders and therefore, it is important to know about users of reproduced architectural contents. From the content analysis, we observed that major users of architectural contents are the development and testing team. In addition to that, organization high-ups (who also have non-technical personals) and end-users also have an interest in produced architecture content. Table III describes architectural contents reproduced from source code, users of reproduced architectural contents, and purpose/goal that the user wants to achieve after having required architectural contents.

C. How are Existing Tools Helpful in Reproducing Required Architectural Contents?

Although existing tools reproduce various architectural contents from source code, however, there are practitioners' needs to be addressed or gap to be fulfilled by existing tools. In the following section, we describe features of tools conformance to practitioners' requirements, future work or opportunities for vendors of reverse engineering tools, and limitations in the selection, adoption, and use of existing reverse engineering tools.

1) Existing Reverse Engineering Tools That Meet Needs of Industrial Practitioners: We observed that most reverse engineering tools focus on producing execution flow from the source code of an application. Function dependencies [8], activity flow, dependency graphs [10, 11, 13, 20], sequence diagram [10, 11, 15, 16, 27], and flow graph [8, 13, 14] are kind of execution flow details which are reproduced by existing reverse engineering tools. We found that our respondents were also reproducing execution flow details from source code, and thus this feature of existing tools has a high tendency toward respondents' aspirations.

One of the respondent's aspirations from reverse engineering tools was to produce ERD from a database schema. There are database reverse engineering tools such as SchemaCrawler [28], MySQL Workbench [29], and SQLDatabaseStudio [30] are a few to name. SchemaCrawler is an open-source database schema discovery and comprehension tool that allows generating diagrams from SQL code. MySQL Workbench generates ERD from a physical schema that a user can use to understand, update and push back changes to the schema through forwarding engineering.

Our respondents also expressed their need for producing services APIs details from the source code of an application. There are integrated development environments (IDE) that provide a view of services by displaying service title, purpose, and contract (input data and output).

Two important respondents' aspirations, i.e., tools to work as a plug-in with IDE and produce architectural contents in their desired format, can make these tools useful for industrial practitioners.

2) Future Work for Reverse Engineering Tools to Address Needs of Industrial Practitioners: Our respondents talked about some architectural contents and were also reproducing them from source code, yet current reverse engineering tools do not reproduce these contents. In the context of database view, our respondents were interested in viewing individual table details: table fields, meta-data, relation to and from one table to another based on the primary and foreign key. Similarly, respondents were interested in examining database schema to highlight redundant table fields which were added due to gradual schema development by different developers. Although there are different database reverse engineering tools, none of them were found to address these specific concerns. Developing a reverse engineering tool that could address these concerns can be possible future work for tool developers.

Some of our respondents have developed tools to capture input system events with input data (if available) and runtime execution flow against the event. Respondents were also capturing GUI and code snippets to document execution flows. Tools are still required to address this challenge. Respondents have also expressed their aspiration that such tools need to work as a plug-in with IDE rather than to migrate the entire code to some new environment.

Respondents were producing system usage scenarios from source code information. They were capturing screenshots along with back-end code snippets. The purpose of these scenarios was to maintain the technical documentation for their clients, helping their developers understand system usage by observing the GUI sequence and back-end code against each GUI frame. Practitioners were also reproducing user stories in the form of sketches and natural language expressions to let non-technical people in the organization understand the purpose of the system. We didn't find any reverse engineering tool that produces user story details from source code information. We believe that developing such a scenario generator tool could have potential use for practitioners.

Our respondents were inspecting the existing codebase to highlight complex code segments due to gradual development activities and code evolution. Tools are required to reorganize complex code by identifying reusable code segments. Our respondents mentioned that they comment on top of code segments, i.e., application code and stored procedures. The objective was to let other people in the team understand the purpose of the code easily. Although it was recommended for each developer to keep working on this activity while developing code, they still have to repeat these tasks months after development. Reverse engineering tools that could help practitioners in producing such documents could have potential use for practitioners.

Similarly, respondents were found inspecting existing code to identify tangled code, complex code bocks, and non-reachable code. Such complexities appeared due to gradual development and code evolution by multiple developers and a developer's premature decision at the early stages of development. Due to these and many other reasons, the code becomes messy and complex and hence, needs to be further modularized. Tools can be developed that could help the development team to highlight code complexities during or after development.

3) Limitations in the Selection, Adoption, and Use of Existing Reverse Engineering Tools: Apart from the architectural contents that current reverse engineering tools reproduce, some other factors influence selection, adoption, and use of reverse engineering tools. We observed that development teams were reluctant to use any off-the-shelf reverse engineering tool unless there is no other way to reproduce architecture contents from source code. The development teams were willing to put effort into unaided browsing or to contact knowledgeable persons rather than exporting code to any reverse engineering tool to produce architecture contents. We suggest that future reverse engineering tools be pluggable with the integrated development environment (IDE) to let practitioners produce architecture content without exporting their code.

Existing reverse engineering tools provide an abstract and lightweight representation of the system either by using UML (accepted as de-facto standard) or tools' own modeling notations. Such models would have good empirical results on the system's understandability in a closed environment; however, after analyzing respondents' data, we found that respondents are reluctant to use any modeling notations or language that requires expertise to use. As one of our respondents mentioned that "architectural contents are reverse engineered from code, but we do not draw UML diagrams". Similarly, another respondent mentioned that "we sketch lines and boxes to document execution flow". Respondents were drawing sketches (having their own notations) on board or paper, and then they were made part of the document. We suggest that tools shall enable users to define their modeling notations, maybe because users are already using these notations in their organizations.

We observed that existing reverse engineering tools produce abstract and lightweight models of the system such as data flow diagrams, package diagrams, class diagrams, etc. These models would be having a high impact on understandability and communication at the early stages of development when code is not developed. However, once an application is developed, these models may not be sufficient for developers to understand the developed system. Once code is available, then the most desirable model for understanding the system is code itself. As our respondents mentioned that, 'execution flows are maintained with the help of code, and GUI snippets,' and 'code is reorganized by inspection of code itself' etc. therefore, we suggest that any reverse engineering tool to be developed in the future should provide close resemblance with code for the ease of users.

5. VALIDITY THREATS

This research aimed to identify architectural contents that industrial practitioners use to produce from source code and their expectations from automated tools to produce these contents from source code. To identify the automated tools and the architectural contents, we performed a literature review. Digital libraries like google scholar, IEEE, ACM, Spring link, etc., as well as some grey literature, were searched for our research purpose. However, there is a chance that we missed some tools due to their unavailability in these libraries. Also, as we included the tools and architectural contents in the English language, therefore, we might have missed some literature that might be available in some languages other than English.

We selected 12 interviewees from different organizations with different expertise, experience, and job responsibilities for our study. Since all these individuals have international projects, thus all of them have experience with a multicultural environment. Therefore, our interview supports the generalization of the result.

Another validity threat to the study was the language barrier as all interviewees have different first languages. This issue was addressed by conducting the English language interview since all interviewees were proficient in the English language. However, there may still be a chance of misinterpretation of some sentences.

Before the start of the interview, all the interviewees were briefed about the interview process and their responsibilities, along with the general terminologies. Thus, before starting the actual interview, all the interviewees and interviewers were on the same page in all these aspects.

6. CONCLUSION

In this research, our efforts were to identify architectural contents that industrial practitioners use to produce from source code, and they also expect automated tools to produce these contents from source code. To meet this goal, we have set our objective as: identify architecture contents that practitioners reproduce from source code, identify users of produced architectural contents, and find out how existing tools help meet the needs of industrial practitioners. To meet our first two research objectives, a qualitative study is performed to select respondents from a homogeneous group. Respondents had experienced producing architecture contents from source code when code had become complex due to continuous development from the last few years. From our analysis, we found that major contents that practitioners produce are: user stories, database views, execution flow, Program APIs, and code organization. We found that major stakeholders of produced architectural content are the development team, testing team, organizational high-ups, and end-users. To address our third research objective, we have studied existing tools and presented their purpose, input artifacts that tools take, and output models that they produce. Then, we compared the tools feature set with our analysis results. We also found architectural contents that existing reverse engineering tools produce, and they are also expected by industrial practitioners, such as program execution flow, ERD from the database schema, etc. However, practitioners require various other architectural contents, but existing tools do not support them. We also observed that existing reverse engineering tools provide architectural contents in modeling notations that are either UML or vendor's defined notations. However, a tool must be flexible in letting its users define their own notations convenient for them to understand easily. It must also be considered that merely providing modeling notations to present high-level design or architecture may not be sufficient for users to understand the system. Once there is code in hand, then the high-level design may also include (or at least provide a mapping to) the code because the codebase is much familiar stuff for practitioners once an application is developed. Any reverse engineering tool to be developed in the future may also focus on practitioners' ease in the development environment. As we stated in the previous section, practitioners are reluctant to export their code to any off-the-shelf tool to reverse engineer architectural content. A tool that could be integrated with IDE could be having various advantages to practitioners such as the ease with respect to use and efforts.

REFERENCES

[1] E. J. Chikofsky and J. H. Cross, "Reverse engineering and design recovery: A taxonomy," IEEE software, vol. 7, no. 1, pp. 13-17, 1990.

[2] D. R. Harris, H. B. Reubenstein, and A. S. Yeh, "Reverse engineering to the architectural level," in Software Engineering, 1995. ICSE 1995. 17th International Conference on, 1995, pp. 186-186: IEEE.

[3] L. Hochstein and M. Lindvall, "Combating architectural degeneration: a survey," Information and Software Technology, vol. 47, no. 10, pp. 643-656, 2005.

[4] S. G. Eick, T. L. Graves, A. F. Karr, J. S. Marron, and A. Mockus, "Does code decay? assessing the evidence from change management data," IEEE Transactions on Software Engineering, vol. 27, no. 1, pp. 1-12, 2001.

[5] C. Stringfellow, C. Amory, D. Potnuri, A. Andrews, and M. Georg, "Comparison of software architecture reverse engineering methods," Information and Software Technology, vol. 48, no. 7, pp. 484-497, 2006.

[6] M. Lindvall, R. Tesoriero, and P. Costa, "Avoiding architectural degeneration: An evaluation process for software architecture," in Proceedings Eighth IEEE Symposium on Software Metrics, 2002, pp. 77-86: IEEE.

[7] D. L. Parnas, "Software aging," in Proceedings of 16th International Conference on Software Engineering, 1994, pp. 279-287: IEEE.

[8] K. Wong, S. R. Tilley, H. A. Muller, and M.-A. Storey, "Structural redocumentation: A case study," IEEE Software, vol. 12, no. 1, pp. 46-54, 1995.

[9] Rigi, A visual tool to understand legacy systems (December 2020). Available: http://www.rigi.cs.uvic.ca/

[10] T. Systä, K. Koskimies, and H. Müller, "Shimba—an environment for reverse engineering Java software systems," Software: Practice and Experience, vol. 31, no. 4, pp. 371-394, 2001.

[11] Imagix 4D (December 2020). Available: www.imagix.com

[12] N. Sangal, E. Jordan, V. Sinha, and D. Jackson, "Using dependency models to manage complex software architecture," in ACM Sigplan Notices, 2005, vol. 40, no. 10, pp. 167-176: ACM.

[13] L. Inc. (January 2021). LDM tool. Available: www.lattix.com

[14] P. Smacchia. (Febuaray 2021). NDepend. Available: http://www.ndepend.com

[15] B. Pages. (Febuaray 2021). BoUML. Available: https://www.bouml.fr

[16] Y. S. Gmb. (Febuaray 2021). UML Lab. Available: https://www.uml-lab.com

[17] U. ObjectAid. (January 2021). Explorer for Eclipse. Available: https://www.objectaid.com

[18] AgileJ StructureViews (Febuaray 2021). Available: https://marketplace.eclipse.org/content/agilej-structureviews

[19] C. Kothapalli. (January 2021). MaintainJ. Available: maintainj.com

[20] ModelGoon (December 2020). Available: www.modelgoon.org

[21] H. Cai and R. Santelices, "Diver: Precise dynamic impact analysis using dependence-based trace pruning," in Proceedings of the 29th ACM/IEEE international conference on Automated software engineering, 2014, pp. 343-348.

[22] Diver (January 2021). Available: https://bitbucket.org/haipeng_cai/subjects/src/master/

[23] S. David, "Doing qualitative research: a practical handbook," ed: SAGE publications Ltd, 2000.

[24] K. Carley, "Coding choices for textual analysis: A comparison of content analysis and map analysis," Sociological methodology, pp. 75-126, 1993.

[25] J. Saldaña, The coding manual for qualitative researchers. Sage, 2015.

[26] R. E. Boyatzis, Transforming qualitative information: Thematic analysis and code development. sage, 1998.

[27] IBM Rational Software Architecture (December 2020). Available: https://www.ibm.com/us-en/marketplace/rational-software-architect-designer

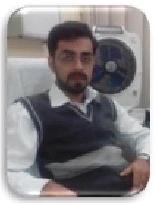
[28] Schemacrawler (December 2020). Available: https://www.schemacrawler.com/

[29] MySQL Workbench (January 2021). Available: https://www.mysql.com/products/workbench/

[30] Sqldatabasestudio (January 2021). Available: https://sqldatabasestudio.com/

Samir Obaid is lecturer in computer science department at Capital University of Science and Technology, Islamabad. Previously, he was working as software engineer at software development organization and his responsibilities were: test planning, test design and automation. He did MS in computer science from ARID university Rawalpindi, Pakistan. His research interests are model driven engineering, model-based testing, and software reengineering.

Ibrar Arshad is currently associated with the Department of Computer Science at Capital University of Science and Technology, Pakistan. He received his Master's degree in 2012 from Blekinge Institute of Technology, Sweden. His area of research is software requirements, software engineering practices in small and medium-scale organizations.



Muhammad Usman Abid did MS in software engineering from Riphah International University Islamabad, Pakistan. Currently, he is working as a lead software engineer at reputed software development organization in Pakistan. His research interests are information systems development, empirical software engineering and software reengineering.

ISSN 2210-142X

Deep Sentiment Approaches for Rigorous Analysis of Social Media Content & Its Investigation

Amit Kumar Sharma[1], Sandeep Chaurasia[2], and Devesh Kumar Srivastava[3]
[1]Department of Computer Science and Engineering, Manipal University Jaipur, India, 303007
[2]Department of Computer Science and Engineering, Manipal University Jaipur, India, 303007
[3]Department of Information Technology, Manipal University Jaipur, India, 303007
E-mail address: [1]amitchandnia@gmail.com, [2]chaurasia.sandeep@gmail.com, [3]devesh988@yahoo.com

ABSTRACT

Social media has a very important contribution to human lives today. Through social media platforms people can share their information, ideas, knowledge, and activities with connecting people in the form of videos, images, texts, and audios. In the context of sharing information, incorrect information is also shared along with the correct information. In this way, unauthentic (fake news), misleading (rumors), abusing, toxic, extremist contents are also shared through social media platforms. This paper reviews the influences of social media contents. In this context, vector representation of the social media sentences, word embedding models has been best applied for better accurate results. Natural language processing (NLP) and text analysis techniques is being used to extract useful information from social media content. The NLP techniques are widely used for correcting the sentences and identifying their meaning also. Currently, machine learning (Decision Tree, Random Forest, SVM, Naïve Bayes) and deep learning (LSTMs, BLSTMs, GRUs, CNNs) models are successfully being implemented to classify social media contents. In the comparative study of different works of literature and results from LSTM deep learning model have been proved that deep learning and the word embedding model provide better accurate results for social media contents categorization.

Keywords: Deep Learning, Machine Learning, NLP, Rumors, Social Media, Text Analysis, Word Embedding.

1. INTRODUCTION

Social networking platforms like Twitter, Facebook, Instagram, and WhatsApp, etc., are widely used by billions of people to interact with each other. Through these platforms, people show their views on the post by sharing, liking, replying to those posts. Also unauthentic, toxic contents, rumors, fake contents, abusive contents, etc., are also shared on social platforms that eventually affects the human mind and divert them from the society [9] [23] [24]. Such contents or post increases negativity and criminal activities in the society. Text, images, videos, and speech contents are widely used on these platforms in different languages. In this review, we are focusing on text and speech contents because of its wide availability in the literature, and mostly contents are circulated in text and speech form in the social media. Social media platforms provide the facility of language plurality, by using these options users can send their message or post in their languages [5] [21].

Analysis of the contents, such as syntax, sentiment, and semantic analysis with huge sparse data pose challenge towards the researchers. There are several models which are present in the literature that perform analysis of different languages and different forms of data. N-gram language model is used to identify language features like spell correction, speech recognition, next word suggestion, text summarization, etc. [9] [16]. Misinformation is spread on social media to create confusions among people as well as they have harmful consequences. Stopping these rumors and misinformation to spread is the biggest problem that researchers are facing. It takes only a second for the public to share the information but validating the information is necessary else it will have the negative impact. Celebrity death rumor, chain mails, falsities about the Social Network, etc., are the examples of the unauthentic news [5].

In the current scenario the novel Corona virus which firstly emerged in the Wuhan city of China in December 2019 is continuing to infect people across all countries. This corona virus (Covid19) has become the biggest threat to all the countries. World Health Organization (WHO) and the leaders of the countries are communicating with their public and doing awareness programs through social platforms. In between this, fake news and rumors are also being spread continuously through social platforms and impacting the society greatly. For example, a rumor on the 14th and 15th march of 2020 started getting viral by Americans that "martial law is coming". This message was hard to stop or even trace. Due to this misleading information viral on social media, the U.S. politician Sen Marco Rubio, R-Fla tweeted and debunked this rumor. Normally, Indian people get daily messages regarding health topics and are receiving more messages in this Corona pandemic. Such type of messages is not sent after fact-checking and those messages impact on people's habits and lifestyle. Some organization works for debunking or destroying such rumors, unauthentic messages, or post by fact-checking. The Indian government also launched a help desk MyGov Corona for awareness about the Coronavirus pandemic and tackles social media services of spreading misinformation.

Social media have contributed a lot to society in the modern era and through which society can share their information and news on public platforms. Many social, educational, governments, and private organizations deliver their messages or news to people through social media. As many people have benefited from social media, they also suffered a lot. Different types of content shared on social networking website which is very harmful and can divert people in the wrong direction. Many researchers have explained the side effects caused by such content in their researches. In this sequence, unauthentic, misuse, toxic, rumors, negative, fake, domestic violation, illicit drug, abuse, extremist, cyber bullying, etc., contents can affect human minds as well as divert people from the society. Following are the social impacts because of social media:

• **Rumor:** Social media have some harmful, unusual, fake, unauthentic contents which are known as Rumors. Rumor is shared to misguide the people. A rumor is propagating to damage the reputation of persons or an organization. Without verification, rumor reaches thousands of people immediately and causes serious damage.

• **Domestic Violence:** It is a very critical issue and harmful to society. The victim sometimes shares their story or health issues on social media. In current years, domestic violence crisis support organizations are very much active on social media and serve support to them.

• **Abusing Contents:** Abusing content is also posted on social media. The effects of such content are on the psychology of teenagers and also it demoralizes them.

• **Extremist Contents:** Jihadist propaganda is spread on social media by extremist organizations and they spread their propaganda to mislead the people and for recruits.

• **Cyber Bullying:** It includes sharing negative, false and harmful messages on social media about someone else causing humiliation and embarrassment them.

• **Toxic Content:** The presence of toxic content has become a major problem for many online communities. It includes racism, sexual predation, and other negative behaviors that are not tolerated in society.

• **Illicit Drug:** In current. Social media has become a popular platform to offering new drugs and alcohol to Youngers and teenagers.

Social media content analysis is very important to stop the rumors and false information. For this, Natural Language Processing (NLP) is used which helps in linguistic text analysis. The linguistic text analysis can read the text of the different languages. In the NLP process, it is required to convert words into numbers or vectors so; word embedding methods are used to convert words and phrases into vectors. In recent years, word embedding shows boom in the performances of text analysis tasks. tf-idf, Word2Vec, Doc2Vec, and GloVe models of word embedding are widely used for vector representation of words and phrases. An NLP language method that is n-gram is used to solve the problem of language identification from social media contents. In recent years, machine learning algorithms are also used to identify the patterns from the text and help to classify the social media contents. The supervised machine learning methods (Decision Tree, Logistic Regression, SVM, Na▪ e Bayes, KNN) are being successfully used for the classification of social media content [9] [13] [18]. In this context, deep learning architectures are also being applied in the field of text analysis, NLP, speech recognition, image processing, etc. Apart from this, RNN, LSTM, BLSTM, GRU, CNN are different methods which are used for text and image analysis [2] [11] [17].

The further paper is organized are: Section 2.0 presents the related works. Section 3.0 presents the methodologies of text analysis. Section 4.0 presents the comparative study of different social media problems. 5.0 presents the experiment evaluation and results analysis of deep learning text classification model. Section 6.0 presents the analysis and discussions of different social media problems and Section 7.0 concludes the paper and presents the future scope of the research work.

2. LITERATURE REVIEW

There are lots of content in the social media platforms in the form of text, videos, images, etc. These platforms provide service-specific applications which are governed by some organizations. Social media facilitated the growth of online social networks by connecting people of similar interests. People hold both kinds of sentiments; positive and negative. Social media content has the text of semantic and sentiment knowledge [11] [17] [18]. These contents have some harmful, unusual, fake, unauthentic contents which misguide people. In the literature, there are some APIs available which helps in extracting social media post's content for sentiments and sematic analysis, like Twitter API [12] [18] [19], Beautiful Soup [20], and Facebook Graph API [6] [7]. The main focus of current research is social media content analysis and to produce authentic content for society. For that researchers are collecting information manually from the news articles, Wikipedia entries, site pages, related magazines [13] [18], and some are using publically available datasets for research purposes, like movie review dataset [17], Twitter Sentiment Corpus [26], Priyo Review [25], Kaggle [9] [15], PHEME [2] etc.

In [1] the author proposed a deep learning model which was based on CNN to detect rumors on Twitter and prove that the existing state of the art methods requires improvement. In this paper, the authors proved that the CNN deep learning architecture obtained more accurate results as compared to existing machine learning algorithms. The model was trained on the publically available PHEME dataset. This research finds that the tanh activation function provides better accurate results as compared with the RELU activation function. Another deep learning model is proposed using RNN classifier for classifying tweets into rumors and non-rumors classes [2], and the results are compared with the machine learning classifiers. In this paper, the authors trained the model with different features. The first model was trained with textual and user characteristics features and traditional machine learning classifiers (SVM, KNN, Gradient boosting, and Random forest). The second model was trained with applied LSTM deep learning architecture on only tweet text, and the third model was trained with tweet text and user metadata features and LSTM deep learning architecture. Second model is performed better than the machine learning-based classifier. This research also suggested that the machine learning approach is a very time-consuming process and cannot preserve the semantic representation and sequential representation of the sentences. This research also suggested that the deep LSTM model learns the hidden information from the tweet text which was difficult to learn from the handcrafted feature and machine learning features.

In [3] the authors focused on the classification of both rumors and non-rumors features and noticed that the classified results detect only from the rumored features, so binary classification not might provide beneficial results. To solve this, existing literature deals with only rumored features and suggests a new approach which is one class classification classifier with one class feature. Rumored features were extracted from already available and detected features of rumors on a social network. In the paper, the data trained on seven class classifiers, namely Autoencoder, Gaussian, K-Means, KNN, SVDD, OCSVM, and PCA which is applied on two major datasets, namely Zubiagaset and Kwonset. Performance of the OCC model was observed by a high level of F1-score.

The approach achieved a 74.30% F1-score for the Zubiagaset dataset and 9398% for Kwonset dataset. In another paper a deep learning model is proposed for detecting breaking news, rumors instead of long lasting rumors [4]. In this, word vectors were generated from the word embedding model and LSTM-RNN deep learning approach is applied for identifying the rumors. This study also suggested that a deep learning approach with word embedding is performed better than the state of the art method in term of precision, recall, and f1 measures. In [5] a model for rumor detection is proposed which is based on the RNN deep learning approach for learning hidden knowledge from the posts. Experimental results compared with the existing machine learning approach with handcrafted features. In this research, the machine learning model trained with a decision tree, SVM, random forest classifiers, and deep learning model was trained with tanh-RNN, LSTM, GRU-1, GRU-2. GRU-2 with multiple hidden layers and provides better accurate results as compared to different models of deep learning and machine learning model. The experimental results have been measured in terms of precision, recall, F1, and accuracy. A deep learning model for automatic content categorization in multi classes on online posts was also proposed [6], which proves that the deep learning model provides the solution for real-world problems over the traditional machine learning techniques. This research provides the results; comparisons between word vectors generated by domain-specific word embedding and pretrained word embedding. The performances were evaluated with 5 deep learning models, namely RNNs, LSTMs, GRUs, BLSTMs, and CNNs. Also, performances were evaluated with machine learning approaches, namely SVM, RF, DT, and LR. With GloVe embedding, GRUs and BLSTMs performed the highest with scores

•of 91:78% and 91: 29%, respectively. Further, researchers has also identified domestic violence problems through content [7], shared by victims on social media and some social service organizations search these types of content and find the victim to help them. In this model, data was extracted from Facebook by using the Facebook Graph API and labeled the data into critical and uncritical posts.

The model, first accurately evaluated with traditional machine learning models using different ML classifiers with different word setting. LR classifier and tf-idf word embedding with stemming of the words obtain 90.74% accuracy. In the second evaluation of deep learning, Word2Vec+LSTMs obtain 93.08% accuracy and Glove+GRUs obtain 94.26% accuracy. In machine learning, evaluation results were evaluated with NB, SVM, RF, LR, and DT and in deep learning, evaluation the results were evaluated with CNNs, RNNs, LSTMs, GRUs, and BLSTMs classifiers. It is concluded that deep learning models achieved better performance results than traditional machine learning models.

A C-BiLSTM (Convolutional Bi-Directional LSTM) deep learning model for automatic identifying inappropriate (abuse, rude and discourteous) comments on language is proposed [8], and is applied in the real-world language which significantly performs better than both handcrafted feature and pattern-based approaches. [9] proposed a machine learning approach for the detection of abusive content on social media. In this, the skip-grams feature improved the results as compared to previous approaches.

Cyberbullying incidents in social media platforms are also detected by the deep learning model [10]. In this, the model was trained on different publically datasets, namely Wikipedia, Formspring, and Twitter, and experimental evaluation is done with four deep neural network models, namely CNN, LSTM, BLSTM, and BLSTM with attention. For vector representation of the sentences, word embedding models (random, GloVe, and SSWE) was applied. After validating findings from Wikipedia, Twitter, and Formspring, the work was expanded on the new YouTube dataset and investigated the performance of the models in the new social media platform. The experimental result shows that the DNN model has successfully been implemented for all social network platforms and the results also suggest that the performance of the DNN model is better than the machine learning model. A machine learning classification model is proposed for detecting bullying and aggression posts on Twitter [11]. In this, the preprocessing process is applied, like removing stop words, URLs, punctuations, repetitive words, stemming, and labeling. For creating word vectors, the Word2Vec model is applied and it detects the sentiments using the SentiStrength tool. For classifying the results, J48, LADTree, LMT, NBTree, RF, and functional Tree is applied and obtained 90% accuracy to detect bullying and aggression or hate speech comments.

Further, literature showcases a deep learning-based sentiment analysis model for classifying the tweets into extremist and non-extremist [12]. In this model, after preprocessing of the sentences, the word embedding model is applied for vector generation of the words. The dataset was trained with LSTM+CNN deep learning network and performance were evaluated in term of accuracy, recall, precision, and f-measures. The model accuracy obtained by 92.66% and its performance is better than machine learning and other deep learning classifiers. Sentiment analysis and identification of the toxic online comment by using the SentiWordNet tool is in focus [15]. In this research, researchers have explored various aspects of sentiment detection and their correlation and obtained the results by using a toxicity detection tool. A CNN deep learning architecture is also proposed for opinion mining [17]. This research mainly focused on sentiment analysis of the movie review. In this process pre-trained Word2Vec model was used for vector representation and information gain, the model was trained with

CNN deep learning architecture and the accuracy of the model was 97.3% which shows the importance of the deep learning concepts. A deep neural network model is proposed for automatically identifying a subset of webpages and social media content that has extremist content [13]. For dealing with different language challenges, the script Unicode was applied to converts all text into the corresponding ASCII characters.

The deep neural network with Doc2Vec was used to classify the text into extremist and non-extremist. Moreover, a modeling-based approach to identify illicit drug-related contents from social networking sites is also proposed [14]. In this, a dataset was created from the NIDA website with 371 hash-tags. Word embedding (Word2Vec) model was used for vector generation and LDA topic modeling algorithm is applied to identify the illicit drug-related contents. The model has obtained 78.1% accurate illicit drug contents. A language identification model [16] is proposed for short segments of the text contents. In this model, the n-gram model was used to correct the sentences of different languages posts and a common n-gram distance based novel model was used for classifying the results. The Common N-Gram text classification is used with different classifiers, namely Logistic Regression, SVM, Na ▪ e Bayes, and Random Forest.

3. METHODOLOGIES OF TEXT ANALYSIS

Social media users are continuously increasing day by day and billions of users are sharing a huge amount of content in different forms like texts, videos, and images, etc., daily. Text is the most common form of content which is used in social media platforms, so it is necessary to find the knowledge from the huge text contents. In text analysis processes, a huge amount of unstructured data is collected from different sources by applying different techniques and methods, and converts these unstructured data into knowledgeable structured data. The text analysis process helps to explore results and identify patterns, keywords, and attributes of the unstructured text. There are some different methods and processes which have been used in previous researches for text analysis:

A. Process of Text Analysis

1) Natural Language Processing (NLP): NLP is widely used in the process of text analysis [19], which explores how a computer system becomes an expert system in the context of understanding natural human languages and develops some tools and techniques that can perform to manipulate human languages to the desired task. In existing literatures, lexical and syntactic analysis of the NLP has been applied in the text analysis process, for example, in a bag of words representation technique, only lexical components of the text are considered. Sentimental and semantic analysis of the words is broadly used in the text analysis process. Semantic analysis has been successfully applied in the text and has improved the results [19, 21]. The n-gram model is a probabilistic model which is applied in many NLP and the text analysis process [9] [16]. In n-gram model, the sequence of words is extracted from the sentences or text and probabilities are assigned to them. The n-gram of the size 1 is known as a unigram, size of 2 is known as bigrams, size of 3 is known as trigrams, and so on. The n-gram model is used in different tasks like sentence correction, spell correction, word breaking, suggestions while typing, text summarization, etc., and also uses supervised machine learning models for developing the best features.

2) Data Preprocessing: In-text analysis process, there is a requirement of cleaning the data because data are very sparse, unstructured, and noisy. So, in data preprocessing, lexical analysis, tokenization, stop word removing, case folding, special character removing, deleting hyperlinks, normalization, stemming etc., is done on the data [19] [25].

3) Word Embedding for data representation: The text analysis deals with huge, raw plain text data, and the machine only understands the numbers. Word embedding is the model that is used to extract features from the text data and convert these texts or word sentences into vectors or numbers so that the machines can understand them.

This model is used to project in continuous of the words and deal with the syntactic and semantic similarities between the words of a sentence. The word embedding model has been very effectively applied in NLP, machine learning, and deep learning processes. Word Embedding methods are generally trying to map a word using a dictionary to a vector. Frequency-based word embedding (tf-idf) and predication based word embedding (Word2Vec, GloVe, Doc2Vec) methods have been successfully applied in the research for representing word vector. In frequency-based embedding (tf-idf), the document's text transforms into numeric vectors, this representation is called the Vector Sparse Matrix (VSM) or Bag of Words model. In tf-idf, a particular document weights each word; this word weight makes the importance of the word in the document. The prediction based Word2Vec method is obtained from two models that are continuous bag-of-words (CBOW) and Skipgram, both models worked with neural network concepts [14] [18] [22].

The Word2Vec CBOW Model uses the surrounding words as inputs to predict a word in a sentence. CBOW is a word embedding model that predicts the target word x0 from the surrounding contextual words, C i.e. the goal is to maximize P(x_0|c) throughout the training set. The distance between the current vectors assigned to x0 and to c is inversely proportional to this probability. The model's goal is to reduce the distance between x0 and 's current vectors (and enhance the probability P(x $_0$|c). By repeating this process over the whole training set, we may build vectors for words that co-occur and tend to be closer together. The input of CBOW is one-hot encoded vectors V, as shown in [Figure 1]. This means that for each vector, only one of the V units will be 1, while the rest will be 0. The CBOW model works as a three-layer basic neural network, with two weight functions in the input layer, hidden layer, and output layer. A V_x N matrix W can be used to represent the weights between the input and hidden layers (i.e V is the number of words and N is the number of neurons in the hidden layer). The N-dimension vector representation v_x of the related word from the input layer is represented in each row of W. To construct the word vectors, the input layer's weight function matrix (i.e. W input) is used.

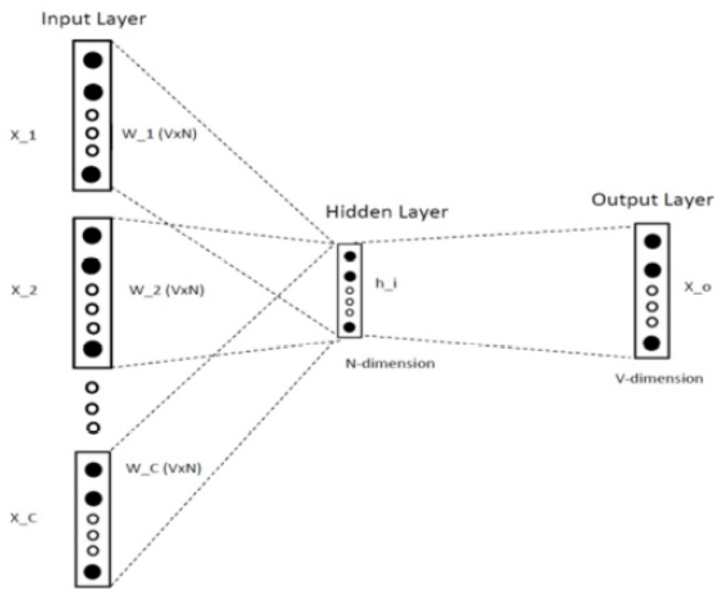


Figure 1. CBOW Model

The skip-gram model is a version of the CBOW model that is reversed. The target word is now at the input layer in the skip-gram model [Figure 2], while the other words in the window are at the output layer. We still utilize v_x as the input vector for the input layer's only word, and the hidden layer outputs *h* are defined in the same way as in CBOW, which means *h* is just copying a row of the input -> hidden weight matrix, W, mapped with the input word x_i.

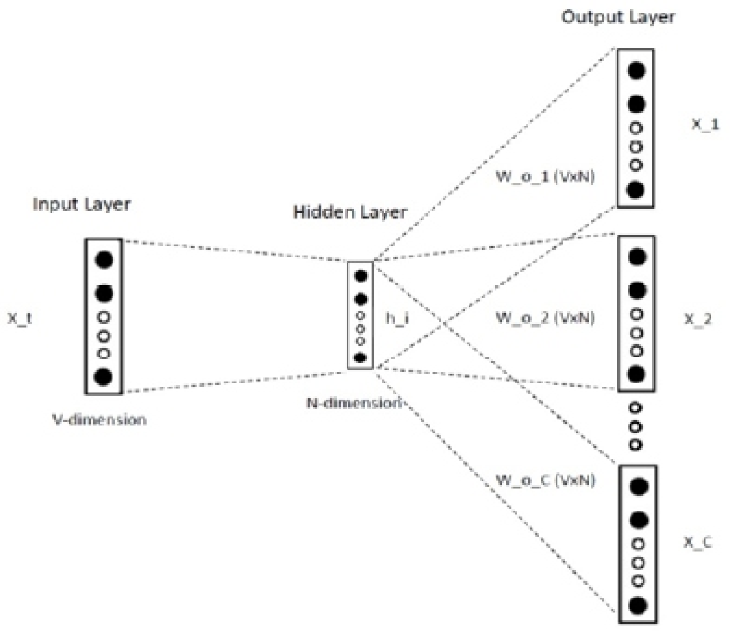


Figure 2. Skip-gram Model

The Skip-gram model attempts to anticipate the surrounding keywords by identifying relevant word representations in a sentence or text. The goal of the Skipgram model is to maximize the average log probability (i.e. $logp(x_1, x_2 \dots x_C|x_t)$) from target words x_t from context, where C $(x_1, x_2, x_3, \dots, x_C)$. A larger C indicates that there are more training examples, which can lead to a greater accuracy. However, it also raises the expense of training time. In the word2vec model, the vector of the words is generated by achieving the process of prediction of surrounding words in a sentence. While in GloVe model, the model learns by constructing a co-occurrence matrix that count how a word frequently appears in a context. In this model, firstly a co-occurrence matrix X is constructed from the training dataset. Where X_{ij} is the frequency of the word I- co-occurring with the word j $X_{ij} = \sum_k^V X_{ik}$ is the total number of occurrences in word i in the dataset. In the second step, the factorization of X get vectors and reduces noise by identifying relevant words.

Another prediction based word embedding model is Paragraph Vector Doc2Vec (PV-Doc2Vec) model. The idea of the model is inspired from the Word2Vec model. This model is the extension of the Word2Vec model concept. In the CBOW model of the Word2Vec, the trained model predicts a center word by using context words of a sentence. Similarly, PV-Doc2Vec selects a sample that is randomly set of consecutive words from a paragraph and predicts a center word from the randomly sampled set of the consecutive words by taking as input paragraph id and context words. The model is divided into three sections: 1) Paragraph matrix: the matrix where each column represents the vector of a paragraph, 2) Average/Concatenate: it checks whether the paragraph matrix and word matrix are concatenated or averaged, and 3) Classifier: the hidden layer vector (the one that was concatenated/averaged) is sent into the classifier, which predicts the center word in [Figure 3].

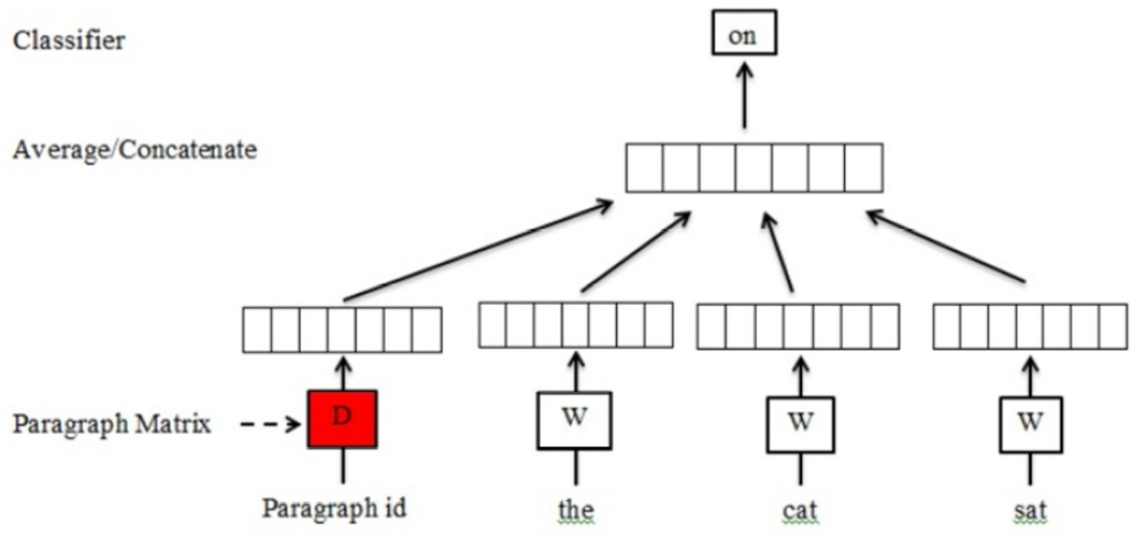


Figure 3. Doc2Vec Model

Generating a new model with word embedding process takes too much time; to solve such issue, a pretrained word embedding model is used which is already trained by someone and is publically available for research. Currently, pre-trained Word2Vec and pretrained GloVe model is widely used for word vector representation [6] [7] [17].

B. Multilingual Text Analysis

Social media user posts their messages in different languages. The NLP task requires a single language platform that can show all text in a single language [16]. By using such single language platform, we can get good results of semantic analysis.

C. Speech to Text Conversion

Currently, artificial intelligence, automation and many more applications has been successfully used in the speech recognition process to develop its applications. Speech recognition is the process in which speech or spoken texts are converted into the written text. Google Speech Recognition is one of the easiest methods used for speech text analysis.

D. Machine Learning

Previously, machine learning concepts are broadly used for social media content categorization with more accurate and effective results. The use of NLP methods with machine learning concepts, the generated model helps to identify patterns from the customer's message. Supervised Machine Learning (SML) is suitable for the classification of social media content. Social media service by machine learning helps in enhancing media quality, helping the brand reach to the target audiences, maintaining security, handling, and automating data. Logistic Regression (LR), Decision Tree (DT), Support Vector Machine (SVM), Random Forest (RF), Na ・ e Bayes, K-Nearest Neighbor (KNN) are some machine learning techniques which have been successfully used in social media applications.

E. Deep Learning

Currently, deep learning approach is used to detect unsupervised and unlabeled data; which is also known as deep neural networks. The method has been successfully implemented to develop different applications of speech recognition, image processing, bioinformatics ∞, text filtering, NLP, etc. In a deep learning process, the trained model automatically learns and performs a classification task from the text, image, and audio. A deep learning model trained with a huge number of labeled data and neural network architecture. In the traditional process of neural network architecture, 2-3 hidden layers were used to detect the feature from the data but now, many hidden layers are being used to train the model for directly learning the features from the data. RNN, LSTM, BLSTM, GRU, CNN model of deep learning are used to detect features and classify the data [18] [27].

1) Recurrent Neural Network (RNN): The feedforward neural network process, cannot predict the next word in a particular sentence because there is no relation between previous output and current output. To overcome this issues the RNN architecture is used for the prediction of the next word in a particular sentence. A RNN is a deep learning concept, in which neuron connection is established with a direct cycle. It means output depends on previous neurons as well as present inputs [Figure 4]. The concept of RNN solved various problems of NLP like handwritten recognition and speech recognition etc

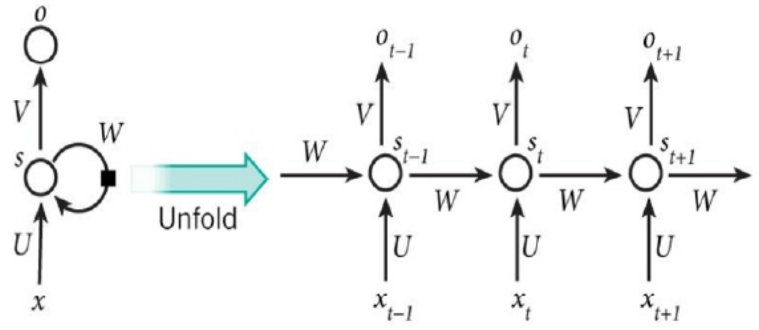


Figure 4. RNN Model

2) Long Short-Term Memory (LSTM): RNN uses a back-propagation algorithm, but it is applied for every timestamp, and back-propagation has a vanishing gradient problem. To solve this problem, a specific RNN architecture is developed that is LSTM, in which a model was designed for learning long term dependencies. In the LSTM process, the activation function is not used for its recurrent components. In the architecture of LSTM, several units of blocks are implemented with four gates which are input, forget, and output gate which uses logistic function to control information flow in the network [Figure 5].

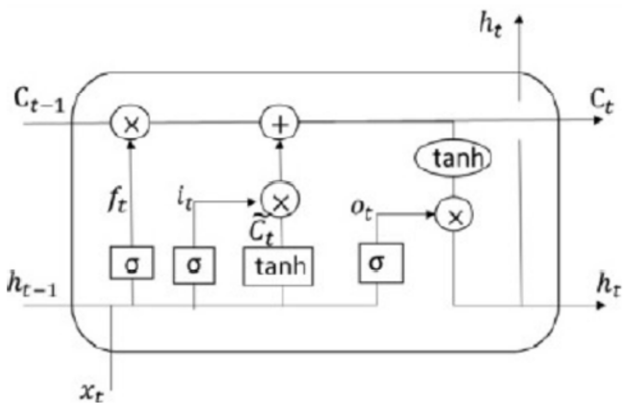


Figure 5. LSTM Model

3) Bidirectional Recurrent Neural Networks (BLSTM): The BLSTMs model is trained with two LSTMs model instead of one LSTM in the input sequence. Implementation of this architecture, the first LSTMs input sequence is to be as it is and the second LSTMs input is to be a reverse copy of this input sequence. The BLSTMs provides additional context to the network and also improve the performance of the results [Figure 6].

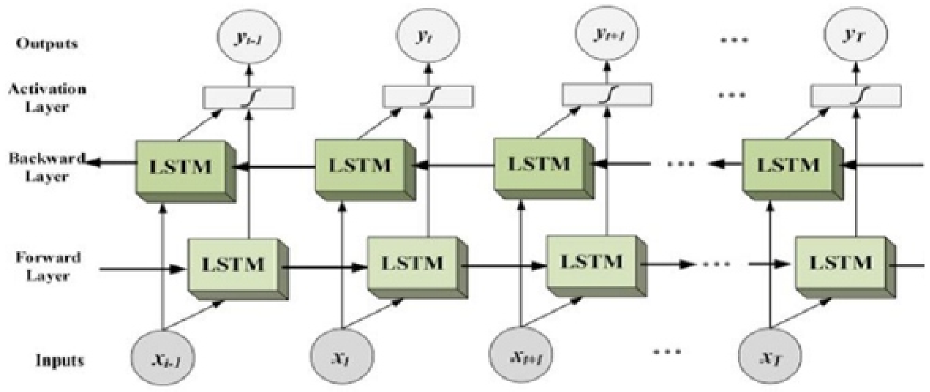


Figure 6. BLSTM Model

4) Gated recurrent unit (GRU): The GRU architecture is similar as LSTM. In GRU architecture, there is no cell state; a hidden state is used to transfer the information [Figure 7]. Only two gates were employed in the GRU's secret state: the reset gate and the update gate. GRU's update gate is similar to the LSTM method's forget and input gates in that it determines what information will be delivered and what information will be added. The reset gate of this method is used to decide how much previous information to leave or forget which a GRU is. Currently, it is not clear that which architecture is better, but GRU's tensor operations are little fast to train data than LSTM. The researcher usually tries both the architectures to identify which is better for their information.

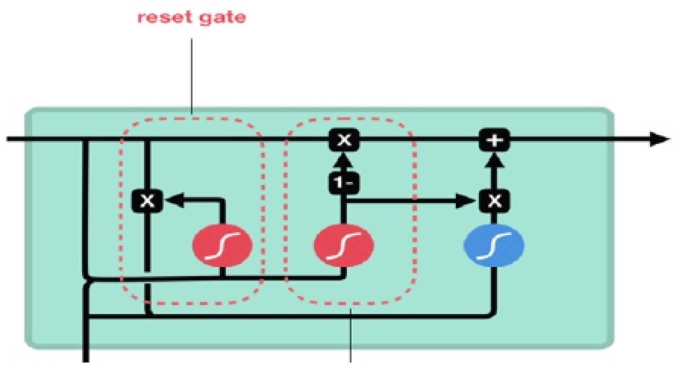


Figure 7. GRU Model

5) Convolutional Neural Network (CNN): The CNN architecture has been proven to provide outstanding results for speech and image processing. The CNN architecture includes: a) convolutional layer/s, b) pooling layer, and c) a multilayer perceptron variation [Figure 8]. The result of the convolutional layer is passed to the next layer via the convolution technique. This procedure allows for a much deeper network with many fewer parameters. A CNN model is proposed for short sentence classification [21]. The CNN model uses the Word2Vec model for feature vector representation and conducts a series of experiments and demonstrated that the suggested model performs excellently.

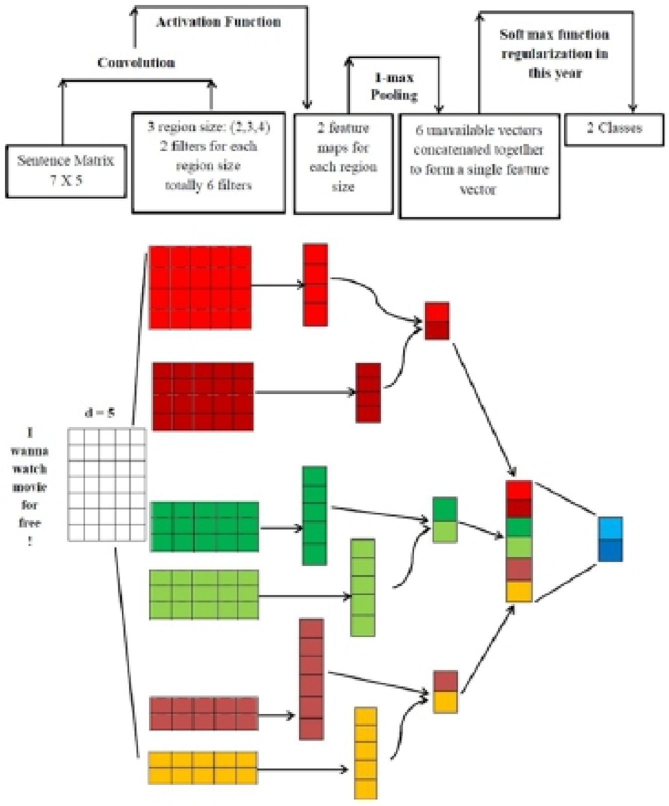

Activation Function
Convolution
1-max Pooling
Soft max function regularization in this year
Sentence Matrix 7 X 5
3 region size: (2,3,4) 2 filters for each region size totally 6 filters
2 feature maps for each region size
6 unavailable vectors concatenated together to form a single feature vector
2 Classes
d = 5
I wanna watch movie for free !

Figure 8. CNN Model

4. COMPARISON OF DIFFERENT SOCIAL MEDIA PROBLEMS

The comparative study table in [Table I] showcases different social media problems suggested by the different researchers and also found the objectives of those researches for such problems. Further, the comparative study of social media problems provides the detail of the different datasets, used methods, and techniques to solve those identified problems.

TABLE I. Comparison of Different Social Media Problems

Problems	Dataset	Feature Extraction	Methodologies	Results Analysis	Authors Name & Year
Rumors detection on twitter	PHEME	Word embedding	CNN deep learning	91.0% Accuracy	Abdullah Alsaeedi, et all., 2020
Rumor identification	Theme	User verified, sentiment, followers, hashtags, length, count status, and retweets of tweets	LSTM deep learning	0.86 F1-score for no-rumors and 0.72 F1-score for rumor	Jyoti Prakash Singh, et all., 2019

Rumor identification	Zubiagaset and Kwonset	Linguistic & language features, User profile features, Metadata features	Autoencoder, Gaussian, K-Means, KNN, SVDD, OCSVM, and PCA	74.30% F1-score for Zubiagaset with KNN and 93.98% F1-score for Kwonset with KNN	Amir Ebrahimi Fard et all, 2019
Breaking news rumors detection	PHEME	Word2Vec word embedding	LSTM-RNN deep learning	0.791 F1-measure	Sarah A. Alkhodair et all, 2018
Rumors identification	Twitter & Weibo	Refine the keywords, replicate handcrafted features.	tanh-RNN, LSTM & GRU deep learning	Twitter dataset: 88.1% Accuracy with the GRU2 method. Weibo dataset: 91.0% Accuracy with GRU2.	Jing Ma, et all., 2016
Domestic Violation content categorization	Facebook (Extracted from Graph API)	Pre-trained Word2Vec and Glove word embedding, Domain-specific embedding.	CNNs, RNNs, LSTMs, GRUs, and BLSTMs deep learning	GRUs+ GloVe 91.78% Accuracy	Sudha Subramani, et. all., 2019

Automatic ally detection of Domestic Violation on social media	Facebo ok (Extrac ted from Graph API)	Google 's Pre-trained Word2 Vec and Twitter 's crawl of Pre-trained GloVe.	RNNs, LSTMs, BLSTMs, GRUs, and CNNs deep learning.	LSTM + Word2 Vec 93.08% , GRU+ GloVe 94.26% Accura cy.	Sudha Subram ani, et. all., 2018
Detection of offensive comments	Kaggle website	N-gram, Count, TF-IDF score, Occurr ence of pronou ns, Skip-grams	Support vector machine (SVM), Logistic Regression of machine learning	86.92% Accura cy	Vikas S Chavan, et all., 2015
Automatic detection of inappropri ate language (abuse contents)	Web search queries	Rando mly initializ e the DSSM word vectors	Convolutio nal Bi-Directional LSTM deep learning	F1 Score 0.8720	Harish Yenala et all, 2017
Detecting bullying and aggression posts	Twitter data (extract ed from twitter streami ng API)	Word2 Vec	J48, LADTree,L MT, NBTree, RF and Functional Tree of ML	90% Accura cy	Despoin a Chatzak ou,et all., 2017
Detection of cyber-bullying	Wikipe dia, Twitter	Word embed ding	CNN, LSTM, BLSTM	F1-score for	Maral Dadvar et all,

incidents	, Formspring, YouTube	(random, GloVe and SSWE)	and BLSTM with attention deep learning	YouTube dataset: CNN: 0.78, LSTM: 0.14, BLSTM: 0.93, BLSTM with attention: 0.92	2018
Identification of extremist contents.	Twitter (extracted from Twitter Streaming API)	Word embedding	LSTM with CNN deep learning model.	92.66% Accuracy	Shakil Ahmad, et all., 2019
Automatically identification of extremist contents on social media and webpages	Manually collected from different sources	Doc2vec vectors	Deep neural network classifier.	93.2% Accuracy	Andrew H. Johnston, et all., 2017
Toxic content detection	Kaggle, Subversive Kaggle, Wikipedia, Subversive Wikipedia, Reddit, Subversive Reddit.	Preprocessing, labeling.	SentiWord Net	Accuracy: Kaggle 93.7, Subversive Kaggle 80.1, Wikipedia 85.5, Subversive Wikipedia 82, Reddit 94.3, Subversive Reddit 83.9.	Eloi Brassard-Gourdeau et all., 2018

Illicit Drug-related content identification	Extract data from NIDA website	Word embedding	LDA topic modeling	78.1% Accuracy	Tao Ding, et all., 2016

5. EXPERIMENT EVALUATION AND RESULTS ANALYSIS OF DEEP LEARNING TEXT CLASSIFICATION MODEL

To measure the effectiveness of deep learning approaches, we measure the performance with LSTM model of deep learning. In this process we applied a deep text classification model on PHEME rumor ottawashooting event dataset. This dataset contains total 12284 tweets, in which 5848 tweets was non-rumors and 6436 was rumors. In this research, pre-trained Word2Vec model have been used for generating word vectors. In this experiment, 90% of tweets were used to train the model and 10% of tweets were used for testing purpose. In this research, deep rumor tweet classification model was trained on different parameters which are extracted from different layers of model [Table II]. The results were analyzed from LSTM deep learning models.

TABLE II. Parameters Received from Different Layers of LSTM Model

Layer (type)	*Output Shape*	*Param #*
Embedding_1 (Embedding)	(None, 30, 300)	15000000
lstm_1 (LSTM)	(None, 150)	270600
Dropout_1 (Dropout)	(None, 150)	0
Dense_1 (Dense)	(None, 1)	151

Experiment results show the model has trained 5 times and calculates the accuracy and loss values [Table III]. The LSTM model provides the 98.59% accuracy to train the model. After every epoch the loss value is decreasing and accuracy is increasing it means the model trained very well. The model provides the 90.56% testing accuracy [Table IV, V].

TABLE III. LSTM Model Accuracy

Epoch	*Execution Time (s)*	*Loss*	*Model Trained Accuracy*
1	107	0.4965	0.7479
2	107	0.1934	0.9309
3	114	0.0946	0.9696
4	123	0.0592	0.9806
5	118	0.0371	0.9859

TABLE IV. Confusion Matrix of Test Results from LSTM Model

	Actual Rumors	*Actual Non-rumors*
Predicted Rumors	526 (TP)	54 (FP)
Predicted Non-rumors	62 (FN)	587 (TN)

TABLE V. Results Analysis from LSTM Model

	Precision	*Recall*	*f-score*	*Accuracy (%)*
Rumors	0.89	0.91	0.90	90.68
Non-rumors	0.92	0.90	0.91	90.44
Average	0.91	0.91	0.91	90.56

ROC curve diagnose the ability of classifiers. In this results, classifier gives curves near to the top of left corner, it indicate the better performance of the classifiers [Figure 9].

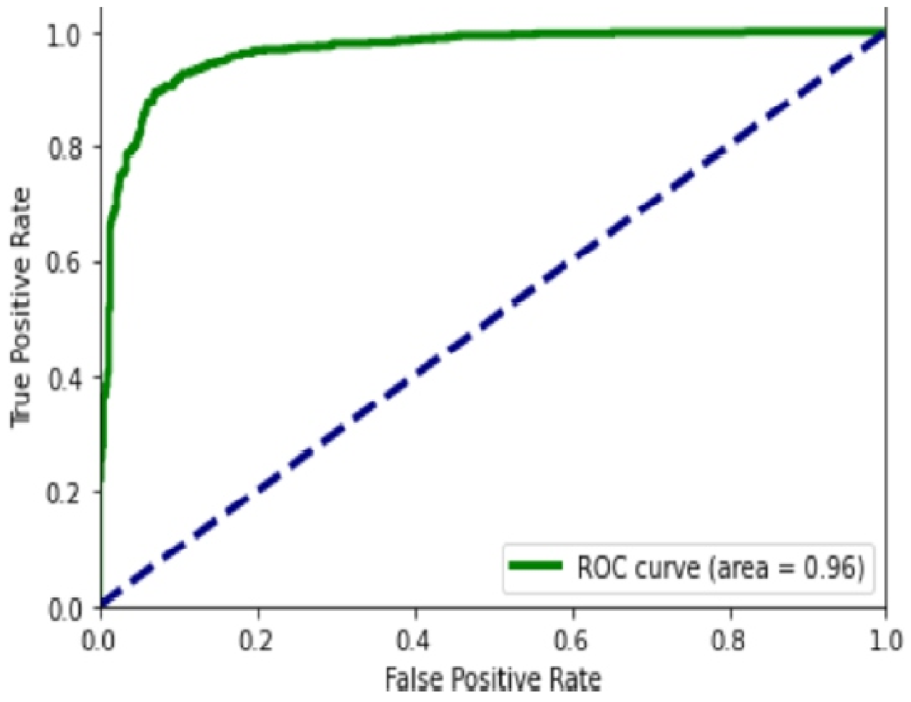


Figure 9. Performance Analysis of LSTM Model using ROC curve

6. ANALYSIS AND DISCUSSION

Manual identification of social media problems is a very tedious process and this process; increase labor and time cost, and degrade the result. To solve such problems, many researchers have been suggested different techniques and models for automatic identification, detection, and stopped them from spreading globally. As per the researchers, the meaning of the information is hidden in the text content and mostly users share the post in the form of text. For the text analysis, it is required to apply the NLP process to the text. Data has to be indifferent and diverse form, so there is also required to clean them by using common and important methods like stop word removing, case folding, steaming, removes hyperlinks, special character removing, normalization, etc.

For processing the data, a vector representation of the words is required. For vector representation of the sentences, researchers suggested some important and useful methods of word vector representation that as word embedding (tf-idf, GloVe, Word2Vec). In between some researchers suggested the process of the multilingual text which converts multilingual text into a single language by using GoogleTrans API. After the word vector representation, it is required to classify the data into different labels that is why researchers have been now focusing on machine learning and deep learning concepts.

In this study, we have just analyzed the previous researches which were based on text analysis by using text preprocessing, NLP process, and applying machine learning and deep learning for text classification [Table1]. This study also suggests that how researchers are focusing on the deep learning concept on social media content categorization.

After analysis of these models present in the literature, it can be concluded that the deep learning models are best for identifying all social media related text content problems. Deep learning concepts help to identify the hidden information from the text content as well as sequential text analysis. Deep learning approaches provide accurate and fast results as compared to traditional machine learning approaches; also provides better results for high dimensional datasets. A deep learning network works better with word embedding techniques and it also provides better performance with different features like linguistic and sentiment features, user profile features, and user metadata features of the tweeter post. So, researchers are now more focused and inclined towards deep learning approaches for the identification of social media problems.

7. CONCLUSIONS AND FUTURE SCOPE

Social media platforms have become a more important part of our daily life for sharing ideas, opinions, knowledge, and news with the people. It is surely a boon in many contexts, but it can be a terrible curse when rumors, offensive contents, abusing contents, misinformation, wrong information, and fake news are also shared by the users. Social media spread harmful, unauthentic, unwanted contents regularly; due to such type of content it has a dangerous influence on society. Many researchers are now focusing on preventing such types of content on the social media platform and for this; they are using many algorithms and methods to analyze and classify such contents and to spread only relevant information. In the existing literature, lexical analysis and syntactic analysis were used for more accurate results and now, sentimental and semantic analysis of NLP helps to extract more features from the sequential sentences.

Previously, researchers were more focused on classification with both rumors and non-rumors features and noticed that the classified results detect only from the rumored features, so binary classification might not provide beneficial results. To solve this, a new approach is used i.e., one class classification classifier with one class feature. Rumor features were extracted from the already available and detected features of rumors on a social network. The researchers used the concept of machine learning and deep learning for determining social media platform problems and also compared these two technologies. The results show the effectiveness of the deep learning concept over the machine learning concept. Also, machine learning concepts are very time-consuming approach as compared to deep learning approaches. Deep learning concept help to solve such issue and useful with the semantic representation of the sentences, its hidden layer learning concept provides the automatically learn from the text with high performance. Currently, deep learning technique like CNN and LSTM has become more popular for text classification. Natural language processing and word embedding methods play the most important role and help to provide better accurate results.

Pre-trained word embedding model used for generating efficient word vectors which help to the low processing time as well as provide efficient results. The GoogleTrans API is also used for classifying multilingual content. Twitter streaming API, Facebook Graph API, BeautifulSoup API is used for streaming content from social media platforms. The Word2Vec & GloVe model of word embedding is used for representing word vectors in a sentence; these vectors are used input for deep learning models to provide better accurate results. As per the comparative study and LSTM deep learning model's evaluation proves that deep learning models are the best for text categorization. In the future hybrid deep learning models will be effective for social media content categorization.

REFERENCES

[1] A. Alsaeedi and M. Al-Sarem, "Detecting Rumors on Social Media Based on a CNN Deep Learning Technique", Arabian Journal for Science and Engineering, Springer, 2020, https://doi.org/10.1007/s13369-020-04839-2.

[2] J. P. Singh, N. P. Rana, and Y. K. Dwivedi, "Rumour Veracity Estimation with Deep Learning for Twitter" ウ IFIP International Federation for Information Processing 2019, Published by Springer Nature Switzerland AG 2019, Y. Dwivedi et al. (Eds.): TDIT 2019, IFIP AICT 558, pp. 351–363, 2019, https://doi.org/10.1007/978-3-030-20671-0_24.

[3] A. E. Fard, M. Mohammadi, and Y. Chen, "Computational Rumor Detection without Non-Rumor: A One-Class Classification Approach", 2329-924X ウ 2019 IEEE, IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS.

[4] S. A. Alkhodaira, S. H.H. Dingb, B. C.M. Fung, and J. Liuc, "Detecting breaking news rumors of emerging topics in social media", 0306-4573/ウ2019 Elsevier Ltd., https://doi.org/10.1016/j.ipm.2019.02.016.

[5] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K. Wong, and M. Cha, "Detecting Rumors from Microblogs with Recurrent Neural Networks", Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16).

[6] S. Subramani, S. Michalska, H. Wang, J. Du, Y. Zhang, and H. Shakeel, "Deep Learning for Multi-Class Identification from Domestic Violence Online Posts", 2169-3536, 2019 IEEE, Volume 7, 2019.

[7] S. Subramani, H. Wang, H. Q. Vu, and G. Li, "Domestic Violence Crisis Identification from Facebook Posts Based on Deep Learning", Volume 6, 2018, 2169-3536, 2018 IEEE.

[8] H. Yenala, A. Jhanwar, M. K. Chinnakotla, and Jay Goyal, "Deep learning for detecting inappropriate content in text", International Journal of Data Science and Analytics, Springer, 2017. https://doi.org/10.1007/s 4106 0-017-0088-4.

[9] V. S. Chavan, and S. S. Shylaja, "Machine Learning Approach for Detection of Cyber-Aggressive Comments by Peers on Social Media Network", 978-1-4799-8792-4/15/$31.00 ウ2015 IEEE.

[10] M. Dadvar, and K. Eckert, "Cyberbullying Detection in Social Networks Using Deep Learning Based Models; A Reproducibility Study", arXiv:1812.08046, 2018.

[11] D. Chatzakou, N. Kourtellisz, J. Blackburnz, E. D. Cristofaro, G. Stringhini, and A. Vakaliy "Mean Birds: Detecting Aggression and Bullying on Twitter", arXiv:1702.06877 [cs.CY].

[12] S. Ahmad, M. Z. Asghar, F. M. Alotaibi, and I. Awan, "Detection and classification of social media-based extremist affiliations using sentiment analysis techniques" Hum. Cent. Comput. Inf. Sci. (2019), https://doi.org/10.1186/s13673-019-0185-6.

[13] A. H. Johnston, and G. M. Weiss, "Identifying Sunni Extremist Propaganda with Deep Learning", 978-1-538 6-2726- 6/17/$31.00ウ2017 IEEE.

[14] T. Ding, A. Roy, Z. Chen, Q. Zhu, and S. Pan, "Analyzing and Retrieving Illicit Drug-Related Posts from Social Media", IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 978-1-5090-1610-5/16/$31.00 ©2016 IEEE.

[15] ꞮE. Brassard-Gourdeau, and R. Khoury, "Impact of Sentiment Detection to Recognize Toxic and Subversive Online Comments", arXiv:1812.01704 [cs.CL].

[16] D. Kosmajac, and V. Keselj, "Language Identification in Multilingual Short and Noisy Texts using Common N-Grams", IEEE International Conference on Big Data (BIGDATA), 978-1- 5386-2715-0/17/$31.00 ©2017 IEEE.

[17] A. H. Ombabi, O. Lazzez, W. Ouarda, and A. M. Alimi, "Deep Learning Framework based on Work2Vec and CNN for Users Interests Classification", 978-1-5386-0667-4/17/$31.00©2017 IEEE.

[18] M. Kanakraj, and Guddeti R. M. R., "NLP Based Analysis on Twitter Data Using Ensemble Classifiers", 3rd International Conference on Signal Processing, Communication and Networking (ICSCN), 978-1-4673-6823-0/15/$31.00 ©2015 IEEE.

[19] M. V. G. Aziz, A. S. Prihatmanto, D. Henriyan, and R. Wijaya, "Design and Implementation of Natural Language Processing with Syntax and Semantic Analysis for Extract Traffic Conditions from Social Media Data", IEEE 5th International Conference on System Engineering and Technology, Aug. 10 - 11, UiTM, Shah Alam, Malaysia, 978-1-4673-6713-4/15/$31.00 ©2015 IEEE.

[20] M. L. Ȝakir, and S. G ▪ damlasioglu, "Text analysis Analysis in Turkish Language Using Big Data Tools", IEEE 40th Annual Computer Software and Applications Conference, 0730-3157/16 $31.00 © 2016 IEEE, DOI 10.1109/COMPSAC.2016.203.

[21] Y. A. Putra, and M. L. Khodra, "Deep Learning and Distributional Semantic Model for Indonesian Tweet Categorization", 978-1- 5090-5671-2/16/$31.00 ©2016 IEEE.

[22] M. A. Zahran, A. Magooda, A. Y. Mahgoub, H. Raafat, M. Rashwan, and A. Atyia, "Word Representations in Vector Space and their Applications for Arabic", © Springer International Publishing Switzerland 2015, DOI: 10.1007/978-3-319-18111- 0_32.

[23] S. H. Yadav, and P. M. Manwatkar, "An Approach for Offensive Text Detection and Prevention in Social Networks", IEEE Sponsored 2nd International Conference on Innovations in Information Embedded and Communication Systems (ICIIECS'15), 978-1-4799-6818-3/15/$31.00 ©2015 IEEE.

[24] C. Buntain, and J. Golbeck, "Automatically Identifying Fake News in Popular Twitter Threads", IEEE International Conference on Smart Cloud, 978-1-5386-3684-8/17 $31.00 © 2017 IEEE, DOI 10.1109/SmartCloud.2017.40.

[25] F. M. T. Hossain, M. I. Hossain, and S. Nawshin, "Machine learning based class level prediction of restaurant reviews", IEEE Region 10 Humanitarian Technology Conference (R10-HTC), DOI: 10.1109/R10-HTC.2017.8288989.

[26] C. Baydogan, and B. Alatas, "Sentiment analysis using Konstanz Information Miner in social networks", 6th International Symposium on Digital Forensic and Security (ISDFS), DOI: 10.1109/ISDFS.2018.8355395.

[27] Y. Luan, and S. Lin, "Research on Text Classification Based on CNN and LSTM", 978-1-7281-1223- 7/19/$ 31.00 ©2019 IEEE.

Amit Kumar Sharma received his M. Tech Degree in Computer Science & Engineering with specialization in Information Security from Central University of Rajasthan, India. He is currently pursuing the Ph.D. degree in the department of Computer Science & Engineering, Manipal University Jaipur, India. His research interests in the fields of Machine Learning, Deep Learning & Text Analysis.

Sandeep Chaurasia is Associate Professor in Department of Computer Science and Engineering, School of Computing & Information Technology at Manipal University Jaipur, India. He has 10 years of teaching experience. His research interests in the fields of Deep Learning and AI.

Devesh Kumar Srivastava is Professor in department of Department of Information Technology, School of Computing & Information Technology, Manipal University Jaipur, India. His current research interests include Software Engineering, Operating System, Data Mining, big data, DBMS, Web Tech, Computer Architecture, Computer Network, Oops Technology.

ISSN 2210-142X

Fault Tolerance Directed by Service Level Agreement in Cloud Computing Environments

Samir Setaouti[1], Djamel Amar Bensaber[2], Reda Adjoudj[1], Mohammed Rebbah[3]
[1]Evolutionary Engineering & Distributed Information Systems Laboratory (EEDIS), Computer Science Department, Djillali Liabes University of Sidi Bel Abbes, Sidi Bel Abbes, Algeria
[2]LabRI-SBA Lab., Ecole Superieure en Informatique, Sidi Bel Abbes, Algeria
[3]Computer Science Department, Mustapha Stambouli University of Mascara, Mascara, Algeria
[1]samir.setaouti@univ-sba.dz, [2]setao.samir@gmail.com, [3]d.amarbensaber@esi-sba.dz, [4]reda.adjoudj@univ-sba.dz, [5]rebbahmed@univ-mascara.dz

ABSTRACT

The Cloud Computing environments are susceptible to frequent failures as a result of their dynamic and unstable nature. Failures have a significant effect on cloud performance and on the benefits that customers and providers expect. Therefore, fault tolerance is necessary to mitigate the impact of failures and preventing revenue losses due to service level agreements (SLA) violation penalties. In this paper, we propose a fault tolerance directed by the SLA contracts established between cloud providers and customers. The proposed fault tolerance includes two phases, the first is based on the use of idles VMs according to selection strategies. The second phase is based on both advanced operation of degradation of quality of service and on VMs selection strategies. The advanced operation of degradation consists of beneficial combinations of VMs deallocation and allocation between customers leading to avoid SLA violation penalties. According to the experimental results, our proposed fault tolerance decreases the considered SLA violations.

Keywords: Cloud Computing, Service Level Agreement, Fault Tolerance, SLA Violation, performance, Availability, Penalties.

1. INTRODUCTION

According to the U.S. National Institute of Standards and Technology (NIST) definition [01]: „„,Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (for example servers, networks, storage, services, and applications) that can be quickly provisioned and released with least management effort or service provider interaction,„„. The cloud resources are delivered to customers as a service in three main categories: infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS) [02]. The relations between Cloud providers and customers are managed by a clear Service Level Agreement (SLA). The SLA is defined as a contract that describes the negotiated service, the quality of service, the QoS metrics, the price and arrangements in cases of violations.

The cloud computing, as a fast evolving technology, is increasingly getting used to host many business or enterprise applications. However, these environments are known for their highly distributed and heterogeneous nature. Consequently, several types of faults may occur in the cloud environment leading to service unavailability and performance degradation [03][04]. Availability and performance of cloud services are essential for maintaining customer‟s confidence and preventing revenue losses due to service level agreement (SLA) violation penalties [05][06]. To make cloud computing viable for both customers and providers, faults should be handled effectively by fault tolerance mechanisms.

Fault tolerance (FT) refers to a system's ability to continue performing its expected operation despite faults.

In other words, FT is related to reliability, successful operation, and absence of breakdowns [07]. In the cloud computing system, fault tolerance awareness has been identified as one of the main issues to ensure reliability, robustness and availability of important services as well as running of applications [08].

In [09], we proposed a basic fault tolerance model that uses a random quality of service degradation. In this work, we go further by proposing a new fault tolerance directed by the SLA contracts established between cloud providers and customers. The proposed fault tolerance includes two phases, the first is based on the use of idles VMs according to selection strategies. The second phase is based on both advanced operation of degradation of quality of service and on VMs selection strategies. The advanced operation of degradation consists of beneficial combinations of VMs deallocation and allocation between customers leading to avoid SLA violations penalties. The execution of the two FT phases depends on their efficiency in avoiding SLA violations. This efficiency is evaluated by a specific function.

This paper makes the following contributions:

- The design of a generic SLA representation model that can be applied to different types of platforms. This model defines the type of resources requested, the margin of degradation accepted and the different regular and irregular states of customers that use the platform resources.
- The proposal of a faults tolerance directed by the SLA contracts to avoid SLA violations penalties. The proposed FT provides three techniques: Faults Tolerance with the strategy Max Available (FT-MA), Faults tolerance with the strategy High Capacity (FT-HC) and Faults Tolerance with the strategy Low capacity (FT-LC).
- Implementation of experimentation to evaluate the effectiveness of our proposed model compared to other existing approaches.

This paper is divided into 5 sections. Section 2 presents related work, followed by the proposed model in section 3, in which we present a generic SLA representation model, the platform model and the proposed fault tolerance. Section 4 shows the experimental setup and the results. Section 5 concludes the paper with a conclusion and future research directions.

2. RELATED WORKS

For the various systems and platforms, the fault tolerance operation ensures a certain level of service in case of failure. This section is divided into two parts, the first part deals with the SLA and the second one discusses work realized on fault tolerance in cloud computing environments.

A. Service Level Agreement (SLA)

A Service Level Agreement (SLA) is defined by Wieder et al. [10] as a formal negotiated agreement between the service provider (SP) and the customer. Its goal is to establish a common understanding about QoS, priorities and responsibilities. SLAs can cover a many aspects of the customer-SP relationship, including service performance, customer service, billing, and service provisioning.

Several works about SLA have emerged with the appearance of service oriented architectures (SOA) whose purpose is to ensure the QoS of Web services [10]. WSLA [11] is a framework composed of three sections: the parties, the service definitions and the obligations. WSLA offers the possibility of enriching the directory of performance criteria by creating new metrics. Slang [12] it is a language for the SLA description, it is modeled in UML. In An SLA, Slang defines the responsibilities of customers and suppliers and the mutual responsibilities to determine the obligations of each party. WSOL [13] is a specification based on XML, which allows offering web services with different levels of services via functional constraints, constraints of QoS, price and access rights.

The rSLA [14] is a framework for managing Service Level Agreements in Cloud environments, it contains three components: the rSLA language to formally represent SLAs, the rSLA Service, which interprets the SLAs and implements the behavior specified in them, and a set of Xlets-lightweight, dynamically bound adapters to monitoring and controlling interfaces. In [15], Kouki et al propose a solution for managing SLA Cloud Service Contracts for Models (Iaas, Paas, Saas) where they introduces the Cloud Service Level Agreement (CSLA) as a new SLA language directly integrating some features dealing with QoS uncertainty and Cloud fluctuation.

CSLAM [16] is a Cloud SLA management framework based on WSLA which provides both SLA negotiation language and management framework. CSLAM also manages the SLA aware inter-cloud service provision. In [02] Labidi et al., propose CSLAOnto A Comprehensive Ontological SLA Model in Cloud Computing, whose objectives are : Improving the SLA representation and facilitating its Understanding, Providing an automated means of SLA monitoring for the client, Informing the supplier of the QoS degradation by notification.

B. Faults Tolerance

Fault tolerance in Cloud Computing platforms is a critical issue. Several works with different mechanisms and objectives have been proposed in this area. Uesheng Tan [17] describes a replication solution for VM FT, exclusively managed by the cloud provider. It proposes to improve efficiency by using passive VM replicas (with very few resources), which become active when a failure is detected. A mechanism is introduced to transfer/initialize the state of VM. In [18], a method of VM placement based on adaptive selection of fault tolerant strategy for cloud applications is proposed. The proposed method consists of two phases. The best evaluation function value of VM placement based on every fault-tolerant strategy is determined in the first phase. In the second phase, the VM placement plan is solved according to the solution of the first phase. Amoon.M [19] propose an adaptive framework for reliable cloud computing environments (AFRCE), it consists of two algorithms: the first for selecting VMs that meets customers needs and can carry out their requests. The second algorithm to select the fault tolerance technique, either replication or checkpoint-restart. In case of replication, AFRCE adjust the number of replicas by calculating the values of VMs. The value of a VM is a function of its failure probability and the profit of its use. If checkpoint-restart is selected, the checkpoint frequency is also adjusted with respect to the failure probability.

The authors in [20], focus on improving the reliability and availability in the Iaas model of cloud computing, they propose a new topology aware VM replica placement approach. The approach places the replicas of virtual machines on the optimal candidate servers according to a calculated weight. The weight is calculated for each PM depending on its distance from the VM and its current temperature. A low value of a PM"s weight indicates its appropriateness for hosting a replica of a VM. In [21] Zhou et al.

propose an approach to cloud service reliability enhancement via virtual machine placement optimization. The proposed approach is composed of three phases; each one is elaborated by an algorithm. Based on the network topology, the algorithm of the first phase selects a suitable set of VM hosting servers from a potentially large set of candidate host servers. The second algorithm determines an optimal strategy to place the primary and backup VMs on the selected host servers. The last phase concerns the recovery strategy decision; a heuristic is used to address the task-to-VM reassignment optimization problem, which is formulated as finding a maximum weight matching in bipartite graphs. Dailbag et al. [22] present an approach for providing high availability to the requests of cloud‟s customers based on a failover faults tolerance strategy for cloud computing, using integrated check-pointing algorithms.

Alain Tchana et al [23] propose a fault tolerance method in which both the Cloud customers and providers will collaboratively share their responsibilities in order to provide the required fault tolerance. According to Tchana, application faults can be detected and repaired at the customer level. But the Virtual Machine (VM) and hardware faults can be detected & repaired at the Cloud provider level. The recovery/restoration of the applications running on the refurbished VMs can be requested and performed at the customer level. Checkpointing technique is used to create restore points for the recovered VMs. VFT [24] is a virtualization and fault tolerance technique proposed to reduce the service time and to increase the system availability. It uses a cloud manager (CM) module and a decision maker (DM) to manage the virtualization, load balancing and to handle the faults. In [25], Bashir et al. propose an optimized fault approach in real-time cloud computing Iaas environment where a model is designed to tolerate faults based on the reliability of each compute node (VM) and can be replaced if it‟s performance is not optimal.

The works in [26], [27] and [28] deal with the different services level objective described in the SLA. In [26], proposed spot instance scheme that users can decide a minimum cost according to SLA agreement between users and instances in Amazon's EC2. The scheme is based on a probabilistic model for the optimization of cost, performance and improved the reliability of services by changing dynamically conditions to satisfy user requirements. Yi et al [27] introduced the spot instances of the Amazon EC2 to offer fewer resource costs in exchange for reduced reliability. Based on the actual price history of EC2 spot instances, authors compared several adaptive checkpointing schemes in terms of monetary costs and the improvement of job completion time. In [28] to avoid delays in task completion, a price history based check-pointing scheme based on SLA is proposed. This is achieved by reducing the number of checkpoints and improving the performance of the tasks. Total cost and number of checkpoints are effectively reduced in this scheme. A coordinator component in this scheme supports and manages the SLA between users and instances. The checkpoints are taken in two places. One at the raising edge where the price exceeds the threshold and the other is taken at the failure time foreseen by the average failure time and failure possibility.

3. PROPOSED MODEL

In this section, first, we present a generic SLA model for representing customer's requests. Second, we describe the platform model and finally, we present the proposed fault tolerance.

A. SLA Generic Model

In this paper, we model the customer request based on the SLA contract as the form :

Req(#Res, Ti, d) such as :

- #Res : The number of resources requested by the customer.
- Ti : The type of the requested resources.
- d : The accepted degradation margin in the requested resources.

The requested resource can be material such as computing nodes in a computing grid, VMs of an Iaas cloud platform, software, web services…etc.

The resources are distinguished into different type due to their characteristics which can be the hardware‟s performances such as the frequency of CPU, the memory size, the network bandwidth,...etc, or it could be the type of the Saas or Paas service provided by the platforms. Concerning the degradation, there are two categories. Firstly, the quantity degradation, it‟s defined by a reduction in the number of resources provided compared to that requested. The second category of degradation, concerns the performances such as the CPU, the memory, the response time of a service, etc. The degradation margin can be specified according to several levels or thresholds. Before continuing, we present a table of symbols used.

TABLE I. SYMBOLS USED

Symbols	Description
#Res	The number of resources requested by the customer
Ti	Resource of type *i*
Ci	Resource of class *i*
#ResCi	The number of the resource from the class *Ci* assigned to the customer
MaxCi	The maximum accepted number of the resource of class *Ci*
MinCi	The minimum accepted number of the resource of class *Ci*
#VMs	The number of requested VMs by the customer.
d	The degradation margin accepted in the requested resources

Based on both the types of resources and the degradation margin specified by the customer, we classify the resources as follows:

$$C_1 < C_2 < C_3 < \ldots\ldots < C_{m-1}$$

- C_1: Resources that match the customer's choice, a resource of type *Ti*.
- C_i $(2 <= i <= m-1)$: Resources with an accepted degradation margin(the degradation<=*d*).
- C_{m-1} : Resources with a capacity below the accepted margin. Such that: *m* is the levels or thresholds number of the degradation margin.

Depending on the resource classes provided to the customer according to the parameters specified in the SLA model, the customer can be in two states:

1) RS: is the regular state that defines the levels from L1 to Lk , such as :

a) L1 : the highest level of quality, the customer receives all his resources from the requested type (from class C1) : #ResC1=#Res.

b) [L2,....Lk]: the interval of degraded QoS accepted levels and Lk is the minimum level of accepted QoS : #ResC1 + #ResC2 +…+ #ResCm-1 = #Res , such as : MinCi <= ResCi < MaxCi.

2) IS: is the irregular state that contains the levels of Potential Violation (PV) and Recorded Violation (RV), such as:

a) PV: this is the level of the Potential Violation (Lk+1). The QoS is below the accepted minimum level. The time spent by the customer in this state must not exceed the resolution time (TR).

b) RV: if the potential violation is not resolved within the TR delay, the customer ends up in the level Lk+2 that represents the status of the violation recorded (considered) involving penalties.

The cloud provider seeks to maximize his profit, firstly, by avoiding the recorded SLA violation involving penalties. Secondly, by minimizing the time spent by the customer in irregular states to reduce the rate of penalties. Therefore, for each customer, the objective is formulated as follow: Min(T) = T.PV+T.RV Under the constraints:

- #Res*d <= (#ResC1 + #ResC2…+ #ResCm-1)<#Res
- T.PV<=TR
- T.PV : Time spent by the customer in the level of potential violation.
- T.RV : Time spent by the customer in the level of recorded violation.

B. Platform Model

As shown in the figure 1, the platform model adopted provides N different types of resources which are classified in M categories; each category is intended for a well-defined use. The customer can thus choose the type of resources that meet his needs.

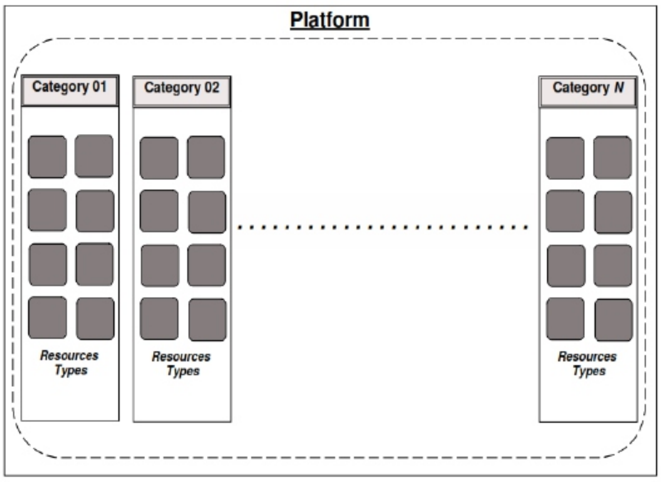


Figure 1. Platform Model

In our work, we focus on Cloud computing Iaas infrastructure that provides a set of VMs to customers.

Known Cloud providers such as: Amazon EC2 [29], Micosoft Azure [30] use this model where they propose to their customers different types of instances optimized for different use case. Instance types are different combinations of CPU, memory, storage, and network capacity that give the flexibility to choose the appropriate resources.

C. Fault Tolerance Parameters

In this work, we focus on ensuring the availability and performance stability of the provided instances (Vms):

1) The Availability: High Availability (HA) in cloud computing services is some of the hot challenges. The availability of a system at time „t" is referred to as the probability that the system is up and functional correctly at that instance in time [31][32]. In IaaS cloud computing infrastructure, unavailability can affect a limited number of VMs or the entire system where the service is completely unavailable for a period of time. Most of the cases, the unavailability is caused by physical faults that mainly occur in hardware resources, such as faults in CPUs, in memory, in storage, failure of power etc. [33].

2) Performance: the decrease in performance of Vms affects their capacities. Generally it's due to the resource consolidation [34]. For example, it occurs when a VM cannot get the requested amount of MIPS. This can occur when multiple VMs sharing the same host require higher CPU performance that cannot be provided due to VM consolidation [35]. Furthermore, the IaaS clouds platforms to their complex nature are prone to performance anomalies due to various reasons such as resource contentions, software bugs, or hardware failures [33][36].

Cloud service providers are required to explain how the availability and performance are measured, as well as refunds in case of SLA violations. Amazon EC2 and Microsoft Azure define the downtime as the total of the accumulated minutes in which the service was in a state of unavailability. A service credit is reimbursed to customers according to the monthly uptime percentage [37][38].

D. Proposed Faults Tolerance

When failures affect a customer"s VMs and cause potential violations, fault tolerance operation is initiated to tolerate the occurred faults and resolve the potential violation as quickly as possible before being recorded. The flow chart presented in figure 2, explains the functioning of the fault tolerance. It is held in two phases:

1) Phase 01 : FT using available free (idle) VMs

- When faults affect the cloud computing platform and cause SLA potential violations, a recovery operation is automatically initiated by restarting the failed VMs. Recovered VMs are added to the list of available VMs. This recovery operation is performed continuously. The available free VMs are VMs that are not assigned to customers and which are in the idle state.
- Immediately after launching the VMs recovery, we evaluate for the customer in potential violation (level PV), the possibility of fault tolerance for its recovery in a regular state level by using the function Check_In_RS() (Algorithm 1). The function Chech_in_RS() evaluates the efficiency of this 1st FT phase by checking the possibility of fault tolerance to resolve the potential violation according to a level in the regular state, starting with the higher level L1

until the last accepted level LK. It compares the number of failed VMs that can be replaced by available free VMs of different classes Ci taking into consideration the maximum number of VMs allowed in each class (MaxCi). The function Check_in_RS() returns the resolution level index possible L_RS, if no resolution is possible for the potential violation by the fault tolerance, it returns the value 0.

Algorithm 01 : Check_In_RS ( )

```
Input  :  Cust_Id, AvailableVmsC1List, AvailableVmsCkList, AssignedVMsList.
Output :  L_RS
  ValidVMs ← AssignedVMsList.nb
  i ← 1,  L_RS ← 0,  Cont ← true
  While ( i <= K  AND Cont = true ) do
      If ( (ValidVMs + MaxcCi ) < #VMs ) then
          If  ( AvailableVmsCiList.nb <= MaxCi  ) then
              ValidVMs ← ValidVMs +AvalableVMsList.nb
          Else
              ValidVMs ← ValidVMs + MaxCi
      Else
          If (AvailableVMsCiList.nb > =MaxCi  OR  (ValidVMs + AvailbaleVMsList.nb ) >= #VMs  ) then
              L_RS ← i
              Cont ← false
              Break
      i←i+1
  return  L_RS
```

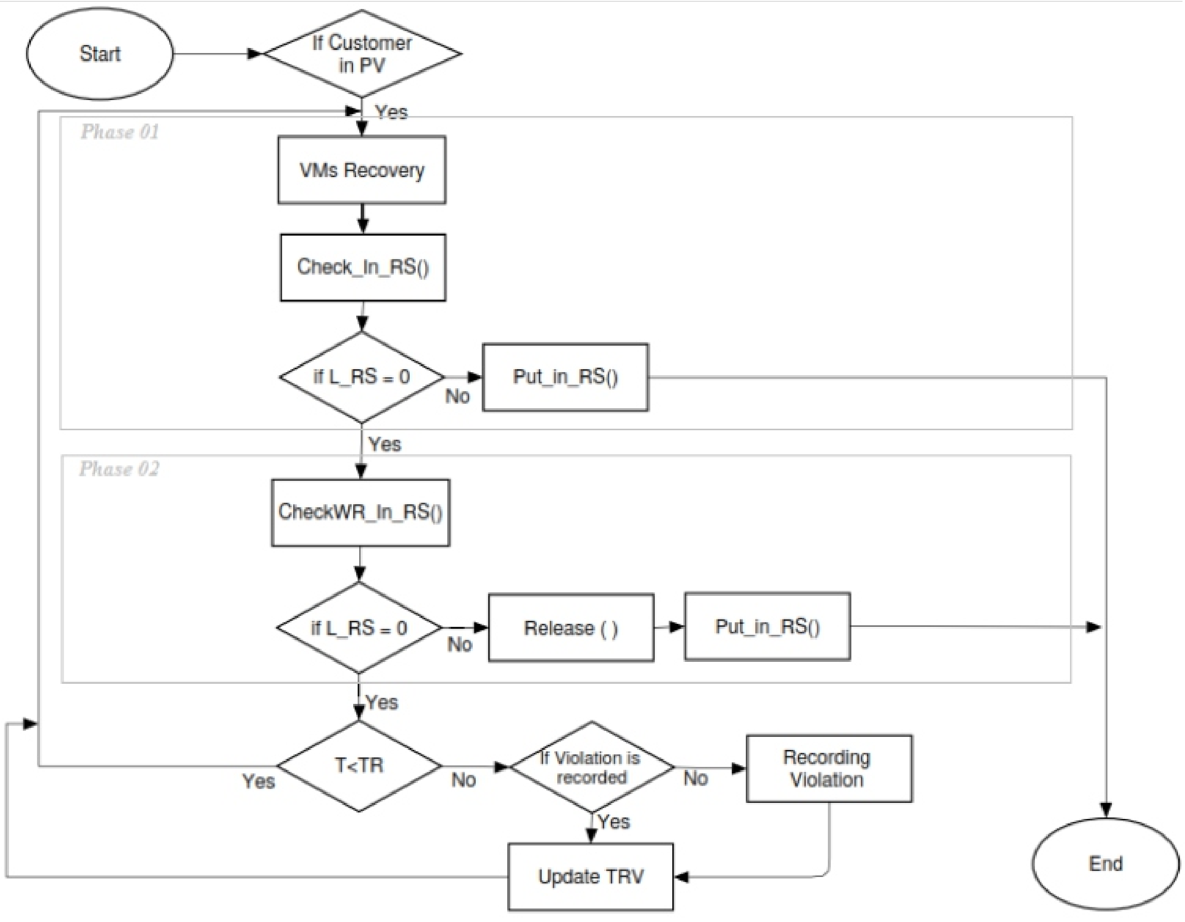


Figure 2. Flow Chart « Fault Tolerance »

- If the evaluation of the possibility to recover the customer in a regular state level by the FT proves to be positive, the customer is put in the level L_RS returned by the function

Check_in_RS() using the function Put_In_RS() (Described in Algorithm 2). The function Put_In_RS() replaces the customer's failed VMs by assigning new VMs based on the L_RS level. VMs are selected according to three different strategies (MA : Max Available, HC : Higher Capacity and LC : Lower Capacity) (Described in the next part) from the different classes Ci, such that i is including between 1 and L_RS and respecting the MaxCi in the allocation of VMs. The content of the customer"s failed VMs is restored in the new assigned VMs based on saved image.

Algorithm 2 : Put_In_RS()

```
Input   : Cust_Id, L_RS, AvailableVmsC1List, AvailableVmsCKList, AssignedVmsC1List, AssignedVmsCKList.
OutPut : AssignedVmsList.
i ← 1
While (AssignedVmsList.nb< #Vms  AND  i <=L_RS ) do
    While (AvailableVmsCiList.nb ≠ 0  AND  AssignedVmsCiList.nb < MaxCi ) do
        SelectVm(AvailableVmsCiList)
        Allocate Vm to Customer Cust_Id
        AssignedVmsList.add(Vm)
        AssignedVmsCiList.add(Vm)
        AvailableVmsCiList.remove(Vm)
        If (AssignedVmsList.nb = #Vms ) then
            Break
    i ← i+1
return AssignedVmsList
```

2) Phase 2 : FT using released VMs

If the violation is not resolved by the first phase of fault tolerance, the second phase is lunched. It uses the VMs released (deallocated) by the degradation of a number of selected customers from the regular state levels.

- Firstly, we evaluate using the function CheckWR_In_RS( ) (Described in Algorithm 03) the efficiency to proceed the second phase of fault tolerance to resolve the potential violation. The function CheckWR_In_RS( ) checks for the possibility of fault tolerance that leads to a potential violation resolution, starting with the second level of regular state until the last accepted level. It search a possible VMs reallocation by comparing if the available VMs and those which can be released by degrading a set of targeted customers (PotentialReleased VMsList), makes it possible to replace the failed VMs. The function CheckWR_In_RS( ) returns the level of possible resolution L_RS.
- If the evaluation realized by the function CheckDW_In_RS( ) is positive, we proceed to releasing the VMs of the classes searched by the degradation of targeted customers from the regular state levels. The degradation operation is elaborated using the function Release( ) (Described in Algorithm 4). The function Release()) performs degradations of the targeted customers specified in CustTargetedList, one by one from their current levels L_init to the new levels of the specified regular state L_new. The VMs are deallocated from the customers respecting the minimum assigned number from each class. The customers concerned by the release of VMs, pass temporarily to the potential violation level, waiting the allocation of their new VMs using the function Put_In_RS() to regain a regular state level. The content of the VMs deallocated from the customers concerned by the degradation are migrated to their new assigned VMs. The released VMs are added to the list of available VMs AvailableVmsList.
- After the release of the VMs by the function Release()), the customer is put in the level returned by the function CheckWR_In_RS()) using the function Put_In_RS()), which replaces the customer failed VMs by VMs from the list AvailableVMsList, which contains the available free VMs, as well as those recently released. Finally, the content of the customer"s failed VMs is restored in the new assigned VMs based on saved image.

Algorithm 3 : CheckWR_In_RS ( )

```
Input  :  Cust_Id, AvailableVmsC1List, AvailableVmsCkList, AssignedVmsC1List, AssignedVmsCkList.
Output :  L_RS
L_RS ← 0, i ← 2
ValidVMs ← AssignedVMsList.nb,  Cont ← true
While  ( i < =k  AND Cont = true)
  If ( ValidVMs + MaxCi < #VMs ) then
    If ( (AvailableVmsCiList.nb + PotentialReleasedVMsList .nb) <= MaxCi ) then
      ValidVMs ← ValidVMs + AvailableVmsCiList.nb + PotentialReleasedVMsList .nb
    Else
      ValidVMs ← ValidVMs + MaxCi
  Else
    If((AvailableVmsCiList.nb + PotentialReleasedVMsList .nb)>=MaxCi OR ( ValidVMs + AvailableVmsCiList.nb + PotentialReleasedVMsList .nb )>= #VMs) then
      L_RS ← i
      Cont ← false
      Break
  i ← i+1
return  L_RS
```

Algorithm 04 : Release( )

```
Input   : Cust_Id,  AssignedVMsCiList, L_init,  L_dg
Output : AvailableVMsList
i ← L_init
If  (Check_In_RS(Cust_Id) = true)  then
  While ( i >=L_Dg ) do
    While ( AssignedVMsCiList.nb > MinCi ) do
      AssignedVMsCiList.remove(VM)
      ReleasedVMsList.add(VM)
    i ← i-1
  Put_In_RS(Cust_Id)
return AvailableVMsList
```

Algorithm 05 : SelectVm( )

```
Input   : AvailableVMsList , Strategy
Output : SelectedVm
If (Strategy =MA)  then
  Select Vm of Type Tj  where AvailableVmsCiTjList.nb  is Max
  Vm←SelectedVm
If (Strategy =HC ) then
  Select Vm of Type Tj where Tj is the Higher capacity Available Type Belonging to the class Ci
  Vm←SelectedVm
If  (Strategy =LC)   then
  Select Vm of Type Tj where Tj is the Lower capacity Available Type Belonging to the class Ci
  Vm←SelectedVm
return SelectedVm.
```

Throughout this process, the VMs recovery operation is performed in the background and the recovered VMs are added to the available VM list. In case of the potential SLA violation is not resolved using the second phase and the time TR is not expired, the process is resumed from the beginning to avoid recording the violation. If the resolution time has expired, the fault tolerance process is restarted to minimize the time spent by the customer in the state of recorded SLA violation. To select the type of available VMs that will be assigned to the customer among the types belonging to a class Ci, we use the function SelectVm( ) (Described in Algorithm 05). This function defines 03 different strategies for the VMs selection:

• **Max Available (MA):** The VMs selected first are those of the most available type compared to other types in the same class

• **Higher capacity (HC):** VMs with the higher capacities compared to the others VMs belonging to the same class are selected first.

• **Lower Capacity (LC):** VMs with the lower capacities compared to the others VMs belonging to the same class are selected in first.

4. PERFORMANCE EVALUATION

To verify the effectiveness of our proposed model, we have conducted the following experiments. First, we have evaluated and compared the results of the proposed fault-tolerance by adopting 03 Vms selection strategies (MA, HC, and LC):

- FT-MA: Faults Tolerance technique with the strategy Max Available VMs.
- FT-HC: Faults Tolerance technique with the strategy Higher Capacity VMs.
- FT-LC: Faults Tolerance technique with the strategy Lower Capacity VMs.

Second, we have compared our model to the following techniques:

- **No FT:** no fault tolerance techniques are used.

- **AFRCE [19]:** The framework is adaptive. It consists of two algorithms. The first for selecting VMs that meet customers‟ needs and can carry out their requests and the second algorithm for selecting the replication or checkpoint-restart fault tolerance technique. In the case of replication, AFRCE adjusts the number of replicas by calculating the values of VMs. The value of a VM is a function of both its probability of failure and the benefit of its use. If checkpoint-restart is selected, the checkpoint frequency is also adjusted with respect to the failure probability [19].

To compare the techniques proposed with the AFRCE technique, we consider all the VMs with an accepted degradation margin, that they satisfy the customer's needs. The price cost used is the same as that of Amazon EC2 [39].

A. Performance metrics

As an endeavor to evaluate the efficiency of the proposed techniques, we have used these metrics:

• **The Rate of recorded Violations (RV):** this metric is the ratio between the numbers of the recorded SLA violations compared to the full number of the customer‟s SLA contracts.

$$\%RV = \frac{\#\,Recorded\ \ SLA\ violations}{\#\,SLA\ contracts} \qquad (1)$$

Our main objective is to maximize the rate of potential violations resolved and therefore, minimizing the number of violations considered involving penalties (#Recorded Violations).

• **Quality of Resolution:** this metric concerns The rate of customers in the highest regular state level %Cust_L1:

$$\%Cust\_L1 = \frac{\#\,Customers\ \ in\ L1}{\#\,All\ \ Customers} \qquad (2)$$

• **Violation Resolution Time (VRT):** It represents the time needed to tolerate the faults to avoid having recorded violations (RV). Optimizing this metric minimizes the time spent by the customers in the state of potential violation (PV).

$$VRT = \sum_{i=1}^{n} VRTC_i \quad (3)$$

Such as:

$VRTC_i$: violation resolution time for the customer i.

n: number of potential violations resolved.

B. Experiment setup

The platform targeted in our work is a generic Cloud Computing environment (Iaas infrastructure). To approve the performance of our proposed model and study its efficiency, we have resorted to simulation. We have chosen the CloudSim toolkit [40] as a simulation platform. It provides a generalized and extensible simulation framework that enables seamless modeling, simulation, and experimentation of emerging Cloud computing infrastructures and application services. An extra package was created to support the fault-tolerance techniques. The created package provides a set of classes that permit simulating the VMs failure, the fault tolerance techniques as well as the detection of the potential violations and recorded violations.

In our experiments, we have generated a cloud of 10000 virtual machines that are connected with fast Ethernet technology (100Mb/s). The Virtual machines are provided by 500 Hosts contained in a set of data centers. The size of each host's RAM range from 16 to 64 GB and the storage is 1TB. The frequency of the host‟s processors is given in MIPS and is assumed to range from 3000 to 10000 MIPS. The virtual machines are classified into 03 categories: general use, optimized calculation, and optimized storage. For each category, there are five models: nano, micro, small, medium, and large, which provide in all 15 types of VMs. The number of customer‟s requests is fixed at 300. Each request expresses the type and the number of VMs required by the customer, as well as the degradation margin accepted. All the experiments are conducted using a random workload.

To perform experiments, we have adopted a checkpointing based on the probability of VM failures. This probability is obtained from the VMs failures history. The checkpointing interval is shortened in case of the high probability of VM failures. Otherwise, it is prolonged in case of a low probability. The interval is calculated as follows:

$INT_i = T(t_j, VM_i) * (1-PF(VM_i))$ such as:

- INT_i : The checkpointing interval for the VM i.
- $T(t_j, VM_i)$: The execution time of the task j on the VM i.
- $PF(VM_i)$: Failure probability of the VM i.

The check-pointing files are stored in the available free VMs candidate to replace the failed VMs. This significantly reduces the restoring time. The candidate VMs are selected according to the VMs types accepted by the customer.

C. Experiment Results

1) Impact of degradation margin

To study the impact of the degradation margin %d, experiments are carried out with a degradation margin of 10% and then 20%. The rate of available free VMs is set at 10% and that of recovered VMs at 05%. The rate of failed VMs varies from 10% to 50%.

Figure 3 shows the comparison between the 03 proposed fault-tolerance techniques (FT-MA, FT-LC, and FT-HC). The recorded violation rate %RV gradually increases with that of failed VMs. The results show that FT-MA minimizes the %RV with %5 compared to FT-LC and FT-HC. This is explained by the variation made by FT-MA in the selection of the available free VMs. Each time, FT-MA selects the most available types of VMs unlike FT-LC and FT-HC, which select the VMs of lowest (FT-LC) or highest (FT-HC) capacities. This makes VMs of these types unavailable for direct replacement of failed VMs or by the combination of deallocation and reallocation of VMs between customers to resolve potential violations detected. In figure 3 (a), we observe that the increase in degradation margin %d from 10% to 20% reduces the %RV by an average value of 5% for FT-MA and 03% for FT-LC and FT-HC. The increase in the degradation margin %d extends the list of VM types accepted by the customer that improves the probability of replacing failed VMs, either directly from the list of available free VMs or by the combinations of deallocation and reallocation so that we can avoid recorded violations.

Concerning the rate of customers in level L1 %Cust_L_1 presented in figure 3 (b), FT-MA provides the best rates comparing with FT-LC and FT-HC. These rates are improved with a value between 2% and 3% by increasing the degradation margin %d to 20%. The improvement is noted for the rates of failed VMs that range from 10% to 30% due to the increase in the deallocation of VMs from the same customers. Therefore, we minimize the number of customers degraded from L_1 to resolve the potential violations.

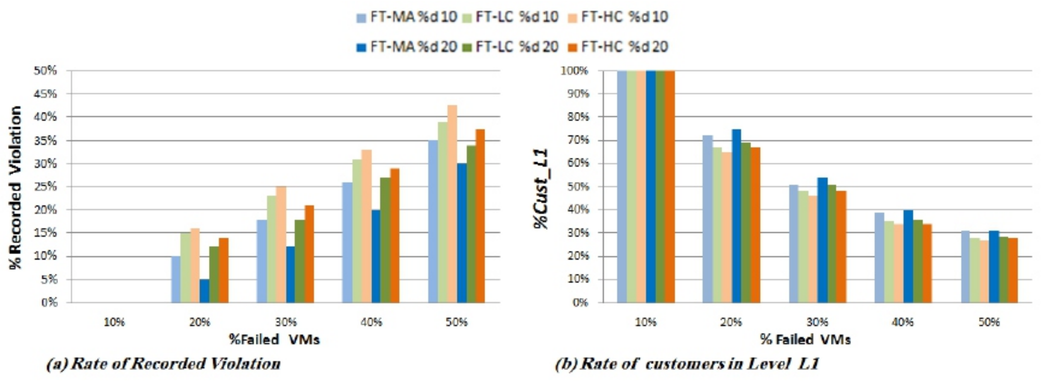


Figure 3. Impact of degradation margin

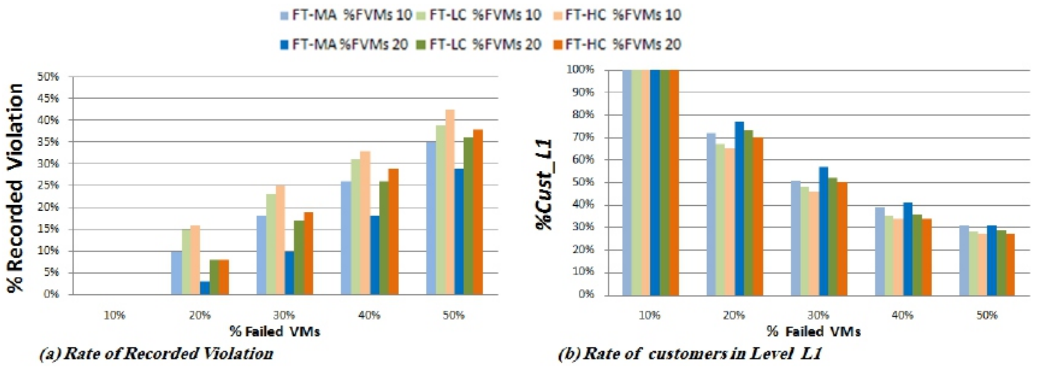


Figure 4. Impact of Available free VMs

2) Impact of Available Free VMs

This experiment is carried out with a rate of free VMs of 10% and then 20%. The degradation margin is set at 10% and that of recovered VMs at 05%.

The results presented in figure 4 (a), demonstrate that FT-MA provides better %RV compared to FT-LC and FT-HC. The increase in the rate of free VMs allows an average reduction of 07% in %RV for FT-MA and 05% for FT-LC and FT-HC. The reduction in %RV is explained by the increase in the availability of VMs for direct replacement of failed VMs, as well as by the improvement in the possible number of customers‟ degradations by the combination of deallocation and reallocation of Vms.

In figure 4 (b), with the increase in the rate of failed VMs, we notice at the beginning of the experiment (rate of failed VMs from 10% to 30%) an improvement in %Cust_L1 that ranges from 03% to 05 %. This improvement is caused by the availability of more free VMs to replace failed VMs without the need to degrade customers from level L1. The improvement in %Cust_L1 decreases as the rate of failed VMs increases in which more customers are degraded from L1 to allow replacing a large number of failed Vms.

3) Comparison with existing techniques

In the previous experiments, we demonstrate that FT-MA provides the best results compared to FT-LC and FT-HC. In this part, we compare FT-MA with existing techniques.

Figure 5 presents the effect of the degradation margin factor in the comparison between the proposed technique FT-MA with AFRCE and No-FT. In this experiment, the rate of available free VMs is set at 10% and that of recovered VMs at 05%. Figure 5 (a) illustrates that FT-MA produces the best results that avoid more than half of violations recorded by No-FT when no FT mechanism is in place, as well as it reduces by more than 10% the %RV obtained compared to AFRCE. This is explained by the fact that AFRCE uses the available free VMs that satisfies the customers‟ needs to elaborate replication or check-pointing to the valuables VMs. However, with the increase in the rate of failed VMs, it becomes extremely difficult to find VMs according to the customer‟s needs. On the other hand, our proposed technique uses the customer‟s degradation by developing combinations of deallocation and reallocation of VMs between the customers so as to replace the failed VMs with other VMs according to the customer‟s needs. +

The increase of the degradation margin %d to 20% minimizes the %RV for FT-MA by 5% and for AFRCE by an average value of 2%. For FT-MA, the minimization is due to the expansion of the list of VMs accepted by the customers with other classes of VMs, as well as by the improvement of the possibility of customer‟s degradations seeking to replace failed VMs. On the other hand, for AFRCE, it is only due to the expansion of the list of VMs that satisfied the customer‟s needs.

Figure 5 (b) presents the rate of customers in level L1. The proposed FT-MA technique provides better results compared to AFRCE. This is due to the functioning of the FT-MA technique which seeks in the first phase of FT to recover the customer in the level L1 by replacing the failed VMs from class C1 before moving to other classes of VMs accepted. For No-FT, we cannot discuss the results of customers in the level L1 since the customers are either in the L1 level or in the recorded violation level.

In figure 6 (a), increasing the rate of available free VMs to 20% enables AFRCE to find more VMs that satisfy customer requests. This diminished the %RV by 04% compared to the same experimentation with a rate of free VMs of 10%. Our proposed FT-MA technique produces minimized %RV of 15% on average compared to AFRCE.

The rates of customers in the level L1 presented in Figure 6 (b) demonstrate an improvement for FT-MA by a value between 03% and 05% as explained in the section impact of available free VMs. For AFRCE, the improvement is between 02% to 03%. It is due to the availability of more VMs of class C1 in the list of VMs that satisfy the customer's needs.

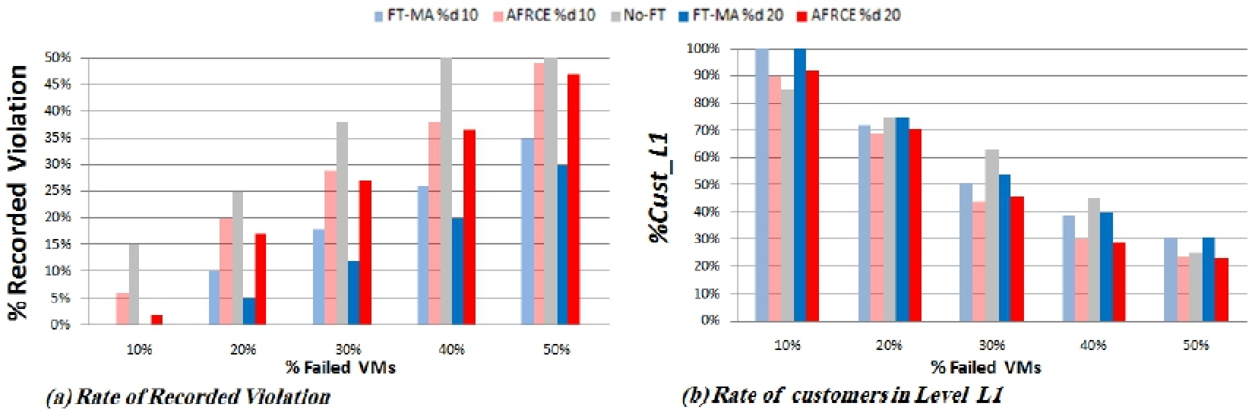


Figure 5. Comparison Results 01

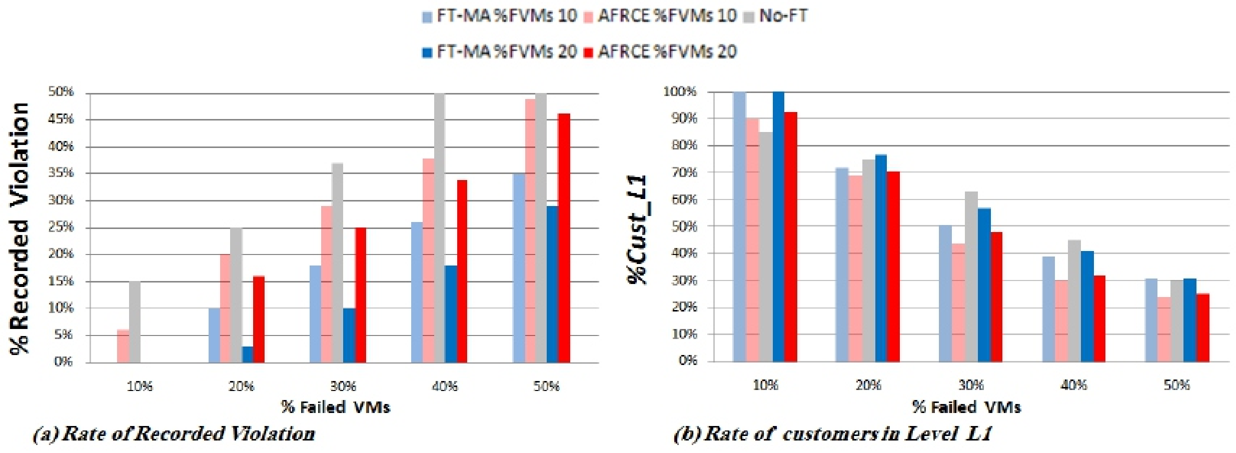


Figure 6. Comparison Results 02

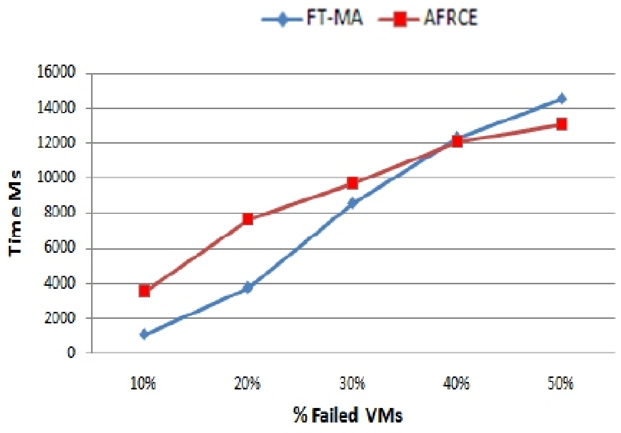


Figure 7. Execution Time.

Figure 7 illustrates the time required to perform the FT-MA and AFRCE techniques. At the beginning of the experiment, AFRCE necessitates more time compared to FT-MA. This is explained by the number of replication performed when more than one VMs are available in the list of VMs that meets the customer's needs. On the other hand, FT-MA, benefits from the available free VMs to develop the FT by the 1st phase which corresponds to the direct replacing of the failed VMs. With the increase in the rate of failed VMs, the time required by FT-MA exceeds that of AFRCE. This is due to the number of potential violations resolved by FT-MA which is higher than that of AFRCE.

5. CONCLUSION

In this paper, we propose a fault tolerance directed by the SLA contracts in the cloud computing environments. The proposed model provides three fault tolerance techniques: FT-MA, FT-HC and FT-LC. These techniques take into consideration the parameters described in a generic SLA model to ensure availability and performance that lead to avoid SLA violation penalties. To verify the effectiveness of our proposed fault tolerance, experiments were conducted using CloudSim. The three FT techniques were evaluated using three metrics: rate of considered SLA violations, rate of customer in the highest level and time. The FT-MA technique provides performance superiority compared to FT-HC, FT-LC and to related work in terms of considered SLA violations and rate of customers in the highest level. In the future work, heuristic algorithms can be incorporated to improve the VMs assignment, and we will endeavor to consider more SLA parameters.

REFERENCES

[1] Mell, P., Grance, T., 2011. The NIST definition of cloud computing recommendations of the national institute of standards and technology. Nist Spec. Publ. 145, 7.

[2] Labidi, T., Mtibaa, A. & Brabra, H. CSLAOnto: A Comprehensive Ontological SLA Model in Cloud Computing. J Data Semant 5, 179–193 (2016). https://doi.org/10.1007/s13740-016-0070-7

[3] Nazari Cheraghlou, M., Khadem-Zadeh, A., Haghparast, M., 2016. A survey of fault tolerance architecture in cloud computing. J. Netw. Comput. Appl. 61, 81–92.

[4] Amin, Z., Singh, H., Sethi, N., 2015. Review on fault tolerance techniques in cloud computing. Int. J. Comput. Appl. 116 (18), 11–17.

[5] Dantas J et al (2015) Eucalyptus-based private clouds: availability modeling and comparison to the cost of a public cloud. Computing 97:1121–1140

[6] Son S, Jung G, Jun SC (2013) An SLA-based cloud computing that facilitates resource allocation in the distributed data centers of a cloud provider. J Supercomput 64(2):606–637

[7] Dubrova, E., 2008. Fault tolerant design: an introduction. Microelectron. Inf. Technol. R.

[8] Abdulhamid, S.M., Abd Latiff, M., Madni, S.H.H. et al. Fault tolerance aware scheduling technique for cloud computing environment using dynamic clustering algorithm. Neural Comput & Applic 29, 279–293 (2018). https://doi.org/10.1007/s00521-016-2448-8

[9] S. Setaouti, D. A. Bensaber, R. Adjoudj and M. Rebbah, "Fault tolerance model based on service delivery quality levels in cloud computing," 2017 International Conference on Mathematics and Information Technology (ICMIT), 2017, pp. 84-91, doi: 10.1109/MATHIT.2017.8259700

[10] Wieder P, Butler Joe M, Theilmann W, Yahyapour R (2011) Service level agreements for cloud computing. Springer, New York. Library of Congress Control Number:2011939783, ISBN 978-1-4614-1613-5, e-ISBN 978-1-4614-1614-2. doi:10.1007/ 978-1-4614-1614-2

[11] Alexander Keller and Heiko Ludwig. The wsla framework : Specifying and monitoring service level agreements for web services. J. Netw. Syst. Manage., 11(1) :57–81, March 2003.

[12] D.D. Lamanna, J. Skene, and W. Emmerich. SLAng A language for defining service level agreements. The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems, pages 100–106, 20O3.

[13] Tosic V, Pagurek B, Patel K, Esfandiari B, Ma W (2005) Management applications of the web service offerings language. Inf Syst 30(7):564–586

[14] M. Mohamed, O.Anya , S. Tata, N. Mandagere, N. Baracaldo, and H. Ludwig , rSLA: An Approach for Managing Service Level Agreements in Cloud Environments, International Journal of Cooperative Information Systems Vol. 26, No. 2 (2017) 1742003 (29 pages) , DOI: 10.1142/S0218843017420035.

[15] Kouki Y, Ledoux T (2012) CSLA: a language for improving cloud SLA management. In: 2nd international conference on cloud computing and services science

[16] M. Torkashvan and H. Haghighi, "CSLAM: A framework for cloud service level agreement management based on WSLA," 6th International Symposium on Telecommunications (IST), Tehran, Iran, 2012, pp. 577-585, doi: 10.1109/ISTEL.2012.6483055.

[17] Uesheng Tan, Dengliang Luo, and Jingyu Wang, "Cc-vit: Virtualization intrusion tolerance based on cloud computing," in 2nd International Conference on Information Engineering and Computer Science (ICIECS), pp. 1-6. Wuhan, China, 2010.

[18] Xiao Chen & Jian-Hui Jiang (2016): A method of virtual machine placement for fault-tolerant cloud applications, Intelligent Automation & Soft Computing, DOI:10.1080/10798587.2016.1152775.

[19] M. Amoon, "Adaptive Framework for Reliable Cloud Computing Environment," in IEEE Access, vol. 4, pp. 9469-9478, 2016, doi: 10.1109/ACCESS.2016.2623633.

[20] Kumari, Priti and Kaur, Parmeet. „Topology-aware Virtual Machine Replication for Fault Tolerance in Cloud Computing Systems". 1 Jan. 2020 : 193 – 206.

[21] A. Zhou et al., "Cloud Service Reliability Enhancement via Virtual Machine Placement Optimization," in IEEE Transactions on Services Computing, vol. 10, no. 6, pp. 902-913, 1 Nov.-Dec. 2017, doi: 10.1109/TSC.2016.2519898.

[22] Dilbag Singh, Jaswinder Singh, Amit Chhabra, "High Availability of Clouds: Failover Strategies for Cloud Computing using Integrated Checkpointing Algorithms", IEEE International Conference on Communication Systems and Network Technologies, 2012.

[23] A. Tchana, L. Broto, and D. Hagimont. "Approaches to cloud computingfault tolerance." In Computer, Information and Telecommunication Systems (CITS), 2012 International Conference on, pp. 1-6. IEEE, 2012.

[24] P. Das and P. M. Khilar, "VFT: A virtualization and fault tolerance approach for cloud computing," 2013 IEEE Conference on Information & Communication Technologies, Thuckalay, Tamil Nadu, India, 2013,pp.473-478. doi: 10.1109/CICT.2013.6558142.

[25] Mohammed, B., Mariam, K.,, Irfan-Ullah, A., & Kabiru, M. M. (2016). Optimising fault tolerance in real-time cloud computing IAAS environment. Proceedings of 2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud) (pp. 363-370). IEEE.

[26] Andrzejak, A., Kondo, D., Yi, S.: Decision Model for Cloud Computing under SLA Constraints.In: Proceedings Of the 2010 IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, pp. 257–266. IEEE Computer Society,Los Alamitos (2010).

[27] Yi, S., Kondo, D., Andrzejak, A.: Reducing Costs of Spot Instances via Checkpointing in the Amazon Elastic Compute Cloud. In: Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing, pp. 236–243. IEEE Computer Society, Los Alamitos (2010).

[28] Jung D., Chin S., Chung K., Yu H., Gil J. (2011) An Efficient Checkpointing Scheme Using Price History of Spot Instances in Cloud Computing Environment. In: Altman E., Shi W. (eds) Network and Parallel Computing. NPC 2011. Lecture Notes in Computer Science, vol 6985. Springer, Berlin, Heidelberg.

[29] https://aws.amazon.com/ec2/instance-types/?nc1=h_ls

[30] https://docs.microsoft.com/en-us/azure/virtual-machines/sizes

[31] Mesbahi, M.R., Rahmani, A.M. & Hosseinzadeh, M. Reliability and high availability in cloud computing environments: a reference roadmap. Hum. Cent. Comput. Inf. Sci. 8, 20 (2018). https://doi.org/10.1186/s13673-018-0143-8

[32] Mina Nabi, Maria Toeroe and Ferhat Khendek, Availability in the Cloud: State of the Art, Journal of Network and Computer Applications, http://dx.doi.org/10.1016/j.jnca.2015.11.014

[33] Kumari, P., Kaur, P. A survey of fault tolerance in cloud computing. Journal of King Saud University – Computer and Information Sciences (2018), https://doi.org/10.1016/j.jksuci.2018.09.021.

[34] A. Corradi, M. Fanelli, L. Foschini, VM consolidation: a real case based on OpenStack cloud, Future Generation Computer Systems 32 (2014) 118–127.

[35] A. Beloglazov, B. Rajkumar, Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers, in: Concurrency and Computation: Practice and Experience, Wiley Press, New York, USA, 2011, http://dx.doi.org/10.1002/cpe .1867. ISSN:1532-0626.

[36] D. J. Dean, H. Nguyen, and X. Gu, "Ubl: Unsupervised behavior learning for predicting performance anomalies in virtualized cloud systems," in Proceedings of the 9th international conference on Autonomic computing, ser. ICAC "12. New York, NY, USA: ACM, 2012, pp. 181–190.: http://doi.acm.org/10.1145/237 1536.2371571

[37] https://aws.amazon.com/compute/sla/?nc1=h_ls

[38] https://www.azure.cn/en-us/support/sla/virtual-machines/
[39] https://aws.amazon.com/ec2/pricing/
[40] http://www.cloudbus.org/cloudsim/

Samir Setaouti Received his Engineer Degree in computer science from the University of Mustapha Stambouli of Mascara, Algeria in 2010. Got the Magister degree in computer science from The University of Sciences and Technology of Oran, Algeria in 2015. Actually, he is a Ph.D. student in the University of Djillali Liabes Of Sidi Bel Abbes, Algeria. His research interests include Cloud Computing, Fault Tolerance and Distributed Computing.

Djamel Amar Bensaber is an Associate Professor, at the High School of computer science of Sidi Bel Abbes, Algeria. He is currently Head of Research Team "Data and Software Engineering" at the LabRI Laboratory. His research interests include Information Systems (Business Informatics), Software Engineering, Big data; Linked open data, Ontology, Semantic Computing, Fault Tolerance and Distributed Systems.

Reda Adjoudj is a Full Professor in Computer Science, at Computer Science Department, University of Sidi Bel-abbes, Algeria. In 2014, Reda Adjoudj obtained his HDR & in 2006 he received Ph.D in computer science from Djillali Liabes University, Sidi Bel Abbes, and also he earned Magister from the same university in 2002, and his Engineer degree from the same university in 2000. His Major area of research is biometrics, pattern recognition, image processing, paradigms of artificial intelligence and fault tolerance. He is a member of the EEDIS Laboratory, Computer Science Department, University of Sidi Bel-abbes.

Mohammed Rebbah obtained his MSC in computer science from the University of Sciences and Technology of Oran (USTO), Algeria. After that he received his PhD in computer science in 2015, in USTO, Algeria. Actually, he is an Assistant Professor in Computer Sciences Department at the University of Mascara, Algeria. His research interests are in grid computing, cloud computing and distributed data mining.

ISSN 2210-142X

RNNCore: Lexicon Aided Recurrent Neural Network for Sentiment Analysis

Nikita Taneja[1], Hardeo K Thakur[2]
[1]Computer Science and Technology, Manav Rachna University, Faridabad, India
[2] Computer Science and Technology, Manav Rachna University, Faridabad, India…
E-mail address: [1]nikita@mru.edu.in, [2]hkthakur@mru.edu.in

ABSTRACT

Sentiment Analysis (SA) or Opinion Mining can help in identifying subjective information conveyed by user reviews for various automation tasks such as building better recommendation systems, identifying user trends, monitoring and customer support. This paper focus on the sentiment score detection. Traditional SA algorithms suffer from low accuracies in identifying true user intents. However, with the advent of Deep Learning many NLP tasks including Sentiment Analysis have become feasible with accuracies comparable to that of human experts. Additional advantage of Deep Learning in contrast to supervised learning is that in deep learning a manually tuned features set is not required. Deep Learning algorithm such as Convolution Neural Networks (CNN), Long Short Term Memory (LSTM), Recurrent Neural Networks (RNN) and various other have successfully been applied to SA. RNN in particular is well suited for this task, however most the works done over RNNs require large supervised training sets which are usually not available for all domains. This work proposes a new method called RNNCore which can make use of the pre-trained word embedding from Stanford Core NLP in conjunction with RNN to improve on accuracy and reduce time complexity. Comparison between the results of RNNCore, RNN and OneR method on the IMDB review dataset suggests that RNNCore yield 92.60% F1-measure which is a marked improvement of 17.74% as compared with a simple RNN approach for Sentiment Analysis.

Keywords: Deep learning, Sentiment Analysis, Recurrent Neural Networks.

1. INTRODUCTION (HEADING 1)

With the great progression of social media, user reviews are constantly generated from all over the internet every second which are easily accessible for further analysis. Increasingly companies have been trying to use this information to evaluate client satisfaction, preferences and provide users with recommendations [1][2]. As a consequence, there is a need to build computational models to analyze these data. These models are needed in order to detect the user opinions with respect to products or services under consideration. The Online user can provide readily available textual information which can be used for various NLP tasks [3], sentiment analysis being one of them. This information is usually subjective expressions that describe reviewer's feelings toward given products and services. The user provides a claim about the products and services associated and often associates a sentiment, which can be 'positive' or 'negative' or even neutral toward the topic; Sentiment always involve the users desires and intents. The Goal of Sentiment analysis task is to deal with the computational understanding of the provided reviews using textual analysis. It tries to determine the mind-set of a user towards certain products and services. The user mind-set can reflect his state of mind and the intended emotional communication requirements. The SA (Sentiment analysis) can be used for identifying critical beliefs of the user about products or services by mining online user reviews. Also, it can be applied in tracking the shifting attitudes and interest [4][5]of the user toward products, services, public topics etc through mining reviews which is also quite useful. This extracted information can then be used to notify other customers about the emotional attitude (positive or negative) towards the product. Tracking user trends of product reviews is also gathering research support, as the track record can be utilized for the changing consumer preferences. The detection of "flames"[6], Sentiment classification will also benefit too heated or aggressive language in e-mails or on social networking platforms. Monitoring newsgroups and forums, where quick and automatic flaming detection is required, will likewise see significant gains. Sentiment analysis can also aid the creation of new types of search engines and recommendation systems, as these systems should not propose something that has received unfavorable feedback. However, ambiguity [7] is one of the most serious issues in the field of Computational Linguistics. This difficulty can only be solved if computational systems have some type of world knowledge, or at the very least a rudimentary dictionary or any artificial intelligence based decision support becomes possible. The Semantic ambiguity present in the user reviews is very strongly related to vagueness, and can never have well-defined meanings in all senses. Another problem is inference [8] i.e. the process to find and identify implicit relations. SA becomes even more complicated when its task also is to identify the 'neutral' sentiments. Additional problem of SA is the availability of large datasets, large supervised training sets which are usually not available for all domains. This work tries to improve on the SA task by reducing the ambiguity using a pre-trained lexicon i.e. the Stanford CoreNLP [9] and inference by using an efficient deep learning architecture of RNN [10] on IMDB review dataset [11]. Most existing deep learning algorithms need manually labeled data source for the training of the algorithm, this data is required ascertain that the algorithm can deliver good accuracy. Thus a huge text/review corpus is needed for training of the deep learning algorithm for human like prediction of the input reviews. However, collection and then manually labeling such large corpus requires lot of resources in terms of time and cost as well. Also, all deep learning models require very large no. of parameter or the weight adjustments of the underlying neural network layers [12], the time complexity of these models is quite high. Furthermore, deep learning algorithms require very high end hardware, multiple GPUs, CPU and large RAM. Initializing the neural networks using already trained word-vectors taken from a supervised language model can be a good method for improving the existing neural networks such as RNNs performance when large corpus is unavailable. Hence, this work uses the pre-trained word embedding from Stanford CoreNLP in conjunction with RNN. This method is able to capture both the semantic and syntactic information from the user reviews, this information is very important for generalization of sentiment analysis task. Simply put: RNN requires structured information about the language, which is difficult to get from the raw data. This work combined the RNN with Word embedding model from CoreNLP to get better results faster.

In addition to this general introduction, this present paper is organized into five sections. Section 1 provides the Introduction to Sentiment Analysis problem. Section 2 gives a detailed Literature Survey of Sentiment Analysis methods which includes the current knowledge of Sentiment Analysis. Section 3 focuses on the proposed methodology of RNNCore method for sentiment analysis. Result of the proposed RNNCore algorithm are discussed in Section 4. Based on the results of experimentation and its analysis, concrete conclusions have been derived in this results and analysis is shown in section 5.

2. RELATED WORK

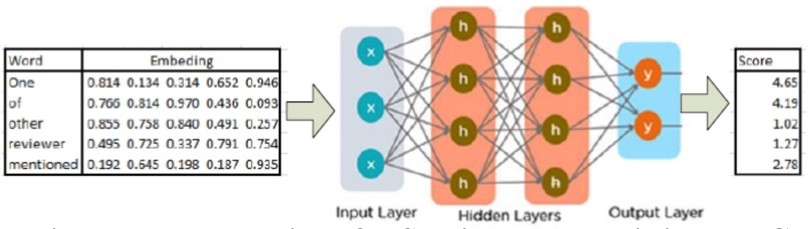
The most basic method of sentiment classification is to employ a thesaurus [13] that has information about which words and phrases are good but which are negative. This thesaurus can be individually compiled or automatically obtained. Annotating corpus is normally done manually, and subsequently methods are used to discover sentiments about new batches of words or phrases using enormous collections of data. Instead of relying on the polarity of words, other techniques can focus on mining phrases or entire reviews. In Lexicon Based approaches [14][15][16]the lexicon-based method employs a sentiment thesaurus comprised of opinion terms. By comparing these terms to the remainder of the data, the polarity is established. To comprehend how a sentiment value or score is allocated to input text using neutral, negative, and positive terms from the thesaurus. The lexicon-based systems employ a language model, which is a collection of recognized and pre-compiled collection of sentiment terms and concepts. The Lexicon-Based methods are divided into two subcategories: Dictionary-Based and Corpus-Based. The words that are routinely gathered and then manually annotated are used [17][18]. The number of synonyms for a given term in the dictionary is steadily increasing. WordNet [19] is just an illustration of a lexicon that may be used to create a lexicon corpus called SentiWordNet [20]. However, these approaches cannot handle subject specific (domain) and context-based texts, which is their biggest flaw. The corpus-based method [21][22] provides dictionary relevant to a certain domain. These dictionaries generate a collection of initial judgment phrases based on the search for appropriate words utilizing quantitative or linguistic methodologies. Latent Semantic Analysis (LSA) [23] and similar methods are based on semantics using statistics [24].The more recent classification based techniques uses training and testing of text. Generate a training set by hand and categorize it as neutral positive, or negative. After that, a test is run upon the test data to ensure that the method is reliable. The fresh reviews may then be classified using this approach. The most extensively employed machine learning algorithms for sentiment categorization are Naive Bayes (NB) [25], Maximum Entropy (ME) [26], and Support Vector Machines (SVM) [27][28].Although providing a collection of labelled reviews to train the classifier is implausible, semi-supervised and unsupervised algorithms are meant to work around this. Also different kinds of ensemble classifiers [29][30] are also being developed for sentiment analysis. For performing the best classification, all the features of the best classifiers are utilized in this classifier.

The voting rule is used to create an ensemble classifier depending upon the output of larger parts of classifiers, their classification is done.

With the advent of deep learning many NLP tasks including Sentiment Analysis have become feasible with accuracies comparable to that of human experts. Likewise advantage of deep learning contrary to the supervised learning approach deep learning need manually tuned features. Deep learning methods such as Convolution Neural Networks (CNN) [31][32], Recurrent Neural Networks (RNN) [33][34][35], Long Short Term Memory (LSTM)[36] and various other have successfully been applied to SA.

3. METHODOLOGY

The proposed RNN architecture as in figure1, is trained over word embedding, which are numeric representations of the text. The RNN does not require manual selected features, it can identify its own set of features from the word embedding. Also RNN can utilize pre trained word embedding, which are provided by CoreNLP [37]. Thus the proposed RNN can take input corpora and word embedding (pre-trained) as inputs to generate a sentiment score as shown in figure below. RNN is trained to generate a Sentiment Score in the range of (1-5) instead of polarity (-1, 1). This is done to generalize the RNN output so that it can be used in other applications such as recommendation systems as input.



Word	Embeding				
One	0.814	0.134	0.314	0.652	0.946
of	0.766	0.814	0.970	0.436	0.093
other	0.855	0.758	0.840	0.491	0.257
reviewer	0.495	0.725	0.337	0.791	0.754
mentioned	0.192	0.645	0.198	0.187	0.935

Score
4.65
4.19
1.02
1.27
2.78

Figure 1:Using word embeddings for Sentiment analysis in RNNCore

Recurrent networks take into account not only the present input but also what they have seen in the past. There is an input layer, a hidden layer containing activations, and an output layer in the generalized architecture of a neural network illustrated above in Figure 1. It is possible to expand the number of layers in an RNN by having one input layer take input from another, i.e., after receiving information from the input layer, the first hidden layer activates the next hidden layer, and so on. Finally, the information arrives at the output layer, which generates the result (sentiment score). The weights of each hidden layer in RNN are trained separately in each epoch of training when varying lengths of reviews arrive at the input layer to generate sentiment score at the output layer. Before sentiment analysis is done using the RNN, pre-processing is one of the vital method to get efficient results. Following steps are followed in pre-processing stages

3.1 Data preprocessing

There is certain amount of irrelevant data available within the data which is collected from the user reviews. All arbitrary characters (such as special characters) or unreadable information is filtered out from the reviews. A NLP pipeline is applied for removal of this useless data. All kind of grammatical associations which can exist in between the reviews is outputted by the NLP pipeline. The reviews that contain meaningful information are recognized using this pipeline. The outcomes aren't aided through the facilitating filtering. Nouns, adjectives and verbs are also discovered using this NLP pipeline. The NLP pipe line after preprocessing has further tasks as described below:

3.1.1 Vectorization

The data has been changed to vector form by using the TF-IDF [38] function and find out the unique words from all the reviews which have been provided as input. It provides a single text file containing all the data needed for the tweet classification. Vector space model is the most widely used method in tweet representation. Vectorization model uses feature entries with associated weights for expressing the review information. To represent document object vector space model have been introduced. Vector $d = w_1, w_2, w_3, \dots w_m$ means that there view words and its weight in reviewed, the number of all word vectors. $w_i \ (i = \bar{1} \dots m)$ is the weight of the entry i in review d. The review vector set is the pattern or data object of the review clustering.

3.1.2 Part-of speech Tagging (POS)

Part-of-speech (PoS) tagging lets the system identify that what Part-of-speech each word in the text belongs to; it may be out of following noun, pronoun, adjective, verb, and interjection and so on. The purpose of PoS tagger is to uncover patterns in reviews using relative frequency assessment of the current segment of review.

3.1.3 Stemming and lemmatization

The process of stemming usually involves words with their roots also known as word stems. As the stem related words for example "speak," "speaker," and "speaking" are rooted from one single word, "read". The bag of word dimensionality however is decreased. While using stemming, nevertheless, care must be exercised because it may exacerbate prejudice. Whenever the words "experiment" and "experience" are combined into one word, the skewed impact of stemming is visible.

3.1.4 Stops-Word removal

Prepositions, articles, and other words that serve as connectors in a sentence are examples of stop words. While there is no definitive list of stop words, several search engines employ several of the most popular, short word forms, like "the," "is," "at," "which," and "on." These words can be eliminated from the review before categorization since they appear often in the review and have not much effect on the sentence's final emotion.

3.1.5 Tokenization into N-grams

Tokenization is the technique of extracting a collection of words from the reviews. Words as well as other items are separated from the input string. Whitespace is a typical divider for recognizing separate words. Because user review data comprises multiple emoticons, URL links, and acronyms that can be easily distinguished as full entities, tokenization of social media data is far more challenging than tokenization of normal text. Combining adjacent words into phrases or n-grams, which can be unigrams, bigrams, trigrams, and so on, is standard procedure.

3.2 RNN Based Classification

The unique aspect of Sentiment analysis and related data is its temporal aspect associated with it. Each word in a sentence depends greatly on what came before and what comes after it. In order to account for this dependency, we need to use a sequence processing neural network. Recurrent Neural Networks in particular are well suited for this task. RNN are the feed-forward neural networks rolled out over time as such they deal with sequence data where the input has some defined ordering which gives rise to several kinds of architectures[39][40]. One of which is, vector to sequence models- these neural nets take in a fixed size vector as input and outputs a sequence of any length in image captioning like the image's vector representation can be an input and its output sequence is a sentence that explains the image. Now Sequence to vector model- is the second type neural networks in which neural nets takes a sequence as input and spits out a fixed length vector. As in sentiment analysis the film evaluation is considered as an input and the output is a fixed size vector representing how good or bad this person thought the movie was. Next and third one, sequence to sequence models- it is the more famous model among all and these neural networks takes input as a sequence and outputs another sequence. Thus Applied to Sentiment Analysis the input could be a sentence user query and the output can be the agent response.

4. RESULTS AND ANALYSIS

For the Analysis of the proposed RNNCore method IMDB review dataset [11] is utilized containing 50,000 reviews with equal polarities. The IMDB review dataset is in the area of movie reviews which is ideal for testing as there are enormous online sources of such user reviews, typically with metadata that offers easily extractable class labels, i.e. polarity (positive or negative) . It is crucial to note that our approaches are not domain-specific and therefore can be easily transferable to other domains if adequate training data is available. Furthermore, movie reviews were shown to be more challenging to categorize than some other product reviews, with movie reviews being simpler to categorize than book and user reviews. For the analysis of the RNNCore the dataset was divided in three sizes, 10,000 reviews (10K), 20,000 reviews (20K) and 50,000 reviews (50K) randomly. These sub datasets were further evaluated against OneR Classifier, Simple RNN without word embedding and the proposed RNNCore, RNN with pre-trained word embedding. The OneR Classifier [42] simply assigns the most frequent (i.e. the class with maximum no of instances) class to the instance without any learning. The OneR acts as a minimum base line for the classifiers. The simple RNN based classifier uses weighted sum of output vector for estimating the sentiment.

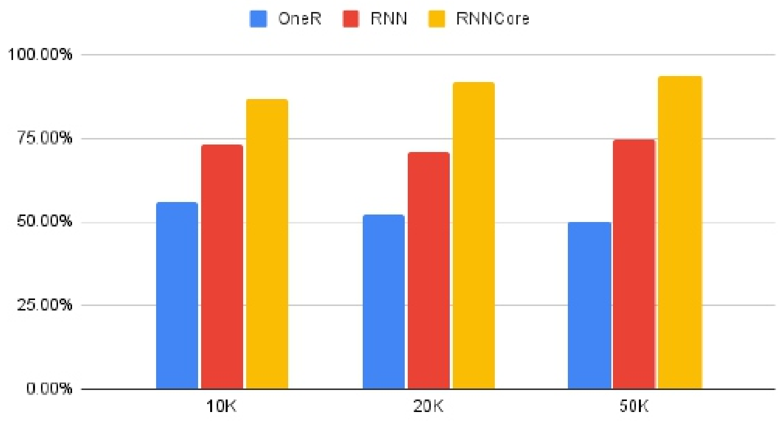
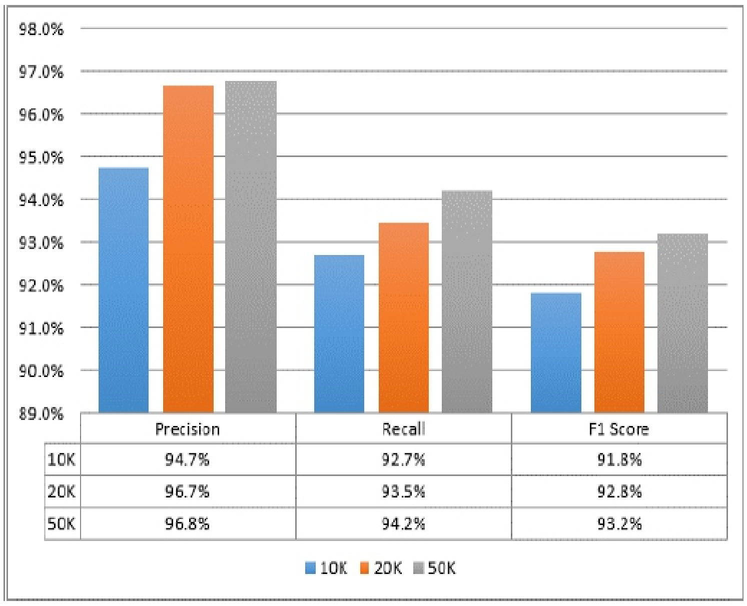


Figure 2: Validation accuracy of the OneR, RNN and RNNCore classifiers

4.1 Validation Accuracy

During the training, validation is done simultaneously to estimate the validation accuracy using loss per epoch (iteration). Mean Squared Error loss function is used to optimize a machine learning algorithm. This Validation Accuracy can be used to compare the models on various subsets of the datasets.

As we can see from the results in figure 3 above that the RNNCore outperforms both baselines in all three subset of the datasets (10K, 20K and 50K). RNNCore works well for all the subsets providing 86.80%, 91.80% and 93.70% validation accuracy respectively which is higher than both OneR (56.10%, 52.07%, and 50.00%) and Simple RNN (73.10%, 71.10% and 74.90%). On Average RNNCore provides 90.77% accuracy compared to 52.67% by OneR and 73.03% by RNN. RNNCore is38.1% better than OneR and 17.74% better than RNN implementations.



	Precision	Recall	F1 Score
10K	94.7%	92.7%	91.8%
20K	96.7%	93.5%	92.8%
50K	96.8%	94.2%	93.2%

Figure 3: Precision, Recall and F1-Score of the RNNCore classifier with 10K, 20K and 50 instances

Experiments were further done to estimate the Precision, Recall and F1-Score [43] of the RNNcore algorithm figure 4. In all the metrics (Precision, Recall and F1-Score) RNNCore performed really well providing on average 96.07% Precision, 94.10% Recall and 92.60% F1-Score on all three datasets.

4.2 Comparative Time Analysis

In terms of time complexity, RNNCore outperforms the simple RNN with twice as much efficiency. Table below shows the comparison between the RNN and RNNCore over the epochs used for training over the IMDB dataset. From 10 epochs to 100 epochs RNNCore works very well against the RNN implementation on average the RNN used around 19.5 minutes to train whereas the RNNCore took around 10.46 minutes, making RNN 46.5% slower than the RNNCore implementation.

The chart of data as in Table 1 is used to show the difference in performance and also to show the shape of the curves. The shape of the curves for both RNN and RNNCore are not exponential as seen in figure 5 and are reaching towards a peak, which is important because it shows that the algorithms take lesser amount of time as they achieve good accuracy. RNNCore however does this faster than the RNN implementation.

Table 1: Comparison of RNN and RNNCore sentiment analysis algorithms in terms of time complexity (minutes)

EPOCHS	RNN	RNN CORE
10	10.02447	3.6168
20	12.27535	6.1665
30	15.39875	7.8672
40	18.77193	10.3005
50	20.64936	11.346
60	22.02725	12.4299
70	23.09473	12.8586
80	23.78683	12.3231
90	24.10644	13.2915
100	25.3592	14.4135
Mean	**19.54943**	**10.46136**

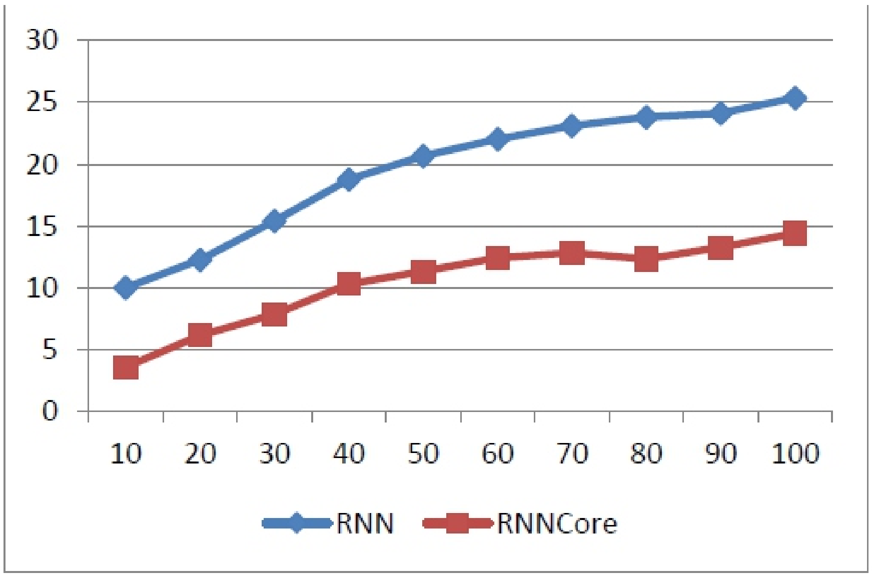


Figure 4: Time Complexity of RNN and RNNCore sentiment analysis algorithms

5. CONCLUSION

In this paper an improved RNN based sentiment analysis method (RNNCore) was developed which utilized word embedding (pre-trained) to increase the efficiency of the traditional RNN network. This paper is focused around a novel method which can make use of such word embedding from Stanford Core NLP in conjunction with RNN to improve on accuracy and reduce time complexity. The RNNCore is applied to achieve better sentiment analysis and finds sentiment cues more accurately than the conventional RNN up to 17% more. For upcoming development, we would like to incorporate RNNCore for aiding the recommendation models, especially cross domain recommendation systems (CDRS). The movie review information can be used to fill the sparse cross domain recommendation matrix for increasing the efficiency of the traditional CDRS.

6. REFERENCES

[1]. Yadav, Ashima, and Dinesh Kumar Vishwakarma. "Sentiment analysis using deep learning architectures: a review." Artificial Intelligence Review 53.6 (2020): 4335-4385.

[2]. Dang, N. C., Moreno-García, M. N., & De la Prieta, F. (2020). Sentiment analysis based on deep learning: A comparative study. Electronics, 9(3), 483.

[3]. Maulud, D. H., Zeebaree, S. R., Jacksi, K., Sadeeq, M. A. M., & Sharif, K. H. (2021). State of art for semantic analysis of natural language processing. Qubahan Academic Journal, 1(2), 21-28.

[4]. Seo, S., Kim, C., Kim, H., Mo, K., & Kang, P. (2020). Comparative study of deep learning-based sentiment classification. IEEE Access, 8, 6861-6875.

[5]. Habimana, O., Li, Y., Li, R., Gu, X., & Yu, G. (2020). Sentiment analysis using deep learning approaches: an overview. Science China Information Sciences, 63(1), 1-36.

[6]. Agüero-Torales, M. M., Salas, J. I. A., & López-Herrera, A. G. (2021). Deep learning and multilingual sentiment analysis on social media data: An overview. Applied Soft Computing, 107373.

[7]. Jouravlev, O., & Jared, D. (2020). Native language processing is influenced by L2-to-L1 translation ambiguity. Language, Cognition and Neuroscience, 35(3), 310-329.

[8]. Heinz, J., De la Higuera, C., & Van Zaanen, M. (2015). Grammatical inference for computational linguistics. Synthesis Lectures on Human Language Technologies, 8(4), 1-139.

[9]. Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., &McClosky, D. (2014, June). The Stanford CoreNLP natural language processing toolkit. In Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations (pp. 55-60).

[10]. Ahmed, A., & Yousuf, M. A. (2021). Sentiment Analysis on Bangla Text Using Long Short-Term Memory (LSTM) Recurrent Neural Network. In Proceedings of International Conference on Trends in Computational and Cognitive Engineering (pp. 181-192). Springer, Singapore.

[11]. Liu, J., Zheng, S., Xu, G., & Lin, M. (2021). Cross-domain sentiment aware word embeddings for review sentiment analysis. International Journal of Machine Learning and Cybernetics, 12(2), 343-354.

[12]. Pathak, A. R., Pandey, M., & Rautaray, S. (2021). Topic-level sentiment analysis of social media data using deep learning. Applied Soft Computing, 108, 107440.

[13]. Esuli, A., & Sebastiani, F. (2007). SentiWordNet: a high-coverage lexical resource for opinion mining. Evaluation, 17(1), 26.

[14]. Hamouda, A., & Rohaim, M. (2011, January). Reviews classification using sentiword net lexicon. In World congress on computer science and information technology (Vol. 23, pp. 104-105). sn.

[15]. Kumar, K. N., & Uma, V. (2021). Intelligent sentinet-based lexicon for context-aware sentiment analysis: Optimized neural network for sentiment classification on social media. The Journal of Supercomputing, 1-25.

[16]. Eng, T., Nawab, M. R. I., & Shahiduzzaman, K. M. (2021). Improving Accuracy of The Sentence-Level Lexicon-Based Sentiment Analysis Using Machine Learning. International Journal of Scientific Research in Computer Science Engineering and Information Technology, 3307, 57-68.

[17]. Khoo, C. S., &Johnkhan, S. B. (2018). Lexicon-based sentiment analysis: Comparative evaluation of six sentiment lexicons. Journal of Information Science, 44(4), 491-511.

[18]. Fellbaum, C. (2010). WordNet. In Theory and applications of ontology: computer applications (pp. 231-243). Springer, Dordrecht.

[19]. Baccianella, S., Esuli, A., &Sebastiani, F. (2010, May). Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In Lrec (Vol. 10, No. 2010, pp. 2200-2204).

[20]. Rice, D. R., & Zorn, C. (2021). Corpus-based dictionaries for sentiment analysis of specialized vocabularies. Political Science Research and Methods, 9(1), 20-35.
[21]. Kumar, A., & Sebastian, T. M. (2012). Sentiment analysis on twitter. International Journal of Computer Science Issues (IJCSI), 9(4), 372.
[22]. Mihalcea, R., Corley, C., &Strapparava, C. (2006, July). Corpus-based and knowledge-based measures of text semantic similarity. In Aaai (Vol. 6, No. 2006, pp. 775-780).
[23]. Moreno-Ortiz, A., &Fernández-Cruz, J. (2015). Identifying polarity in financial texts for sentiment analysis: a corpus-based approach. Procedia-Social and Behavioral Sciences, 198, 330-338.
[24]. Govindarajan, M. (2013). Sentiment analysis of movie reviews using hybrid method of naive bayes and genetic algorithm. International Journal of Advanced Computer Research, 3(4), 139.
[25]. Appel, O., Chiclana, F., Carter, J., & Fujita, H. (2016). A hybrid approach to the sentiment analysis problem at the sentence level. Knowledge-Based Systems, 108, 110-124.
[26]. Ren, R., Wu, D. D., & Liu, T. (2018). Forecasting stock market movement direction using sentiment analysis and support vector machine. IEEE Systems Journal, 13(1), 760-770.
[27]. Xia, H., Yang, Y., Pan, X., Zhang, Z., &An, W. (2020). Sentiment analysis for online reviews using conditional random fields and support vector machines. Electronic Commerce Research, 20(2), 343-360.
[28]. Dashtipour, K., Ieracitano, C., Morabito, F. C., Raza, A., & Hussain, A. (2021). An ensemble based classification approach for persian sentiment analysis. In Progresses in Artificial Intelligence and Neural Systems (pp. 207-215). Springer, Singapore.
[29]. Rani, S., & Gill, N. S. (2020). Hybrid Model using Stack-Based Ensemble Classifier and Dictionary Classifier to Improve Classification Accuracy of Twitter Sentiment Analysis. International Journal, 8(7).Convolution Neural Networks
[30]. Giménez, M., Palanca, J., &Botti, V. (2020). Semantic-based padding in convolutional neural networks for improving the performance in natural language processing. A case of study in sentiment analysis. Neurocomputing, 378, 315-323.
[31]. Huang, M., Xie, H., Rao, Y., Liu, Y., Poon, L. K., & Wang, F. L. (2020). Lexicon-Based Sentiment Convolutional Neural Networks for Online Review Analysis. IEEE Transactions on Affective Computing.
[32]. Kurniasari, L., &Setyanto, A. (2020, February). Sentiment Analysis using Recurrent Neural Network. In Journal of Physics: Conference Series (Vol. 1471, No. 1, p. 012018). IOP Publishing.
[33]. Sachin, S., Tripathi, A., Mahajan, N., Aggarwal, S., &Nagrath, P. (2020). Sentiment analysis using gated recurrent neural networks. SN Computer Science, 1(2), 1-13.
[34]. Goud, A., & Garg, B. (2021). Sentiment Analysis Using Long Short-Term Memory Model in Deep Learning. In 2nd EAI International Conference on Big Data Innovation for Sustainable Cognitive Computing (pp. 15-23). Springer, Cham.
[35]. Song, M., & Chambers, T. (2014). Text mining with the Stanford CoreNLP. In Measuring scholarly impact (pp. 215-234). Springer, Cham.
[36]. Aizawa, A. (2003). An information-theoretic perspective of tf–idf measures. Information Processing & Management, 39(1), 45-65.
[37]. Hussain, S., Sianaki, O. A., & Ababneh, N. (2019, March). A survey on conversational agents/chatbots classification and design techniques. In Workshops of the International Conference on Advanced Information Networking and Applications (pp. 946-956). Springer, Cham.
[38]. Radhakrishnan,P, Introduction to Recurrent Neural Network (2017,20August)TowardsDataScience,https://towardsdatascience.com/introductionto-recurrent-neural-network- 27202c3945f3
[39]. Understanding LSTM networks (2017, 27August) Colah's blog, https://colah.github.io/posts/2015-08-Understanding-LSTMs/
[40]. Jason Brownlee, (2020), One-Class Classification Algorithms for Imbalanced Datasets, https://machinelearningmastery.com/one-class-classification-algorithms/
[41]. Goutte, C., & Gaussier, E. (2005, March). A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In European conference on information retrieval (pp. 345-359). Springer, Berlin, Heidelberg.

Nikita Taneja is working as an Assistant Professor at Department of Computer Science and Technology of Manav Rachna University (MRU), Faridabad. She has more than 13 years of teaching and industry experience in reputed institutions of India. She is pursuing her Ph.D. (Computer Engineering) from Manav Rachna University in the field of Cross Domain Recommender systems.

Dr. Hardeo K Thakur is working as an Associate Professor at department of Computer Science and Technology of Manav Rachna University (MRU), Faridabad. He has more than 12 years of teaching and research experience in leading institutions of India. He has earned his Ph.D (Computer Engineering) from University of Delhi in 2017 in the field of data mining. Dr. Thakur has published 14 research papers in international journal of repute, 13 papers in international conferences and 1 book chapter. His current research interests are Data Mining, Dynamic Graph Mining, Machine Learning and Big Data analytics. He is an active referee for many international Journals and Conferences.

ISSN 2210-142X

Load Balancing Implementation Strategy for Various Services in Software Defined Network using ONOS Controller

[1]Rohmat Tulloh, [2]Muhammad Iqbal, [3]Mohammad Rifki Baihaqi, [4]Aditya Bayu Aji Pamungkas

[1,2,3,4]School of Applied Science, Telkom University, Bandung, Indonesia

E-mail address: rohmatth@telkomuniversity.ac.id

ABSTRACT

The popularity of Software Defined Network (SDN) is due to its ability to simplify and automate operational network processes to improve service performance. SDN is not only related to the separation of control plane and data plane, but its architecture also includes network services at other top tiers such as load balancing, security, and application performance. Load balancing is a technique to divide traffic loads equally. The correct load balancing algorithm will make the network better stable because network resources must always be operational and available to cope with the increasing demand for services and service users. Scalability and diversity of services on the network such as HTTP, FTP, and VoIP are also concerned with selecting the proper load balancing algorithms. SDN-based software makes it easy to develop and integrate with other software and hardware. This study implemented a load balancing algorithm using an L4-L7 load balancer connected with an SDN controller, ONOS, which automates traffic load sharing on a Web server, FTP server, and VoIP server. The scheduling algorithms used are round robin, least connection, dynamic ratio, and ratio. The test parameters used are Throughput, Response time, Request loss, Block call, and CPU Utilization.

Keywords: Software Defined Network, Load Balancing, Least Connection, Round Robin, Dynamic Ratio, Ratio.

1. INTRODUCTION (HEADING 1)

Computer network technology is developing very rapidly, and the impact of this rapid development makes the human need for services through internet access increase sharply. The increase in service requests and the number of users of internet services make the server often overloaded. A common way to overcome overload is to add a new server or add a hard disk to the database, but this costs a considerable amount. The solution that can solve it is the application of load balancing techniques. Load balancing is a mechanism for dividing compute loads into multiple servers. The goal is to optimize resources and increase throughput to not overload [1]–[3].

However, load balancing algorithms are very diverse. Therefore, selecting the proper load balancing algorithm will make the network have better stability because network resources must always be available to cope with the increasing demand for services and the number of service users. In addition, scalability and diversity of services on the network such as HTTP, FTP, and VoIP are also essential benchmarks in selecting the proper load balancing algorithms [4].

Software-Defined Network (SDN) is an innovation in network architecture. The concept of SDN is to separate the Control Plane and the Data Plane through the use of the OpenFlow protocol [5], [6]. Control Plane is moved out of the network device so that only the Data Plane is inside the network device [7], and the load balancing policy can be managed centrally through a controller. SDN centralized network control makes network setup easier, flexible, and faster [8]–[10]. SDN-based software makes it easy to develop and integrate with other software and hardware. SDN can also be used to change network behavior, make such changes automatically, and maximize network devices such as load balancing[11].

2. CURRENT STATE AND CHALLENGES

The load balancing application works by dividing the load in requests given by the client to the server [12], [13]. The algorithms observed are dynamic ratio, ratio, round robin, and least connection.

The dynamic ratio is an algorithm whose weight ratio is based on continuous monitoring of the server and, therefore, is constantly changing and performs connection sharing based on the monitored server performance parameters in real-time. In comparison, the ratio algorithm uses a fixed ratio parameter calculation for each server. The server with the highest ratio is given a more significant load. on the other hand, servers with smaller ratio values will be given less load [14]. The Round-Robin algorithm on load balancing works when receiving requests from clients will be directed to multiple servers in turn and sequentially from one server to another to receive load evenly [15]. While the least connection algorithm serves to do load sharing based on the number of connections being served by the server. The server with the least connection will be given the next load [16].

In research [1], [17], it has been proven that the selection of the correct load balancing algorithm provides improved performance of Web Server, FTP Server, and VoIP services on conventional networks. In SDN-based networks dedicated to high scalability networks, the use of load balancing algorithms is interesting to research in more depth.

Additionally, research [6], [18] uses Mininet to create virtual network environments for evaluation. However, it turns out that the use of Mininet does not reflect the actual overhead. Therefore, this research spread experimental environments and built experimental platforms using real hardware [19]. Sdn controller implementation handles datalink layers to network layers only, while integrating with F5 can add SDN controller capabilities to handle from transport layer to application layer [14].

However, SDN implementation has a challenge: how the strategy is to provide good performance results. This study implemented load balancing in the transport layer to the application layer using the F5 application. F5 is used to enable applications and networks to communicate with each other. Then, analyze four load balancing algorithms on SDN-based networks consisting of MikroTik switch that act as data plane and ONOS as controller. ONOS is a variant of the SDN controller that uses the Java programming language that takes advantage of the Open Service Gateway (OSGi) framework initiative. With this framework, it will be more accessible when installing and updating applications [20], [21].

3. SYSTEM DESIGN AND IMPLEMENTATION

A. System Design

The network topology design implemented in this study can be seen in figure 1. There are two MikroTik routers set as OpenFlow switches and connected to the ONOS controller. The switch only serves as a data plane, while the control plane function becomes centered in the controller. The ONOS controller feature used in this study uses a reactive forwarding application that serves to allow or reject data traffic on SDN networks. On servers 1 and 2, each has three virtual servers with different services. Each servers consist of web services with http protocol, FTP for file transfers, and SIP for VOIP services. For testing, the number of clients accessing the server will increase to see network performance. The addition of the client is done by a traffic generator that will provide a request to the server.

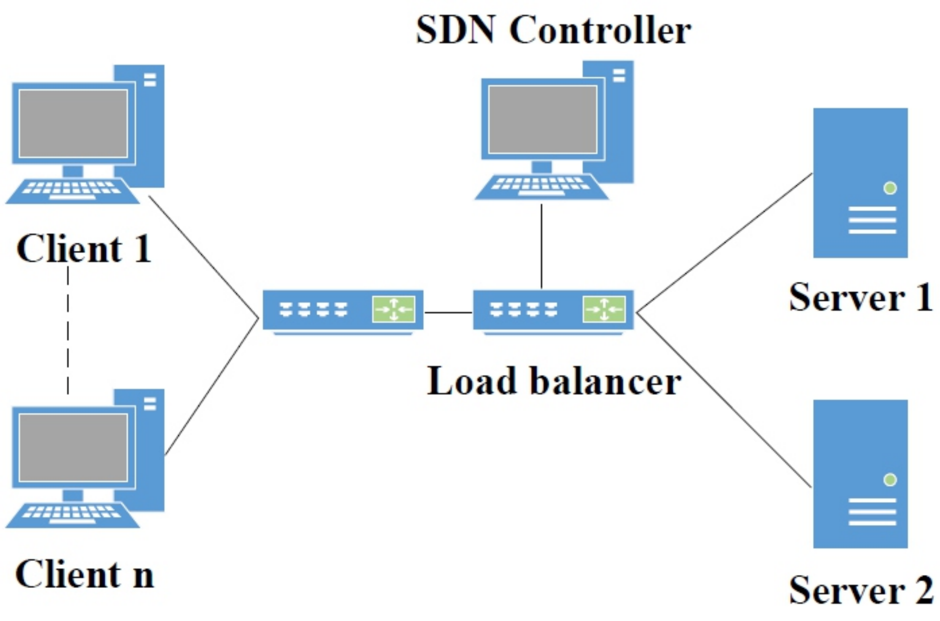


Figure 1. System Design

The software used in this study includes:

- ONOS-2.0.0 as SDN controller.
- F5-BIG-IP-12.1.4 LTM sebagai virtual load balancer.
- Ubuntu 16.04 as a server and client operating system.
- VMware is software for virtual machines that run many operating systems in one hardware.
- Nginx as Web server software.
- Proftpd as FTP server software.
- Asterisk as a VoIP server software.
- Httperf is the software used to generate requests on the webserver service.
- Jmeter Application serves to generate traffic for the FTP service.
- SIP serves as a call generator on the VoIP server service.

B. Load Balancing Algorithms

The load-feeding scenario on the server using four compared algorithms is as follows:

1) Round Robin. Server load on the round-robin algorithm is done by dividing the request load and sequentially from server 1 to server 2.
2) Least Connection. Server load on the least connection algorithm is done by dividing the load based on the number of requests served by the server. The server with the least requests will be given the next load. Otherwise, the server with many requests will be redirected load to the server with a lower load.
3) Dynamic Ratio. Server load on dynamic ratio algorithm is done dynamically by looking at the fastest node response. Dynamic ratio algorithm testing disposes of requests to servers with the same significant CPU and memory specifications. The dynamic ratio algorithm will load dynamically by looking at the CPU and memory conditions of both servers. Dynamically loading is helpful for the more balanced performance of each server.
4) Ratio. The ratio algorithm distributes requests to servers that have the same significant CPU and memory specifications. The ratio algorithm performs load sharing evenly between the two servers used. C. Test Scenarios

We are testing four load balancing algorithms on SDN using web servers, FTP servers, and VoIP servers. Testing is conducted to obtain throughput parameters, response time, and request loss on web server services and FTP servers. While the measurement of services on VOIP server using block call parameters.

As in Table 1, it appears that in the webserver service is done testing by giving load to the server as many as 200 requests, 400 requests, 600 requests, 800 requests, and 1000 requests. step-by-step request generation using HTTP software.

The FTP server service is done testing by doing file transfer upload and download with a file size of 653.5 KB as much as 10 file transfers, 20 file transfers, 30 file transfers, 40 file transfers, and 50 file transfers. This scenario using Jmeter application.

The VoIP server service will be tested by creating calls as much as 10 cps, 15 cps, 20 cps, 25 cps, and 30 cps within one minute using SIPp software. In the test, 30 tests were conducted to get maximum results.

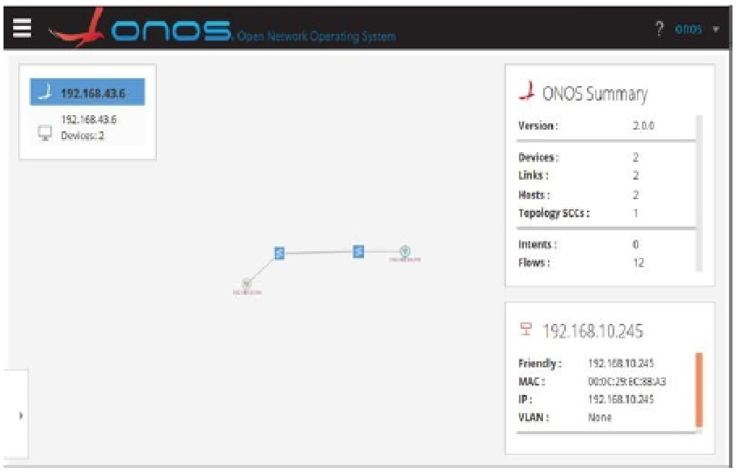
TABLE I. TEST SCENARIOS

Service Type	Number of testing				
	1	*2*	*3*	*4*	*5*
Web (request)	200	400	600	800	1000
File Transfer (file)	10	20	30	40	50
VoIP (call per second)	10	15	20	25	30

1. PERFORMANCE EVALUATION AND EXPERIMENTAL RESULT

A. Controller Testing

In testing, the controller is done to determine if all devices can already communicate with the controller, in Figure 2. Controllers can recognize all devices connected to the controller according to the topology used in this study, such as clients, switches, and server load balancers. For Web server, FTP server and VoIP server are integrated by load balancer server, so the controller does not monitor it. Then done test connectivity to know the data plane device and controller is running well so that the host connected with the data plane device can communicate.



192.168.43.6
Devices: 2

Version : 2.0.0
Devices : 2
Links : 2
Hosts : 2
Topology SCCs : 1
Intents : 0
Flows : 12

Friendly : 192.168.10.245
MAC : 00:0C:29:EC:88:A3
IP : 192.168.10.245
VLAN : None

Figure 2. Onos controller interface

B. Web Service Performance Evaluation

1. Throughput

Throughput measurement in web service aims to know the server's ability to provide services to the client. In this parameter, the more significant the throughput value, the better the load balancing performance. Figure 3 shows that the dynamic ratio algorithm provides better throughput when the number of requests is still below 600, although it is not very significant compared to the ratio algorithm. However, when the number of requests is getting bigger, it turns out that the ratio algorithm can outperform other algorithms. This condition proves that sharing the load to other servers when the service request increases has a significant effect on throughput value. In addition, the more accessible load balancing settings performed at SDN make the load balancing algorithm based on ratio settings superior to others.

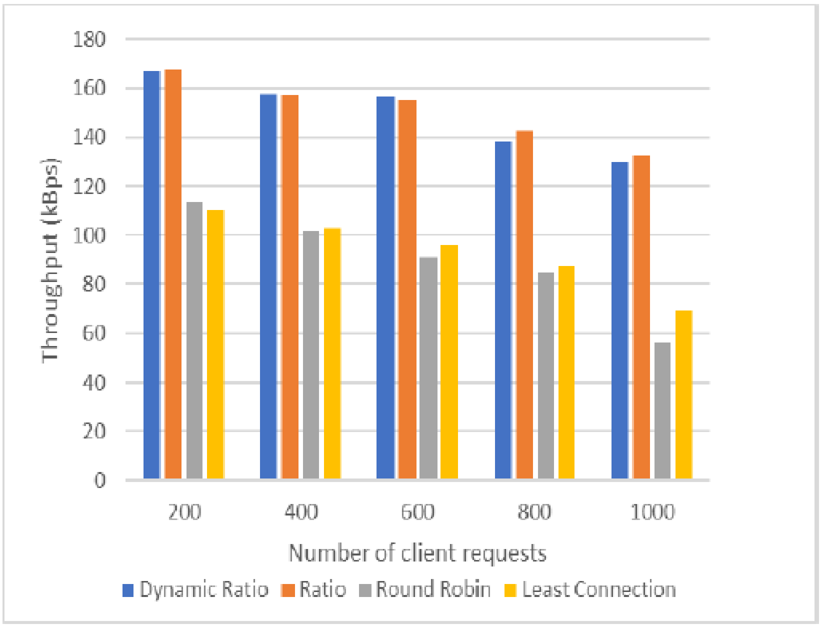


Figure 3. Throughput measurement results on web service

2. Respon Time

Measurement of response time on the web service aims at how quickly a server can respond to request packets from the client. The smaller the response time number, the faster the server responds to clients. Figure 4 shows that the round robin algorithm is faster when the number of requests is still below 800, although it is not very significant compared to the least connection algorithm. However, when the number of requests is getting bigger, it turns out that the least connection algorithm can outperform other algorithms. This condition proves that when a service request increases, it dramatically affects how it takes the server to respond to user requests.

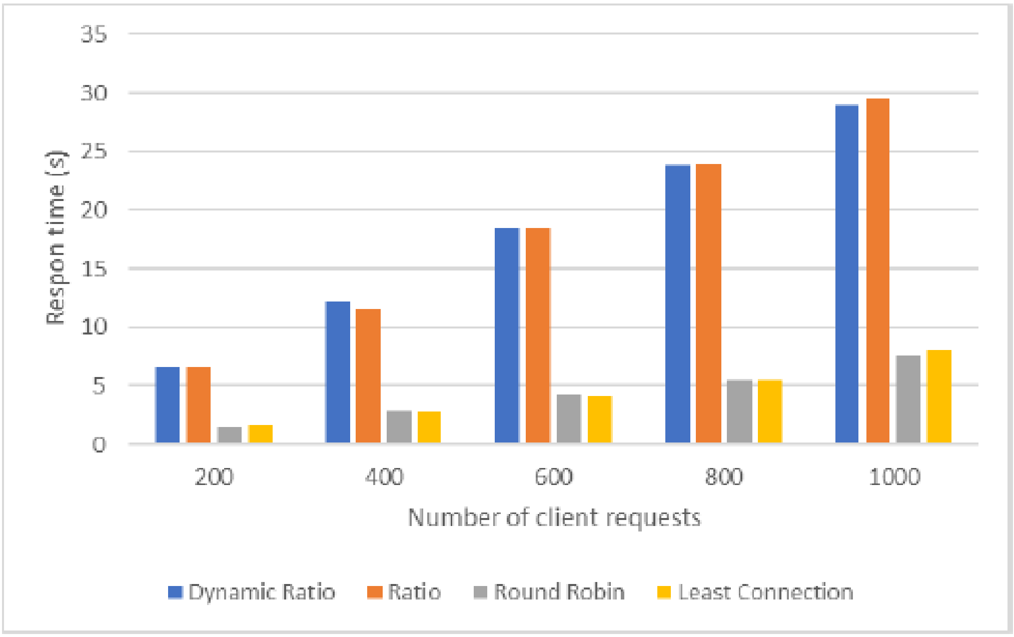


Figure 4. Measurement results of time response on web service

3. Request Loss

The measurement of request loss on the web service aims to determine the number of request failures that the server cannot serve. The smaller the number of the request loss, the more reliable the server responds to requests from the client. Figure 5 shows that the dynamic ratio and ratio algorithms are more reliable because 1000 requests did not occur until the experiment. While at the time of the request above 800, the failure rate o,46% occurs in the round-robin algorithm but is still below the threshold of request loss allowed.

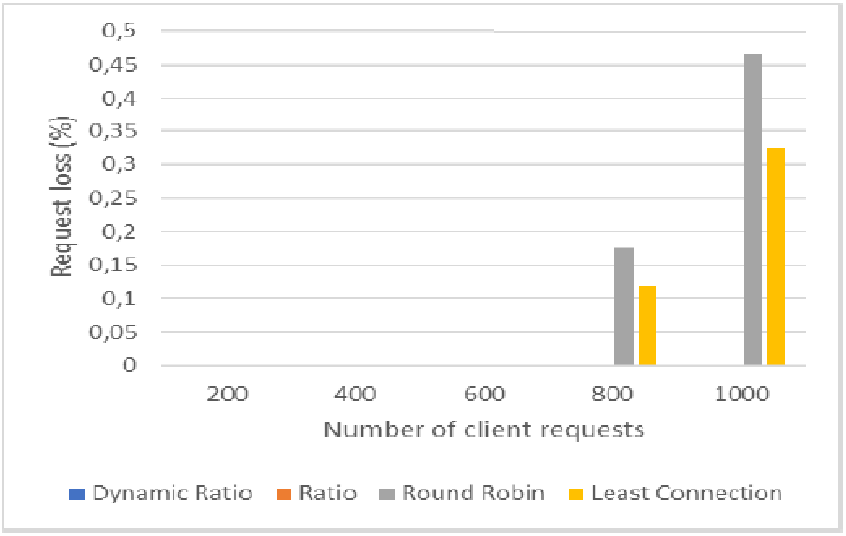


Figure 5. Measurement results of request loss on web service

4. CPU Utilization

CPU utilization measurement in web service aims to observe resources used to run the client's request service process. The smaller the number of CPU utilization, the fewer resources used.

Figure 6 shows that the dynamic ratio and ratio algorithm is more wasteful although still below 8.5%, compared to algorithms round robin and least connection that looks more optimal. The least connection algorithm can reduce a load of each server more optimally than other algorithms. The smaller the CPU value obtained, it can reduce the occurrence of overload in the server.

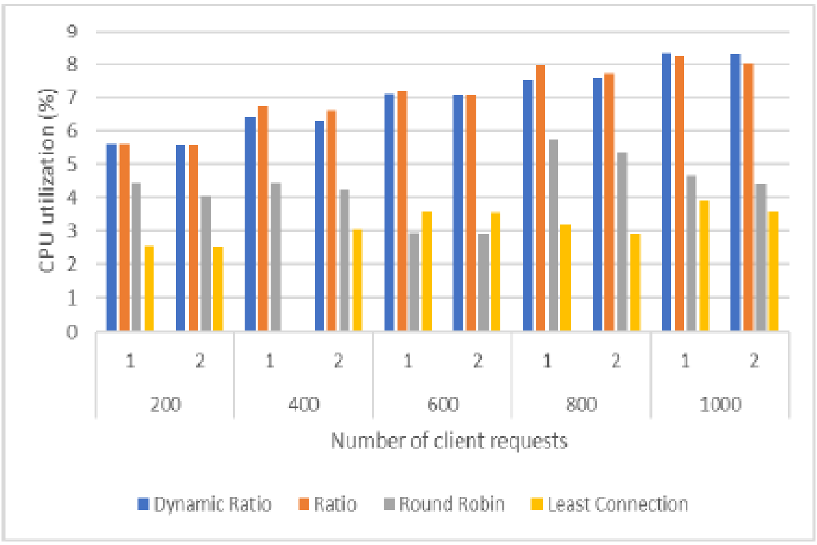


Figure 6. Results of measuring CPU utilization on web services

C. FTP Service Performance Evaluation

1. Throughput

Throughput measurement in FTP service aims to know the server's ability to provide services to the client. In this parameter, the more excellent the throughput value, the better the load balancing performance. Figure 7 shows that the round-robin algorithm provides better throughput than other algorithms as the number of transfer files increases. It is because the time it takes to serve all file transfers is more stable and constant on the appeal of web servers.

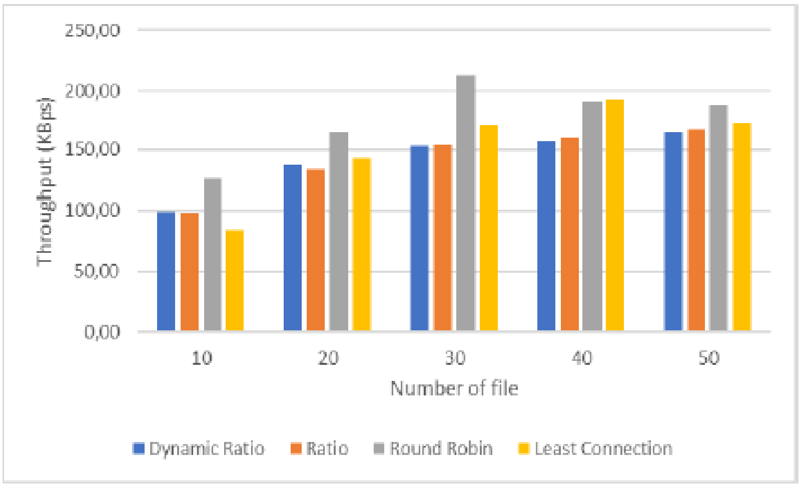


Figure 7. Throughput measurement results on FTP Service

2. Respon Time

Measurement of response time in FTP service aims at how quickly a server can respond to the client. The smaller the response time level, the faster the server responds to the clients. Figure 8 shows that the value

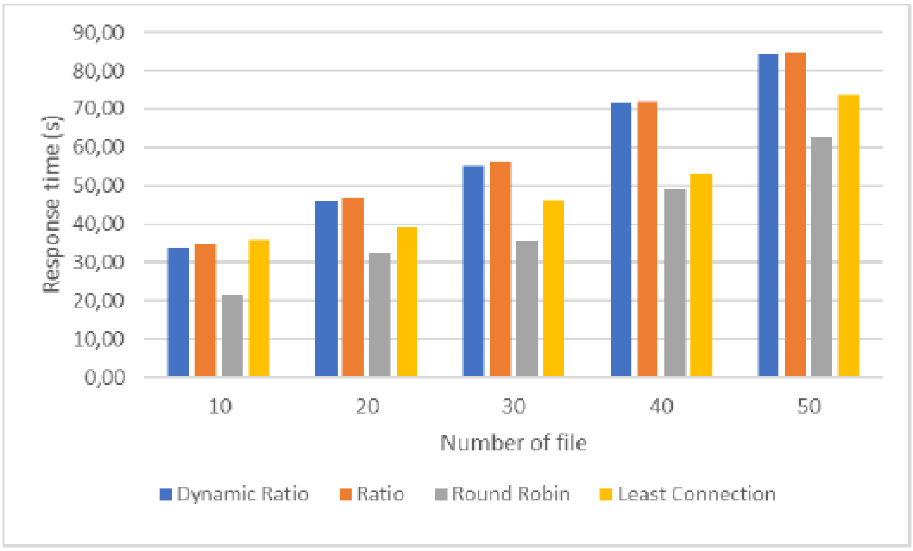


Figure 8. Response time measurement results on FTP Service

3. Request Loss

The measurement of request loss on the FTP service aims to determine the total of request failures that the server unserved. The smaller the value of the request loss, the more reliable the server responds to requests from the client. Figure 9 shows that the algorithm shows that dynamic ratio and ratio algorithms are more reliable because there are no failures. While the highest request loss when generating 50 transfer files occurred in the round-robin algorithm of 0.66 %, while the algorithm at least connected 0.4% but still below the threshold of request loss allowed.

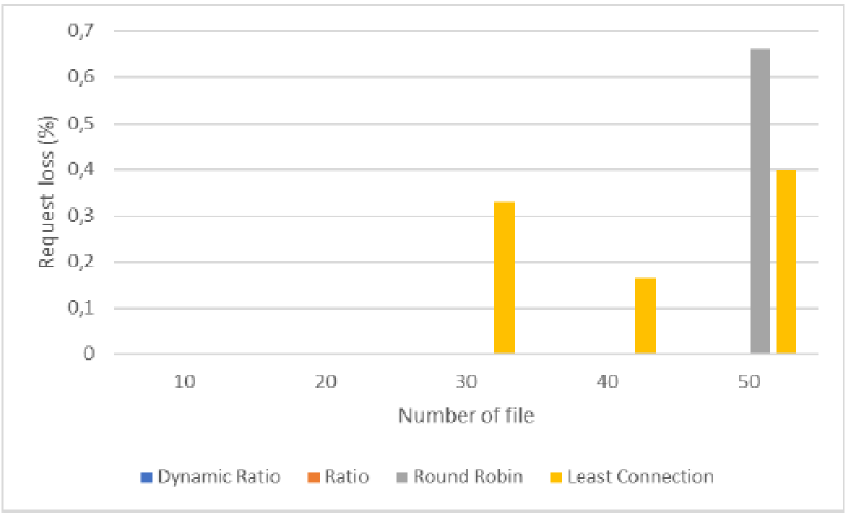


Figure 9. Measurement result of request loss on FTP Service

5. CPU UTILIZATION

CPU utilization measurement in FTP service aims to observe resources used to run the client's request service process. The smaller the number of CPU utilization, the fewer resources used. Figure 10 shows that dynamic ratio and ratio algorithms are better when compared to around robin and least connection that looks more wasteful. Dynamic ratio algorithms can reduce a load of each server more optimally than other algorithms. The smaller the CPU value obtained, it can reduce the occurrence of overload in the server.

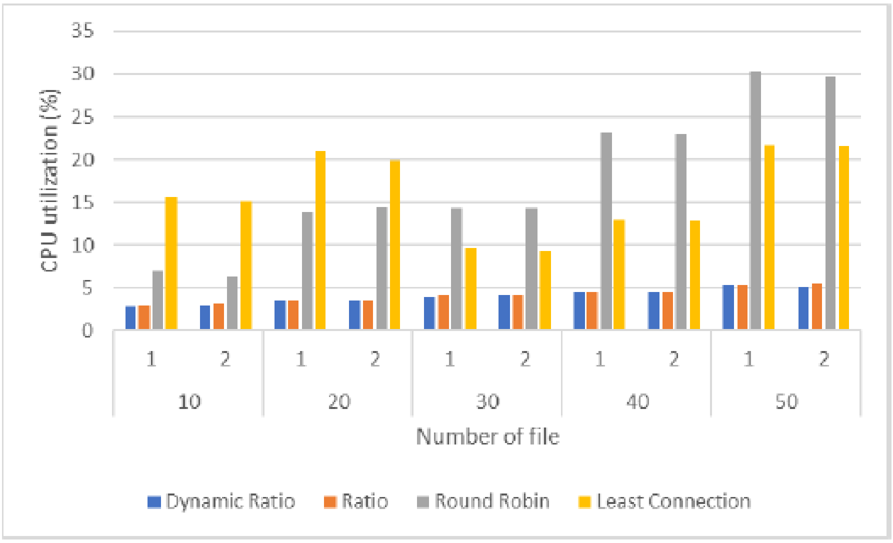
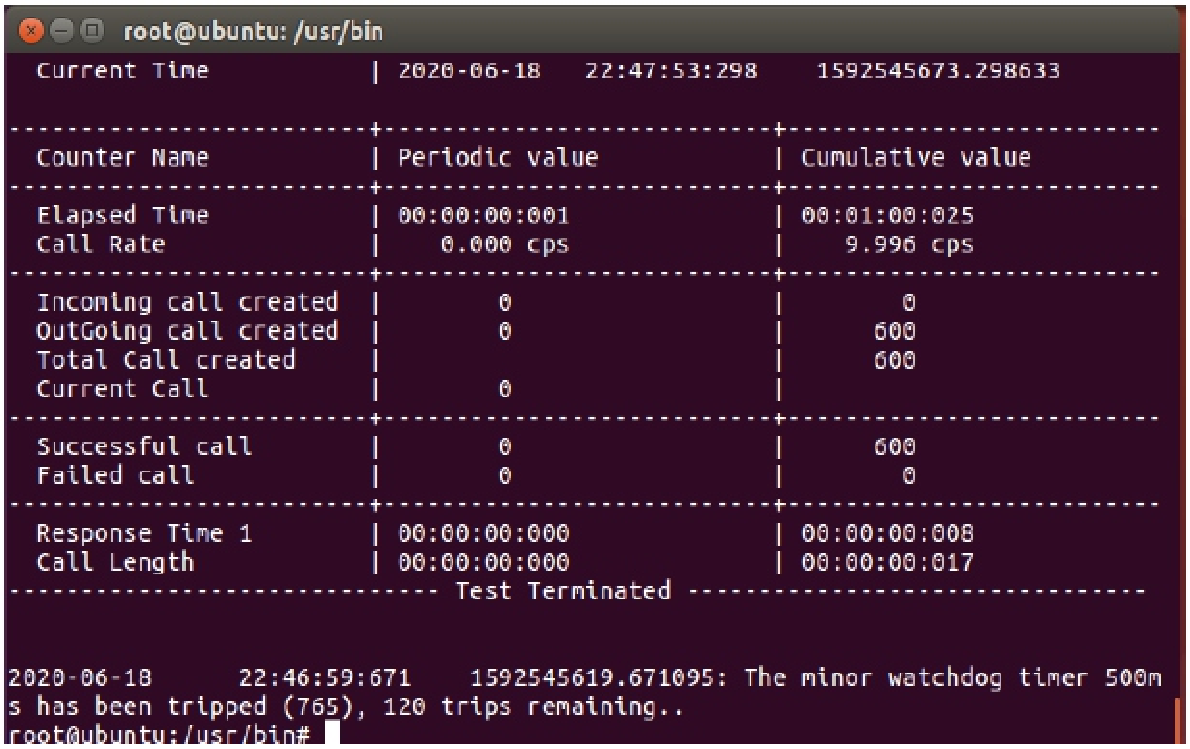


Figure 10. CPU Utilization measurement results on FTP Service

D. VoIP Service Performance Evaluation

1) Block Call

To generate traffic load on the VOIP server service, we used the SIPp Tester application. Traffic load in the form of calls per second is generated in stages to see the server's performance. The process of using the SIPp Tester can be seen in Figure 11.



```
root@ubuntu: /usr/bin
  Current Time              | 2020-06-18   22:47:53:298    1592545673.298633

-----------------------------+---------------------------+--------------------------
  Counter Name              | Periodic value            | Cumulative value
-----------------------------+---------------------------+--------------------------
  Elapsed Time              | 00:00:00:001              | 00:01:00:025
  Call Rate                 |    0.000 cps              |    9.996 cps
-----------------------------+---------------------------+--------------------------
  Incoming call created     |        0                  |        0
  OutGoing call created     |        0                  |      600
  Total Call created        |                           |      600
  Current Call              |        0                  |
-----------------------------+---------------------------+--------------------------
  Successful call           |        0                  |      600
  Failed call               |        0                  |        0
-----------------------------+---------------------------+--------------------------
  Response Time 1           | 00:00:00:000              | 00:00:00:008
  Call Length               | 00:00:00:000              | 00:00:00:017
------------------------------ Test Terminated --------------------------------


2020-06-18      22:46:59:671    1592545619.671095: The minor watchdog timer 500m
s has been tripped (765), 120 trips remaining..
root@ubuntu:/usr/bin#
```

Figure 11. Loading Process Using SIPp Tester

Figure 12 shows that the increasing number of VoIP calls and the number of blocks calls also increased but still below 1% for all algorithms used. From the test results, it can be seen that at the most significant traffic load (30 cps), the Round robin algorithm has the highest block call, which is 0.96 percent, then the

Dynamic Ratio is 0.66 percent, and the Algorithm Ratio is 0.53 percent. The least connection algorithm produces the smallest call block among the other four algorithms, which is 0.46 percent. It can be seen that the Round robin algorithm will experience a significant increase when the traffic load on VOIP services increases

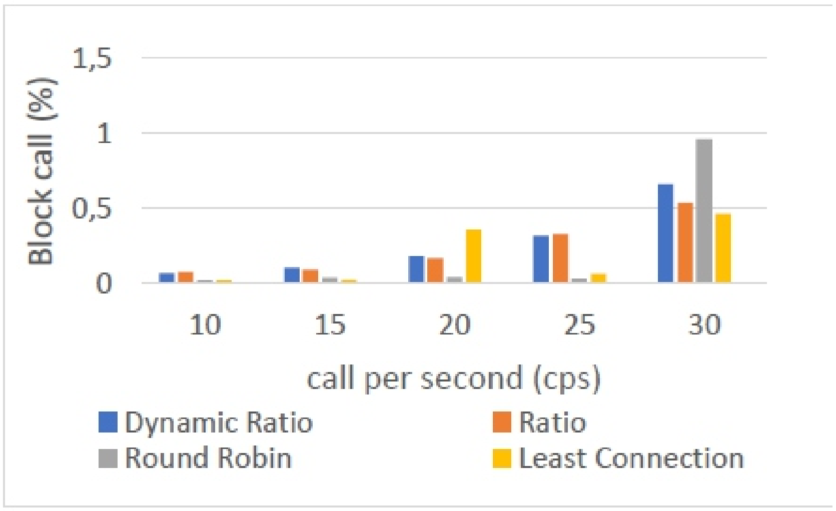


Figure 12. Block call measurement results on VOIP Service

2. CPU Utilization

CPU utilization measurement in VoIP service aims to observe resources used to run the client's request service process. The smaller the number of CPU utilization, the fewer resources used. Figure 13 shows that the dynamic ratio, least connection, and ratio algorithms are better when compared to around robin Algoritma that looks more wasteful. The least connection algorithm can reduce a load of each server more optimally than other algorithms. The smaller the CPU value obtained, it can reduce the occurrence of overload in the server.

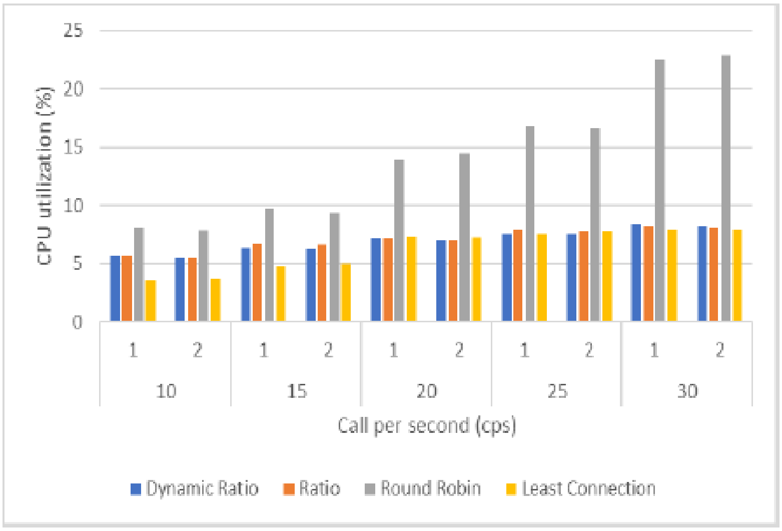


Figure 13. CPU Utilization measurement results on VOIP service

4. CONCLUSION

The selection of the correct load balancing algorithm will make the network have better stability because network resources must always be operational and available to cope with increasing service requests and the number of service users. Based on the tests done, the service on the web server will provide better throughput when using the dynamic ratio algorithm, but if the number of requests is more than 800, it is better to use the ratio algorithm. However, it turns out that even though the throughput is high, the response time of the ratio and dynamic algorithms is still slower than the least connection, but it is still below the permitted threshold. As for the FTP service, it can be seen that the round-robin algorithm is superior to the throughput parameter but has a weakness in overcoming request loss at a high number of requests. Then for VoIP services, it is shown that the dynamic ratio and ratio algorithms provide more reliable performance in controlling the increase in the number of service requests. From the measurement results, it can be concluded that the load balancing algorithm selection is strongly influenced by the characteristics of the service passed and also the parameters that are the priority. There is no superior algorithm in all measurement strategies. Currently, the implementation of SDN is still using a single controller architecture. Subsequent research will explore a larger scale of SDN implementation in the distributed data plane and control plane layers.

REFERENCES

[1] K. W. Murti, "Comparative Analysis of Load Balancing Dynamic Ratio and Server Ratio Algorithms," in 2020 FORTEI-International Conference on Electrical Engineering (FORTEI-ICEE), 2020, pp. 162–167, doi: 10.1109/FORTEI-ICEE50915.2020.9249815.

[2] K. Rishabh, "Analysis of Load Balancing Algorithm in Software Defined Networking," in 2017 2nd International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS), 2017, pp. 1–4.

[3] S. K. Ejaz, Analysis of the trade-off between performance and energy consumption of existing load balancing algorithms, no. November. 2011.

[4] R. Tulloh, J. G. Amri Ginting, A. Mulyana, and M. Lutfi, "Performance Comparison of File Transfer Protocol Service between Link State and Distance Vector Routing Protocol in Software Defined Network," in IOP Conference Series: Materials Science and Engineering, 2020, vol. 982, no. 1, doi: 10.1088/1757-899X/982/1/012026.

[5] H. A. Kar and G. M. Rather, "Multilayer Software Defined Networking Architecture for the Internet of Things," Int. J. Comput. Digit. Syst., vol. 90, no. 4, pp. 735–745, 2020, doi: 10.12785/ijcds/090420.

[6] H. Tussyadiah, R. M. Negara, and D. D. Sanjoyo, "Distributed gateway-based load balancing in software defined network," Telkomnika (Telecommunication Comput. Electron. Control., vol. 18, no. 5, pp. 2352–2361, 2020, doi: 10.12928/TELKOMNIKA.v18i5.14851.

[7] D. B. Rawat and S. R. Reddy, "Software Defined Networking Architecture, Security and Energy Efficiency: A Survey," IEEE Commun. Surv. Tutorials, vol. 19, no. 1, pp. 325–346, 2017, doi: 10.1109/COMST.2016.2618874.

[8] O. Marzuqi, A. Virgono, and R. M. Negara, "Implementation model architecture software defined network using raspberry Pi: A review paper," Telkomnika (Telecommunication Comput. Electron. Control., vol. 17, no. 3, 2019, doi: 10.12928/TELKOMNIKA.V17I3.8859.

[9] R. Dewanto, R. Munadi, and R. M. Negara, "Improved Load Balancing on Software Defined Network-based Equal Cost Multipath Routing in Data Center Network," J. Infotel, vol. 10, no. 3, p. 157, 2018, doi: 10.20895/infotel.v10i3.379.

[10] H. T. Zaw and A. H. Maw, "Traffic management with elephant flow detection in software defined networks (SDN)," Int. J. Electr. Comput. Eng., vol. 9, no. 4, pp. 3203–3211, 2019, doi: 10.11591/ijece.v9i4.pp3203-3211.

[11] S. Ejaz, Z. Iqbal, P. A. Shah, B. H. Bukhari, A. Ali, and F. Aadil, "Traffic Load Balancing Using Software Defined Networking (SDN) Controller as Virtualized Network Function," IEEE Access, vol. 7, pp. 46646–46658, 2019, doi: 10.1109/ACCESS.2019.2909356.

[12] R. Jain and S. Paul, "Network Virtualization and Software Defined Networking for Cloud Computing : A Survey," IEEE Commun. Mag., no. November, pp. 24–31, 2013.

[13] D. Kreutz, F. Ramos, M.V, P. Verissimo, Esteves, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking : A Comprehensive Survey," in Software-Defined Networking: A Comprehensive Survey, 2015, vol. 103, no. 1.

[14] K. Hikichi, T. Soumiya, and A. Yamada, "Dynamic Application Load Balancing in Distributed SDN Controller," in The 18th Asia-Pacific Network Operations and Management Symposium (APNOMS) 2016 Authorized, 2016, pp. 1–6.

[15] A. A. Alsulami, Q. A. Al-Haija, M. I. Thanoon, and Q. Mao, "Performance Evaluation of Dynamic Round Robin Algorithms for CPU Scheduling," in Conference Proceedings - IEEE SOUTHEASTCON, 2019, vol. 2019-April, doi: 10.1109/SoutheastCon42311.2019.9020439.

[16] L. Zhu, J. Cui, and G. Xiong, "Improved dynamic load balancing algorithm based on Least-Connection Scheduling," in Proceedings of 2018 IEEE 4th Information Technology and Mechatronics Engineering Conference, ITOEC 2018, 2018, no. Itoec, pp. 1858–1862, doi: 10.1109/ITOEC.2018.8740642.

[17] A. B. Prasetijo, E. D. Widianto, and E. T. Hidayatullah, "Performance Comparisons of Web Server Load Balancing Algorithms on HAProxy and Heartbeat," in 2016 3rd Int. Conf. on Information Tech., Computer, and Electrical Engineering (ICITACEE), 2016, pp. 393–396, doi: 10.1109/ICITACEE.2016.7892478.

[18] F. Chahlaoui and H. Dahmouni, "A Taxonomy of Load Balancing Mechanisms in Centralized and Distributed SDN Architectures," SN Comput. Sci., vol. 1, no. 5, pp. 1–16, 2020, doi: 10.1007/s42979-020-00288-8.

[19] M. Chiang, "SDN-based server clusters with dynamic load balancing and performance improvement," Cluster Comput., vol. 24, no. 1, pp. 537–558, 2021, doi: 10.1007/s10586-020-03135-w.

[20] R. C. Kurniawan, R. Tulloh, and I. D. Irawati, "VPLS on Software Defined Network Using ONOS Controller Based on Raspberry-Pi 3," in The 2021 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob), 2021, pp. 19–24.

[21] A. H. Eljack, A. H. M. Hassan, and H. H. Elamin, "Performance analysis of ONOS and floodlight SDN controllers based on TCP and UDP traffic," 2019, doi: 10.1109/ICCCEEE46830.2019.9071189.

Rohmat Tulloh received B.Eng. in Telecommunication Engineering in 2008 and M.Eng. Electrical Engineering in 2012 from Telkom University Indonesia. Currently, he works as a lecturer at Telkom University, Faculty of Applied Sciences, in the Diploma of Telecommunication Technology study program. His current research interests include Software Defined Networks, Named Data Networking, and Internet of Things.

Muhammad Iqbal received B.Eng. Telecommunications Engineering and M.Eng. Electrical Engineering from Telkom University Indonesia. Currently, he works as a lecturer at Telkom University, Faculty of Applied Sciences, in the Diploma of Telecommunication Technology study program. His current research interests include Telecommunications Networks and Network Security.

Mohammad Rifki Baihaqi received a diploma in telecommunications engineering from School of Applied Science, Telkom University Indonesia in 2020. Now he is pursuing a bachelor's degree in telecommunications engineering at the same University and currently researching software defined networks and the Internet of Things.

Aditya Bayu Aji Pamungkas received a diploma in telecommunications engineering from School of Applied Science Telkom University Indonesia in 2020. His research interests include Software Defined Network, Internet of Things, Big Data, and cloud computing.

ISSN 2210-142X

Philippine Banknote Counterfeit Detection through Domain Adaptive Deep Learning Model of Convolutional Neural Network

Marwin B. Alejo[1], Josh Lawrenz D. Villanueva[1], Marcus Philip E. Garchitorena[1], Shannen C. Reyes[1], John Michael B. Delos Reyes[1], Quinne Adonis L. Marasigan[1]

[1] Computer Engineering Department, National University, Manila, Philippines

E-mail address: mbalejo@national-u.edu.ph, villanuevajd@students.national-u.edu.ph, garchitorenampe@students.national-u.edu.ph, reyessc@students.national-u.edu.ph, delosreyesjmb@students.national-u.edu.ph, marasiganqal@students.national-u.edu.ph

ABSTRACT

Money counterfeiting is the illegal duplication of any currency for the use of deceiving any entity in exchange for a real-world value. Due to the advancements in computer vision in digital computing and the ill-effects of money counterfeiting, it had become one of the most prevalent issues in the fiscal system of any country that needs to be progressively solved. This paper investigated the use of ResNet18 through transfer learning for the task of Philippine banknote counterfeit detection. The used dataset of this study consisted of 391 counterfeited and 391 authentic images of 500 and 1000 Philippine peso bills. The trained model achieved a testing accuracy of 99.59%. Despite achieving a lower training accuracy, the trained model of this study achieved a validation accuracy, specificity, precision, sensitivity, and F1-score of 100% on live testing with the developed web-based money counterfeit detection system.

Keywords: Philippine money counterfeit, transfer learning, deep learning, resnet18, domain adaptive learning, convolutional neural network.

1. INTRODUCTION

Money counterfeiting is defined as the illegal duplication of currency to deceive recipients [1]. It is a prevalent threat and problem to any fiscal and economic system due to its ill effects. Counterfeited money deflates the value of a genuine currency, which inflates the prices of goods due to an unbalanced money distribution within the economy [2]. Due to the emergence of advanced technologies like computer vision and digital imaging in modern money counterfeiting, there is a need to design a solution in combatting it [1].

Among the many modern solutions in combatting money counterfeit is ultraviolet rays [3]. Production of modern banknotes includes anti-copying features like unique lines and decorations that glow when exposed to ultraviolet light. Although this method is simple and effective at small scale, it is tedious at large scale and heavily rely on manual labor and the subjective money counterfeit perception of human [4].

Image processing techniques are among the most studied and proposed solutions in combatting money counterfeiting [5]. One of these techniques is the Canny Edge Detection that processes a grayscale image to detect edges and suppress noise. In the paper of Ballado et al. [6], the suspected bill image is compared to an authentic reference image of a Philippines peso bill to check if the bill is genuine or not. Similarly, the study of Ankush Singh [7] improved the Canny Edge Detection for money counterfeit by adding segmentation and feature extraction. Although image processing techniques are effective for counterfeit money detection, they are computationally inefficient, tedious, and rely on the subjective perception of humans for feature extraction.

The disadvantages of using image processing techniques for counterfeit money detection turned most of the recent studies to convolutional neural networks (CNN) [6, 8]. CNN is a deep learning technique that allows a system or model to learn from a large dataset through its heuristic approach [9]. The study of Kamble et al. [10] uses CNN to model counterfeit money detection in Indian rupees. Their model achieved an accuracy of 86.5% by designing a handcrafted CNN architecture. The paper of Kumar et al. [11] achieved an accuracy of 96.6% in detecting counterfeited Indian currency notes by adapting a three-layered CNN architecture. However, their methods are inadaptable to several currency platforms due to the scarce of well-studied banknote datasets. Furthermore, Larsen-Freeman and Alejo et al. [12, 13] suggested using transfer learning or domain adaptive learning methods for classification modeling on a limited dataset.

Transfer learning is a domain adaptive deep learning technique that reuses a pre-trained model of one task to a new task with a different or small dataset [12, 14]. In the process, transfer learning uses the upper layers of the convolutional neural network as feature extractors without changing the weights except for the class values in the lower-most layers [13, 15].

Recently published banknote counterfeit detection and recognition studies utilized transfer learning as their main method. One of these is the paper of Pachon et al. [16] that utilizes the pre-trained (a)AlexNet, (b)SqueezeNet, (c)ResNet-18, (d)Inception-v3, and (e)customized CNN architecture for Colombian banknote counterfeit detection. Their transfer learning method achieved a validation and training accuracy of (a)99.86%, (b)99.88%, (c)100.00%, (d)99.97%, and (e)99.86% respectively. The transfer learning methods of Linkon et al [17] uses the pre-trained (a)ResNet-152-v2, (b)MobileNet, (c)NASNetMobile for Bangladeshi banknote counterfeit detection and achieved a testing/validation accuracy of (a)98.88%, (b)88.65%, and (c)100.00% accordingly. Rajendran and Anithaashi [18] transfer learning on pre-trained (a)AlexNet, (b)GoogLeNet, and (c)VGG-16 for Indian banknote counterfeit detection. Their method achieved a validation/testing accuracy of (a)95.00%, (b)88.00%, and (c)100.00%. On a similar note of the works in [18], the papers of Yildiz et al. [19] and Almisreb and Saleh [20] use the same pre-trained architectures on transfer learning for Bosnian Mark banknote counterfeit detection. The works of Yildiz et al. achieved a validation/testing accuracy of (a)99.68%, (b)97.36%, and©)99.88%, while the works of Almisreb and Saleh achieved a validation/testing accuracy of (a)95.24%, (b)88.65%, and (c)100.00%. Pham et al. [21] use the pre-trained (a)AlexNet and (b)ResNet-18 on transfer learning for the detection of the counterfeited Indian rupee, Korean won, and U.S. dollar bills. Their study achieved a validation/testing accuracy of (a)97.93% and (b)97.69%. Schulte et al. [22] use transfer learning with pre-trained Inception-v3 on several training configurations to detect counterfeited Euro bills. Their paper achieved a validation/testing accuracy of 100.00%. Although these studies show promising results, no studies use the transfer learning method on Philippine banknotes, and real-time counterfeited money detection and application. Moreover, there is no existing study or published dataset that contains Philippine peso bills.

This paper explored the convolutional neural network model of ResNet-18 for a counterfeited money detection system through a limited image-set and domain adaptive learning method in the context of the Philippine banknotes. Moreover, this paper selected ResNet-18 architecture due to its performance in the preliminary study of Villanueva et al. [21] over AlexNet and VGG16.

The main goal of this paper is to determine the adequacy of ResNet18 for Philippine banknote counterfeit detection. Specifically, this study aimed to (1) produce a banknote image-set consisting of five-hundred and one-thousand Philippine peso bills; (2) investigate CNN in the context of counterfeited Philippine banknote detection through ResNet-18 and transfer learning; (3) determine the training and validation accuracy of the CNN model for counterfeited Philippine banknote detection and; (4) determine the testing/validation accuracy, specificity, precision, sensitivity, and F1-score of the trained model on a real-time counterfeited Philippine banknote detection system or application, and (5) provide a fair comparison of the transfer learning method of this study in terms of testing/validation accuracy over the recently published transfer learning methods in [16 – 22].

The rest of this paper is structured as follows: Section 2 tackles the used procedures and methodology of the paper. Section 3 discusses the study results and the performance comparison of the proposed method towards other existing methods. Lastly, section 4 shows the conclusion of the study.

2. METHODOLOGY

This study's methodology consisted of four successive phases: (1) Input Dataset, (2) Pre-processing, (3) Training and Modeling, and (4) Classification, as shown in figure 1. The following subsections below briefly discuss each of these phases as used in this study.

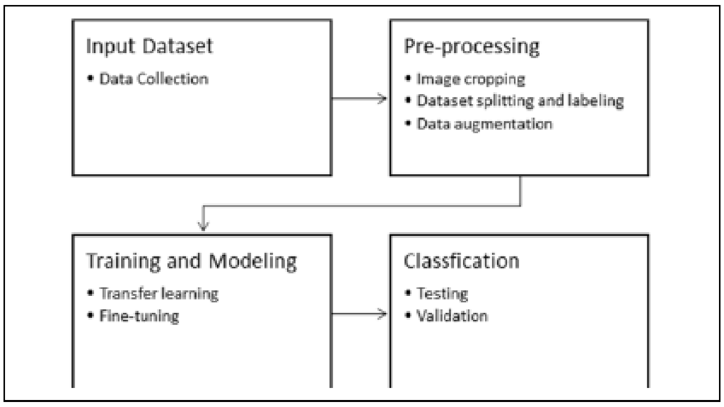
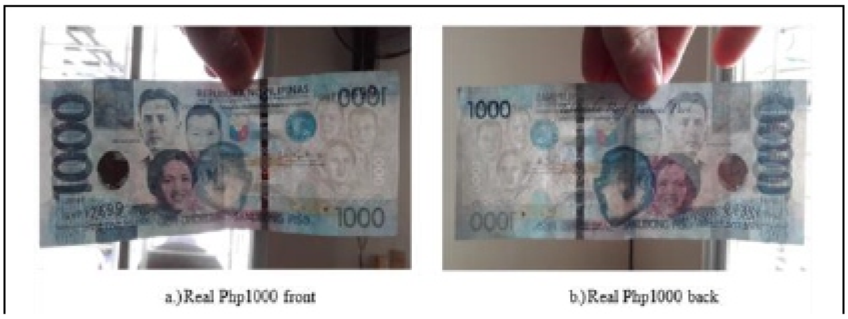


Figure 1. Deep learning pipeline of this study.

A. Dataset and image collection

Dataset and image collection is the first phase of the methodology of this study. The dataset of this study consisted of 782 raw images with 391 each are genuine and counterfeited five-hundred and one-thousand Philippine peso bills. The method of this paper manually extracted and collected these images as JPEG from the web. Figure 2 shows a sample raw image of both genuine and counterfeited Philippine peso bills of the used dataset. Additionally, figure 3 shows the security features of the current genuine Philippine peso bill.



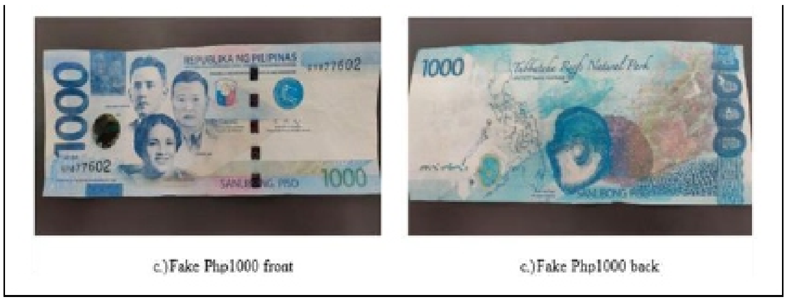


Figure 2. Deep learning pipeline of this study.

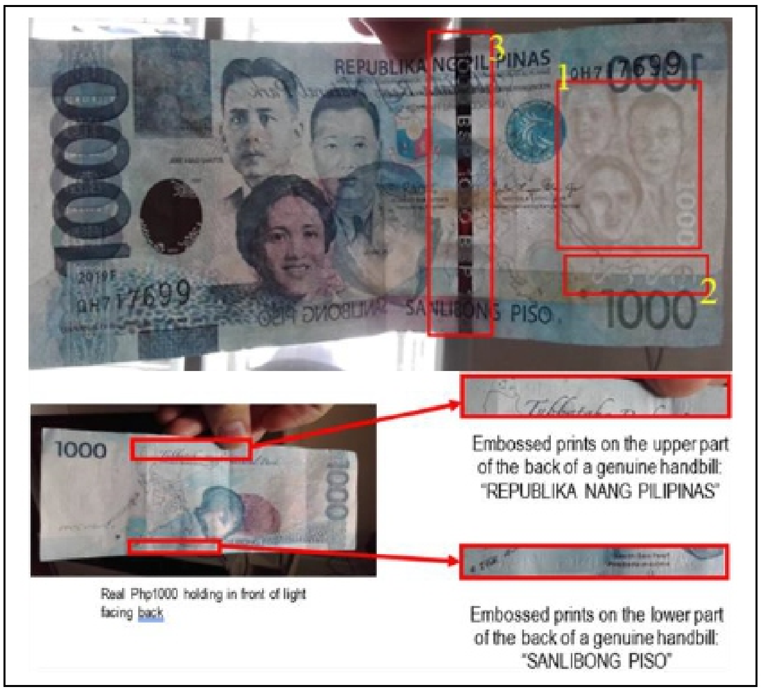

1
2
3
Embossed prints on the upper part of the back of a genuine handbill: "REPUBLIKA NANG PILIPINAS"
Real Php1000 holding in front of light facing back
Embossed prints on the lower part of the back of a genuine handbill: "SANLIBONG PISO"

Figure 3. Front (top: 1. Watermark, 2. Baybayin script, and 3. Security thread) and back (bottom) security features of a Philippine peso bill.

B. Image cropping

Image cropping of this study aimed to eliminate all the unwanted information or pixels of the images in the dataset.

The proponents of this study manually accomplished this process using Adobe Photoshop. The output of this process is a dataset of Philippine banknote images without unnecessary pixels, as shown in figure 4.

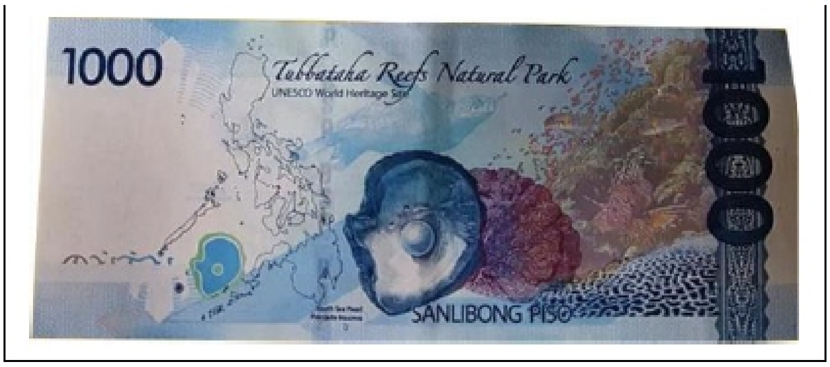


Figure 4. SamplePhilippine banknotedataafter image cropping.

C. Dataset splitting and labeling

This study randomly partitioned the used dataset by 80% training or 626 images, 10% testing or 156 images, and 10% validation or 156 images. Each partitioned image-set contains an equal number of authentic and counterfeited images of the five-hundred and one-thousand Philippine banknotes.

This study labeled the images in a CSV file which contains the filenames of the images and their respective classes. Counterfeited images are represented by zero (0), while authentic images are represented by one (1). This study uses Google Colab, Python programming language, and Microsoft Excel to accomplish this process.

D. Dataset augmentation

The purpose of this process in this study is to increase the statistics of the used training dataset and minimize the occurrence of overfitting on the trained models. This study resized the training dataset by 229×229 pixels, cropped in all sections by 224×224 pixels to satisfy the input requirements of ResNet-18 architecture, and randomly rotated by 1 to 10 degrees with default. This study performed these processes on Google Colab with Python programming language. Figure 5 shows the sample output of these processes.

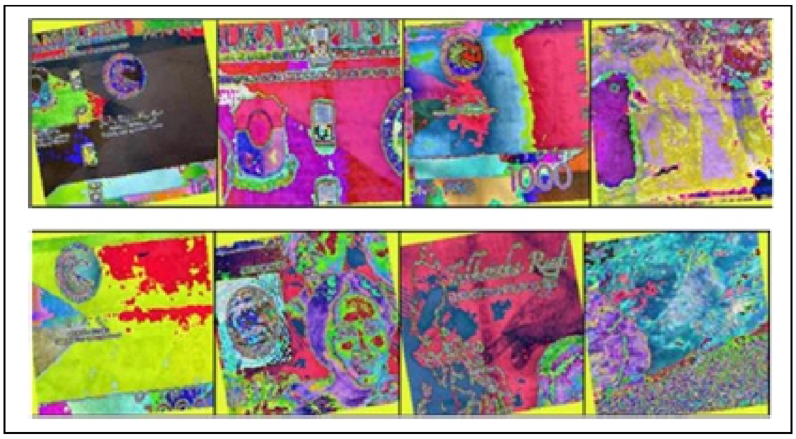


Figure 5. Sample output of dataset augmentationon the use dataset.

E. Training and Modeling

This study uses a pre-trained ResNet-18 model to develop a Philippine banknote counterfeit detection model through the domain adaptive approach of CNN or transfer learning. The proponents of this paper considered only the ResNet-18 architecture due to the initial findings of Villanueva et al. [23] in their study over AlexNet and VGG-16. This study accomplished the transfer learning modeling of this phase by fine-tuning the last three layers of ResNet-18 and replacing its final layer with a new sequential layer, as shown in Table I.

TABLE I. STRUCTURAL LAYERS OF RESNET-18

Child No.	Layer Type	Status
0	Convolutional	Frozen
1	Batch Normalization	Frozen
2	ReLU	Frozen
3	Max Pooling	Frozen

4	Sequential	Frozen
5	Sequential	Frozen
6	Sequential	Frozen
7	Sequential	Not Frozen
8	Adaptive Average Pooling	Not Frozen
9	Sequential	Not Frozen / Replaced

The standard ResNet18, as used in this paper, consisted of only nine layers. Through transfer learning, the frozen layers of ResNet-18 are layers 0 to 6, while layers 7 to 9 remain unfrozen to learn and extract the features of the preprocessed and augmented 626 training images. Furthermore, this study modified the last layer of the pre-trained ResNet-18 to recognize only the classifications of the used dataset, which are genuine and counterfeited Philippine bills. This study accomplished this training and modeling process on Google Colab-GPU with Python programming language, Tensorflow2.0/Keras, and the modeling configuration as shown in Table II.

TABLE II. TRAINING CONFIGURATION OF MONEY COUNTERFEIT CLASSIFIER TRAINING

Training Parameters	Configuration Settings
Batch size	8
Maximum epoch	15
Learning rate	0.001
Shuffle	True
Training function	Adam

F. Classification and the web-based application

The classification phase of this study consisted of two procedures: (1) validation and (2) testing. This phase aimed to determine the performance of the developed Philippine banknote counterfeit detection model using the 156 testing and validation images. This study determined the testing accuracy of the developed model coincidentally of the model training. In contrast, this study determined the validation accuracy using the output generated from the developed application with the developed model embedded in it and confusion matrix scores by Equations 1 to 5.

$$\text{Accuracy} = (TP + TN) / (P + N) \quad (1)$$

$$\text{Specificity} = TN / (FP + TN) \quad (2)$$

$$\text{Precision} = TP / (TP + FP) \quad (3)$$

$$\text{Sensitivity} = TP / (TP + FN) \quad (4)$$

$$F1 = 2TP / (2TP + FP + FN) \quad (5)$$

where,

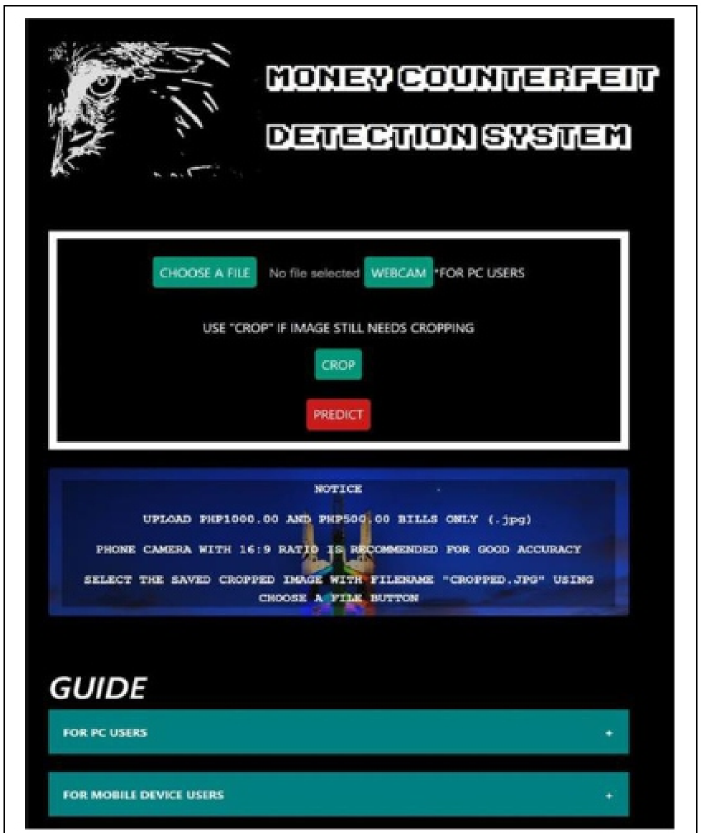
T.P. is True Positive,

TN is True Negative,

F.P. is False Positive,

F.N. is False Negative,

P is Positive, and N is Negative



USE "CROP" IF IMAGE STILL NEEDS CROPPING

NOTICE

UPLOAD PHP1000.00 AND PHP500.00 BILLS ONLY (.jpg)

PHONE CAMERA WITH 16:9 RATIO IS RECOMMENDED FOR GOOD ACCURACY

SELECT THE SAVED CROPPED IMAGE WITH FILENAME "CROPPED.JPG" USING CHOOSE A FILE BUTTON

Figure 6. Counterfeit Money Detection Application

The developed counterfeit money detection system is a web-based application that can run on any camera-equipped computing device. The proponents of this study developed the web application using HTML, Python programming language, and Flask. Testing using the web application may be done by uploading an image from the test dataset or an image captured by an embedded camera. Furthermore, this paper uses the validation dataset to assess the trained model on the developed system. Figure 6 shows the interface of the developed web application of this study.

3. RESULTS AND DISCUSSION

A. Training and testing result

This study took 11 minutes and 44 seconds to train and model the Philippine banknote counterfeit detection model using the train dataset. The trained model achieved a training accuracy of 100.00% and a testing accuracy of 99.59% . These statistics, despite of the currency of the used dataset, are lower than the best performing models of the papers of Pachon et al. [16], Linkon et al. [17], Rajendran and Anithaashri [18], Yildiz et al. [19], Almisreb and Saleh [20], Pham et al. [21], and Schulte et al. [22], as shown in Table III. Among of the factors that affect these results are the number of the used dataset or the training configurations.

TABLE III. COMPARATIVE TESTING ACCURACY RESULTS OF THIS STUDY OVER OTHER STUDIES' RESULTS

Transfer Learning Method		Currency	Statistics (%)
This study	ResNet-18	Philippine pese	99.59
Pachon et al. [16]	(a)AlexNet (b)SqueezeNet (c)ResNet18 (d)InceptionV3 (e)Custom	Colombian peso	(a)99.86 (b)99.88 (c)100.00 (d)99.97 (e)99.86
Linkon et al. [17]	(a)ResNet152v2 (b)MobileNet (c)NASNetMobile	Bangladeshi banknote	(a)98.88 (b)100.00 (c)97.77
Rajendran and Anithaashri [18]	(a)AlexNet (b)GoogLeNet (c)VGG-16	Indian Rupee	(a)95.00 (b)88.00 (c)100.00
Yildiz et al. [19]	(a)AlexNet (b)GoogLeNet (c)VGG-16	Bosnian mark	(a)99.68 (b)97.36 (c)99.88
Almisreb and Saleh [20]	(a)AlexNet (b)GoogLeNet (c)VGG-16	Bosnian mark	(a)95.24 (b)88.65 (c)100.00
Pham et al. [21]	(a)AlexNet (b)ResNet18	Indian rupee, Korean won, U.S. dollar	(a)97.93. (b)97.69
Schulte et al. [22]	Inception-v3	Euro banknotes	100.00

B. Validation result

This study validated the trained Philippine banknote counterfeit detection model using nine (9) random counterfeited bill images and 11 authentic bill images in the validation dataset and uploaded each onto the developed web application. The trained model achieved perfect true positive and negative scores of nine and 11, respectively. Extensionally, the trained model achieved a score of 100% in terms of validation accuracy, specificity, precision, sensitivity, and F1-score despite having a lower testing accuracy than the results of relevant literature. Table IV shows the classification results of the model through the developed web application over the actual label of the 20 extracted validation images and the computed performance metrics of the trained model using Equations 1 to 5. Moreover, figure 7 shows the samples of the actual validation results using the developed web application of this study.

TABLE IV. CONSOLIDATED VALIDATION RESULTS OF THE MODEL

Image No.	Actual Label	Predicted Label	Remarks
1	Fake	Fake	True
2	Real	Real	True
3	Real	Real	True
4	Fake	Fake	True
5	Fake	Fake	True
6	Fake	Fake	True

7	Real	Real	True
8	Real	Real	True
9	Real	Real	True
10	Fake	Fake	True
11	Fake	Fake	True
12	Fake	Fake	True
13	Real	Real	True
14	Real	Real	True
15	Fake	Fake	True
16	Real	Real	True
17	Real	Real	True
18	Real	Real	True
19	Real	Real	True
20	Fake	Fake	True
(TP)	11	**Validation Accuracy**	**100%**
(FP)	0	**Specificity**	**100%**
(TN)	9	**Precision**	**100%**
(FN)	0	**Sensitivity**	**100%**
(P)	11	**F1-score**	**100%**
(N)	9		

4. CONCLUSION

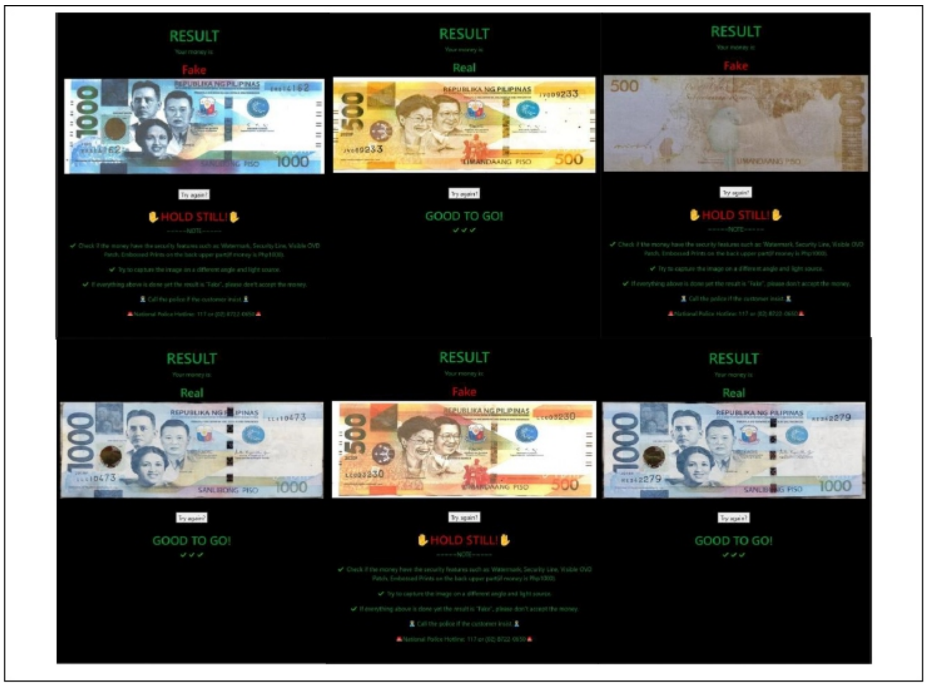


Figure 7. Samples of the actual testing results of the model on the developed web application.

Presented in this paper is a method for detecting counterfeited money in the context of Philippine peso bills. Due to the disadvantages of previous approaches and the absence of studies that utilize the domain adaptive learning approaches and well-studied money counterfeit datasets, this paper proposed a counterfeit banknote detection using transfer learning with a Philippine peso bill dataset.

This study investigated the use of transfer learning through ResNet18 for Philippine banknote counterfeit detection. It also determined that the trained model from ResNet18 achieved a training and testing accuracy of 100.00% and 99.59%, respectively. This study also determined that the trained model scored 100% on validation accuracy, specificity, precision, sensitivity, and F1-score while using a web application. Overall, ResNet18 is adequate for Philippine banknote counterfeit detection despite having a lower testing result than the other relevant studies.

For further studies, investigating the same method using the other pre-trained CNN architectures and transformer networks using the same Philippine banknotes dataset and alike should be explored. Additionally, the scarce Philippine money counterfeit dataset hinders other researchers from conducting further the same study; hence, there is a need to establish a dataset consisting of different banknote images of each country.

ACKNOWLEDGMENT

The authors would like to acknowledge the Engineering College of National University-Manila for its support and guidance towards completing this study. Furthermore, this paper does not have a conflict of interest nor is funded by any governing bodies or organizations.

REFERENCES

[1] P. Purpura, Security and Loss Prevention. Elsevier Inc., 2013. doi: 10.1016/C2010-0-69612-X.

[2] F. van der Horst, J. Snell, and J. Theeuwes, "Finding counterfeited banknotes: the roles of vision and touch," Cognitive Research: Principles and Implications, vol. 5, no. 1, pp. 1–14, Dec. 2020, doi: 10.1186/s41235-020-00236-3.

[3] M. A. Zamalloa Jara, C. Luízar Obregón, and C. Araujo Del Castillo, "Exploratory analysis for the identification of false banknotes using portable X-ray Fluorescence spectrometer," Applied Radiation and Isotopes, vol. 135, pp. 212–218, May 2018, doi: 10.1016/j.apradiso.2018.01.043.

[4] T. Agasti, G. Burand, P. Wade, and P. Chitra, "Fake currency detection using image processing," in IOP Conference Series: Materials Science and Engineering, Dec. 2017, vol. 263, no. 5. doi: 10.1088/1757-899X /263/5/052047.

[5] S. v. Viraktamath, K. Tallur, R. Bhadavankar, and Vidya, "Review on detection of fake currency using image processing techniques," in Proceedings - 5th International Conference on Intelligent Computing and Control Systems, ICICCS 2021, May 2021, pp. 865–870. doi: 10.1109/ICICCS51141.2021.9432111.

[6] A. H. Ballado et al., "Philippine currency paper bill counterfeit detection through image processing using Canny Edge Technology," Jan. 2016. doi: 10.1109/HNICEM.2015.7393184.

[7] Ankush Singh, "Detection of Fake Currency using Image Processing," International Journal of Engineering Research and, vol. V8, no. 12, Dec. 2019, doi: 10.17577/ijertv8is120143.

[8] M. Laavanya and V. Vijayaraghavan, "Real Time Fake Currency Note Detection using Deep Learning," International Journal of Engineering and Advanced Technology (IJEAT), no. 9, pp. 2249–8958, 2019, doi: 10.35940/ijeat.A1007.1291S52019.

[9] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," Insights into Imaging, vol. 9, no. 4, pp. 611–629, Aug. 2018, doi: 10.1007/s13244-018-0639-9.

[10] K. Kamble, A. Bhansali, P. Satalgaonkar, and S. Alagundgi, "Counterfeit Currency Detection using Deep Convolutional Neural Network," Dec. 2019. doi: 10.1109/PuneCon46936.2019.9105683.

[11] S. Naresh Kumar, G. Singal, S. Sirikonda, and R. Nethravathi, "A Novel Approach for Detection of Counterfeit Indian Currency Notes Using Deep Convolutional Neural Network", doi: 10.1088/1757-899X/ 981/2/022018.

[12] D. Larsen-Freeman, "Transfer of Learning Transformed," Language Learning, vol. 63, no. SUPPL. 1, pp. 107–129, Mar. 2013, doi: 10.1111/j.1467-9922.2012.00740.x.

[13] M. Alejo and C. P. G. Hate, "Unconstrained ear recognition through domain adaptive deep learning models of convolutional neural network," International Journal of Recent Technology and Engineering, vol. 8, no. 2, 2019, doi: 10.35940/ijrte.B2865.078219.

[14] M. Hussain, J. J. Bird, and D. R. Faria, "A study on CNN transfer learning for image classification," in Advances in Intelligent Systems and Computing, 2019, vol. 840, pp. 191–202. doi: 10.1007/978-3-319-97982-3_16.

[15] K. Weiss, T. M. Khoshgoftaar, and D. D. Wang, "A survey of transfer learning," Journal of Big Data, vol. 3, no. 1, pp. 1–40, Dec. 2016, doi: 10.1186/s40537-016-0043-6.

[16] C. G. Pachón, D. M. Ballesteros, and D. Renza, "Fake banknote recognition using deep learning," Applied Sciences (Switzerland), vol. 11, no. 3, pp. 1–20, Feb. 2021, doi: 10.3390/app11031281.

[17] A. H. Md. Linkon, Md. M. Labib, F. H. Bappy, S. Sarker, Marium-E-Jannat, and M. S. Islam, "Deep Learning Approach Combining Lightweight CNN Architecture with Transfer Learning: An Automatic Approach for the Detection and Recognition of Bangladeshi Banknotes," Proceedings of 2020 11th International Conference on Electrical and Computer Engineering, ICECE 2020, pp. 214–217, Dec. 2020, Accessed: Jul. 30, 2021. [Online]. Available: https://arxiv.org/abs/2101.05081v1

[18] P. S. Rajendran and Dr. T. P. Anithaashri, "CNN based framework for identifying the Indian currency denomination for physically challenged people," IOP Conference Series: Materials Science and Engineering, vol. 992, no. 1, p. 012016, Nov. 2020, doi: 10.1088/1757-899X/992/1/012016.

[19] A. Yildiz, A. A. Almisreb, Š. Dzakmic, N. M. Tahir, S. Turaev, and M. A. Saleh, "Banknotes counterfeit detection using deep transfer learning approach," International Journal of Advanced Trends in Computer Science and Engineering, vol. 9, no. 5, pp. 8115–8122, Sep. 2020, doi: 10.30534/ijatcse/2020/172952020.

[20] A. A. Almisreb and M. A. Saleh, "Transfer Learning Utilization for Banknote Recognition: a Comparative Study Based on Bosnian Currency," Southeast Europe Journal of Soft Computing, vol. 8, no. 1, Apr. 2019, doi: 10.21533/SCJOURNAL.V8I1.172.

[21] T. D. Pham, D. T. Nguyen, C. Park, and K. R. Park, "Deep Learning-Based Multinational Banknote Type and Fitness Classification with the Combined Images by Visible-Light Reflection and Infrared-Light Transmission Image Sensors," Sensors 2019, Vol. 19, Page 792, vol. 19, no. 4, p. 792, Feb. 2019, doi: 10.3390/S19040792.

[22] J. Schulte, D. Staps, and A. Lampe, "A feasibility study of deep neural networks for the recognition of banknotes regarding central bank requirements," Jul. 2019, Accessed: Jul. 30, 2021. [Online]. Available: https://arxiv.org/abs/1907.07890v2

[23] Josh Lawrenz D. Villanueva, Marcus Philip E. Garchitorena, Shannen C. Reyes, John Michael B. Delos Reyes, and Quinne Adonis L. Marasigan, "Money Counterfeit Detection System using Convolutional Neural Network and Transfer Learning," Manila, Philippines, 2021.

Instructions for Authors

Essentials for Publishing in this Journal

1 Submitted articles should not have been previously published or be currently under consideration for publication elsewhere.

2 Conference papers may only be submitted if the paper has been completely re-written (taken to mean more than 50%) and the author has cleared any necessary permission with the copyright owner if it has been previously copyrighted.

3 All our articles are refereed through a double-blind process.

4 All authors must declare they have read and agreed to the content of the submitted article and must sign a declaration correspond to the originality of the article.

Submission Process

All articles for this journal must be submitted using our online submissions system. http://enrichedpub.com/ . Please use the Submit Your Article link in the Author Service area.

Manuscript Guidelines

The instructions to authors about the article preparation for publication in the Manuscripts are submitted online, through the e-Ur (Electronic editing) system, developed by **Enriched Publications Pvt. Ltd**. The article should contain the abstract with keywords, introduction, body, conclusion, references and the summary in English language (without heading and subheading enumeration). The article length should not exceed 16 pages of A4 paper format.

Title

The title should be informative. It is in both Journal's and author's best interest to use terms suitable. For indexing and word search. If there are no such terms in the title, the author is strongly advised to add a subtitle. The title should be given in English as well. The titles precede the abstract and the summary in an appropriate language.

Letterhead Title

The letterhead title is given at a top of each page for easier identification of article copies in an Electronic form in particular. It contains the author's surname and first name initial .article title, journal title and collation (year, volume, and issue, first and last page). The journal and article titles can be given in a shortened form.

Author's Name

Full name(s) of author(s) should be used. It is advisable to give the middle initial. Names are given in their original form.

Contact Details

The postal address or the e-mail address of the author (usually of the first one if there are more Authors) is given in the footnote at the bottom of the first page.

Type of Articles

Classification of articles is a duty of the editorial staff and is of special importance. Referees and the members of the editorial staff, or section editors, can propose a category, but the editor-in-chief has the sole responsibility for their classification. Journal articles are classified as follows:

Scientific articles:

1. Original scientific paper (giving the previously unpublished results of the author's own research based on management methods).
2. Survey paper (giving an original, detailed and critical view of a research problem or an area to which the author has made a contribution visible through his self-citation);
3. Short or preliminary communication (original management paper of full format but of a smaller extent or of a preliminary character);
4. Scientific critique or forum (discussion on a particular scientific topic, based exclusively on management argumentation) and commentaries. Exceptionally, in particular areas, a scientific paper in the Journal can be in a form of a monograph or a critical edition of scientific data (historical, archival, lexicographic, bibliographic, data survey, etc.) which were unknown or hardly accessible for scientific research.

Professional articles:

1. Professional paper (contribution offering experience useful for improvement of professional practice but not necessarily based on scientific methods);
2. Informative contribution (editorial, commentary, etc.);
3. Review (of a book, software, case study, scientific event, etc.)

Language

The article should be in English. The grammar and style of the article should be of good quality. The systematized text should be without abbreviations (except standard ones). All measurements must be in SI units. The sequence of formulae is denoted in Arabic numerals in parentheses on the right-hand side.

Abstract and Summary

An abstract is a concise informative presentation of the article content for fast and accurate Evaluation of its relevance. It is both in the Editorial Office's and the author's best interest for an abstract to contain terms often used for indexing and article search. The abstract describes the purpose of the study and the methods, outlines the findings and state the conclusions. A 100- to 250-Word abstract should be placed between the title and the keywords with the body text to follow. Besides an abstract are advised to have a summary in English, at the end of the article, after the Reference list. The summary should be structured and long up to 1/10 of the article length (it is more extensive than the abstract).

Keywords

Keywords are terms or phrases showing adequately the article content for indexing and search purposes. They should be allocated heaving in mind widely accepted international sources (index, dictionary or thesaurus), such as the Web of Science keyword list for science in general. The higher their usage frequency is the better. Up to 10 keywords immediately follow the abstract and the summary, in respective languages.

Acknowledgements

The name and the number of the project or programmed within which the article was realized is given in a separate note at the bottom of the first page together with the name of the institution which financially supported the project or programmed.

Tables and Illustrations

All the captions should be in the original language as well as in English, together with the texts in illustrations if possible. Tables are typed in the same style as the text and are denoted by numerals at the top. Photographs and drawings, placed appropriately in the text, should be clear, precise and suitable for reproduction. Drawings should be created in Word or Corel.

Citation in the Text

Citation in the text must be uniform. When citing references in the text, use the reference number set in square brackets from the Reference list at the end of the article.

Footnotes

Footnotes are given at the bottom of the page with the text they refer to. They can contain less relevant details, additional explanations or used sources (e.g. scientific material, manuals). They cannot replace the cited literature.
The article should be accompanied with a cover letter with the information about the author(s): surname, middle initial, first name, and citizen personal number, rank, title, e-mail address, and affiliation address, home address including municipality, phone number in the office and at home (or a mobile phone number). The cover letter should state the type of the article and tell which illustrations are original and which are not.

<u>**Address of the Editorial Office:**</u>

Enriched Publications Pvt. Ltd.
S-9,IInd FLOOR, MLU POCKET,
MANISH ABHINAV PLAZA-II, ABOVE FEDERAL BANK,
PLOT NO-5, SECTOR -5, DWARKA, NEW DELHI, INDIA-110075,
PHONE: - + (91)-(11)-45525005