ISSN 2521-0122 (Online)
ISSN 2519-7991 (Print)
Vol (8): Issue (2)
April - June 2024

Lahore Garrison University Research

LGURJCSIT

Journal of Computer Scienceand Information Technology

www.lgurjcsit.edu.pk

Lahore Garrison University,
Lahore, Pakistan

LGURJCSIT ISSN: 2519-7991(Print), 2521-0122 (Online)

SCOPE OF THE JOURNAL

The LGURJCSIT is an innovative forum for researchers, scientists and engineers in all domains of computer science and technology to publish high quality, refereed papers. The journal offers articles, survey and review from experts in the field, enhancing insight and understanding of the current trends and state of the art modern technology.

Coverage of the journal includes algorithm and computational complexity, distributed and grid computing, computer architecture and high performance, data communication and networks, pattern recognition and image processing, artificial intelligence, cloud computing, VHDL along with emerging domains like Quantum Computing, IoT, Data Sciences, Cognitive Sciences, Vehicular Automation. Subjective regime is not limited to aforementioned areas, Journal policy is to welcome emerging research trends in the general domain of computer science and technology.

SUBMISSION OF ARTICLES

We invite articles with high quality research for publication in all areas of engineering, science and technology. All the manuscripts submitted for publication are first peer reviewed to make sure they are original, relevant and readable. Manuscripts should be submitted through online journal submission system (https://lgurjcsit.lgu.edu.pk/index-.php/lgurjcsit/user/register).

To submit manuscripts Online with attach file is strongly encouraged, provided that the text, tables, and figures are included in a single Microsoft word/pdf file. Submission guidelines along with official format is available on the following link:

https://lgurjcsit.lgu.edu.pk/index.php/lgurjcsit

Contact: For all inquiries, regarding call for papers, submission of research articles and correspondence, kindly contact at the folloing address:
LGURJCSIT, Sector C, DHA Phase-VI, Lahore, Pakistan.
Phone: +92-042-37181823
Email: lgurjcsit@lgu.edu.pk

Published by: Faculty of Computer Sciences, Lahore Garrison University, Lahore, Pakistan

Hassan et al. LGURJCSIT 2024

ISSN: 2521-0122 (Online)
ISSN: 2519-7991 (Print)

LGU Research Journal of Computer Science & IT

doi: 10.54692/lgurjcsit.2024.082558

Vol (8): Issue (2), April – June 2024

Use of Code Refactoring Transformation in Software Advancement

Taimoor Hassan[1*], Abrar Ahmed[2], Mehmood Anwar[3], Shahzaib Afzal[4], Muhammad Mohsan[5], Muhammad Basit Ali Gilani[6]

[1]Department of Software Engineering, University of Central Punjab, Lahore, Pakistan.
[2,3]Department of Software Engineering, The University of Lahore, Lahore, Pakistan.
[4,5]Riphah Institute of Computing & Applied Sciences, Riphah International University, Islamabad, Pakistan.
[6]Department of Computer Science, University of Central Punjab, Lahore, Pakistan.

Email: taimoor.hassan01@ucp.edu.pk

ABSTRACT:

In this paper, the refactoring of Object-oriented code affects the software quality in some ways. Reengineering, which means refactoring, is nearly always a good idea and is usually cheap; it optimizes the structure where the behavior does not appear to the user to have changed. This advantage makes it a subject of much research when as a tool for measuring the quality of software. It is proposed that software requirements be addressed at this stage since rather specific requirements are associated with higher-quality work. Refactoring is cost-effective when done during the development phase, this is because requirements that may need refactoring are identified early. It is necessary to note that our paper consists of five research questions and their answers. Some of the best practices followed in case of requirements management include Collection of requirements, choosing relevant requirements from the pool, further subversion of the selected requirements, placing the requirements in order of priority, and numbering and documentation of the requirements. This methodology is then implemented and results are attained at each phase of the study in the case of the Hotel Management System.

KEYWORDS: Refactoring, Transformation, Primary Studies, Systematic Mapping Study, Structured Query Language.

1. INTRODUCTION

In the eyes of the spectators, the plot of the topic indicates the significant role of code refactoring in quality assurance, growth in agility, and ease in maintenance and sustainability. With the expansions of software systems complexity, it is expected that the implementation of code refactoring as a regular practice also increases, which would improve speed and the spirit of innovation.

Refactoring as a process control concept has been a component of software engineering for decades. Originally, it was introduced by Martin Fowler and it refers to the process of refactoring the code that already exists to enhance its internal organization. This has become important especially in agile, which is characterized by frequent cycles and improvement. Moreover, Code refactoring is the process by which an existing code is changed for the better in terms of quality standards but the external interface remains the same. It is a way to decrease the technical debt, make updates less complex, and guarantee the software's further maintainability. Essentially, the main reason to refactor is the dynamic nature of the requirements and technologies in use. Another disadvantage of using software systems is that as the complexity

of the systems increases the management also becomes a herculean task. Refactoring facilitates the enhancement of the source code system in various ways such as; enhancing flexibility, reducing the number of defects, and increasing the capacity of production among developers. Furthermore, the problems of refactoring are usually long and can make creating features take significantly longer in the short term. Modifications to the code regarding its structure might create new problems in the program. It is relatively easier for stakeholders to get a handle on why the refactoring has to be done as distinct from seeing feature addition benefits.

Furthermore, this research aims to execute inclusive methodical mapping research on previous experimental studies to evaluate the consequence of code refactoring activities for object-oriented to measure the quality attributes of the software.

Software maintenance is one of the exclusive and intense struggles for activities of development of software [1] [2]. Software maintenance higher in cost shows the bad design quality of software [3]. In the maintenance phase of software, several code modules are by mistake presented.

the developers [4]. These modules of code or parts of code show the bad quality of software at a later time due to several code changes [5] [6]. In this research, the author identified 13,283 articles relevant to the study from six digital libraries. Later, they scrutinized research that is based on several viewpoints, which include abstract, title, and full text, using searching manually and by reference. Further, they analyze the quality of the study by selecting 142 Primary Studies (PSs). They classify these primary studies on behalf of the refactoring activities influences. The author also discusses the numerous refactoring activities such as quality measures of software, statistical practices, datasets, and quality attributes, working using nominated PSs. Further, the author extracts the testified software tools that forecast or evaluate the influence of refactoring actions. They also provide current and previous studies on the outcome of code refactoring methods of object-oriented to enhance the attributes of the software quality.

In conclusion, they organize the distributed and inconsistent discoveries to determine a list of exposed problems and challenges that are required to be addressed in the future. The research problem which is addressed in this research is that author extracts a clear picture by constructing systematic research that analyzes and reports the conclusions on the association amid refactoring methods and the enhancement of quality for object-oriented applications or software. This paper is divided into different sections, section II explains the related work, III section elaborates on the research methods, section IV defines the problem statement, V section explains the proposed solution, section VI shows the results, VII section explores the discussion, and the last, summarize the overall paper in conclusion section VIII.

2. LITERATURE REVIEW

Code refactoring impacts the software quality, performance, and maintainability deeply, adding the crucial components of a successful outcome. It has a multitude of benefits such as better readability, increase in performance, and flexibility but at the same time, it has its shortcomings, i.e. introducing new bugs from other resource consumptions. Hence, we need to come up with a well-designed plan, communicate with the team, and test the refactoring result before we decide to move it to the main codebase [7].

Furthermore, the objective for recovery-focused is just to enhance the level of technology as well as the competitiveness of developers. Over the previous two decades, several longitudinal experiments were performed to examine the effect of recompilation operations on technology efficiency. The objective of this project [8] would be to conduct an expected to be stable vulnerability assessment of current academic research on the role of expression application regression testing practices on application performance parameters. According to the findings, scientific work observed a greater detrimental effect of recompilation with application efficiency whereas research was published across sectors. Except for consistency, difficulty, succession, mistake propensity, and thermal dissipation, both consistency parameters improved rather than degraded because of rewriting practices. Besides that, many performance indicators investigated throughout research papers provide dynamic influence on individual requirements engineering operations, meaning whether modification often does not increase certain performance characteristics. The whole research identifies the list of available problems that need to be examined more deeply, such as a scarcity of technical verification, the restricted scope with provocation, and minimal device

funding, just to name a few. Moreover, the study consists of a comprehensive research methodology analysis that describes, evaluates, and summarizes its previous research evidence on the effect of rewriting practices to graphical fidelity, only to evaluate its latest developments while identifying new research directions. To begin, they presented an overview of both the programmer requirements engineering system. They then spoke about the visualization study methodology that was used to perform that applicable study contributes to certain phenomena under investigation are searched using two techniques [9].

On the other hand, they started by searching certain internet sources: Embedded Celia de, Cochrane Library, Walther, Research gate, PubMed, including Greene. Then, in all the five most important famed publications, a physical review was undertaken. We have used comparison-losing games to improve the accuracy of their quest but reduce the chance of losing important information. The distribution of PSs by presentation month revealed that assessing the impact of recompilation practices for system development has become a hot topic throughout academia. Those findings suggest also that the Transfer System and extraction Process, including Extraction Group requirements engineering practices, has received more attention from investigators than many of the other requirements engineering operations [10].

Meanwhile, scholars have contributed less time and effort towards designing machine learning that can forecast their advantages from programmed regression testing until it is implemented. The impact of recompilation practices on user-friendliness was measured and found using a variety of various shaped samples applied across many instrument computer languages. Since only a few other Rag used computational methods to investigate the importance of expected returns, they used a cast-a-ballot method to combine their results more about the impact of recompilation with computer efficiency. This same effect from cumulative regression testing practices for various system integration indicators provides contradictory data, indicating that computer modification often does not boost certain application performance objectives. Likewise, various configuration operations have an opposing influence on global device consistency parameters. Eventually, they discovered how analyzing the impact of recompilation over graphical fidelity involves several measurements including the order in which provocation is applied, its user experience metric, and so forth. Every element's variance may influence this report's ultimate result. As a result, when planning the analysis, that element ought to be seriously evaluated [11].

However, the modification would be a popular method of improving software efficiency. The effect on something like a wide variety of computer luxury homes, such as reproducibility, ease of maintenance, and efficiency, was already thoroughly investigated but quantified. They evaluate the effect of some of the more standard programming change management guidelines on privacy laws inside this paper, utilizing protection indicators that can measure safety first from the perspective of future knowledge transfer [12]. Those statistics were estimated with each SQL statement that used a fixed methodology we created to examine extracted Separator dynamically. They used their Java programming language analyzer on several applications that have been remediated by the rules. Which pseudo-code systems' updated indicators data suggest that perhaps the programming updates used to have a meaningful impact on data protection.

Integration testing, [13] the method of enhancing the current program's functionality by modifying its internal configuration whilst altering any dynamic behavior, is often used to increase computer consistency through optimizing semantic layout, usability, and error reduction. This relates to the reorganization of the admin tool (qualities and both techniques) within other categories. Modifying would be a common practice of project implementation. For certain web applications, such as Requirements Elicitation, patching is considered the most important aspect of both the agile methodology [25]. Adjustments may be needed to maintain the computer's reliability. Responding to rising standards, technology keeps evolving throughout its lifespan. The code becomes dysfunctional because of such a development. Reduced technology could have highly amplified transformations of both its architecture. As a result, its expense of commitment to repairs would have been greater. As a result, there may be a critical requirement for reliable applications. Only at the right points, which reflect that greater degree of project management, unacceptable uncertainty may occur. At such a reduced scale, namely the grade level, unnecessary volatility will occur. The fundamental feature regarding product design through

out Expression applications was its category. Category consistency refers to how easily a computer object may develop whilst maintaining that structure, whereas technology stability refers to how adaptable code would be to changing requirements or environments even when maintaining that structure. Computer structures that have been well ought to be able to adapt while requiring significant appropriate adjustments [14].

Since structural modifications become complex to maintain, code should always be built keeping enhanced structural consistency considered in mind. Since groups are indeed an important part of the development infrastructure, programmers must continue to put them in as soon as reasonably practicable. Some criteria for the category of design stabilization were identified. Modifying technology requires a considerable amount of funds. People argue which source code becomes important because it increases programmer efficiency throughout. That requirement engineering process, on the other hand, seems to have a unique effect on user-friendliness. Computer programmers normally create for specific architectural purposes that might or might not be mutually exclusive [15].

As a result, programmers must aim to create robust applications in terms of improving the development process for key calligraphers. There have been no rules for computer programmers that the recompilation techniques to be used in time to retain code security. As a result, they must investigate the impact of recompilation approaches against computer stabilization but classify such source code approaches according to that observable impact upon system instability. Its category becomes aimed at helping computer developers through selecting suitable configuration strategies for maintaining consistent code [12]. The purpose of this article would be to evaluate the effect of recompilation upon category but structure stabilization, as well as to suggest a grading system among rewriting approaches that depend highly upon category but structure stabilization. It will also inform computer programming on whose recompilation techniques are being used to preserve code stability [16].

That benefit of responsive design in terms of readability would be much less apparent inside the brief period that can sometimes be believed, based on the circumstances. In this analysis, they examine how "spotless programming" source code improves programmer efficiency inside a technology platform including data structures inside an educational factory [16]. Significant rises or declines in intelligibility were found, demonstrating that rapid rises in intelligibility are not necessarily apparent. According to their findings, adapting software can lead together in a usability hit within a brief period if indeed the mechanical reinforcement diverges according to what programmers were becoming accustomed to. Huge, lengthy software applications sometimes tackle the issue of technological complexity, which is exacerbated by implementation methods that end in difficult-to-maintain coding [17]. The technological liability may be justified; furthermore, unless the liability also is not forgiven, the cost is always charged, who serves the purpose with extra resources duration by designers either neither learn nor modify complicated programming. Thus, according to Bennett, "that budget deficit of just an undistributed deployment will bring whole organizations to a halt. "Modification is indeed a system software reorganization strategy that could be used to pay back intellectual liability. Reuse is said to boost programmer morale, establish principles architecture, render applications easy to learn, and even assist programmers with finding glitches.

Matthews also makes the case that the composition of software affects high flexibility, but they encourage programmers to stay aware of certain software but write native apps. Even so, it has already been acknowledged that only some research has quantified its reported configuration benefits, significantly about production efficiency. When they evaluate its alleged configuration benefits from just one arm as well as the limited scientific data from the other, there is a lack of information [8]. If additional innovations were applied to both frameworks, several results show how modification will result in improved efficiency for many days, and therefore programmers who do also get a stronger copy of the concept [21]. That ongoing creation of the micro data center also led to significant corporate expansion, including software developers spread over several regions. The program's exponential expansion, and the total number of employees upon this, also resulted in technological borrowing.

3. RESEARCH METHODS

In the research method step, they followed a multi-stage shortlisting process for the selection

of 142 published research articles by December 2017. Proposed research questions are answered by different aspects using the classification of selected research articles. The vote-counting method is applied to syndicate the results and report the analysis in PSs. The proposed research questions for this study are the following:

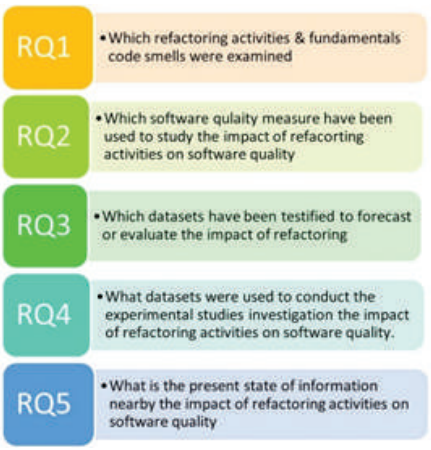

Figure 1: Research Questions

Several electronic databases are used in this study, some of these are the following.

Firstly, they draw major search terms from research questions, later in this research article the author obtains synonyms, abbreviations list, alternative spells, and core research terms. In the final section, they combine the core search terms using Boolean operators. For validation of the initial search, the string author selects 20 research articles from databases that contain the most relevant research articles. The search string is constructed which needs a search for titles, abstracts, and keywords of research articles. For this purpose, they selected 20 research articles, and only 11 out of 20 research articles were selected by their initial string search. Different types of searches are used by authors. These are the following:

3.1. Automatic search

In this step, they execute search strings of the articles in six electronic databases. In the first automation search, 12,996 search results were found. In the second phase, they found 287 more articles, where the total number of search results was updated to 13,283. These searches were organized and managed using Zotero.

3.2. Manual search

For completeness of the articles list, they perform a manual search analysis. This search results in 174 total articles in two phases, 161 and 13 search results, respectively. Manual search results were recorded using MS Excel spreadsheets after the exclusion of unrelated articles. After this search, both results were combined to eliminate the duplicate search results.

3.3. Reference Verification

They independently perform references using the manual examination of the reference list for the 129 applicable gained research articles.

The author set the Inclusion and exclusion standards which were useful for searching the articles that were not significant or related to core work concerning research questions. After phase 1, both authors independently verify this criterion using a set of 90 research articles arbitrarily from automated examination results. After this, they applied Cohen Kappa static [18] on another set of 90 research articles randomly. The author uses 7 stages to screen the articles, at stage 1, the author performs automatic and manual searches from six electronic databases. Search results were captured automatically using Zoterol. In stage 2, the author performs data cleaning of automatic exploration outcomes and separates unrelated entries. At stage 3, the author merges the outcomes of in cooperation automatic and manual explorations in MS Excel spreadsheets.

At stage 4th, the authors rejected research articles based on the article's titles. This process selects 1374 and 52 studies in Phase 1 and Phase 2 individually for further processing in step 5th. In the 5th stage, they screened both authors autonomously on behalf of the abstracts. In stage 6th both authors studied the complete manuscript of 247 and 37 research articles selected in both phases correspondingly and performed an inclusion and exclusion process on articles.

In the 7th stage, they verify references of selected articles. Results from this step were additionally distinguished on the behalf of abstracts and full texts. Finally, 142 primary studies were involved in the concluding list, and they extracted the relevant data from these selected research articles. Information obtained from a reading of these articles was documented in a data abstraction form. Items that are extracted from the individual primary study are the following. Full references from the PSs with title, author, publication title, and year. For study classification, the author uses 4 phase approach for the classification of selected PS into 7 facets which are research involvement

technique, study framework, refactoring activities, quality measures of the software, search method, focus, and datasets.

4. PROBLEM STATEMENT

Numerous software initiatives advance daily in our sector of the software business. Given the size of the investment in our nation, that is advantageous for the industry. These days, practically every firm uses a distinct software system to carry out daily office operations and activities. Thus, it is safe to state that not using and installing software systems in any kind of organization or business leads to a great deal of issues. The nation's largest investor and contributor to the growth of our businesses is the software sector.

As time goes on, the organization encounters a growing number of software requirements-related issues when developing software systems. It can be very challenging to grasp requirements when they are communicated by clients in their natural language. Clients without literacy are the primary problem with this. As a result, individuals fail to clearly state and articulate their needs about the suggested system. These clients can be found all over the place [19] to produce software from software companies. Because of the client's ambiguous criteria, the suggested program will not be excellent in every way in the end. The software development process is prone to numerous fluctuations due to the significant communication and conversation gaps that exist between the project team and the client. The client's lack of literacy and improper clearance of the requirements are the primary causes of this.

5. PROPOSED SOLUTION

It is very important to exemplify the applicability of the proposed approach as well as to prove its efficiency for practical usage, so we reveal the application and evaluation of the proposed approach based on the case study of the Hotel Management System. That is why in each stage of the development when applying our approach, we obtain tangible outcomes that describe the effectiveness of requirements-driven strategies with refactoring. The given case can be considered as a pilot for the demonstration of the results that can be achieved by implementing suggested strategies as the improvement of software quality and enhancement of development productivity are evident. Moreover, the suggested remedy for a particular issue is discussed in section IV of this section. Figure 2 below provides an overview of the suggested solution, detailing how we will create a software system that is error-free and intelligible to both the customer and the company's end user.

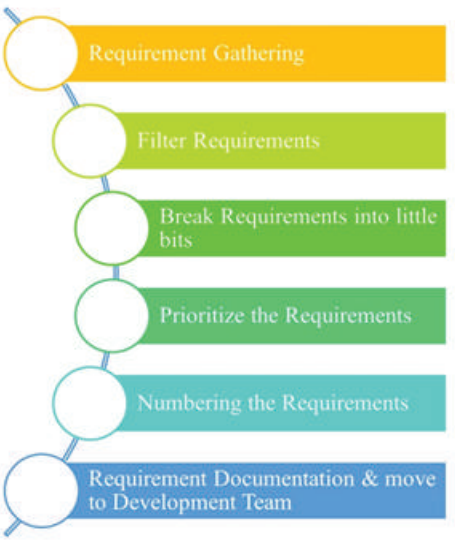

Figure 2: Used Approach

5.1. Requirement Gathering

Gathering the requirements should be the first step of the suggested solution that has been provided. requirements gathered using various methods or sources, as section I explains. Since we are aware of the client's lack of education and skill, we study the client's business system, staff members, and most importantly the end user who will be using the suggested system firsthand. We also meet with other staff members and company stakeholders to precisely determine what the proposed system's requirements are.

5.2. Filter Requirements

Filter the requirements once they have been gathered. Requirements are separated into functional and non-functional forms, sorted sequentially, and filtered to remove false, unclear, and unnecessary information.

5.3. Break Requirements

Once the requirements have been filtered, break them down into smaller, more manageable pieces. The software team found the requirement formulation and analysis to be quite simple to comprehend.

5.4. Prioritize Requirements

Sort the needs according to functionality at this stage. Requirements that are basic and essential

are given priority over non-essential requirements. Prioritize the development of the software's functionality. The early completion of priority needs results in increased client satisfaction.

4.5. Numbering the Requirements

The allocation of the numbering to slightly prioritize the criteria is the most crucial phase in this strategy. Typically, numerals like 1, 2, 3, 5, 6... n, are used for numbering. This will make it simple for us to compare the requirements to their quantity and begin working on them in the form of development.

4.6. Requirement Documentation

Ultimately, all of the criteria will be documented during this phase. Written requirements will be very helpful in understanding the system, tracking needs, all functionality, and how it is used, in the future (should the requirements change). The document will then be forwarded to the development team so that the software system may be developed. All processing needs to be recorded since, upon project completion, the client receives the recorded file, which explains each step, its use, and its potential future use.

6. RESULTS

To determine the pertinent outcomes, we apply the suggested solution to the Hotel Management System case study.

6.1. Requirement Gathering

Initially, we gather all 12 requirements for this case study, which are listed in table 1 below. (NADRA: National Database and Registration Authority).

Table 1: HMS Requirement Gathering

Hotel Management System	
Requirements	User login and password
	Book room on ID card
	Check authentication with NADRA
	Payment through cash/credit card
	Bank validation
	External reservation guest
	Check-out/ Check in
	Time scheduling
	Room cleaning staff
	Booking confirmation message/email
	Automated generated slip
	Food facility record

6.2. Filter Requirements

There are a total of 12 requirements, however, after sorting through them, we remove any unnecessary information and separate them into the functional and non-functional requirements for the hotel management system, as shown in table 2 below.

Table 2: HMS Functional/Non-Functional

Functional Requirements	Non-Functional Requirements
User login and password	External reservation guest
Book room on ID card	Booking confirmation message/email
Check authentication with NADRA	Room cleaning staff
Payment through cash/credit card	Time scheduling
Bank validation	Food facility record
Check-out/ Check in	-----
Automated generated slip	-----

6.3. Break Requirements

In this stage, we dissect each criterion into the smaller components listed in Table 2 above. But functional requirements are the main emphasis of our major.

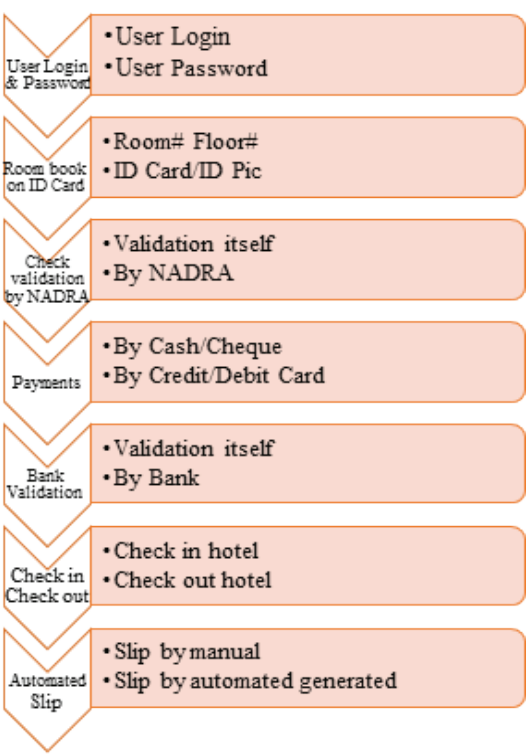

Figure 3: Break HMS Requirements

6.4. Prioritize Requirements

We will be ranking the seven functional needs of the HMS (Hotel Management System) shown in Table 2 above at this step of the suggested process.

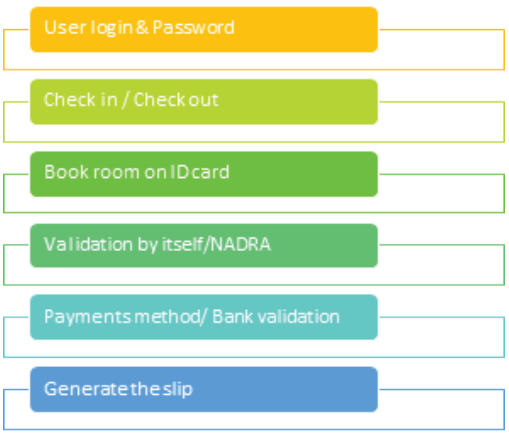

Figure 4: Prioritize Requirements

6.5. Numbering Requirements

The functional needs of the Hotel Management System, which are listed in Table 3 below, are being numbered in the virtually last stages of the proposed system.

Table 3: Numbering Requirements

Number Allocation	Requirements
1	User Login/Password
2	Check in/ Check out
3	Book the room
4	Validation by NADRA
5	Payments/Bank checking
6	Generate Slip

6.6. Requirement Documentation

At this stage, we will ultimately record every need listed in Table 3 and hand the project over to the Hotel Management software system development team.

7. DISCUSSION

This research paper assesses the process of code refactoring and its significance in improving the quality of software. Concerning code refactoring which is understood to be a constructive and effective approach to enhance program codes and performance, authors have not underestimated that it has variable effectiveness [22]. Therefore, the paper recommends that this approach be adjusted because the selection of the refactoring technique should relate to the nature and clarity of the requirements of the software. This way, it is seen that the majority of the issues can be handled in the early phases of the development life-cycle, provided that only the necessary and well-defined functional requirements are targeted, thus making the process of refactoring more effective [23].

The paper's section 4 considers the problems found in software companies today, especially the issue of code refactoring. Furthermore, in the foremost part of Section V, a solution is presented and it is highlighted the necessity of the refactoring strategies that are matched with the nature of the software project. This section takes a look into various methods used for verifying and fine-tuning the software requirements, along with keeping stakeholders involved in the procedure. As the last section of the paper, Section VI is to demonstrate results by using a case study. Thus, a hotel management system is chosen to apply the approach. The case study here demonstrates that the solution offered brings into being specific advancements in software quality, return on investment, and achievement of the project goals.

8. CONCLUSION

In this section, we conclude the paper that is staring at the effects of code refactoring on software quality. Everyone knows that who belongs to the software industry software quality is a big challenge for software survival and its credibility purpose [20]. Apply code refactoring in such a way that does not affect the software behavior and its output functionality. In another section, we will move to code refactoring to software requirements, because we know very well that every time code refactoring does not produce authentic results [24].

In the software requirement's structure, the whole development is based on it. It is an earlier phase of software development that uses the requirements-divided approach. Starting with gathering the requirements from actual sources, then filtering them, breaking them into relevant requirements, then allocating the numbers to break or filter requirements, and then all these requirements convert into documented form and send to the software development team. So, we can say that the requirements phase is the most valued in that we summarize the overall software project in the earlier phase in the requirements form. Due to the starting phase, it is not very expensive. If all

the requirements are valid and do not exist in any ambiguity, then the developed software is most probably error-free and based on client requirements.

9. FUTURE WORK

The future of code refactoring in software engineering will be mainly driven by automation, integrating these tasks with DevOps and investment into technical debt reduction, mature architectures modularization, security intensification, AI-based solution building, and educational ecosystem construction. These will be the key elements of more effective, fast-moving, and secure software development, at the end stage of improving the quality of software systems.

REFERENCES

[1] B. N. Van Vliet et al., "Direct and indirect methods used to study arterial blood pressure," *Journal of pharmacological and toxicological methods,* vol. 44, no. 2000, pp. 361–373, 2001.

[2] A. Hochstein et al., "Evaluation of Service-Oriented IT Management in Practice," *In Proceedings of ICSSSM'05. 2005 International Conference on Services Systems and Services Management,* Vol. 1, pp. 80-84, 2004.

[3] S. M. H. Dehaghani, and N. Hajrahimi, "Which Factors Affect Software Projects Maintenance Cost More?," *Acta Informatica Medica,* vol. 21, no. October 2012, pp. 63–66, 2013, doi: 10.5455/aim.2012.21, , 2013.

[4] M. Kessentini et al., "Many-Objective Software Remodularization using NSGA-III," *ACM Transactions on Software Engineering and Methodology (TOSEM),* vol. 24, no. 3, pp.1-45, 2015.

[5] E. F. Brown et al., "CRUSTAL HEATING AND QUIESCENT EMISSION FROM TRANSIENTLY ACCRETING NEUTRON STARS Edward F. Brown, Lars Bildsten, and Robert E. Rutledge 1," *The Astrophysical Journal,* vol. 504, no. 2, pp. 1996–1999, 1998.

[6] A. Yamashita and L. Moonen, "Do code smells reflect important maintainability aspects?," I*n 2012 28th IEEE international conference on software maintenance (ICSM),* pp. 306-315

[7] M. Abebe and C. Yoo, "Trends , Opportunities and Challenges of Software Refactoring : A Systematic Literature Review," *international Journal of software engineering and its Applications,* vol. 8, no. 6, pp. 299–318, 2014.

[8] S. Kaur and P. Singh, "The Journal of Systems and Software How does object-oriented code refactoring influence software quality ? Research landscape and challenges," *Journal of Systems and Softwar,* vol. 157, pp.110394, doi: 10.1016/j.jss.2019.110394, 2019.

[9] F. Coelho et al., "Refactoring-Aware Code Review : A Systematic Mapping Study," *In 2019 IEEE/ACM 3rd Int. Work. Refactoring*, pp. 63–66, doi: 10.1109/IWoR.2019.00019, 2019.

[10] A. Ouni et al., "Multi-Criteria Code Refactoring Using Search-Based Software Engineering : An Industrial Case Study," *ACM Transactions on Software Engineering and Methodology (TOSEM),* vol. 25, no. 3, pp.1-53, 2016.

[11] M. Paixão et al., "Behind the Intents : An In-depth Empirical Study on Software Refactoring in Modern Code Review," I*n Proceedings of the 17th International Conference on Mining Software Repositories,* pp. 125–136, 2020.

[12] B. Alshammari et al., "Security Assessment of Code Refactoring Rules," *In WIAR 2012; National Workshop on Information Assurance Research*, pp. 1-10, 2012.

[13] M. Alshayeb, "The Impact of Refactoring on Class and Architecture Stability," J*ournal of Research and Practice in Information Technology,* vol. 43, no. 4, pp.269-284, 2011.

[14] G. Szoke et al., "Designing and Developing Automated Refactoring Transformations : An Experience Report," *In 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER),* Vol. 1, pp. 693-697, doi: 10.1109/SANER.2016.17, 2016.

[15] H. Mumtaz, et al., "An empirical study to improve software security through the application of code refactoring," *Inf. Softw. Technol.,* vol. 96, pp. 112–125, doi: 10.1016/j.infsof.2017.11.010, , 2018.

[16] E. Ammerlaan and A. Zaidman, "Old Habits Die Hard: Why Refactoring for Understandability Does Not Give Immediate Benefits," *In 2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER),* pp. 504–507, 2015.

[17] P. Techapalokul and E. Tilevich, "Code Quality Improvement for All: Automated Refactoring for Scratch," *2019 IEEE Symp. Vis. Lang. Human-Centric Comput.,* pp. 117–125, 2019.

[18] J. Cohen, "A COEFFICIENT OF AGREEMENT FOR NOMINAL SCALES," *Educational and psychological measurement,* vol. 20, no. 1, pp. 37–46, 2016.

[19] D. Damian et al., "An Industrial Case Study of Immediate Benefits of Requirements Engineering Process Improvement at the Australian Center for Unisys Software," *Empirical Software Engineering,* pp. 45–75, 2007.

[20] G. Lacerda et al., "Code smells and refactoring: A tertiary systematic review of challenges and observations," *J. Syst. Softw.,* vol. 167, pp.110610, doi: 10.1016/j.jss.2020.110610, 2020.

[21] A. Almogahed, Mahdin et al., "A refactoring categorization model for software quality improvement," *Plos one,* vol. 18, no. 11, pp. e0293742, 2023.

[22] B. Nyirongo et al., "A Survey of Deep Learning Based Software Refactoring," a*rXiv preprint arXiv:2404.19226,* 2024.

[23] E. A. AlOmar et al., "How to refactor this code? An exploratory study on developer-ChatGPT refactoring conversations," *In Proceedings of the 21st International Conference on Mining Software Repositories,* pp. 202-206, April 2024.

[24] A. C. Bibiano et al., "Composite refactoring: Representations, characteristics and effects on software projects," *Information and Software Technology,* 156, pp. 107134, 2023.

[25] N. Raeesinejad et al., *Refactoring. In The Ignite Project: A Journey in Scrum,* pp. 168-179, Singapore: Springer Nature Singapore, 2023.

Asghar et al. LGURJCSIT 2024

LGU Research Journal of Computer Science & IT

ISSN: 2521-0122 (Online)
ISSN: 2519-7991 (Print)

doi: 10.54692/lgurjcsit.2024.082573

Vol (8): Issue (2), April – June 2024

Integrative Machine Learning Framework for Accurate COVID-19 Forecasting

Muhammad Asghar[1*], Nida Anwar[2], Manahil Hassan[3], Komal Saleem[4]

[1]Department of Computer Science and Information Technology, Virtual University of Pakistan, Lahore, Pakistan.

[2]Department of Computer Science and Information Technology, Virtual University of Pakistan, Lahore, Pakistan.

[3]Department of Computer Science and Information Technology, Virtual University of Pakistan, Lahore, Pakistan.

[4]Department of Computer Science and Information Technology, Virtual University of Pakistan, Lahore, Pakistan.

Email: ms190400021@vu.edu.pk

ABSTRACT:

This research paper introduces a machine learning (ML) forecasting model for COVID-19 cases to investigate the performance of machine learning algorithms and develop a new procedure to improve prediction efficiency. Utilizing the Multilayer Perceptron (MLP), Linear Regression (LR), K-Nearest Neighbours (KNN), Support Vector Machines (SVM), and the proposed approach, the study considers the COVID-19 data to forecast case numbers. The study contextualizes its findings through a systematic methodology of dataset compilation, algorithm interpretation, and framework development. It uses measures such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) to evaluate the predictive performance. The trend in COVID-19 data and predictions from different algorithms is shown through graphical illustrations. The same goes for the proposed framework predictions. This study demonstrates that the proposed LR approach and the framework outperform previous MLP, KNN, and SVM models, suggesting the relevance of explainable and robust modelling solutions for COVID-19 risk assessment. Our suggested framework, which outperforms individual algorithms by averaging their results after combining them, significantly reduces prediction errors. Discussion of implications of forecasts for interventions, resource allocation, and policy decisions is done, with the need for accurate forecasting in the pandemic response highlighted. Further research can be expected to improve the framework, incorporate new machine learning (ML) techniques, and deploy real-time adaptive modelling systems. Collaboration between researchers, policymakers, and healthcare practitioners is crucial for adopting research results into practice and promoting evidence-based decision-making in outbreak response and preparedness. Generally, this study shows the progress of epidemiological forecasting and provides important information in fighting COVID-19 and future epidemics.

KEYWORDS: Covid-19 Forecasting, Machine Learning Algorithms, Neural Networks, Support Vector Machine, Linear Regression, KNN.

1. INTRODUCTION

This template is an example of formatting a paper for the Journal of Southwest Jiaotong University. The template is available online on the page for all authors on the official website of the Journal of Southwest Jiaotong University.

1.1. Background information on the COVID-19 pandemic

The COVID-19 pandemic, caused by the SARS-CoV-2 virus, has posed an unprecedented challenge to global health systems, economies, and societies since its emergence in late 2019. Initially detected in Wuhan, China, in December 2019, the virus spread rapidly across international borders.[1]

The World Health Organization (WHO) declared COVID-19 a pandemic in March 2020. Efforts to curb the spread of the virus have included large-scale testing and tracing, movement restrictions, and quarantine protocols.[2] During the pandemic, a clear understanding of the necessity for accurate prognostications of the severity of the COVID-19 spike appeared to be instrumental in organizing the subsequent public health measures, resource management, and decisions. This research aims to develop a better way of improving the COVID-19 prediction models so that a lot of effort is achieved in fighting the pandemic.

1.2. Importance of Predicting COVID-19 Cases

Prediction of COVID-19 cases has several reasons, as follows. It informs resource allocation by health authorities and policymakers, ensuring better mobilization of resources. Reliable projections enable epidemiologists and public health managers to implement effective strategies to minimize virus transmission. Forecasting also helps optimize vaccine distribution to areas with the greatest need. Furthermore, sharing projected trends with the public raises' awareness and compliance with health measures, such as self-isolation and social distancing [3].

In addition to aiding immediate pandemic response, evaluating the performance of various machine learning algorithms for COVID-19 case prediction is essential. This study systematically assesses the performance of these algorithms, proposing a new approach that leverages the strengths of multiple models to improve predictive accuracy.

1.3. Objectives and Structure

This study aims to improve forecasting tools and provide a reliable information base for evidence-based decision-making in response to COVID-19. It introduces a novel framework integrating multiple machine-learning algorithms to enhance predictive accuracy. The paper is structured as follows: The Methodology section describes the dataset and the employed machine learning algorithms, followed by a detailed explanation of the proposed framework. The Results and Discussion section presents the findings, including graphical illustrations and comparative analyses. The Conclusion summarizes the essential findings and suggests directions for future research.

By advancing the field of epidemiological forecasting, this study contributes valuable insights for combating COVID-19 and future epidemics.

2. RELATED WORK

Manuscripts should be written in English or Chinese. The title of the paper, information about the authors, abstract and keywords, and bibliography must be written in English and Chinese. You can submit an article online at the journal's website. To submit a paper, the author will first need to register. All manuscripts are peer-reviewed. The first decision is given to authors about 10-50 days after submission; acceptance for publication after revisions is done within 7-10 days (averages for articles published in this journal in the first half of 2021).

The Total Article Processing Charge (APC) is USD 600. Local VAT or Sales Tax will be added if applicable. Many national and private research funding organizations and universities explicitly cover APCs for articles from funded research projects. Waivers may be granted at the editorial office's discretion and should be discussed with the Editor when submitting the article. The editorial decision-making is decoupled from the authors' ability to pay the Processing Charges. However, authors should consider whether they have sufficient funds to cover the full APC. Journal of Southwest Jiaotong University also offers discount vouchers to selected reviewers.

Forecasting techniques are designed and implemented in various ways to avoid major disasters and make better decisions [4]. This chapter will review the literature on COVID-19 and its predictions; forecasting and its results will be described in this literature work [5]. The proposed approach aimed to predict the future course of COVID-19 using a straightforward yet innovative technique. Based on their analysis, they expect a continuous increase in confirmed COVID-19 cases, assuming the data's reliability and a continuation of the current infection pattern. They share the results of a real-time prediction exercise with important

implications for planning without any association with pledges or commitments. Their work involves forecasting the number of confirmed COVID-19 cases using powerful time-series models and examining the trend of recovered cases. The most recent predictions, which cover the period from 02 March 2020 to 21 March 2020, indicate a notable increase in the pattern of cases globally and a growth in the associated susceptibility.

Pandemic forecasting approaches have been researched thoroughly and implemented to reduce destructive effects and enhance decision-making. Now, the forecasts of COVID-19 impact contain different methods for control and prediction proposed by various authors. One of them was based on time series models aimed at predicting cases of COVID-19, and it was stated that there would be an increase in the number of confirmed cases in the future if other factors remained constant. Data on the statistics of the identified infection is reliable. The journal also had an article of Chinese origin and examined Italian and French cases to explain the possibility of managing the rate of infections to stop the virus [6].

Mathematical models like the Susceptible Exposed Infectious Removed (SEIR) model have been adjusted to include age groups and catchment areas to predict the daily new case Numbers, hospitalizations, deaths, and probable ICU bed demand with COVID-19 [7].

Other works developed artificial intelligence and machine learning models for COVID-19 prediction, namely the additive regression models, dynamic maps, MLP, VAR, and LR [8]. These studies further prove that machine learning and artificial intelligence are instrumental in combating pandemics.

Thus, a prediction model based on MLP, VAR, and LR will be developed as the machine learning method [9]. These algorithms are employed to forecast the number of COVID-19 patients in India. The author employed a free tool called WEKA and another tool called Orange to apply these models. Finally, the author analyzed the results of the three algorithms and proposed that MLP gives better outcomes than LR and VAR in terms of accuracy.

Modern machine learning and artificial intelligence techniques are elaborated to tackle the COVID-19 pandemic. The author briefly describes these techniques along with their applications in the form of a table, which helps a lot in understanding the power of machine learning, artificial intelligence, and other modern technologies. The author also explained that researchers are utilizing/practising these techniques to fight against COVID-19 more effectively [10].

In this paper, the authors have discussed different safety measures that can be taken using machine learning predictive techniques like predicting an outbreak, screening of patients, vaccine development, contact tracing, etc. It is also described that ML and AI can enhance treatment, forecasting and prediction, contact tracing, and vaccine development significantly [11].

In this paper, the researcher suggests employing a non-linear machine learning approach, specifically the Partial Derivative Regression (PDR-NML) method, for predicting the COVID-19 epidemic. An exponential second derivative linear regression model was utilized to implement this, and a computer-assisted analysis was conducted to determine the appropriate parameters within the dataset. The results of this study indicate that the proposed machine learning approach surpasses the current state-of-the-art methods in predicting COVID-19 trends within the Indian community [12]. This study suggests a unique strategy for forecasting the COVID-19 smoothed daily new cases per million. The nations' time-series data were utilized to generate GMM representations [13].

3. METHODOLOGY

3.1. Description of the Dataset

The dataset used in this research comprises a complete collection of COVID-19-related papers from sources such as national health systems, global health organizations, and research institutions. This dataset covers the epidemiological parameters and characteristics, demography, healthcare facilities & infrastructure, and socio-economic factors that are part of the countries and regions that are the epicentres of the pandemic. The key variables utilized in this study are:

- Total confirmed cases
- Total deaths
- Total recoveries
- Daily new cases
- Daily new deaths
- Testing rates

These variables contain the necessary information to study the dynamics of the pandemic and the actions taken in detail.

3.2. Explanation of Machine Learning Algorithms

In this study, four distinct machine learning algorithms are employed to predict COVID-19 cases: Among them, the most common are – Multilayer Perceptron (MLP), Linear Regression (LR), K-Nearest Neighbors (KNN), and Support Vector Machines (SVM). Each algorithm introduces its properties, which qualify it to perform this task.

A. Multilayer Perceptron (MLP)

MLP is one of the feedforward neural networks used to train the data sets containing features with non-linear correlations. It is especially recommended for analyzing large data sets in which the nature of relationships is unknown.

B. Linear Regression (LR)

LR is the most basic type of algorithm where a linear equation represents the correlation between features and target variables. Due to its interpretability and applicability in situations with linear relationships can be used in COVID-19 prediction.

C. K-Nearest Neighbors (KNN)

KNN is considered one of the non-parametric methods of classification where the nearest data points directly influence the model's flexibility. This flexibility makes it quite valuable in dealing with non-linear relationship situations.

D. Support Vector Machines (SVM)

SVM is well known for its high efficiency in building the decision boundary that gives the most significant class separation. However, it is not always efficient and depends on the data's characteristics. These algorithms were chosen based on many properties so that the framework could use one strength and minimize the others' weaknesses.

Figure 1: Proposed Methodology

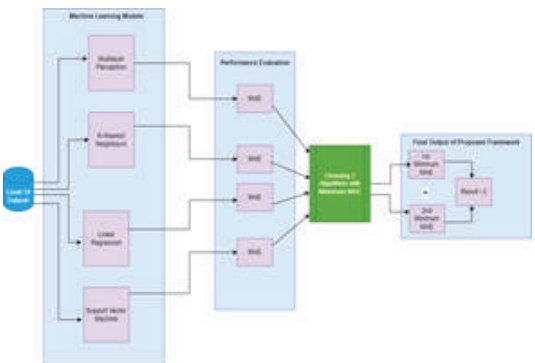

Figure 2: Our Proposed Framework

3.3. Detailed Explanation of the Proposed Framework for COVID-19 Prediction

Several Machine Learning Models are used in the COVID-19 prediction to maximize the prediction scores. This framework systematically makes predictions about the task, chooses the correct algorithm, merges the result, and assesses the performance.

A. Algorithm Selection

The four depicted algorithms, MLP, LR, KNN, and SVM, are chosen in a way that their characteristics will complement each other to improve the overall prediction ability of the model.

B. Prediction Generation

Each algorithm is used to build models and establish correspondence in the given data set with the help of epidemiological, demographic, and healthcare system indicators to forecast COVID-19 cases. With the help of iterations conducted on training and testing the models, the current models are increasingly accurate.

C. Combination of Predictions

After passing through the updating process and outlier elimination, the final forecasts are then averaged using a weighted summation model. The weights influence algorithms' accuracy levels, whereby algorithms that record high accuracy are given high weights in the final prediction.

4) Performance Evaluation:

These integrated estimations are then checked

with the help of indicators like Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). These give quantitative measures of the degree of prediction and measures of reliability so the framework can be compared against accurate COVID-19 data.

3.4. Mathematical Formulation

The suggested framework has been mathematically defined as the average of the minimum MAE errors of the two superior models. The formula is as follows:

$$\text{Proposed Framework}=(\text{Min1}(\sum_{i=1}^{n}|yi-y\tilde{i}|/n) + \text{Min2}(\sum_{i=1}^{n}|yi-y\tilde{i}|/n))/2 \quad (1)$$

In this formula:
Min1(∑ |yi-y ĩ| /n) is showing the minimum MAE of 1st algorithm.
Min2($\sum_{i=1}^{n}$ |yi-y ĩ| /n) is showing the minimum MAE of 2nd algorithm.
This approach takes advantage of the robust characteristics of an array of algorithms, making the prediction framework more accurate and dependable. Therefore, this methodology presents the study process, the methods to be used in this research, and the objectives and questions posed in the research study. The general outline of the research, the description of the dataset, the machine learning algorithms, and the proposed framework are pretty comprehensive and coherent, which helps in understanding all the research design and analysis procedures used in the study.

4. RESULTS AND DISCUSSION

4.1. Presentation of graphical results depicting COVID-19 data trends

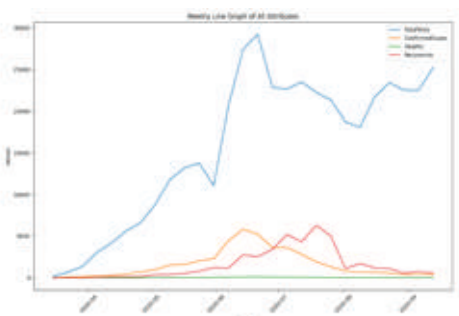

Figure 3: Weekly Graph of all Attributes

Figure 3 shows the line graph in which all dataset attributes are drawn. The highest line shows the total number of tests conducted (blue colour). Confirmed cases are shown in yellow, and recoveries are shown in red. The line at the bottom shows the number of deaths in the green line.

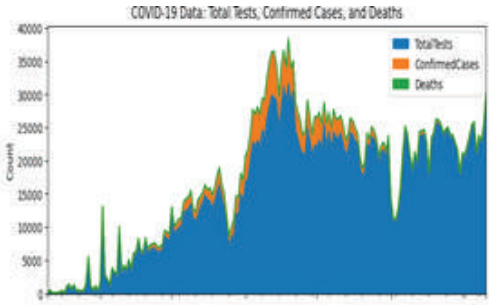

Figure 4: Stacked Graphs of all Attributes

In Figure 4, the stacked graph shows three attributes. Attributes like Total Tests, confirmed cases, and deaths are shown in different colours. Legend is also printed on the graph.

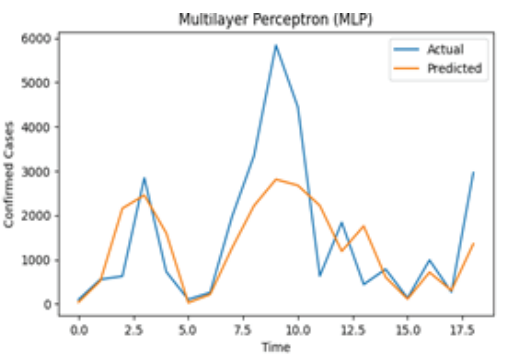

Figure 5: Prediction using MLP

In Figure 5, we can see the output of the multilayer perceptron algorithm. The actual value of confirmed cases is shown in the blue line, and the orange line shows the predicted values of confirmed cases. The X-axis shows the period. As we can see, there is an enormous gap between the blue line and orange line in the centre of the graph, so by looking at the graph, we can say that this model is not performing well.

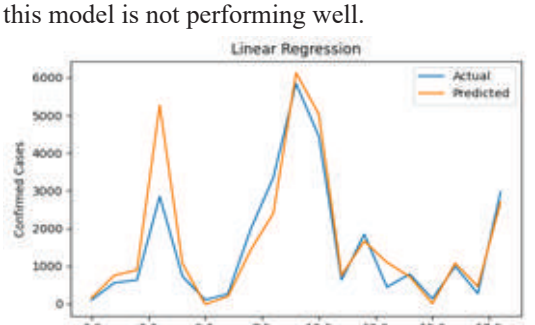

Figure 6: Prediction using Linear Regression

In Figure 6, we can see the output of the linear regression algorithm. The actual value of confirmed cases is shown in the blue line and the orange line is showing the predicted values of confirmed cases. The X-axis shows the period. As we can see both lines are very close, so by looking at the graph, we can say that this model is performing well.

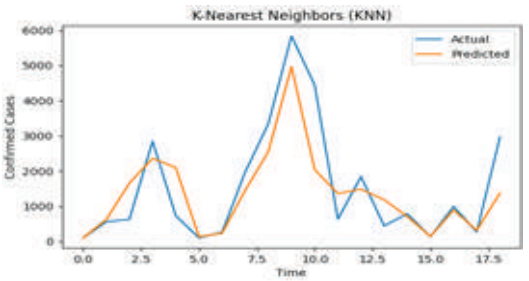

Figure 7: Prediction using KNN

In Figure 7, we can see the output of the KNN algorithm. The actual value of confirmed cases is shown in the blue line, and the orange line shows the predicted values of confirmed cases. The X-axis shows the period. As we can see, both lines are very close to each other, so by looking at the graph, we can say that this model is performing well.

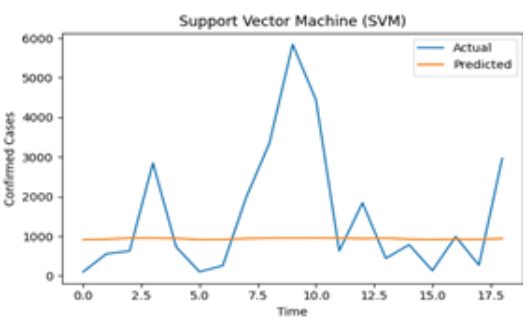

Figure 8: Prediction using SVM

In Figure 8, we can see the output of the support vector machine algorithm. The actual value of confirmed cases is shown in the blue line and the orange line is showing the predicted values of confirmed cases. The X-axis shows the period. As we can see, there is a big gap between the blue line and the orange line in the center of the graph, so by looking at the graph, we can say that this model is not performing well.

So, by analyzing all previous graphs, we can say that the two models named KNN and linear regression are performing well compared to the other two models named support vector machine and multilayer perceptron, which are not performing well. Based on our proposed model, we will develop a machine-learning framework based on the two algorithms that are performing well.

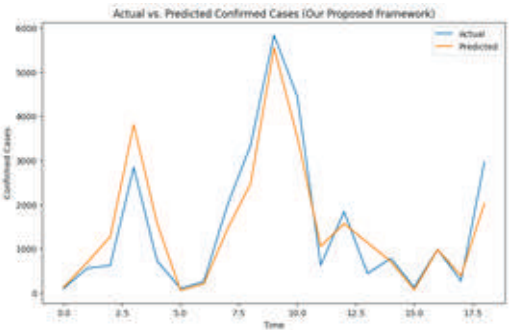

Figure 9: Prediction using our Proposed Framework

In Figure 9, we can see the output of our proposed algorithm. The actual value of confirmed cases is shown in the blue line and the orange line shows the predicted values of confirmed cases. The X-axis shows the period. As we can see both lines are very close to each other.

Table 1: Output of Different Models

Algorithm	MAE	MSE	RMSE
MLP	805.93	1300931.94	1140.58
LR	396.27	437986.66	661.80
KNN	592.29	756687.85	869.87
SVM	1172.32	2881699.18	1697.55
Our Proposed Framework	416.38	299751.13	547.49

4.2. Discussion

This section interprets the results in Table 1, providing a detailed analysis of each model's performance, its suitability for the task, and implications for COVID-19 prediction.

4.3. Analysis of Results

Results shown in Table 1 depict that the Linear Regression (LR) and also the K-Nearest Neighbors (KNN) algorithms yielded better results as compared to the Multilayer Perceptron (MLP) and the Support Vector Machine (SVM) algorithms in terms of COVID-19 case prediction. Hence, the Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) of each model are the parameters that are applied as criteria for comparing their accuracy of

predictions.

A. Linear Regression (LR)

Out of all the models, LR came out on top with the least MAE of (396. 27); MSE of (437, 986. 66); and RMSE of (661. 80); hence, recommended for use in the prediction of employee attrition. Due to LR's basic structure and ease of interpretability, it is quite suitable for modelling linear association between the predictors and the response variable. Concerning COVID-19 prediction, whereby some epidemiological patterns could be linear, the ability of LR to estimate these relations is paramount to its better performance.

B. K-Nearest Neighbors (KNN)

The predictive ability of KNN was also very desirable and reasonably accurate regarding MAE of 592. 29, MSE of 756,687. 85, and RMSE of 869. 87. KNN is mainly a non-parametric model; thus, it can always strategically adjust to catch various complicated data patterns, which is useful in non-linear data. This could have boosted its performance in the role of predicting the numbers of COVID-19 cases since the patterns may fluctuate and depend on various circumstances.

C. Multilayer Perceptron (MLP)

Regarding the error metrics, MLP recorded slightly higher values than LR and KNN; thus, MAE = 805. 93, MSE = 1,300,931. 94, RMSE = 1, 140. 58. MLP excelled due to its sensitivity to the architecture and the fact that hyperparameter optimization is generally demanding. The structure, the number of layers in the neural network, and the related non-linear setting might not have been suitable for the proffered data, implying that this particular predictive task was executed with lower accuracy.

4.4. Support Vector Machine (SVM)

SVM came out as the worst-performing model with the highest MAE (1,172.32), MSE (2,881,699.18), and RMSE (1,697.55). SVM is also good at determining the classification decision boundaries but may not perform well in the case of regression tasks, and this was seen especially when it comes to COVID-19 prediction, where the data distribution, especially considering volatility, is irregular. It shows a clear decline in SVM generalizing capabilities while further demonstrating the complications and variations associated with analyzing pandemic-related data.

4.5. Performance of MLP

The following variables are known to be associated with MLP's lack of efficiency. Neural networks like MLP require significant amounts of data and careful tuning of hyperparameters. In this study, the available data and the specific characteristics of the COVID-19 dataset might not have provided the ideal conditions for MLP to excel. Additionally, the potential for overfitting and the complexity of training deep networks could have contributed to its high error metrics.

4.6. Implications and Broader Context

The findings of this study have several implications for the field of COVID-19 prediction and beyond. The superior performance of LR and KNN suggests that more straightforward, interpretable models can be highly effective for predicting pandemic trends, especially when dealing with linear or near-linear relationships. This contrasts with the often assumed necessity for complex models in all predictive tasks.

The proposed framework, which combines predictions from multiple algorithms, demonstrated the lowest error metrics (MAE: 416.38, MSE: 299,751.13, RMSE: 547.49), validating the approach of leveraging the strengths of individual models to enhance overall predictive accuracy. This ensemble method can be valuable for public health organizations, providing more reliable forecasts to inform policy and resource allocation.

5. LIMITATIONS AND FUTURE RESEARCH

As with most studies, there are some hindrances, and this work is no exception despite the positivity of the findings. The variables chosen may not include all possible factors affecting COVID-19 spread, like social interactions or policies and government measures worldwide. It is suggested that future studies investigate the inclusion of other characteristics and enhanced machine learning algorithms to attain a higher level of prediction.

In addition, the existence of real-time update systems and progressive modelling procedures is also helpful in increasing the effectiveness of the prediction models in response to the changing epidemiological status. This study highlights the importance of cooperation between academic individuals, authorities, and clinicians to implement the results acquired into practice, thus

improving the effectiveness of planning further pandemic control.

6. CONCLUSION

It is shown that the machine learning approaches, namely Linear Regression and K-nearest neighbours, can be practical for COVID-19 prediction. The proposed framework helps improve the system's accuracy with the help of many base models. The widespread findings of this study help develop public health policies and illustrate the need for building practical and understandable models to address pandemics.

Subsequent research should aim to improve and enhance the presented framework; besides that, new machine-learning approaches should be integrated, and new datasets of patients' cases should be used to enhance the accuracy of the prediction. Thus, by using such models to advance through collaborative research, it is possible to work towards figuring out how to prevent and respond to future epidemics of infectious diseases, therefore retaining the well-being of the populations in question.

REFERENCES

[1] M. Ciotti et al., "The COVID-19 pandemic," *Critical reviews in clinical laboratory sciences,* vol. 57, no. 6, pp. 365-388, 2020.

[2] B.S. Mohan and V. Nambiar, "COVID-19: an insight into SARS-CoV-2 pandemic originated at Wuhan City in Hubei Province of China," *J Infect Dis Epidemiol*, vol. 6, no. 4, pp. 146, 2020.

[3] J. Devaraj et al., "Forecasting of COVID-19 cases using deep learning models: Is it reliable and practically significant?," *Results in Physics,* vol. 21, pp. 103817, 2021.

[4] G. R. Shinde et al., "Forecasting models for coronavirus disease (COVID-19): a survey of the state-of-the-art," *SN computer science,* vol. 1, pp. 1-15, 2020.

[5] F. Petropoulos and S. Makridakis, "Forecasting the novel coronavirus COVID-19," *PloS one,* vol. 15, no. 3, p. e0231236, 2020.

[6] D. Fanelli, F. Piazza, Solitons, and Fractals, "Analysis and forecast of COVID-19 spreading in China, Italy and France," Chaos, Solitons & Fractals, vol. 134, pp. 109761, 2020.

[7] C. Massonnaud et al., "COVID-19: Forecasting short term hospital needs in France," *medrxiv,* pp. 2020.03. 16.20036939, 2020.

[8] A. N. Roy et al., "Prediction and spread visualization of COVID-19 pandemic using machine learning," 2020.

[9] R. A. A. Sujath et al., "A machine learning forecasting model for COVID-19 pandemic in India," vol. 34, pp. 959-972, 2020.

[10] A. Kumar et al., "A review of modern technologies for tackling COVID-19 pandemic," *Diabetes & Metabolic Syndrome: Clinical Research & Reviews,* vol. 14, no. 4, pp. 569-573, 2020.

[11] S. Lalmuanawma et al., "Applications of machine learning and artificial intelligence for Covid-19 (SARS-CoV-2) pandemic: A review," *Chaos, Solitons & Fractals,* vol. 139, pp. 110059, 2020.

[12] D. P. Kavadi et al., "Partial derivative non-linear global pandemic machine learning prediction of covid 19," *Chaos, Solitons & Fractals,* vol. 139, pp. 110056, 2020.

[13] E. Külah et al., "COVID-19 forecasting using shifted Gaussian Mixture Model with similarity-based estimation," *Expert Systems with Applications,* vol. 214, pp. 119034, 2023.

Sharif et al. LGURJCSIT 2024

LGU Research Journal of Computer Science & IT

ISSN: 2521-0122 (Online)
ISSN: 2519-7991 (Print)

doi: 10.54692/lgurjcsit.2024.082569

Vol (8): Issue (2), April – June 2024

A Machine Learning-Based Approach for the Detection of DDoS Attacks on the Internet of Things Using CICDDoS2019 Dataset – PortMap

Hanan Sharif[1], Sardar Usman[2], Muhammad Hasnain[3], Shagufta Anwar[4], Mohammed Nawaf Altouri[5], Fahad Mohammed Sharahili[6], M. Usman Ashraf [7*]

[1,3,4]Department of Computer Science, Leads University Lahore, Punjab, Pakistan.

[2]Department of Computer Science Software Engineering & IT, Grand Asian University, Sialkot, Punjab, Pakistan.

[5]University of prince muqrin, Madinah, Saudi Arabia.

[6]Imam mohammad Bn Saud Islamic University, Riyadh, Saudi Arabia.

[7]Department of Computer Science, GC Women University Sialkot, Punjab, Pakistan,

Email: usman.ashraf@gcwus.edu.pk

ABSTRACT:

In today's technological era, the Internet has become ubiquitous, playing a vital role in our daily lives. With the exponential growth of IoT innovation, millions of interconnected IoT-enabled devices rely on cloud services to communicate over the Internet. However, this rapid development also exposes these devices to various threats, with DDoS (Distributed Denial of Service) and DoS (Denial of Service) attacks being particularly potent and destructive. DDoS attacks present a unique challenge as they are tough to detect using conventional intrusion detection frameworks and traditional methodologies. Fortunately, advancements in machine learning have provided a promising solution by enabling accurate differentiation between DDoS attacks and other forms of data. This study proposes a DDoS detection model based on machine learning algorithms. We used the most recent and freely available online dataset called CICDDoS2019 to conduct this study. Various machine learning-based techniques were explored to identify the characteristics associated with accurate classification. Among the algorithms tested, AdaBoost and XGBoost demonstrated exceptional performance. A hybrid approach will be incorporated into this model as part of future work, further improving its capabilities. It is worth noting that this model will be continuously updated with new data on DDoS attacks, ensuring its relevance and effectiveness in combating emerging threats. By leveraging machine learning techniques, this approach enhances the detection of DDoS attacks on Internet of Things networks, safeguarding the integrity and security of connected devices and the overall IoT ecosystem.

KEYWORDS: DDoS, Internet of Things, Machine Learning, Classification, DDoS Detection, CICDDoS2019.

1. INTRODUCTION

The Internet of Things (IoT) continues revolutionizing our world, bringing numerous benefits and advancements. Today, IoT devices play a pivotal role in our daily lives, permeating various aspects such as smart cities, electricity grids, homes, vehicles, construction machinery, and hospitals. This exponential growth in digital technology aims to enhance our lives by seamlessly integrating physical devices with digital intelligence, creating a more comfortable, intelligent, and manageable environment. IoT devices collect vast amounts of data, which can be shared through the Internet, enabling access from anywhere at any time. These data streams are typically stored and accessed through integrated

cloud platforms, facilitating communication among IoT devices. Research indicates that by 2030, the number of IoT devices is projected to reach 20 billion, with the current count already at 10.07 billion, all interconnected through the web [1]. However, with this extensive proliferation of interconnected devices comes the need to protect the data they generate. Cyber security is crucial to prevent unauthorized access and safeguard our valuable assets and personal privacy [2]. As the volume of data transferred among these devices continues to grow, robust security measures must be in place to mitigate the risk of cyber-attacks.

The cyber security threats faced by IoT devices can be classified into six types: denial of service, impersonation, eavesdropping, hardware tempering, bogus information, and message suspension. [3]. DoS and its more advanced version, DDoS, the abbreviation of distributed denial of service, are complicated and much more complex attacks to detect or mitigate than the other five types [4]. In this category of attack, a lot of information is sent through the servers, which brings about prevention of administration given by the specialist co-op. Due to this, consumers or users won't be able to use services properly and face problems in receiving proper service [5]. DDoS attacks are also classified into different types depending on different characteristics. A DDoS attack is classified into the following types [6]: 1) SYN Flood 2) TCP Flood 3) Ping of Death 4) DNS Flood 5) Zero-Day DDoS 6) HTTP Flood 7) ICMP Flood 8) SYN Flood 9) UDP Flood.

The primary objective of this study is to identify the most dependable, precise, and accurate algorithm for detecting DDoS attacks on IoT devices. The research utilizes the CICDDoS2019 dataset, depicted in Figure 1 (Model Architecture). The dataset is divided into two categories: "harmful" and "harmless" classes. This study introduces a machine learning model that rapidly detects DDoS attacks, comparing thoroughly with existing detection models. By utilizing the CICDDoS2019 dataset, the study refines dimension reduction and feature selection techniques for effective DDoS detection. Rigorous testing identifies the most suitable algorithm for detecting distributed denial of service attacks proficiently. The subsequent sections of the paper are structured as follows: Section II provides a concise introduction to the background, followed by a comprehensive literature review in Section III. Section IV outlines the intricate methodology employed in the study. The findings and their analysis are expounded upon in Section V. Finally, and Section VI encapsulates the conclusions drawn from the analysis and points towards potential future directions.

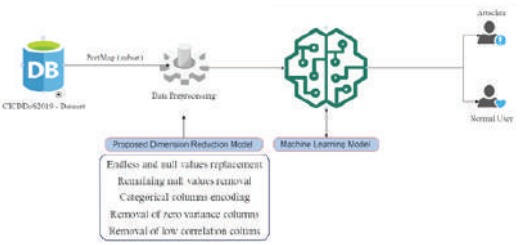

Figure 1: Architectural framework of the proposed model

2. BACKGROUND

The term "IoT" stands for the Internet of Things, encompassing all internet-connected devices with specific functionalities in our daily lives. These devices include actuators, sensors, and microcontrollers. The IoT comprises countless physical, digital devices worldwide that collect, exchange, and analyze data. The applications of IoT are diverse and include connected communities, electric grids, electric mobility, smart homes, healthcare systems, and smart living, among others. [7]. Due to this extensive use and exponentially increasing data of IoT device users, it has become a significant threat to the privacy and usability of these devices. Many attacks have been focused on IoT gadgets since their development in the advanced world. Denial of Service and its advanced form, DDoS, are among today's most serious security threats. In this attack, the attacker aims to bypass an organization's resources by manipulating the incoming and outgoing network traffic. This leads to service disruptions and prevents authorized users from exercising their administrative rights effectively. DDoS attacks come in various forms, each with its consequences. It is crucial to clearly understand IoT and its functioning to distinguish between different types of attacks and their distinct characteristics. With diverse detection methods and mitigation strategies available for each type of DDoS attack, this knowledge becomes essential in effectively countering such threats.

Every web-enabled computer device that can transmit and receive data via sensors, communicate using a network, and have processing capability by using embedded processors is considered an IoT (Internet of Things). Existing and emerging technologies are utilized for sensing, network

ing, and robotics. This allows the user to achieve deeper analysis, integration, and automation within a system. With the increase in networking capabilities of machines and appliances used in different fields daily, such as homes, offices, transportation, buildings, and industries, they open a world of opportunities for the betterment of business and better customer satisfaction. Some of the key features of the IoT are communication, sensors, artificial intelligence, small devices, and active engagement [8].

2.1. How IoT Works?

IoT technology is classified into four basic categories: IoT gadgets and devices, cloud-based data storage systems, remote controls used through mobile applications, and gateway systems. Combining systems can make connecting two or more devices possible [9]. In Fig. 2, we can see how the IoT layers are connected through a general diagram. The following are some key components of IoT technology that have a vital role in IoT device performance.

- **Sensor-based Technology:** Important information about gadgets can be detected from an extensive range of sensors attached to the devices. The collected information can be location, temperature, gases, any industrial machine's function, or sensory data for plant health [10].
- **IoT Gateways:** Gateways bridge the gap by providing a link between the end user and the IoT device, thus allowing them to connect and communicate [11].
- **Storage of Data and Cloud Server:** The cloud stores and analyses data. Collected data reaches the cloud after passing through the gateway. The data is then processed and transferred to the user for further proceedings. A user executes different actions on data depending on the achieved information [12].
- **Usage of Mobile Applications for Remote Controls:** Remote controls are used by end users through mobile phones, laptops, tablets, etc., and they have different applications installed on them. These applications control, monitor, retrieve data, and perform different actions on a user's remote-controlled IoT gadgets [13].

2.2. DDoS Attacks - Detection

There are a variety of strategies for successfully detecting DDoS attacks. However, the new confounded types of attacks make traditional ones increasingly difficult to distinguish. The most proficient method for distinguishing between these attacks is to utilize information mining and machine learning strategies. In these sorts of methods, a lot of information is gathered in a reproduced environment or genuine attack; then, at that point, analysts separate key features from crude information. From that point forward, ML-based methods are used to create the detection model, and the model's performance is evaluated to determine whether the method is appropriate for detecting DDoS attacks. A rundown of standard machine learning algorithms for DDoS detection is accessible as follows:

- **Random Forest:** This is a decision tree-based algorithm that is also used primarily for the classification of datasets and some other tasks, which are carried out by constructing many decision trees from a training set that was randomly selected. It combines the votes from numerous decision trees to determine the object's exact class. A separate loss for each class label for each observation is made for a multiclass classification.
- **AdaBoost:** This machine-learning approach works by assigning weights to observations. Cases are given weightage based on their identification. New weak learners are introduced sequentially, allowing them to focus on increasingly challenging patterns. Boosting has two primary challenges.
- How can the training set be adjusted so the weak classifier can learn from it?
- How do we make a strong classifier out of the poor classifiers created during training?

In 1995, the Adaboost (adaptive boosting) strategy to address these difficulties was introduced by Freund and Schapire, which worked by altering weightage without requiring any prior knowledge of learner learning. The algorithm can solve many of the early boost method's practical issues and adjust voting weights. [14].

- **XGBoost:** Xtreme Gradient Boosting is a technique that employs an iterative strategy to improve model accuracy by reducing errors. The augmentation of gradient boosted decision trees (GBM) provides adaptable, easy, and DGBL results (Distributed gradient boosting library). XGBoost may improve performance and efficiency for classification, regression, and parallel computing issues. It tends to be utilized as an implicit system with the assistance of SageMaker to accomplish better versatility and a way to deal with more further developed construction, for example, K-overlay cross approval, as you can

alter your preparation scripts.

The three essential components of XGBoost are a loss function to assess model prediction percentage, a weak learner to categorize data while guessing incorrectly, and an additive function to reduce the loss function's value through repeated and sequential processing [15] [16].

• **Naïve Bayes:** This basic Bayes' Theorem-based probabilistic classifier works well with huge datasets. When the characteristics in the datasets are independent of one another, the Naive Bayes model is simple to construct. The classifier is quick and unaffected by irrelevant characteristics. In binary scenarios, such as when the goal of classification is to determine if arriving packets are DDoS or not, the Nave Bayes algorithm works exceptionally well. The model learns by computing the probability of training data. The naive Bayes classifier simplifies learning by assuming that attributes are independent of class. The class conditional mutual information between characteristics is defined as component dependence and does not affect naive Bayes accuracy. On the other hand, the amount of information about the class is lost because the independence assumption is a better predictor of the success of Naive Bayes classification [17].

• **SVM:** To solve the problems of regression and classification, a supervised ML approach is known as SVM (Support Vector Machine). However, it is often utilized in grading. A hyperplane in N-dimensional space is used to classify the data points (N is the number of characteristics). With roots in Statistical Learning Theory (SLT) and optimization methods, support vector machines have evolved into powerful ML problem-solving tools with finite training points, overcoming some traditional challenges, including over-fitting, the curse of dimensionality, and so on. Implementation techniques and theoretical foundations for SVMs have been established due to several appealing features, including good generalization abilities, enticing mathematical representations, geometrical explanations, and promising empirical performance. SVMs are gaining popularity and development at a rapid pace [18].

• **KNN:** It is a simple and easily applicable supervised ML-based algorithm that is best for problem-solving in regression and classification. The k-Nearest-Neighbors (kNN) technique is a basic yet successful classification method. In many cases, kNN is a basic yet effective non-parametric classification algorithm. To classify an x number of data records, the k closest neighbours are collected, forming a neighbourhood of x. The category of x is usually determined by a majority vote amongst collected data in the neighborhood, with or without distance-based weighting [19]. We use the distance metric to compare and find relativity between existing K examples of a training dataset and the upcoming input.

3. LITERATURE REVIEW

Jie Xue et al. [20] presented a model that utilized 7 features extracted from the user's application layer to distinguish between regular users and bots. Their primary objective was to identify and differentiate bot behaviour. They incorporated a one-class SVM algorithm into their collected database and concluded that their model effectively detected denial of service attacks at the application layer. Shrikhand Wankhede et al. [21] conducted an analysis and proposed a detection model based on machine learning. In addition to utilizing machine learning techniques, the researchers incorporated neural networks to enhance the accuracy of their model by optimizing a set number of parameters. They employed the 500-tree random forest algorithm to train their model on half of the dataset and achieved an impressive accuracy of 99.95 percent. It is important to note that an earlier version of the CIC IDS was utilized in this study.

Jakula et al. [22] researched detecting DDoS attacks using supervised and unsupervised machine learning approaches previously explored by other researchers. Their study introduced a novel approach for identifying DDoS attacks, incorporating a new parameter called P (A), which significantly contributed to their research. They enhanced their algorithm's accuracy and performance by determining the optimal number of hyperparameters. The parameter P (A) was utilized as a benchmark to make informed decisions during model training. Through experimentation on the "NSL-KDD" dataset, they found that certain algorithms such as Naive Bayes, Gradient Boost, and Random Forest yielded the best precision and training time results.

Dong et al. [19] introduced two innovative algorithms, DDADA and DDAML, based on the concepts of K-Nearest Neighbors (KNN) and attack intensity. They collected datasets from a simulation environment to generate DDoS traffic. Then they evaluated the performance of their proposed algorithms in comparison to traditional AI algorithms such as KNN, SVM, and Naïve

Bayes. By examining the ROC curve results, they discovered that their new algorithms outperformed the existing ones, showcasing improved detection capabilities for DDoS attacks.

Patil et al. [23] offered a DDoS location framework in light of solicitation package header connections. For testing purposes, researchers used the Caida dataset and the real separated information that utilized the ideas of modularity, SVM calculation, and entropy. Researchers concluded that higher precision can be achieved by adding the concept of entropy to UDP association and modularity to TCP association.

R. Boss et al. [24] contrasted various calculations for conventional learning and crossover strategies. They tried these calculations in the DARPF and KDDcup99 informational collections, observing that decision trees and C-Mean work well compared to other researchers' work. There is a 98.7% chance that the Fuzzy C-Mean calculation can tell if there is DDoS traffic with 0.15 seconds of identifying season.

Doshi et al. [25] were encouraged to develop new ways to automatically detect IoT consumer traffic attacks. Researchers showed that using IoT-specific network behaviour to guide feature selection can lead to the acquisition of accurate attack detection using a range of ML methods, which include neural networks. These findings suggest that using cheap machine-learning strategies and protocol-agnostic-based traffic data by home gateways or other internal network routers may automatically identify IoT device resources for DDoS attacks. Machine learning is extensively utilized to tackle various challenges in diverse domains [26-30]. The versatility of machine learning is evident in its application across fields such as cybersecurity, agriculture, energy management, and more.

This research aims to create a machine-learning model capable of swiftly and accurately identifying DDoS attacks. The study delves into optimizing dimension reduction and feature selection techniques for DDoS detection using the CICDDoS2019 dataset. Rigorous testing determines the optimal algorithm for proficient and efficient detection of DDoS attacks.

4. METHODOLOGY

4.1. Proposed Study

Initially, an optimal machine learning (ML) algorithm commonly employed for Dos/DDoS attack detection is identified through a comprehensive review of related works by various authors. Utilizing observed data, a model is formulated to assess the effectiveness and execution speed of various algorithms. The "CICDDoS2019" dataset is utilized for training and testing the models. The model encompasses multiple phases, with preprocessing involving the extraction of an effective feature set for model training. Subsequent testing evaluates the performance of different algorithms to determine the optimal solution. Ultimately, the study identifies crucial features within the CICDDoS2019 dataset that significantly influence accurate DDoS predictions.

4.2. Dataset Preparation

This research employs the latest available dataset, named CICDoS2019, for studying DDoS attacks, addressing the limitations of previous datasets. This dataset encompasses two types of DDoS attacks: reflection-based and exploitation-based, utilizing TCP/UDP protocols at the application layer. Notably, the dataset introduces a novel classification method with new attack types, which is a significant advantage. Diverse DDoS attack categories, including "WebDDoS," "SNMP," "NTP," "DNS," and more, are categorized, while regular traffic is labeled as "BENIGN." These labeled network traffic data and associated features are stored in an accessible CSV file. The traffic features are extracted using CICFlowMeter-V3.

4.3. Pre-processing of Data

Direct utilization of the CICDDoS2019 dataset for model training and testing is hindered due to its large size (around 3 GB), necessitating robust processing capabilities. Consequently, only the Portmap section of the dataset is employed, focusing on a streamlined set of essential features. Data preprocessing is conducted using Google Colab, a web-based Python coding environment developed by Google, which is highly regarded among data analysts and ML researchers. The preprocessing phase involves the utilization of various libraries, including Pandas, Numpy, and Scikit-learn, to accomplish tasks effectively.

4.4. Dimension Reduction of Dataset

The vital phases for proposed dimension reduction model are concluded as:

- **Replace Infinite and Null Values:** To prevent the mixing of SI and CGS units, like using amperes for current and oersteds for magnetic field, which often leads to dimensional

imbalances in equations, it's essential to avoid such combinations. If mixed units are necessary, ensure clear unit definitions for each quantity in the equation.

- **Eliminate Remaining Null Values:** String-type null values in the dataset can disrupt experiments. Since these values can't be replaced, they are removed from the dataset, which contains a sufficient number of records.
- **Encode Categorical Columns:** As part of data preprocessing, convert all string values in the dataset into numerical values. This encoding ensures precise experimental outcomes.
- **Discard Zero Variance Columns:** Columns with zero variance are eliminated since they contain identical values, offering no impact on results.
- **Prune Low Correlation Columns:** Removing insignificant features is crucial for dimension reduction, preventing overfitting, and enhancing the model's execution speed. This step involves eliminating columns with low correlation.

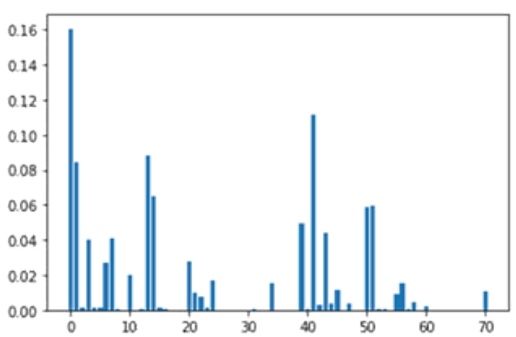

Figure 2: Complete Features and their Importance

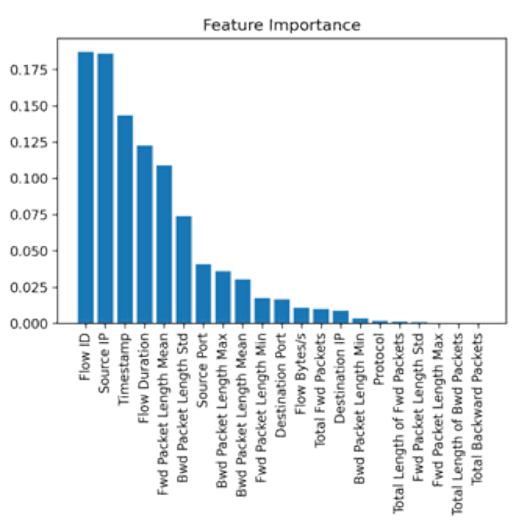

Figure 3: Top 20 Important Features

Figure 2 visually depicts the significance of all the features extracted from the dataset during the data preprocessing stage. Figure 3 presents the top 20 significant attributes influencing network class prediction. The X-axis delineates these key features, which hold sway over both the dataset and model performance. The Y-axis represents the proportional impact of each feature. Flow ID and Source IP are the most influential factors in DDoS attack detection systems.

4.5. Machine Learning Based Algorithms

Numerous studies by fellow researchers focusing on detecting DDoS attacks through machine learning techniques are explored. Prominent among these techniques are Random Forest, SVM, Naive Bayes, KNN, XGBoost, and AdaBoost—widely recognized ML algorithms known for their efficacy in identifying distributed denial of service attacks. The proposed model is subjected to training and testing utilizing these algorithms to determine the most proficient performer. Subsequent sections delve into the assessment metrics used for evaluation.

4.6. Evaluation

Several evaluation metrics are employed to compare the performance of ML algorithms and extract valuable features:

- **Classification Accuracy:** This measures the proportion of accurate predictions out of the total predictions made. However, solely relying on accuracy might be inadequate due to potential imbalances in the dataset.
- **F1-Score:** The F1-Score combines precision and recall into a single metric, offering a comprehensive assessment of false positives and false negatives. It is a more robust testing measure.
- **Training Time:** This metric gauges a model's efficiency and speed during training.
- **Feature Importance:** This assesses the correlation between each feature and its predicted labels, aiding in identifying influential attributes.

5. RESULTS AND ANALYSIS

This section uses various algorithms to present the outcomes of the comparative analysis between the proposed model and the CICDDoS2019 dataset. The results are meticulously examined to determine the optimal algorithm for DDoS attack detection. As depicted in Figure 4, the performance of all algorithms is evaluated, with Naïve Bayes being the exception, achieving an

F1-score of 0.6802, indicating true positive predictions. However, Naïve Bayes displays limitations in DDoS attack detection due to its elevated rate of false positives. This implies that Naïve Bayes tends to misclassify benign traffic as malicious, potentially hindering effective attack prevention.

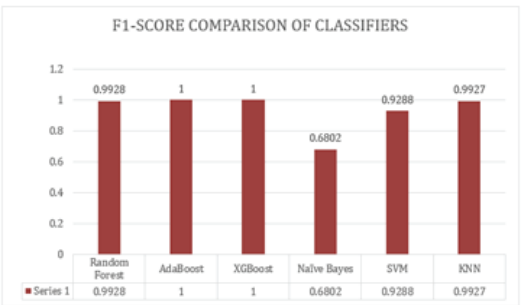

Figure 4: F1-Score Comparison of All Classifiers

In Figure 6, the outcomes reveal that Naïve Bayes excels in terms of training time, requiring a mere 0.414 seconds to complete model training. Following that, Random Forest concluded training in 4.645 seconds, while XGBoost and AdaBoost took 10.325 and 14.853 seconds, respectively. In contrast, KNN and SVM exhibited the lengthiest training times, consuming 97.795 and 1711.976 seconds, respectively.

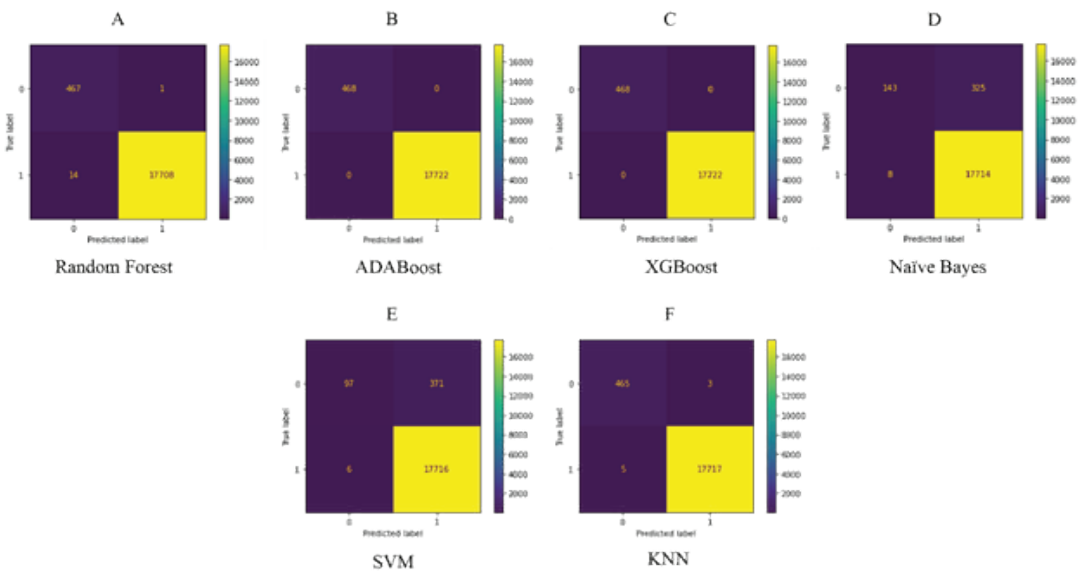

Fig.ure 5: Confusion Matrix of all algorithms

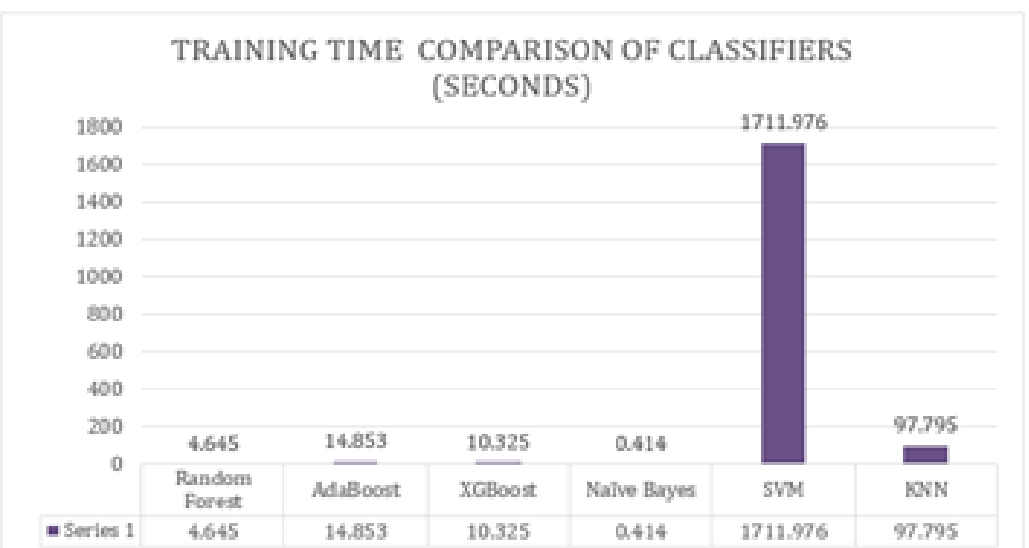

Figure 6: Execution Time Comparison of All Classifiers

The results from TABLE 1 highlight significant achievements, with AdaBoost and XGBoost showcasing exceptional performance at 100% accuracy, a notable milestone for DDoS attack detection. Following closely, KNN achieves a respectable accuracy of 99.93%, demonstrating its effectiveness. Random Forest secures the third position with 99.928% accuracy, establishing its capability for DDoS attack detection. Conversely, SVM and Naïve Bayes exhibit the lowest accuracies compared to other algorithms, registering 99.334% and 98.16%, respectively, indicating inadequate performance for DDoS attack detection.

The evaluation techniques for each algorithm used in this model are outlined in Table I. These techniques encompass accuracy as a percentage, F1-score, and training time measured in seconds.

Table 1: Evaluation Metrix

ALGORITHMS	EVALUATION		
	ACCURACY (percentage)	F1-SCORE	TRAINING TIME (seconds)
Random Forest	99.928	0.9928	4.645
AdaBoost	100	1	14.853
XGBoost	100	1	10.325
Naïve Bayes	98.16	0.6802	0.414
SVM	99.334	0.9288	1711.976
KNN	99.93	0.9927	97.795

The confusion matrix provides a concise summary of predicted outcomes in a classification scenario. It tabulates correct and incorrect predictions, assigning them to specific classes and quantifying their occurrences. This aids in the model training phase and subsequent evaluation of performance. A comparative analysis is conducted among different algorithms to detect DDoS attacks efficiently. Each classifier undergoes training and is evaluated, leading to the presentation of confusion matrices for each algorithm in this section. The overarching aim of the evaluation is to pinpoint the optimal classifier for the problem-solving model. Confusion matrices for each classifier are depicted in Figure 5 (A-F). Remarkably, all the considered ML algorithms, except Naïve Bayes, surpass our experimentation's expectations regarding accuracy, efficacy, and efficiency. It was observed that an imbalanced dataset skews Naïve Bayes' accuracy, causing it to classify more attacks than the count of harmless records predominantly. F1-score and recall metrics play a crucial role in comprehending false positive classifications.

Naïve Bayes' underperformance stems from its assumption of feature independence, grounded in Bayes' theorem, which contrasts with the dataset's actual feature interdependencies. Evaluation metrics, including F1-score, accuracy percentage, and training time in seconds for each algorithm utilized in this research experiment, are outlined in the Table I.

6. CONCLUSION AND FUTURE DIRECTIONS

This research introduces a robust DDoS detection model utilizing popular ML algorithms, including Random Forest, AdaBoost, XGBoost, Naïve Bayes, SVM, and KNN. The CICDDoS2019 dataset is categorized into "harmless" and "harmful" classes. All algorithms except Naïve Bayes effectively classify the dataset into these classes, displaying exceptional accuracy, efficiency, and training speed.

XGBoost and AdaBoost shine among these algorithms, exhibiting superior accuracy and F1 scores. Their training times show minor disparities. The proposed model achieves high accuracy and swiftness and identifies the top 20 influential features within the CICDDoS2019 dataset. By discarding irrelevant attributes, the research enhances accuracy, speed, and training efficiency. As future work, the model can be refined further by incorporating newer DDoS attack datasets and exploring hybrid mechanisms for improved detection capabilities. Building upon the current research, several promising avenues for further exploration and enhancement emerge:

- **Integration of Hybrid Approaches:** To elevate the model's detection capabilities, combining the strengths of multiple algorithms or integrating hybrid mechanisms could prove fruitful. Hybrid models that synergize the unique advantages of different algorithms may result in even more accurate and efficient DDoS attack detection.
- **Incorporation of Real-Time Analysis:** Expanding the model's applicability to real-time analysis can enhance its practical utility. The model could play a pivotal role in proactive threat mitigation by continuously monitoring network traffic and swiftly identifying potential DDoS attacks in real time.
- **Leveraging Advanced Feature Engineering:** Exploring advanced feature engineering techniques can further refine the model's predic-

tive power. Incorporating domain-specific knowledge and extracting more relevant features may contribute to more nuanced and accurate predictions.

- **Utilization of Deep Learning Techniques:** Integrating deep learning methodologies, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), can provide the model with enhanced capabilities to capture complex patterns and relationships within network data, potentially leading to improved accuracy.
- **Evaluation with Diverse Datasets:** Testing the model against a wider range of diverse datasets containing various types of DDoS attacks and network scenarios can validate its robustness and generalizability. Incorporating data from different sources and attack scenarios can ensure its effectiveness across different contexts.
- **Optimization for Scalability:** Optimizing the model's scalability becomes crucial as network infrastructures continue to grow in complexity and scale. Developing strategies to handle large-scale networks and big data environments while maintaining accuracy and efficiency will be essential.
- **Continuous Model Enhancement:** The model should undergo ongoing refinement and updates to stay relevant in the face of evolving attack strategies. Regularly updating the model with new attack patterns, techniques, and datasets ensures its effectiveness against emerging threats.
- **Collaboration and Knowledge Sharing:** Collaborating with other researchers, practitioners, and industry experts can lead to innovative insights and solutions. Sharing knowledge and experiences can drive the advancement of DDoS detection techniques.

These future directions can further elevate the DDoS detection model's capabilities, making it a more potent tool in safeguarding networks against the evolving landscape of cyber threats.

REFERENCES

[1] M. A. Khan and K. Salah, "IoT security: Review, blockchain solutions, and open challenges," *Futur. Gener. Comput. Syst.,* vol. 82, pp. 395–411, doi: 10.1016/j.future.2017.11.022, 2018.

[2] M. Al-Sarem et al., "An aggregated mutual information based feature selection with machine learning methods for enhancing iot botnet attack detection," *Sensors,* vol. 22, no. 1, doi: 10.3390/s22010185, 2022.

[3] J. Li et al., "RTED-SD: A Real-Time Edge Detection Scheme for Sybil DDoS in the Internet of Vehicles," *IEEE Access,* vol. 9, pp.11296-11305, doi: 10.1109/ACCESS.2021.3049830, 2021.

[4] A. Aljuhani, "Machine Learning Approaches for Combating Distributed Denial of Service Attacks in Modern Networking Environments," *IEEE Access,* vol. 9, pp. 42236–42264, doi: 10.1109/ACCESS.2021.3062909, 2021.

[5] T. Mahjabin et al., "A survey of distributed denial-of-service attack, prevention, and mitigation techniques," *Int. J. Distrib. Sens. Networks,* vol. 13, no. 12, doi: 10.1177/1550147717741463, 2017.

[6] E. SÖĞÜT et al., "Farklı Türde Dağıtık Hizmet Dışı Bırakma Saldırılarının Tespiti," *Gazi Üniversitesi Fen Bilim. Derg. Part C Tasarım ve Teknol.,* vol.9, no.1, doi: 10.29109/gujsc.840126, March, 2021.

[7] S. N. Thanh et al., "Survey on botnets: Incentives, evolution, detection and current trends," *Futur. Internet,* vol. 13, no. 8, doi: 10.3390/fi13080198, 2021.

[8] M. A. Khan et al., "HCRNNIDS: Hybrid Convolutional Recurrent Neural," *Multidiscip Digit. Publ Inst.,* 2021.

[9] S. Chen et al., "A vision of IoT: Applications, challenges, and opportunities with China Perspective," *IEEE Internet Things J.,* vol. 1, no. 4, pp. 349–359, doi: 10.1109/JIOT.2014.2337336, 2014.

[10] R. Santos et al., "Machine learning algorithms to detect DDoS attacks in SDN," *Concurr. Comput. Pract. Exp.,* vol. 32, no. 16, pp. 1–14, doi: 10.1002/cpe.5402, 2020.

[11] A. E. Bouaouad et al., "The key layers of IoT architecture," Proc. 2020 *5th Int. Conf. Cloud Comput. Artif. Intell. Technol. Appl. CloudTech 2020,* pp. 20–23, doi: 10.1109/CloudTech49835.2020.9365919, 2020.

[12] P. Lou et al., "An anomaly detection method for cloud service platform," *ACM Int. Conf. Proceeding Ser.,* pp. 70–75, doi: 10.1145/3340997.3341005, 2019.

[13] M. A. Underwood et al., "nternet of things: Toward smart networked systems and societies," *Appl. Ontol.,* vol. 10, no. 3–4, pp. 355–365, doi: 10.3233/AO-150153, 2015.

[14] K. Sha et al., "A survey of edge computing-based designs for IoT security," *Digit. Commun. Networks,* vol. 6, no. 2, pp. 195–202, doi: 10.1016/j.dcan.2019.08.006, 2020.

[15] C. L. Zhong et al., "Study on the IOT architecture and gateway technology," *Proc. - 14th Int. Symp. Distrib. Comput. Appl. Business, Eng. Sci. DCABES 2015,* pp. 196–199, doi: 10.1109/DCABES.2015.56, 2016.

[16] I. Grønbæk, "Architecture for the Internet of Things (IoT): API and interconnect," *Proc. - 2nd Int. Conf. Sens. Technol. Appl., SENSORCOMM 2008, Incl. MESH 2008 Conf. Mesh Networks; ENOPT 2008 Energy Optim. Wirel. Sensors Networks, UNWAT 2008 Under Water Sensors Syst.,* pp. 802–807, doi: 10.1109/SENSORCOMM.2008.20, 2008.

[17] Soma Bandyopadhyay et al., "Role Of Middleware For Internet Of Things: A Study," *Int. J. Comput. Sci. Eng. Surv.,* vol. 2, no. 3, pp. 94–105, doi: 10.5121/ijcses.2011.2307, 2011.

[18] V. Jakkula, "Tutorial on Support Vector Machine (SVM)," *Sch. EECS, Washingt. State Univ.,* pp. 1–13, *[Online],* Available: http://www.ccs.neu.edu/course/cs5100f11/resources/jakkula.pdf, 2011.

[19] S. Dong and M. Sarem, "DDoS Attack Detection Method Based on Improved KNN with the Degree of DDoS Attack in Software-Defined Networks," *IEEE Access,* vol. 8, pp. 5039–5048, doi: 10.1109/ACCESS.2019.2963077, 2020.

[20]. J. Xue et al., "Bound maxima as a traffic feature under DDOS flood attacks," *Math. Probl. Eng.,* vol. 2012, no.1, pp. 1–5, doi: 10.1155/2012/419319, 2012.

[21] S. Wankhede and D. Kshirsagar, "DoS Attack Detection Using Machine Learning and Neural Network," *Proc. - 2018 4th Int. Conf. Comput. Commun. Control Autom. ICCUBEA 2018, pp. 1-5* doi: 10.1109/ICCUBEA.2018.8697702, 2018.

[22] M. A. Prriyadarshini and S. R. Devi, "Detection of DDoS attacks using supervised learning technique," *J. Phys. Conf. Ser.,* vol. 1716, no. 1, doi: 10.1088/1742-6596/1716/1/012057, 2021.

[23] N. V. Patil et al., "E-Had: A distributed and collaborative detection framework for early detection of DDoS attacks," *J. King Saud Univ. - Comput. Inf. Sci.,* doi: 10.1016/j.jksuci.2019.06.016, 2019.

[24] R. Bose, "Virtual labs project: A paradigm shift in internet-based remote experimentation," *IEEE Access,* vol. 1, pp. 718–725, doi: 10.1109/ACCESS.2013.2286202, 2013.

[25] R. Doshi et al., "Machine learning DDoS detection for consumer internet of things devices," *Proc. - 2018 IEEE Symp. Secur. Priv. Work. SPW 2018, no. Ml*, pp. 29–35, doi: 10.1109/SPW.2018.00013, 2018.

[26] M. Sajawal et al., "A Predictive Analysis of Retail Sales Forecasting using Machine Learning Techniques," L*ahore Garrison University Research Journal of Computer Science and Information Technology,* vol. 6, no. 4, pp. 33-45, https://doi.org/10.54692/lgurjcsit.2022.0604399, 2022.

[27] M. U. Ashraf et al., "Comparative Analysis of Machine Learning Techniques for Predicting Air Pollution," L*ahore Garrison University Research Journal of Computer Science and Information Technology,* vol. 6, no. 2, pp. 40-54, https://doi.org/10.54692/lgurjcsit.2022.0602270, 2022.

[28] M. Ahmed et al., "A Machine Learning-Based Tool for Performance Optimization of Parallel SPMV Computations Using Block CSR," *Appl. Sci.*, vol. 12, no. 14, pp. 7073. https://-doi.org/10.3390/app12147073, 2022.

[29] S. Usman et al., "ZAKI: A Smart Method and Tool for Automatic Performance Optimization of Parallel SpMV Computations on

Distributed Memory Machines," *Mobile Netw Appl (2019)*. https://-doi.org/10.1007/s11036-019-01318-3, 2023.

[30] S. Usman et al., "ZAKI+: A Machine Learning Based Process Mapping Tool for SpMV Computations on Distributed Memory Architectures," *in IEEE Access,* vol. 7, pp. 81279-81296, doi: 10.1109/ACCESS.2019.2923565, 2019.

[31] M. U. Ashraf, "A Survey on Data Security in Cloud Computing Using Blockchain: Challenges, Existing-State-Of-The-Art Methods, And Future Directions," *Lahore Garrison University Research Journal of Computer Science and Information Technology,* vol. 5, no. 3, pp. 15-30, 2021.

[32] M. U. Ashraf et al., "A Survey on Emotion Detection from Text in Social Media Platforms," *Lahore Garrison University Research Journal of Computer Science and Information Technology,* vol. 5, no. 2, pp. 48-61, 21 Jun 2021.

[33] K. Shinan et al., "Machine learning-based botnet detection in software-defined network: a systematic review," *Symmetry* vol. 13, no. 5, pp. 866, 2021.

[34] A. Hannan et al., "A decentralized hybrid computing consumer authentication framework for a reliable drone delivery as a service," *Plos one,* vol. 16, no, 4, pp. e0250737, 2021.

[35] S. Fayyaz et al., "Solution of combined economic emission dispatch problem using improved and chaotic population-based polar bear optimization algorithm," *IEEE Access,* vol. 9, pp. 56152-56167, 2021.

[36] I. Hirra et al., "Breast cancer classification from histopathological images using patch-based deep learning modeling," *IEEE Access*, vol. 9, pp. 24273-87, 2 Feb 2021.

[37] M. U. Ashraf et al., "AAP4All: An Adaptive Auto Parallelization of Serial Code for HPC Systems," *INTELLIGENT AUTOMATION AND SOFT COMPUTING,* vol. 30, no. 2, pp.615-39, 1 Jan 2021.

[38] T. Hafeez et al., "EEG in game user analysis: A framework for expertise classification during gameplay," *Plos one,* vol. 16, no. 6, pp. e0246913, 18 Jun 2021.

[39] N. Siddiqui et al., "A highly nonlinear substitution-box (S-box) design using action of modular group on a projective line over a finite field," *Plos one*, vol. 15, no.11,pp. e0241890, 12 Nov 2020 .

[40] M. U. Ashraf et al., "Detection and tracking contagion using IoT-edge technologies: Confronting COVID-19 pandemic," *2020 international conference on electrical, communication, and computer engineering (ICECCE),* pp. 1-6, IEEE, 2020.

[41] K. Alsubhi et al., "MEACC: an energy-efficient framework for smart devices using cloud computing systems," *Frontiers of Information Technology & Electronic Engineering,* vol. 21, no. 6, pp. 917-930, 2020.

[42] S. Riaz et al., "A Comparative Study of Big Data Tools and Deployment PIatforms.," *In 2020 International Conference on Engineering and Emerging Technologies (ICEET),* pp. 1-6, IEEE, 22 Feb 2020.

[43] M. U. Ashraf et al., "Empirical investigation: performance and power-consumption based dual-level model for exascale computing systems," *IET Software*, vol.14, no. 4, pp. 319-27, 27 Jul 2020.

[44] M. U. Ashraf et al., "IDP: A Privacy Provisioning Framework for TIP Attributes in Trusted Third Party-based Location-based Services Systems," *International Journal of Advanced Computer Science and Applications (IJACSA),* vol. 11, no. 7, pp. 604-617, 2020.

[45] A. Manzoor et al., "Inferring Emotion Tags from Object Images Using Convolutional Neural Network," *Applied Sciences,* vol. 10, no.15, pp. 5333, 2020.

[46] K. Alsubhi et al., "A Tool for Translating sequential source code to parallel code written in C++ and OpenACC," *In 2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA)*, pp. 1-8, IEEE, 2019.

[47] M. U. Ashraf et al., "H2E: A Privacy

Provisioning Framework for Collaborative Filtering Recommender System," *International Journal of Modern Education and Computer Science*, vol.11, no. 9, pp. 1,1 Sep 2019.

[48] M. U. Ashraf et al., "A Roadmap: Towards Security Challenges, Prevention Mechanisms for Fog Computing," *In 2019 International Conference on Electrical, Communication, and Computer Engineering (ICECCE),* pp. 1-9, IEEE, 24 Jul 2019.

[49] M. U. Ashraf et al., "State-of-the-art Challenges: Privacy Provisioning in TPP Location Based Services Systems," *International Journal of Advanced Research in Computer Science (IJARCS)*, vol. 10, no. 2, pp. 68-75, 20 Apr 2019.

[50]. M. U. Ashraf et al., "Improving Performance In Hpc System Under Power Consumptions Limitations," *International Journal of Advanced Research in Computer Science*, vol. 10, no. 2, Mar 2019.

[51] R. Javed et al., "Prediction and monitoring agents using weblogs for improved disaster recovery in cloud," *Int. J. Inf. Technol. Comput. Sci.(IJITCS),* vol.11, no. 4, pp. 9-17, 2019.

[52] M. Ali et al., "Prediction of Churning Behavior of Customers in Telecom Sector Using Supervised Learning Techniques," *In 2018 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*, pp. 1-6, IEEE, 2018.

[53] M. U. Ashraf et al., "Performance and power efficient massive parallel computational model for HPC heterogeneous exascale systems," *IEEE Access*, 6, pp. 23095-107, 9 Apr 2018.

[54] M. U. Ashraf et al., "Toward exascale computing systems: An energy efficient massive parallel computational model," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 2.

[55] M. U. Ashraf., "Provisioning quality of service for multimedia applications in cloud computing," *Int. J. Inf. Technol. Comput. Sci.(IJITCS)*, vol. 10, no. 5, pp.40-7, 2018.

[56] M. U. Ashraf et al., "Efficient Execution of Smart City's Assets Through a Massive Parallel Computational Model," *InInternational Conference on Smart Cities, Infrastructure, Technologies and Applications,* pp. 44-51, *Springer,* Cham, 27 Nov 2017.

[57] M. S. Alrahhal et al., "AES-route server model for location based services in road networks," *International Journal Of Advanced Computer Science And Applications,* vol. 8, no. 8, pp. 361-368, 2017.

[58] M. U. Ashraf et al., "High performance 2-D Laplace equation solver through massive hybrid parallelism," *In 2017 8th International Conference on Information Technology (ICIT),* pp. 594-598, IEEE, 17 May 2017.

[59] M. Mumtaz et al., "Iteration Causes, Impact, and Timing in Software Development Lifecycle: SLR," *IEEE Access,* vol. 10, pp. 65355-65375, 2022.

[60] M. Ahmad et al., "Efficient Liver Segmentation from Computed Tomography Images Using Deep Learning," *Computational Intelligence and Neuroscience 2022,* no. 1, pp.2665283, 2022.

[61] H. Tufail et al., "The Effect of Fake Reviews on e-Commerce During and After Covid-19 Pandemic: SKL-Based Fake Reviews Detection," *IEEE Access,* vol. 10, pp. 25555-25564, 2022.

Khalid et al. LGURJCSIT 2024

LGU Research Journal of Computer Science & IT

ISSN: 2521-0122 (Online)
ISSN: 2519-7991 (Print)

doi: 10.54692/lgurjcsit.2024.082573

Vol (8): Issue (2), April – June 2024

Strategic Customer Segmentation: Harnessing Machine Learning For Retaining Satisfied Customers

Hira Khalid[1*], Shazia Saqib[1, 2], Muhammad Junaid Asif [2], Deshinta Arrova Dewi[2]
[1]Faculty of IT and Computer Sciences (FoIT & CS), University of Central Punjab, Lahore, Pakistan.
[2]Faculty of Data Science and Information Technology, INTI International University, Malaysia.

Email: hkhalid99@gmail.com

ABSTRACT:

This research paper explores the burgeoning field of machine learning and its application in strategic customer segmentation within the aviation industry. Leveraging the Airline Passenger dataset, this study assesses the potential of various machine learning classifiers to enhance customer retention by effectively segmenting satisfied customers. Our methodology involves a comparative analysis of five machine learning classifiers: Random Forest, K-Nearest Neighbors (KNN), Decision Tree, Naive Bayes, and Artificial Neural Network (ANN). Each classifier is rigorously tested and evaluated based on key performance metrics, including accuracy, precision, recall, and F1-score.

The results indicate a diverse range of classifier effectiveness. Notably, the Random Forest classifier outperforms others with outstanding metrics: accuracy, precision, recall, and F1-score of 0.96. Decision Tree follows closely, achieving high performance with a score of 0.95 across all metrics. Naive Bayes and ANN demonstrate respectable performance, with accuracy scores of 0.86 and 0.90, respectively. In contrast, KNN presents lower but consistent performance, with all metrics at 0.75. These quantitative findings highlight the nuanced performance differences among classifiers, emphasizing the critical role of algorithm selection in achieving precise customer segmentation.

This study provides significant insights into the application of machine learning for strategic customer retention in the aviation sector, presenting practical implications for airlines aiming to optimize their segmentation strategies and retain satisfied customers. By showcasing the varying performances of different classifiers, this research contributes to the broader discourse on integrating machine learning into customer-centric strategies, ultimately aiding airlines in engaging and retaining their customer base more effectively.

KEYWORDS: Customer Segmentation, Machine Learning, Aviation Industry, Retention Strategies .

1. INTRODUCTION

In today's competitive business landscape, understanding and retaining satisfied customers are paramount for sustainable growth and success. With the advent of machine learning (ML) techniques, businesses have unprecedented opportunities to delve deeper into customer behaviors, preferences, and patterns. Lewaaelhamd et al. [1] Instead of analyzing the entire customer database, it's more effective to categorize customers. by characteristics like age or location and tailor marketing campaigns to each segment for personalized, relevant offers. One such powerful application is strategic customer segmentation, which enables businesses to tailor their strategies, products, and services to different customer segments, enhancing customer satisfaction and loyalty. Customer segmentation's transformative potential in marketing and decision-making is advocated by Thakkar et al. [2]. Based on behav-

ior and demographics, segmentation allows for customized methods that improve product creation and marketing effectiveness. Division improves customer comprehension, sharpens offerings, and boosts profitability.

This research paper focuses on integrating machine learning algorithms into strategic customer segmentation processes to optimize customer retention efforts. The objective is to explore how various ML classifiers perform in identifying and categorizing customer segments based on their satisfaction levels. By harnessing the capabilities of ML, businesses can gain actionable insights into their customer base, enabling targeted and personalized marketing initiatives, product recommendations, and customer service enhancements. By using this strategy, companies in the tourism sector can also effectively target important client segments, increasing revenue and profitability Vikram et al. [3].

By using this strategy, companies in the tourism sector can also effectively target important client segments, increasing revenue and profitability. The experimentation phase of this study involved the evaluation of several ML classifiers to determine their effectiveness in customer segmentation tasks. Metrics such as accuracy, precision, recall, and F1-score were employed to assess the performance of each classifier comprehensively. The results unveiled a nuanced landscape of classifier effectiveness, providing insights into their suitability for strategic customer segmentation purposes.

Among the classifiers evaluated, Random Forest emerged as a standout performer, showcasing remarkable accuracy, precision, recall, and F1-score. Its robust performance underscores its potential as a reliable tool for identifying and segmenting satisfied customers effectively Conversely, K-Nearest Neighbors (KNN) exhibited lower but consistent performance across metrics, indicating its moderate effectiveness in customer segmentation tasks.

Decision Tree, another commonly used classifier, demonstrated strong performance comparable to Random Forest, highlighting its utility in strategic customer segmentation efforts. While Random Forest and Decision Tree demonstrated top-tier performance, other classifiers such as Naïve Bayes and Artificial Neural Networks (ANN) also presented respectable results. Although Naive Bayes and ANN exhibited slightly lower accuracy scores than Random Forest and Decision Tree, their performance across precision, recall, and F1-score metrics remained consistent. These findings suggest that while certain classifiers may outperform others in specific metrics, their overall effectiveness in strategic customer segmentation depends on various factors such as dataset characteristics, feature selection, and model tuning.

Integrating machine learning techniques into strategic customer segmentation offers several critical advantages for businesses. Firstly, ML algorithms can rapidly analyze vast amounts of customer data, allowing businesses to identify meaningful patterns and segments efficiently. Secondly, ML-based segmentation enables dynamic and adaptive customer categorization, accommodating shifts in customer preferences and behaviors over time. Thirdly, by leveraging ML-driven insights, businesses can tailor their marketing strategies and offerings to specific customer segments, enhancing overall customer satisfaction and loyalty.

In conclusion, strategic customer segmentation powered by machine learning presents a compelling approach for businesses seeking to retain satisfied customers in today's competitive marketplace. By leveraging the capabilities of ML algorithms, businesses can gain deeper insights into their customer base, enabling targeted and personalized strategies for customer retention and satisfaction. The findings from this research contribute to the growing body of knowledge on the integration of ML in marketing and customer relationship management, paving the way for more effective and data-driven approaches to customer segmentation and retention strategies.

Tabianan et al. [4] to understand how new technologies help marketers better fulfil the needs and wants of customers. This project also understands how, in situations where technology has the power to affect and alter consumer behavior, marketers may better satisfy the needs and wants of their target audience thanks to evolving technologies.

As we move forward, the next chapter will delve into the existing literature on strategic customer segmentation, providing insights into the theoretical foundations, previous research findings, and emerging trends in this field. Through a comprehensive review of relevant studies, we aim to contextualize our research within the broader academic discourse and identify gaps and opportunities for further investigation.

2. LITERATURE REVIEW

Throughout history, airlines have strived to

and retain their customers, prompting extensive research in the field. As airports and aircraft have evolved, so too have airlines' strategies for understanding customer needs, building relationships, and fostering loyalty. This journey through past and present research on airline passengers parallels an adventure, blending traditional customer service with modern technological advancements like machine learning. From the simplicity of earlier times to the complexity of today's high-tech era, this exploration unveils the transformation of airlines and their relentless pursuit of satisfied and loyal customers.

2.1. Conventional Methods in Customer Segmentation

Traditional customer segmentation methods, rooted in established practices, have long been fundamental in understanding consumer behavior. This investigation explores their historical context, covering manual testing, traditional clustering, and foundational concepts, laying the groundwork for modern methodologies. Sun et al. [5] introduce GPHC, a heuristic approach tailored for interval customer requirements. GPHC employs entropy for data filtering, Gaussian distributions for preference transformation, and a hybrid clustering method for segmentation, proving effective in handling complex customer preferences. While automated parameter tuning and broader application remain future research areas, GPHC is a robust tool for improving audience understanding and optimizing marketing strategies. Grieve et al. [6] propose a robust two-stage business analytics approach for e-commerce, integrating geographic and behavioral segmentation to understand and target customers effectively. The study uses data mining and machine learning to segment customers based on product preferences and then refine these groups geographically. This method offers actionable insights for targeted marketing campaigns and product promotions. Additionally, it enhances logistics efficiency by optimizing warehouse layouts and inventory distribution.

Collaboration opportunities arise between 3PL companies and retailers, with potential monetization through an "Analytics as a Service" platform. Spoor in [7] addresses B2B customer segmentation challenges by introducing a two-stage approach, identifying and separating "keyaccounts" before clustering remaining customers. This yields more straightforward classifications for targeted strategies and informed decision-making. While acknowledging areas for improvement, both studies provide practical guidance for analysts, empowering businesses to unlock valuable insights and confidently navigate the competitive landscape.

Meng et al. [8] propose a dynamic pricing framework for demand response in the electricity retail market, integrating adaptive clustering-based customer segmentation and multiple pricing strategies. The approach categorizes customers based on consumption patterns, develops customized demand models, and maximizes profits considering market constraints. Simulations show a 2.1% average profit margin improvement compared to uniform pricing, highlighting its scalability and practicality. In CRM-based segmentation, introducing CRM enables a more dynamic and personalized understanding of customer interactions beyond traditional methods.

2.2. Unveiling CRM Technologies in Customer Segmentation

In exploring Customer Relationship Management (CRM) systems, we delve into their foundational role in customer segmentation, a pivotal aspect of contemporary business strategies. CRM goes beyond mere customer management, employing sophisticated algorithms and data analytics to understand and cater to individual customer needs. Monil et al. [9] emphasizes the significance of customer segmentation in e-commerce, advocating for clustering analysis to recognize unique customer traits and preferences.

Businesses gain a competitive edge by effectively segmenting their customer base and customizing marketing strategies. The study suggests hybridizing clustering algorithms for even better results, showcasing the evolving landscape of CRM-driven customer segmentation in optimizing business strategies. Kumar et al. [10] examine the nexus between E-CRM, Customer Experience, Satisfaction, and Loyalty in banking, highlighting their interconnectedness. While E-CRM's impact on satisfaction and loyalty is established, its influence on experience warrants further exploration. Understanding this relationship offers banks insights to enhance customer satisfaction and loyalty, reducing acquisition costs and ensuring profitability. Gomes et al. [11] explore personalized customer targeting in e-commerce, emphasizing a four-phase process and prevalent use of k-means segmentation. Future research avenues include deep learning for segmentation and automation of personalized targeting.Loukili et al.

[12] discuss sentiment analysis for market intelligence in e-commerce, suggesting future directions for enhancing performance and detecting fake reviews. CRM-driven segmentation fosters customer engagement and satisfaction, aligning with modern business strategies and adaptability to evolving customer expectations.

2.3. Sentiment Analysis in Customer Segmentation

Mahesvari et al. [13] explore sentiment analysis (SA) in mobile phone reviews, employing machine learning to categorize sentiments effectively. Using NLTK, they implement Multinomial Naive Bayes (MNB), Support Vector Machine (SVM), and Random Forest (RF) models to classify reviews. Results demonstrate high accuracy in mobile phone classification and user segmentation, showcasing SA's power in unlocking customer insights for tailored marketing strategies and product refinement. The study anticipates further exploring sentiment nuances, real-time feedback loops, and cultural context impacts. Overall, it underscores SA's role in customer segmentation, especially when integrated with IoT for comprehensive behavioral data analysis, offering a holistic understanding of customer preferences.

2.4. IoT Integration

Integrating the Internet of Things (IoT) marks a significant shift in customer segmentation, offering real-time insights into consumer behavior through connected devices. Ilona et al. [14] explore this integration using data from wearable sports devices, demonstrating its potential in user segmentation for sports marketers. The study employs clustering and classification models on data from active joggers and runners, highlighting opportunities for accurate segmentation based on physical activity. Mashabi et al. [15] systematically reviewed using artificial intelligence and natural language processing in customer service, identifying chatbots and question-answering systems as predominant applications across various domains. Evaluation methods predominantly include accuracy metrics.

2.5. K-Means Clustering: A Data-Driven Approach to Customer Segmentation

Sophisticated algorithms, particularly K-Means clustering, are crucial in modern customer segmentation. This method effectively groups customers based on shared traits, providing strategic insights into behaviors, preferences, and interactions.

Nandapala et al. [16] demonstrate the practical application of K-Means clustering in customer segmentation, showcasing its benefits in targeting marketing efforts, optimizing resource allocation, and reducing uncertainty. Alsayat in [17] explores its efficacy in hospitality, leveraging big social data to analyze traveller behavior and satisfaction. Similarly, Tabianan et al. [18] focus on e-commerce, utilizing K-Means clustering to segment customers based on purchasing history, enabling personalized marketing and optimized product offerings. Despite limitations, K-Means clustering is a valuable tool for businesses across various industries, facilitating informed decision-making and improved customer engagement. The "Customer Segmentation Using Machine Learning" [19] project addresses the need for businesses to understand and categorize their customer base effectively. Employing the K-means clustering algorithm on the Mall Customers dataset, the project successfully segments customers based on age, income, and spending habits, yielding actionable insights for tailored marketing efforts.

Similarly, Vieri et al. [20] utilize machine learning, specifically the K-Means algorithm, to analyze consumer behavior in the telecommunications industry. The study identifies hidden patterns in customer transaction history, facilitating targeted decision-making to address customer churn effectively.

2.5. Machine Learning Approaches in Customer Segmentation

The compilation of research papers explores the dynamic field of customer segmentation through machine learning methods, offering fresh perspectives and strategies for tailored services and marketing campaigns. Papers delve into various sectors, approaches, and algorithms, shedding light on advancements from clustering analyses to dynamic pricing strategies.

Dullaghan [21] examines ML's potential in reducing customer attrition in the telecom industry using algorithms like C.5 and decision trees, emphasizing the importance of billing and usage data in predicting churn. Alghamdi [22] proposes a hybrid approach using K-means clustering and an optimized ANN-PSO model to predict customer satisfaction in Saudi Arabian restaurants, highlighting the innovative use of social media data for segmentation. These studies

underscore the transformative potential of machine learning in unlocking customer insights and enhancing business strategies.

The research by Ullah delves into retail customer segmentation using the RFMT model, proposing a novel approach that incorporates hierarchical, k-means, Gaussian, and DB-SCAN algorithms. Through meticulous validation and cluster factor analyses, three stable clusters are identified, paving the way for enhanced customer relationship management and targeted marketing.

Malviya et al. [24] explore machine learning applications for sales prediction and customer segmentation, achieving improved model accuracy by integrating the BIRCH algorithm with time-lagged machine learning. Vikram et al. [25] advocate for adopting machine learning models in tourism to expedite customer segmentation and tailor marketing campaigns. Similarly, Joung et al. [26] propose an interpretable machine learning-based approach for customer segmentation, emphasizing product features derived from online reviews.

3. METHODOLOGY

The study adopts a mixed-methods approach, combining quantitative analysis of an extensive Airline Passenger dataset using advanced machine learning models like K-means clustering and decision trees with qualitative insights from in-depth interviews with industry stakeholders. This integration aims to comprehensively explore strategic customer segmentation and satisfaction dynamics in the aviation sector.

3.1. Dataset

The dataset used in this study encompasses information on 130,000 airline passengers, with a demographic distribution of 51% females and 49% males. Key attributes include customer type (loyal or disloyal), age, travel class, flight distance, and type of travel (personal or business). The dataset also assesses satisfaction across various aspects of the travel experience, such as inflight services, seat comfort, online booking ease, and baggage handling, along with departure and arrival delays metrics. [Source: Airline Passenger Satisfaction Dataset].

3.2. Proposed Method

The methodology employed in this study revolves around the strategic use of machine learning techniques for customer segmentation within the aviation sector, focusing on the "Airline Passenger Satisfaction" dataset.

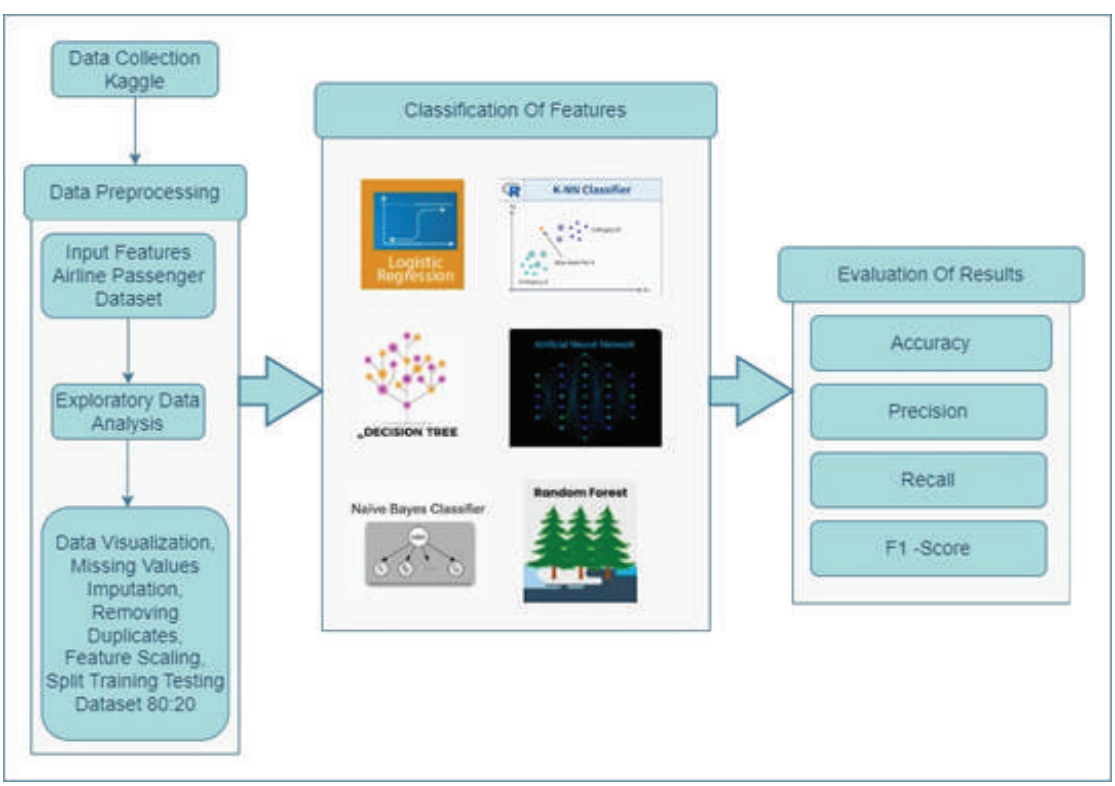

Figure 1: Classification of Airline Passengers by Machine-learning Classifiers

We apply five distinct machine learning classifiers, as shown in Figure 1: Random Forest, K-Nearest Neighbors (KNN), Decision Tree, Naive Bayes, and Artificial Neural Network (ANN) to the training data in our study to explore their effectiveness in customer segmentation.

The Random Forest classifier, which constructs multiple decision trees and aggregates their outputs, helps mitigate overfitting and improves predictive accuracy. In our experiments, Random Forest effectively handled the varied features in the dataset, capturing intricate patterns related to customer satisfaction and behavior. The K-Nearest Neighbors (KNN) algorithm classifies data points based on their proximity to neighboring points. In our study, KNN provided a straightforward approach to segmenting customers by grouping similar passenger profiles based on travel experiences and satisfaction levels. The Decision Tree model's tree-like structure facilitated the segmentation process by making hierarchical decisions about passenger attributes. This model was adept at identifying key factors influencing customer satisfaction, allowing for clear and interpretable segmentation rules. The Naive Bayes classifier, leveraging the independence assumption between features, demonstrated efficiency in handling categorical data within the dataset. Its probabilistic nature enabled quick segmentation of passengers based on their likelihood of belonging to a particular satisfaction category. Finally, with its layers of interconnected neurons, the Artificial Neural Network (ANN) model captured complex, non-linear relationships in the data. The ANN's ability to model intricate patterns contributed to its effectiveness in segmenting customers based on a comprehensive analysis of multiple satisfaction indicators. Each classifier was trained on 80% of the dataset, during which they learned the underlying patterns and relationships. The training process involved feeding input features into the models and adjusting their parameters to minimize prediction errors related to customer segmentation. These trained models were then tested on the remaining 20% of the data to evaluate their performance and generalizability. This approach ensured that the models developed robust predictive capabilities for accurately segmenting and retaining satisfied customers.

3.3. Evaluation

The evaluation phase rigorously assesses classification models developed for airline passenger satisfaction prediction using standard metrics: accuracy, precision, recall, and F1 score. These metrics offer nuanced insights into model performance, which is crucial for effective decision-making in the competitive airline industry.

Accuracy: As shown in Equation 1, accuracy measures the proportion of correctly predicted customer segments to the total number of predictions made by the model. It indicates how well the machine learning classifiers identify and segment satisfied customers based on their travel experiences. High accuracy signifies effective segmentation, which is essential for targeted customer retention strategies.

$$\text{Accruacy} = (\text{Number of Correct Predictions}) / (\text{Total Number of Predictions}) \quad (1)$$

Precision: It measures the model's accuracy in identifying true segments of satisfied customers. It is calculated as the ratio of true positive predictions (correctly identified satisfied customers) to the sum of true positive and false positive predictions (incorrectly identified satisfied customers). High precision indicates that the model effectively minimizes false positives, ensuring that the identified customer segments are highly reliable and relevant for targeted retention strategies. As shown in Equation 2 it's a critical metric for evaluating the performance of our segmentation models.

$$\text{Precision} = (\text{True Positives}) / (\text{TruePositives} + \text{False Positives}) \quad (2)$$

Recall: measures the model's ability to identify all relevant instances of satisfied customers. It is calculated as the ratio of true positive predictions (correctly identified satisfied customers) to the sum of true positive and false negative predictions (un-satisfied customers). High recall indicates that the model effectively identifies most of the satisfied customers, ensuring comprehensive segmentation for targeted retention strategies. As shown in Equation 3, recall is a crucial metric for assessing the completeness of our segmentation models.

$$\text{Recall} = (\text{True Positives}) / (\text{True Positives} + \text{False Negatives}) \quad (3)$$

F1-Score: provides a balance between precision

and recall, offering a metric that considers false positives and false negatives. It is calculated as the harmonic mean of precision and recall, ensuring that both metrics are equally weighted. A high F1 score indicates that the model correctly identifies satisfied customers and minimizes the number of missed satisfied customers, making it a robust measure for evaluating our segmentation models. As shown in Equation 1, the F1 score is a comprehensive metric for assessing the overall performance of our classifiers.

$$\text{F1-Score} = 2 \times (\text{Precision} \times \text{Recall})/(\text{Precision} + \text{Recall}) \quad (4)$$

This study equips stakeholders with tools to refine customer-centric strategies and enhance passenger satisfaction and retention by validating model reliability and providing data-driven insights. The research contributes to airline passenger segmentation by identifying influential factors, comparing algorithm performance, and offering insights into improving passenger experience.

4. RESULTS

The results of the strategic customer segmentation, powered by advanced machine learning techniques applied to the "Airline Passenger Satisfaction" dataset, unfold a nuanced landscape in the realm of airline customer satisfaction. The meticulous classification model has traversed the intricacies of passenger data, revealing distinct segments with unique characteristics and satisfaction patterns. In this section, we delve into the profound insights gleaned from the segmentation process, shedding light on the discernible clusters, their defining features, and the implications these revelations carry for elevating customer satisfaction and retention strategies in the dynamic aviation industry.

4.1. Performance with Random Forest Classifiers

The Random Forest classifier emerges as a robust performer, as shown in Figure 2 in the strategic customer segmentation analysis of the "Airline Passenger Satisfaction" dataset. With an outstanding accuracy of 96%, the model distinguishes and assigns passengers to their respective satisfaction segments. This high accuracy reflects the classifier's proficiency in making correct predictions and suggests its efficacy in capturing the intricate patterns within the dataset. Precision, a crucial metric in classification tasks, is also recorded at an impressive 96%. This indicates that a substantial proportion of all the predicted positive instances (satisfied passengers) are true positives. In practical terms, it highlights the classifier's precision in correctly identifying passengers who genuinely fall into the satisfied segment.

Furthermore, the recall metric, denoting the ratio of true positive predictions to all actual positive instances, stands at 96%. This implies that the Random Forest classifier correctly identifies a substantial portion of satisfied passengers within the dataset. High recall is particularly valuable in scenarios where identifying all positive instances is crucial, such as ensuring that every satisfied customer is recognized. The F1-score, a composite metric considering precision and recall, harmonizes at 96%. This balanced measure signifies the overall robustness of the Random Forest classifier in achieving a blend of precision and recall, indicating its suitability for customer segmentation in the aviation industry. Further, the confusion matrix below explains the results more about how it falls on the testing dataset.

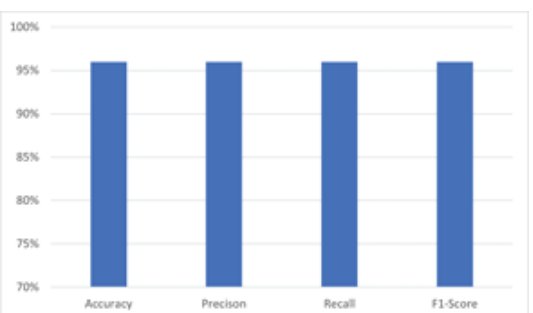

Figure 2: Performance Results with Random-Forest Classifier

The Random Forest classifier's exceptional performance across accuracy, precision, recall, and F1-score metrics underscores its effectiveness in discerning patterns within the "Airline Passenger Satisfaction" dataset. These results position the Random Forest model as a reliable tool for informing targeted strategies to enhance customer satisfaction and retention in the complex landscape of the airline industry.

4.2. Performance with Decision Tree

The Decision Tree classifier exhibits commendable performance, as shown in Figure 3 in the strategic customer segmentation analysis of the "Airline Passenger Satisfaction" dataset. With an accuracy rate of 95%, the model effectively

Precision, a vital metric in classification tasks, is recorded at an impressive 95%. This signifies that among all the instances predicted as positive (satisfied passengers), a significant proportion are true positives. The Decision Tree classifier excels in precisely identifying passengers who genuinely belong to the satisfied segment, demonstrating its reliability in positive predictions.

The recall metric, indicating the ratio of true positive predictions to all actual positive instances, stands at 95%. This implies that the Decision Tree classifier adeptly identifies a substantial portion of satisfied passengers within the dataset. High recall is crucial in scenarios where capturing all positive instances is pivotal, and the model demonstrates effectiveness in achieving this objective. The F1-score, a comprehensive metric that balances precision and recall, achieves a harmonious 95%. This composite measure underscores the Decision Tree classifier's overall robustness in achieving a balance between precision and recall, making it a suitable tool for customer segmentation in the aviation industry. Further the confusion matrix below explains the results more about how it falls on testing dataset.

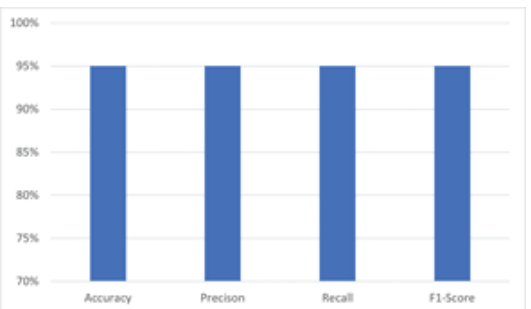

Figure 3: Performance Results with Decision-Tree Classifier

In conclusion, the Decision Tree classifier's strong performance across accuracy, precision, recall, and F1-score metrics underscores its efficacy in deciphering patterns within the "Airline Passenger Satisfaction" dataset. These results position the Decision Tree model as a reliable and interpretable tool for informing targeted strategies to enhance customer satisfaction and retention in the dynamic realm of the airline industry.

4.3. Performance with ANN

The Artificial Neural Network (ANN) emerges as shown in Figure 4 as a robust classifier in strategic customer segmentation using the "Airline Passenger Satisfaction" dataset. Demonstrating an accuracy rate of 90%, the ANN excels in discerning intricate patterns within the dataset, showcasing its proficiency in predicting passenger satisfaction segments. Precision, a pivotal measure of the classifier's accuracy in positive predictions, is reported at a solid 90%. This signifies that the ANN accurately identifies a significant proportion of true positive instances among all predicted positive cases, attesting to its precision in categorizing satisfied passengers effectively. The recall metric, indicating the model's ability to capture true positive instances among all actual positive cases, stands at 90%. This implies that the ANN proficiently recognizes satisfied passengers within the dataset, highlighting its efficacy in capturing positive instances without overlooking significant segments.

The F1-score, a comprehensive metric harmonizing precision and recall, attains a balanced 90%. This composite measure underscores the ANN's equilibrium in achieving accurate positive predictions while capturing a substantial proportion of satisfied passengers. With its neural network architecture, the ANN proves to be a valuable tool for customer segmentation in the airline industry. Further, the confusion matrix below explains the results more about how it falls on the testing dataset.

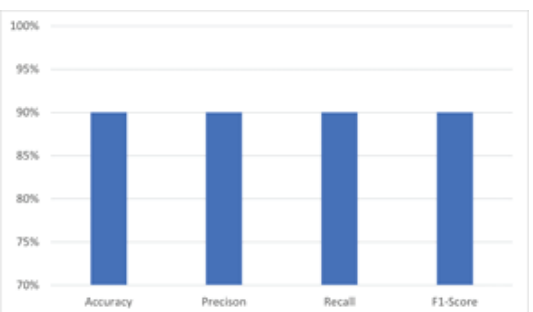

Figure 4: Performance Results with ANN

In conclusion, the ANN's commendable performance across accuracy, precision, recall, and F1-score metrics positions it as a potent classifier for unravelling patterns in the "Airline Passenger Satisfaction" dataset. The results affirm the ANN's potential as an effective tool for informing customer-centric strategies, contributing to enhanced satisfaction and retention strategies within the dynamic aviation landscape.

4.4. Performance with KNN

The K-Nearest Neighbors (KNN) classifier's

performance is highlighted in the context of customer segmentation using the "Airline Passenger Satisfaction" dataset. With an accuracy rate of 75%, as shown in Figure 5, KNN exhibits competency in classifying passengers into satisfaction segments, providing valuable insights for strategic decision-making in the airline industry. Precision, measuring the accuracy of positive predictions, is reported at 75% for KNN. This implies that KNN effectively identifies a substantial proportion of true positive instances among all predicted positive cases, showcasing its precision in accurately categorizing satisfied passengers.

The recall metric, indicating the model's ability to capture true positive instances among all actual positive cases, also stands at 75%. This demonstrates KNN's effectiveness in recognizing satisfied passengers within the dataset, showcasing its ability to capture positive instances without overlooking significant segments.

The F1-score, a comprehensive metric balancing precision and recall, is reported at 75%. This composite measure underscores the equilibrium achieved by KNN in making accurate positive predictions while capturing a significant proportion of satisfied passengers. KNN, with its proximity-based classification approach, contributes meaningfully to customer segmentation efforts in the aviation sector. Further, the confusion matrix below explains the results more about how it falls on the testing dataset.

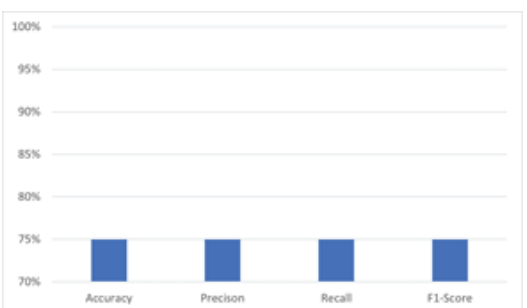

Figure 5. Performance Results with KNN Classifier

In conclusion, KNN's performance across accuracy, precision, recall, and F1-score metrics positions it as a reliable classifier for deciphering patterns in the "Airline Passenger Satisfaction" dataset. While not achieving the same level as some other classifiers, KNN provides a valuable perspective for customer segmentation, offering nuanced insights for enhancing satisfaction and retention strategies within the dynamic aviation landscape.

4.5. Performance with Naïve Bayes

The Naïve Bayes classifier's performance is evaluated within customer segmentation using the "Airline Passenger Satisfaction" dataset. Demonstrating, as shown in Figure 6, an accuracy rate of 86%, Naïve Bayes showcases its ability to effectively classify passengers into satisfaction segments, contributing valuable insights for strategic decision-making in the airline industry. Regarding precision, which measures the accuracy of positive predictions, Naïve Bayes achieves a commendable score of 86%. This implies that the classifier adeptly identifies a significant proportion of true positive instances among all predicted positive cases, underscoring its precision in accurately categorizing satisfied passengers.

The recall metric, indicating the model's ability to capture true positive instances among all actual positive cases, is also reported at 86% for Naive Bayes. This highlights the classifier's effectiveness in recognizing satisfied passengers within the dataset, showcasing its capacity to capture positive instances without overlooking significant segments.

The F1-score, a holistic metric balancing precision and recall, is 86%. This composite measure emphasizes the equilibrium achieved by Naive Bayes in making accurate positive predictions while capturing a substantial proportion of satisfied passengers. Naïve Bayes, with its probabilistic approach, contributes meaningfully to customer segmentation efforts in the aviation sector. Further, the confusion matrix below explains the results more about how it falls on the testing dataset

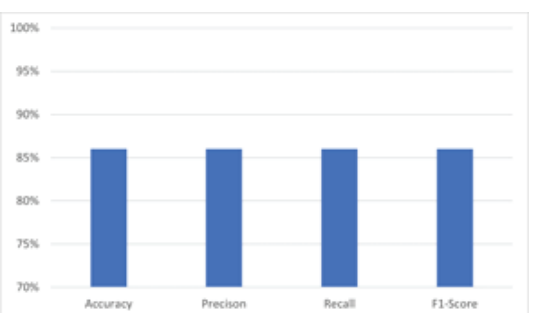

Figure 6: Performance Results with Naïve Bayes

In conclusion, Naïve Bayes performance across accuracy, precision, recall, and F1-score metrics

positions it as a reliable classifier for uncovering patterns in the "Airline Passenger Satisfaction" dataset. While not surpassing the accuracy levels of specific other classifiers, Naïve Bayes provides a valuable probabilistic perspective for customer segmentation, offering nuanced insights to enhance satisfaction and retention strategies within the dynamic aviation landscape.

4.6. Conclusions

In conclusion, the findings of this study emphasize the significance of strategic customer segmentation in the airline industry. The classifiers exhibited commendable accuracy and efficiency, showcasing their applicability in identifying distinct passenger groups with specific preferences and behaviors. The practical implications highlight the potential for airlines to employ these classifiers in tailoring services, marketing strategies, and loyalty programs. The results affirm the value of integrating machine learning into customer relationship management, improving customer satisfaction and strengthening relationships. For future research, it is recommended to delve deeper into refining and expanding the dataset, incorporating additional features and diverse sources of information. Further investigation into ensemble methods and hybrid models could provide insights into optimizing classification performance. Exploring the integration of real-time data and continuous learning models can enhance the adaptability of the classifiers to evolving passenger preferences. Additionally, extending the study to encompass a broader range of airlines and demographic groups would contribute to a more comprehensive understanding of customer segmentation in the aviation sector.

REFERENCES

[1] I. Lewaaelhamd, "Customer segmentation using machine learning model: an application of RFM analysis", J*ournal of Data Science and Intelligent Systems,* vol. 2, no. 1, pp.29-36, 2024.

[2] V. R. Thalkar, "Customer Segmentation Using Machine Learning" *in International Journal of Scientific Research in Computer Science, Engineering and Information Technology,* ISSN DOI: https://doi.org/10.32628/CSEIT217654, 2023.

[3] S. Vikram et al., "Customer segmentation in tourism industry using machine learning models," I*nternational Research Journal on Advanced Science Hub,* vol. 5, no. 5, pp. 43-49, DOI: 10.47392/irjash.2023.S006, 2023.

[4] K. Tabianan et al., "K-means clustering approach for intelligent customer segmentation using customer purchase behavior data," *Sustainability,* vol. 14, no. 12, pp. 7243 DOI: 10.3390/-su14127243, , 2022.

[5] Z. H. Sun et al., "GPHC: A heuristic clustering method to customer segmentation," *in Applied Soft Computing,* 111, pp. 107677, 2021.

[6] A. Griva et al., "A two-stage business analytics approach to perform behavioral and geographic customer segmentation using e-commerce delivery data," J*ournal of Decision Systems,* vol. 33, no. 1, pp.1-29, DOI: 10.1080/12460125.2022.2151071, 2024.

[7] J. M. Spoor, "Improving customer segmentation via classification of key accounts as outliers," *Journal of Marketing Analytics,* vol. 11, no. 4, pp. 747-760, DOI: 10.1057/s41270-022-00185-4, 2023.

[8] F. Meng et al., "Multiple dynamic pricing for demand response with adaptive clustering-based customer segmentation in smart grids," *In Applied Energy,* 333, pp. 120626, DOI: 10.1016/j.apenergy.2022.120626, 2023.

[9] P. Monil et al., "Customer Segmentation using Machine Learning," *In IJRASET,* ISSN: 2321-9653, IC Value: 45.98, SJ Impact Factor: 7.429, Vol. 8, no. 4, pp.2104-2108, Available at www.ijraset.com, June 2020.

[10] P. Kumar and A. K. Mokha, "Relationship between E-CRM, Customer Experience, Customer Satisfaction and Customer Loyalty in Banking Industry: A Review of Literature," *in RESEARCH REVIEW International Journal of Multidisciplinary,* vol. 6, no. 2, pp. 127-137, DOI: 10.31305/rrijm.2021.v06.i02.022, 2021.

[11] M. A. Gomes and T. Meisen, "A review on customer segmentation methods for personalized customer targeting in e-commerce use cases," *In Information Systems and e-Business Management,* DOI: 10.1007/s10257-023-00640-4, 2023.

[12] M. Loukili et al., "Sentiment Analysis of Product Reviews for ECommerce Recommendation based on Machine Learning," *In Int. J. Advance Soft Compu. Appl,* vol. 15, no. 1, DOI: 10.15849/IJASCA.230320.01, March 2023.

[13] T. L. Maheswari et al., "Customer Segmentation Based on Sentimental Analysis," *In 2022 International Conference on Advanced Computing Technologies & Applications (ICACTA),* pp. 1-7, DOI: 10.1109/ICACTA54488.2022.9753207, 2022.

[14] I. Pawełoszek, "Customer segmentation based on activity monitoring applications for the recommendation system," *In 25th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems,* 192, pp. 4751–4761, DOI: 10.1016/j.procs.2021.09.253, 2021.

[15] M. Mashaabi, A. Alotaibi, H. Qudaih, R. Alnashwan, and H. Al-Khalifa, "Natural Language Processing in Customer Service: A Systematic Review," *in Computation and Language (cs.CL); Artificial Intelligence (cs.AI) , preprint arXiv:2212.09523*, 2023.

[16] E. Y. L. Nandapala and K.P.N Jayasena "The practical approach in Customers segmentation by using the K-Means Algorithm," I*n 15th (IEEE) International Conference on Industrial and Information Systems (ICIIS),* pp. 344-349, DOI: 10.1109/ICIIS51140.2020.934263, 2020.

[17] A. Alsayat, "Customer decision-making analysis based on big social data using machine learning: a case study of hotels in Mecca," *In Neural Computing and Applications,* vol. 35, no. 6, pp. 4701–4722, doi: 10.1007/s00521-022-07992-x, 2023.

[18] K. Tabianan et al., "K-Means Clustering Approach for Intelligent Customer Segmentation Using Customer Purchase Behavior Data," *in Data. Sustainability,* vol. 14, no. 12, pp. 7243, DOI: org/10.3390/su14127243, 2022,.

[19] P. S. Krishna and Dr. G. S. Sujatha, "Customer Segmentation Using Machine Learning, " *in © 2023 IJCRT,* vol. 11, no. 9, ISSN: 2320-2882, September 2023.

[20] J. K. Vieri et al., "Exclusive Clustering Technique for Customer Segmentation in National Telecommunications Companies," *In International Journal of Information Technology and Computer Science Applications (IJITCSA),* ISSN 2964-3139, vol. 1, no. 1, pp. 51 - 57, 2021.

[21] C. Dullaghan and E. Rozaki, "INTEGRATION OF MACHINE LEARNING TECHNIQUES TO EVALUATE DYNAMIC CUSTOMER SEG- MENTATION ANALYSIS FOR MOBILE CUSTOMERS," *In International Journal of Data Mining & Knowledge Management Process (IJDKP),* vol. 7, no.1, January 2017.

[22] A. Alghamdi, "A Hybrid Method for Customer Segmentation in Saudi Arabia Restaurants Using Clustering, Neural Networks, and Optimization Learning Techniques," *In Arabian Journal for Science and Engineering,* vol. 48, no. 2, pp. 2021–2039, DOI: 10.1007/s13369-022 07091.

[23] A. Ullah, M. I. Mohmand, H. Hussain, S. Johar, I. Khan, and S.Ahmad, "Customer Analysis Using Machine Learning-Based Classification Algorithms for Effective Segmentation Using Recency, Frequency, Monetary, and Time," *Sensors,* vol. 23, no. 6, pp. 3180, DOI: 10.3390/s23063180, 2023.

[24] P. Malviya et al., "Customer Segmentation and Business Sales Forecasting using Machine Learning for Business Development," *In International Journal on Recent and Innovation Trends in Computing and Communication,* DOI: 10.17762/ijritcc.v11i11s.8170, 2023.

[25] S. Vikram et al., "Customer Segmentation in Tourism Industry using Machine Learning Models," *In International Conference on intelligent COMPUting TEchnologies and Research (i-COMPUTER),* vol. 5, no. 5, pp.43-49, DOI: 10.47392/irjash.2023.S006, 2023.

[26] J. Joung and H. Kim, "Interpretable machine learning-based approach for customer segmentation for new product development from online product reviews," *In International Journal of Information Management,* 70, pp.102641, DOI: 10.1016/j.ijinfomgt.2023.102641, 2023.

Ali et al. LGURJCSIT 2024

LGU Research Journal of Computer Science & IT

ISSN: 2521-0122 (Online)
ISSN: 2519-7991 (Print)

doi: 10.54692/lgurjcsit.2024.082467

Vol (8): Issue (2), April – June 2024

Security Assessment in Software Defined Networks (SDN): Vulnerabilities, Challenges and Research Prospects

Amir Ali[1*], Muhammad Murtaza Yousaf[2], Muhammad Khawar Bashir[1], Fahran Masud[1], Muhammad Rizwan Saleem[3], Asad Ali[4]

[1]Department of Statistics and Computer Science, University of Veterinary & Animal Sciences, Lahore, Pakistan.

[2]Department of Software Engineering, Faculty of Computing & Information Technology, University of the Punjab, Lahore, Pakistan.

[3]Directorate of I.T., University of Veterinary & Animal Sciences, Lahore, Pakistan

4Superior University, Lahore, Pakistan.

Email:amir.ali@uvas.edu.pk

ABSTRACT:

Conventional internet architecture facilitates users for different services and applications. Still, it faces numerous challenges like network management, QoS management for network virtualization, IP multicasting, deployment of IPV6, crucial security measures, end-to-end connectivity, inter- and inter-domain routing. To meet the demand for these services due to the rapid growth of technologies and traffic, an emerging network architecture termed Software Defined Network (SDN) with programmable technology has brought unprecedented management to control networks. Due to the separation of data, control, and application planes, the defined network provides cost-effective openness, centralized automation, programmable features as per user demands, and high resilience for network administrators. OpenFlow evolved as the first standard protocol for software-defined network control and data plane communication to meet changing business requirements. Although Software Defined Network brings enormous advancements in networks to support business applications, it is severely affected by cyber-attacks on data, control and application planes. Middle boxes play a significant role in managing network effectiveness and providing adequate security control from external and internal security threats. However, they require proper management and configuration; otherwise, they can lead to devastating effects. This paper makes significant contributions by (1) Examining potential security attacks on various SDN layers and addressing inconsistent policies, (2) Identifying security concerns within SDN planes and proposing preventive measures against prevalent attacks, and (3) Delving into security threats, challenges, and research opportunities in SDN, with a focus on critical security controls like spam detectors, IDS/IPS, firewalls, and policy management. Researchers can use this article as a resource to comprehend the security landscape, including the current state of development and challenges explored by the scientific community in the realm of SDN.

KEYWORDS: SDN Security Issues, SDN Planes Security, DDoS Attacks in SDN Planes, Open Flow Security, SDN Security Research Opportunities, Future Challenges in SDN, Security Measurements in SDN.

1. INTRODUCTION

The internet state is becoming rapidly ineffective due to substantial traffic patterns produced by various services like big data, mobile devices, server virtualization, network security controls and cloud computing. High performance,

energy efficiency and resilience are essential elements, along with improved network speed, versatile digital services, scalability and good QoS. Compiling these requirements with conventional network devices is impossible because of limited functionalities. Network experts have to manually configure thousands of network equipment to implement network policies and services, which is not an easy task for them. Control and forwarding logic parts they are combined in the same boxes as traditional devices. Vendor supports vary from device to device without any open software platform to experiment with the latest concepts. So, in this regard, SDN provides a more optimised solution which decouples control and forwarding planes with reprogrammable support. As opposed to a distributed control plane in the existing network, SDN comes with the separation of network control and forwarding elements. With the support of applications, underlying network components and services can be accessed. Nevertheless, due to centralized control plane functionality, network management and resilience are much improvised. Severe attacks on applications, data and control planes can abruptly degrade SDN performance, causing a deteriorating impact. Security detection algorithms can be applied as OpenFlow security applications, but malevolent application services can severely disrupt the entire SDN network. Intrusion Detection and Prevention Systems (IDPSs) were also modelled in SDN but could not perform well in attack detection and prevention. In SDN, OpenFlow is the most widely used implementation architecture due to its multi-vendor support. Network services and policies are deployed via controller-level software application that interrelates with the data plane via OpenFlow protocol South-Bound API. At the same time, North-Bound API is used to connect the data plane with a controller with the support of OpenFlow protocol. SDN OpenFlow-based applications are used to implement various services and functions in the underlying network structure. So, the entire control of network traffic is moved from the infrastructure plane to the control plane under the supervision of the administrator, which facilitates a high level of network management, optimization and automation with the support of SDN applications. SDN improvised network management due to centralized control, traffic forwarding rules, policies, dynamic orchestration of network components, and global visibility, but security concerns have created a severe challenge to SDN implementation. SDN architecture is based on three planes. Firstly, Application Plane contains SDN applications for network management, policy deployment and security services. Secondly Control Plane, which is a centralized control manage through the network operating system, built OpenFlow tables, facilitates hardware abstractions to SDN application and provides centralized management along with global view of the entire network. Lastly Data Plane, which is a forwarding element used to forward traffic flows as per control plane instruction and routing decision. Nevertheless, SDN has various limitations and challenges like energy efficiency, scalability and the most crucial challenge among all these is SDN security for each planes. As, a centralized controller is accountable to manage whole network operation, in case of centralized controller vulnerable then entire network will get disrupted due to illegal access and improper network resources utilization by adding and sending suspicious flows. How to secure the entire network from malicious applications is a grave security concern in SDN. Fig. 1 indicates SDN layers or planes simplified architecture. Software defined network brought programmability support which has the functionality to modify and control network devices through software instead of proprietary interfaces and traditional closed boxes [1]. In SDN centralized control manage global view of entire networks and links which connect them. To deploy several numbers of policies in data plane, applications utilize northbound API offered through control plane. However, to communicate with different network devices, the controller used southbound API between control and data planes [2].

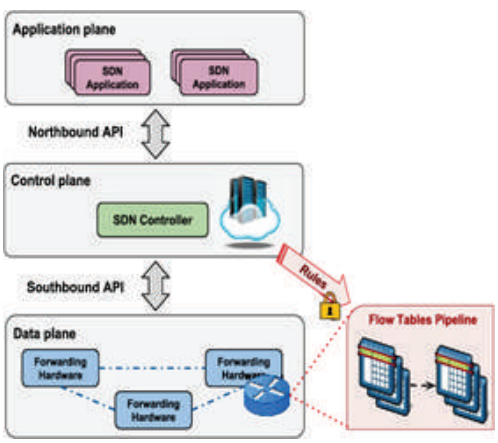

Figure 1: SDN Layers Architecture

Software-defined networks (SDN) are emerging network architectures that support several business applications and are termed next-generation internet architecture for future networks. Software Defined Network (SDN) has emerged as the most dynamic and programmable architecture per user's requirements since its emergence in the industry. The physical separation of infrastructure and control planes brings more network flexibility regarding scalability, cost-effective solutions, consistent policies, and a vendor-independent and global view of the entire network. SDN provides centralized control for network experts to manage their respective network at the data plane level easily. Data packets can be forwarded per control plane instructions provided by the network administrator. This emerging architecture has received significant attention from the network industry and academia. OpenFlow is an open standard designed by Stanford University's clean slate project. The primary scope or concept behind SDN evolution is to provide reprogrammable support with multi-vendor support to network domain researchers. SDN-based networks have already been deployed in NTT's edge gateway, Google's backbone network, and Microsoft's public cloud [3]. With the advent of this promising architecture, one central area of security in SDN has advantages compared to conventional network architecture. Like in SDN, the centralised controller has a complete network view of traffic analysis and can implement new security policies to prevent attack propagation. With OpenFlow support, a centralized global view, and the programmable nature of SDN, vulnerabilities at each layer are increased. Due to the open programmability approach, various serious security threats can damage centralized controller resources, flow tables, communication channels, SDN components and functions. In the last decades, several researchers and industry groups have provided solutions for SDN security challenges for the data, control, and application planes. However, they are still facing failure due to improper deployment and separation of data and control planes. SDN deployment in a single organization or data centre might not succeed without focusing on proper security measures. This article presents a comprehensive analysis of security challenges in SDN, proposed solutions for research communities, open research challenges in SDN security, and impending research guidelines in this domain. This paper is organized as follows: Section 2 describes Early Programmable Networks. Section 3 illustrates the Properties of the Software Defined Network. Security Analysis and Potential Attacks in SDN are defined in Section 4. In Section 5, Security Issues in SDN are elaborated. Solutions to Security concerns are discussed in Section 6. Future Directions and challenges are deliberated in Section 7, and the paper is concluded in Section 8.

2. EARLY PROGRAMMABLE NETWORKS

Software Defined Network (SDN) brought significant prospects for network experts to control and manage third-party applications. The design of SDN covers various aspects like policy management, middlebox placements, vendor independence, simplified algorithms, and low-cost management. Several programmable network efforts have been made in history. 1995, Open Signaling was introduced for Asynchronous Transfer Mode (ATM) and mobile networks to be more scalable and programmable [4]. Researchers decided to separate communication hardware and control software at that period. With the support of a distributed programming platform, a programmable network interface and openness are managed in Open Signaling. To manage Asynchronous Transfer Mode networks and to design scalable control, researchers put their efforts in various domains in the middle of 1990 [5].

Programmable switches in Active Networking emerged in 1990, but lack of security and low performance were not widely used [6]. Routing decisions and protocol communication among different network devices with clean slate design logic was proposed in 2004 and termed a 4D project [7]. A NOX operating system was proposed for an Open flow-enabled network [8]. NETCONF protocol, which used API for network configuration, was proposed in 2006 for network configuration modification. A protocol named SNMP was introduced for performance monitoring and fault management of network equipment configuration [9]. Although SNMP and NETCONF can be used in programmable networking for hybrid and parallel switches, SNMP has significant security concerns. The NETCONF working group is still actively developing new paradigms. Later on, Ethane was proposed to manage security policies at the controller level in 2006 [10]. Ethane emerged as the antecedent of software-defined networks.

Network securities rely on manual middlebox

configuration in conventional network infrastructure, which is vendor-specific devices and must be configured manually. To deploy security policies, network administrators must manually configure every vendor-specific level of equipment, which could cause policy conflicts. Crucial and indispensable security threats can be encountered due to manual configurations of firewalls, IDS/IPS and other security appliances. Centralized security policies can be deployed in underlying network devices through SDN applications and control plane programmability functionalities. The SANE architecture was introduced to perform authentication among different end users. Ethane, an extension of SANE architecture, was proposed to enforce policies at the controller level to forward data packets as per defined flow rules in the flow table [11]. Control and data planes acquired programmable functionality with these inventions. In these programmable networks, security implications led to the invention of OpenFlow, which Stanford University developed. It is a clean slate project whose main objective is to facilitate network researchers in executing experimentations in a real test bed environment [12].

Both data and control planes are tightly coupled in one device in traditional networks, where each device has a different vendor's own designed operating system that needs distributed configurations and high skills. Network administrators faced complex configuration, high cost due to vendor-specific interfaces, and significant overhead. To implement security control in traditional networks, perimeter-based solutions are deployed. Critical security configuration errors and policy conflicts can cause severe security threats and breaches due to the manual configuration of different middlebox controls, such as adaptive security appliances, IDS / IPS, spam detectors, firewalls, and IPSec technologies. SDN with OpenFlow standard brought remarkable evolution, and leading providers like VMWare, Google, Dell, Microsoft, HP and others have made its adoption. SDN architecture provides security management operations to detect network anomalies, and traffic management can be monitored at the controller level. Due to the programmable nature of SDN architecture, a centralized controller provides a global view of underlying network and new security policies can be implemented on a timely basis to prevent various security threats. SDN infrastructure poses many security concerns because of its dynamic updates, cost-cutting solution and reprogrammable technology. Different kinds of attacks on SDN planes, like denial of services, network manipulation, brute force, rap spoofing, brute force, and policy conflicts, create severe deterioration in network operations. The security of a centralized controller can be endangered. Attackers can interrupt the communication between the forwarding plane and the controller, particularly in different autonomous systems. A number of solutions related to different planes of software defined network were offered by several researchers. Nonetheless, if these growing security dangers are not adequately managed, they can cause significant damage. Dealing with such significant security concerns in SDN is difficult for academics and requires careful study. Three functional planes are implemented in a defined network. Application plane is used to maintain different applications like network management, load balancers, implementation of security policies and procedures, and VPNs. To manage the entire network more effectively via SDN application, the control plane provides centralized control with a network operating system. Lastly, after acquiring instructions from the control plane controller, traffic is forwarded among different devices. Network security measures that require knowledge about network resources from the control plane can be configured at the application plane via the north bound interface. Security applications retrieved data packets from the control plane via the southbound interface to forward traffic as per user-defined security policies. Despite being implemented in hardware like in traditional networks, software modules are used for data management and other associated setups in SDN networks. This allows for the efficient implementation of policies at the run-time level.

3. PROPERTIES OF SOFT WARE-DEFINED NETWORKS

Properties of SDN architecture with essential features are presented in Fig. 2, which can cause severe and potential attacks. Centralized control is a physically distributed controller component that is logically centralized. The underlying network infrastructure and policies for network services are managed with this centralized controller. Various numbers of controllers are developed, like NOX, Beacon [13], and Floodlight [14], to function as an OpenFlow [15]. To manage forwarding devices and support multiple

programming interfaces like BGP and NETCONF, more controller implementations like OpenContrail [16] and OpenDaylight [17] are developed. For scalability and reliability requirements, a single controller design to controller cluster is proposed, as shown in Fig. 2. Distributed controllers like Kandoo, SoftCell, Onix [18], HyperFlow, OpenDaylight [19], ONOS [20] are proposed. Every single controller is presented as a master for a set of switches, and controllers are shown as clusters in the Main and enslaved person groups. In Open Programmable Interfaces brings more flexibility for network management, automation and orchestration for novel solutions. OpenFlow adopted an open standard programmable interface in the industry to program multiple forwarding devices like virtual switches, network processors, and FPGA-based ones. So, the complexity of underlying hardware is managed in a centralized way.

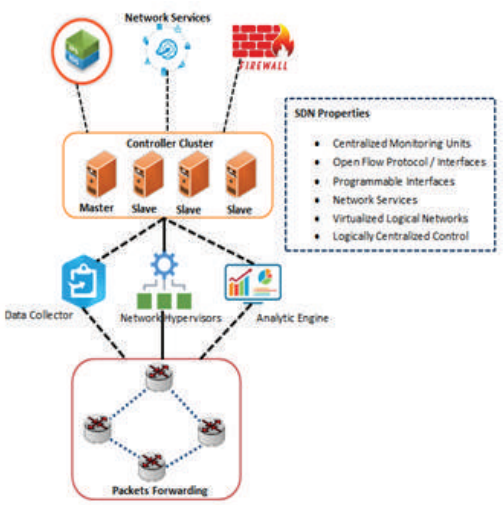

Figure 2: SDN Properties with Network Services

SDN permits the integration of third-party network services. SDN controller implementation in a monolithic architecture like NOX, POX and RYU applications are compiled and executed as a portion of the controller module. Run time execution of applications is done in the OpenDaylight controller without restarting the controller module. It is similar to operating systems in which libraries and software functions are integrated. It brings more flexibility, permits customization of services, and decreases the cost of proprietary services. Virtualization in SDN components plays a more vital role than in a shared physical substrate, where several logical switches can be instantiated, and each entity can characterize individual customers. It provides various services like security, high-end performance and QoS as per customer requirements. The telecommunications industry designs Network Function Virtualization to employ virtualization for network services and functions previously configured in proprietary components. So, it provides a more agile and dynamic network service. NFV and SDN support brought a novel trend to network research communities.

In Fig. 3, different interfaces are shown like Southbound API for Control-Data Interface (Like NETCONF, OpenFlow, OVSDB), Northbound API for Application-Control Interface and East-West interface for bidirectional and lateral communication between controllers. A programmable interface named Switch Management Protocol, like OVSDB and Open Flow-Configure is used to configure and manage programmable hardware [21]. Like to configure OpenFlow and logical switches Open Flow Configure protocol is used. NETCONF is a transport protocol that exchanges switch configuration information between packet forwarding entity and configuration point. Centralized monitoring units like analytic engines and data collectors in SDN architecture are used to automate updates; networking function like the TAP unit provides data to the Deep Packet Inspection engine to highlight intrusion traffic, and then malicious traffic resisted through forwarding table with updated state. Other SDN entities are used for monitoring purposes, such as underlying architecture components and switch state information and status managed through the OpenFlow protocol. NetFlow, sFlow or visibility tools can be deployed to update information about the current status of network devices.

SDN decouples network control and forwarding functions. Control logic is implemented Through a centralised controller and separated from each forwarding device. Forwarding devices in the data plane are managed through a centralized controller. Forwarding devices can be accessed via applications due to the split nature of control and data planes. Network operation for specialized environments is customized by aggregating traditional vertically integrated networking stacks by SDN. To control heterogeneous network resources, applications can be designed at the software layer to manage the underlying hardware [22].

With the changing business requirements, a

software defined network facilitates network administrators with an instant response. Network administrators can perform traffic management in terms of traffic blocks or prioritize individual packet levels with the support of a centralized controller through software without physical access to switches [23]. Some proposals on open flow and security systems are discussed in [24]. Open Flow enabler of SDN and three planes of SDN concerning OpenFlow architecture are defined below.

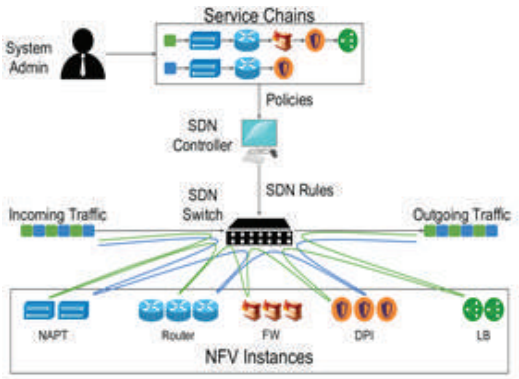

Figure 3: Service Management and SDN Interfaces

3.1. Open Flow Enabler of SDN

OpenFlow stands out as a renowned Software Defined Networking (SDN) technology. It delineates the guidelines for establishing connections among network forwarding devices and orchestrates the centralized software control of entire networks. The OpenFlow controller furnishes a clear and comprehensive perspective of the network, enabling the implementation of security policies and identifying vulnerabilities within the network [24].

3.2. Application Plane

SDN manages network device performance via the control layer, enabling applications to interact with it. Applications can benefit from controllers regarding accessibility and network resource availability, which gives network administrators more flexibility to manage and deploy different policies through SDN-supported applications [25]. The control panel provides a complete logical map and resource information of all network components for the SDN application through the southbound API. For this reason, OpenFlow is a natural choice for deploying network properties in the form of OpenFlow applications. So, as Security applications, a range of network security services are deployed over the top of the OpenFlow controller.

3.3. Control Plane

It is deployed in a separate, logically centralized plane by extracting from individual network entities. To implement flow rules at the switches level and control the entire network through the Network Operating System with global level view SDN controller is mainly responsible. NOS collect information via APIs to control a network. Controller is accountable for flow management in OpenFlow switches. OpenFlow protocol facilitates standard technique for the controller to communicate with switches. For simplicity purposes, at the inception stage, OpenFlow was designed for a single OpenFlow controller. Still, later, its architectures extended to support multiple distributed controllers to acquire high availability and scalability in the network [26].

3.4. Data Plane

Numerous forwarding elements like routers, switches, virtual switches and access points using secure transport channels can be rearranged or reprogrammed for many reasons, such as traffic isolation and virtualization. In SDN forwarding devices, the OpenFlow switch is the most acceptable illustration. OpenFlow switches are simple and, therefore, unlikely to become obsolete since forward strategies are thrust by the controller software instead of switch firmware. An OpenFlow switch using OpenFlow protocol can be programmed. When programmed to an OpenFlow switch, the flow tables of the switches have to manage a set of actions for every flow. The earlier infrastructure does not provide Layer 2 and Layer 3 support, while the latter has OpenFlow interfaces and protocols as the latest properties.

4. SECURITY ANALYSIS AND POTENTIAL ATTACKS IN SDN

Several surveys [27-37] are being conducted to analyze SDN/OpenFlow security issues. In this context, our initial objective and contribution is a detailed discussion of potential attacks in the SDN environment and industry work for more effective security solutions and future research challenges. Various security analyses are focused on OpenFlow technology. An assessment of the OpenFlow protocol using STRIDE threat evaluation was performed [38-39]. This work targets DoS attacks and Information Disclosure, and

various mitigation methods are proposed, like collecting flow rules to minimize the flow table size. Still, such claims are not verified in this work. Fig. 4 shows centralized policies in the SDN controller [39]. Transport Layer Security for authentication between switches and controllers is introduced in OpenFlow switch specification, but the Transport Layer Security standard is not specified with optional security features. Fraudulent rule modification and insertion are highlighted in [39] as most vendors do not support TLS (Transport Layer Security) in switches.

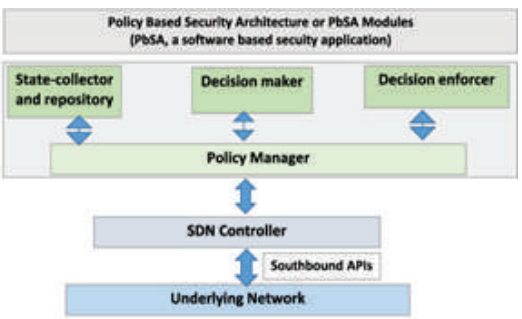

Figure 4.:Centralized Policies in SDN Controller

An analysis is conducted in [40] that timely prevents essential responses for novel threats from network programmability and centralized control. The SDN environment highlights Seven threat vectors related to application, control and data planes. Several preventive solutions are deployed in case of software or hardware failure replication of controllers to avoid any disruption of applications and services, and single controller failure causes errors in software and confidentiality for data. ProtoGENI testbed is analyzed in [41] to measure the attack's effect in the SDN environment. It is realized that several flooding and propagation attacks occur when this testbed is used. A fingerprinting attack in an SDN network is studied in [42], wherein the network is fingerprinted to find whether it employs OpenFlow switches. Then, a DoS attack is implemented on the control plane via NorthBound API from the data plane forwarding table. A DoS attack affects each SDN plane, which attempts to make a machine unavailable for its legitimate users. Flow table insertions in the switches control plane are vital in traffic management and policies. If centralized control is affected by a DoS attack, the entire network will be in trouble. Security management solutions to improve SDN network performance are discussed in [43], in which the primary focus was on protocol security regarding authentication and confidentiality. They concluded that communication protocols between controller to controller and switch to controller need more refined techniques to tackle attacks appropriately. Security management in conventional and SDN networks is discussed [44] regarding availability, consistency, authenticity, and integrity. This study shows that SDN performs better in attack detection, mitigation, cost-effective solutions and network management than conventional networks. In this work, the author claims that threats highlighted in SDN exist in conventional networks. In the context of SDN planes, there is more vulnerability, a vital security challenge in SDN deployment [45].

In the study documented in [46], the author concentrated on enhancing security measures in wireless mobile networks by introducing a comprehensive framework. However, it's important to note that this endeavour had a relatively constrained integration of Software Defined Networking (SDN) based security solutions. The work underscores the diverse requirements for SDN security, emphasizing aspects such as adaptability, interoperability, simplicity, responsiveness, and compatibility. A framework is proposed wherein the wireless mobile security layer coordinates with local agents at the data plane. Figure 5 shows controller management for wireless security. SDN impact is discussed, as well as potential risks and attacks on cloud computing [47]. Various vulnerabilities and prospects are discussed in this work. The author mentioned that SDN expansion from the campus area network to the vast area network requires significant security observation to secure each layer in the SDN environment for better performance. To validate those risk factors to which the SDN network is vulnerable, the traditional STRIDE threat model [48] is used. This model is based on traditional methods, but the risks can also apply to general networks. Another method is to classify SDN functional parts and the parts vulnerable to different threats. Attacks can be differentiated based on resources.

For example, switches and flow tables, including important network information, risk being attacked. The other one is the communication channel between Controller and Switches, which contains important messages that can be accessed.The interface used by the controller to

communicate with high-level applications such as REST is vulnerable to attack; it plays with the controller to allow unauthorized networks and applications to communicate with the controller. Fig.6 [49] shows the attacker's access at each service and plane of SDN.

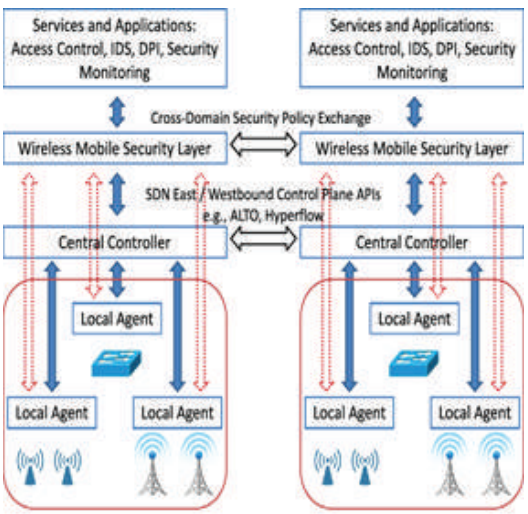

Figure 5: Controllers Management for Wireless Security

4.1. Information Disclosure

This malware does not cause any network disruption, but it does operate as a secret agent in the information it collects. Furthermore, intruders initially seek sensitive information to obtain various network statistics linked with connecting nodes. A controller is required to control all of the switches. If an attacker gains access to the controller, they can take control of the entire network. Flow rules are controlled in switch flow tables,

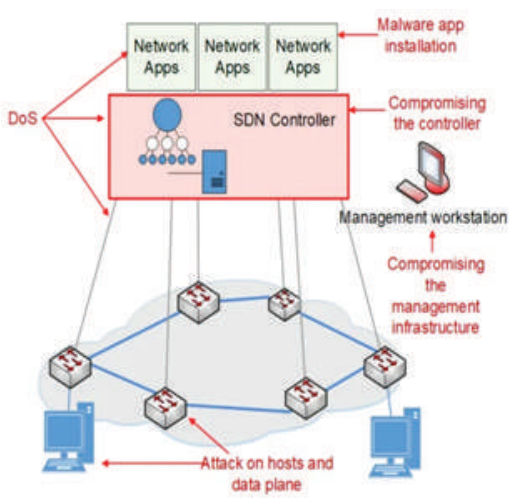

Figure 6: Attackers Access at different Planes

which attackers can hack, allowing them to change flow rules and send traffic to an unapproved network. MitM attacks have just been discovered in the existing open-flow architecture. Transport layer security is an optional feature in a recent encrypted method for open-flow communication.

4.2. Non-Repudiation Attack

Inside the center, an intruder communicates with both parties. As a result, encryption has the potential to be effective in MitM attacks. Encryption techniques are used to ensure that messages are validated from the source and that they are not disrupted in the middle of a conversation. The authors demonstrated transport layer encryption for switches and controller communication in open flow.

4.3. Repudiation

Repudiation occurs when one party engaged in communication denies the involvement of the other in any aspect of the communication process. In contrast, non-repudiation, acknowledged as a legitimate rather than a technical concept, seeks to prevent such disavowals. Figure 7 [49] illustrates the actions associated with repudiation attacks.

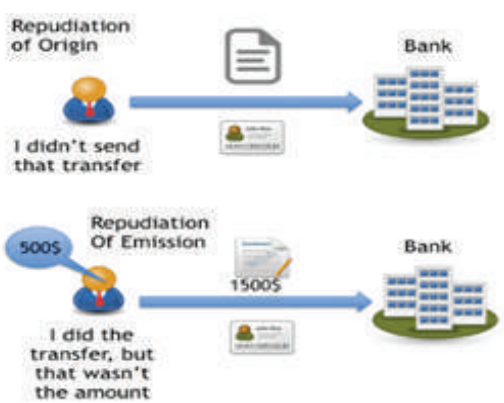

Figure 7: Repudiation Attack Activity

4.4. Spoofing

To determine if an attacker is hiding on purpose by changing (IP, MAC, ARP), etc. in spoofing. For example, an attacker can get access to a network by using a fake or spoofed IP address. It could be a variety of attacks, such as SYN flooding, smurfing, and SDN amplification [50]. Furthermore, when a faked IP is part of a zombie network, it can launch distributed DOS and DDOS attacks. Primary SDN threats include ARP and IP spoofing.

4.5. *ARP Spoofing*

ARP spoofing connects an attacker's MAC address to a valid IP address. As a result, traffic from the genuine user is hijacked, and the genuine user is compromised. IP to MAC mapping tables are used to identify ARP spoofing. An ARP module was added to the controller to track MAC addresses from authentic host users [51]. The controller connects with the ARP module and rejects any invalidated ARP acknowledgements. If SSL encryption is not enabled, ARP poisoning may occur between controller switches. Victims' and attackers' locations at the same subnet lead towards an ARP cache poisoning attack. An attacker can use scanning tools to learn about network traffic and its elements.

An Anti-ARP poisoning switch application is developed in the POX OpenFlow controller, whereas [52] divided detection approaches in lower and higher resolution and not at the packet level. Low-level attacks, such as DOS and DNS amplification, need information at the flow level. In contrast, in the case of cache poisoning and ARP spoofing, higher-level attacks require packet-level information. Fig. 8 [52] depicts several attacks on network resources at various levels.

4.6. *IP Spoofing*

IP spoofing is DNS manipulation or amplification. An attacker can modify the IP DNS directory. The most typical spoofing attack strategy is to relay legal traffic to unauthorized users, which increases the risk of Man in the Middle threats. Spoofing can be prevented with the proper authentication system. Encryption mechanisms and a strong password should be used to prevent non-authentication entries.

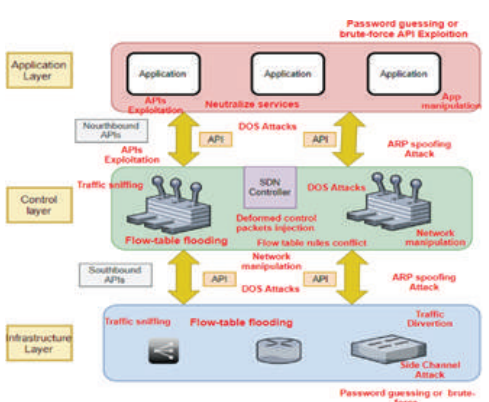

Figure 8: Different Attack Levels to abuse Network Resources

4.7. *Accountability*

In an SDN network, every controller is in charge of managing its own switches and flow tables.The interaction of different controllers in an inter-domain context isn't strengthened. Switchpackets seek to fall through the nearest controller in other controllers. Despite this, different types of cases detail the need for distinct controllers to ensure data interoperability. Controllers should exchange information with the help of well-defined interfaces to minimize interruptions and other security issues.

4.8. *Tampering*

When an attacker attempts to alter flow protocols, significant changes occur in tampering attacks such as access lists, network topological behavior, and network regulations, such as the network being shifted into disruptive mode.

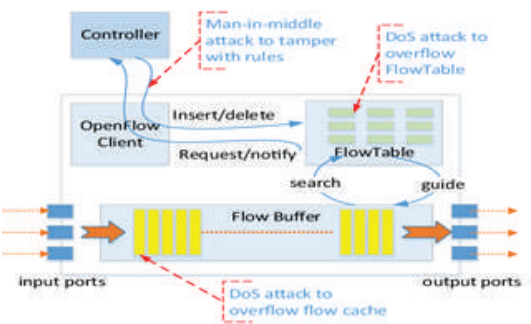

Figure 9: Tampering Attacks to change Network Topological behaviour

They include a flow table that allows or disallows access to specific hosts. An attacker can attempt to interfere with topological information, resulting in the blocking of a few processes. Diverse controllers shared crucial information in SDN controller sharing. It is extremely important to protect this communication link from being intercepted [53]. Security analysis comparisons between OpenFlow and SDN are presented in Table 1.

Within the layers of Software Defined Networking (SDN), this discussion encompasses OpenFlow, the analysis of security vulnerabilities and improvements, as well as significant research findings and contributions identified through our survey, specifically addressing security concerns in both control and data planes. Figure 9 [53] visually represents tampering attacks that manipulate network topological behaviour.

Table 1: Security Analysis comparison between other survey works and our survey

	SDN Layer/ Plane							
	Application Plane	Control Plane	Data Plane	Performance	Security Assessment	Network-wide Security	ITU Recommended	Frameworks for Security
SDN Security Study [29]	Yes	Yes	Yes	No	Yes	No	No	No
Pro. Gen. [33]	No	No	No	Yes	No	No	No	No
Security Op. Flow [30]	No	No	Yes	No	No	No	No	No
Op.Flow Vulnerability [31]	No	No	No	Yes	No	No	No	No
SDN Sec. and Depend. [32]	Yes	No	Yes	No	No	No	No	No
SDN Assessment [33]	Yes	No	Yes	Yes	No	No	No	No
Study on SDN Attacks [57]	No	No	Yes	No	No	No	No	No
Security V.N.E. [36]	No	No	Yes	Yes	No	No	No	No
Study on NFV in SDN [51]	No	Yes	Yes	No	Yes	No	No	No
Security in SDN [52]	No	No	Yes	No	Yes	No	No	No
W.M.Network in SDN [53]	No	Yes	Yes	Yes	Yes	No	No	No
What evolves with SDN [54]	No	No	Yes	No	Yes	No	No	No
Our Survey	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

5. Security Issues in SDN

Each attack is addressed below, with different security vulnerabilities in SDN categorized according to layers.

5.1. Malicious Applications

The controller holds a significant position in the SDN net, where different applications are used to abstract information from the data plane [54]. A decline in network efficiency can result from compromising a controller by a malicious application, steering controller services into an insecure state. Notably, only a limited number of researchers have delved into addressing this particular issue. The visual representation of the impact of malicious applications at both the controller and switches level can be observed in Figure 10 [55].

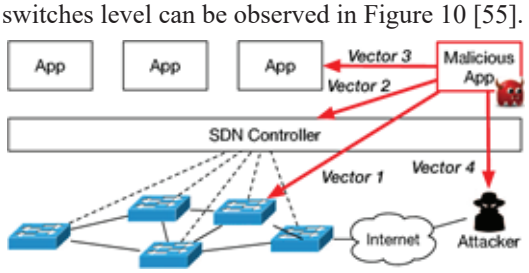

Figure 10: Malicious Applications behaviour at Controller and Switches

5.2. Data Leakage

Several potential events, such as forward, drop,

and transmit to the controller, are highlighted in the OpenFlow switch specification [56]. An attacker can detect such behaviour through packet processing time analysis. Similarly, the time a packet is required to move from the ingress towards the egress port is smaller than the time required for a packet to move from ingress port towards a controller. An attacker can monitor the switch's reactive or proactive status. An attacker can include additional network device information in a forged packet and then send many forged flows towards the controller, resulting in a Denial of Service attack. Data leaking due to a DoS attack is detailed in [57]. They were protecting the storage of credentials such as certificates and keys, which was a crucial challenge at the data plane level. It can result in data leaks, putting overall network resources at risk.

5.3. Unauthorized Access

It's a part of the access management system. A pool of controllers might be reserved or dedicated to applications with different sources. When an attacker impersonates a program or a controller, the entire network's resources and operations might be affected.

5.4. Data Modification

Controllers can program OpenFlow switches to govern traffic flow in SDN. Still, suppose an attacker takes control of the controller. In that case, fake flow rules can be inserted at the controller level to serve the attacker's interests, degrading the performance of all network services and applications. For the provisioning of virtual networks like FlowN [58], OpenVirtex [59], and FlowVisor, proper security methods must be used between each component and interface. TLS (Transport Layer Security) is specified in the OpenFlow switch for shared authentication between controllers and switches in [60], although the TLS version is not given with an optional security feature. The virtualization procedure is shown in Fig.11 [60], which connects the Operating System to the FlowVisor.- Most OpenFlow switch and controller vendors do not supply or enable TLS, which exposes attackers to man-in-the-middle attacks. An attacker can exploit communication channels and intercept messages between controllers and switches due to a lack of authentication. In the split planes design of SDN, modification of data is a major concern for researchers. A network hypervisor for Open-Flow termed FlowVisor has been found to lack adequate isolation mechanisms, allowing attackers to initiate data modification attacks on interconnecting components.

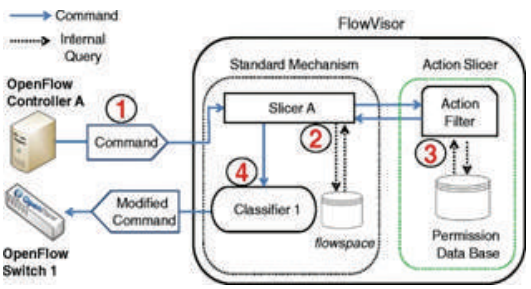

Figure 11: FlowVisor: A Virtual Network via Virtualization Process

Vulnerabilities and potential attacks are depicted in Fig.12. Several attacks are outlined below that can significantly affect the SDN network.

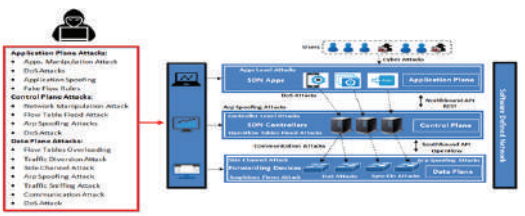

Figure 12: SDN Potential Attacks and Vulnerabilities

5.5. Configuration Issues

Network security policies are continually established and imposed on SDN layers due to the discovery of network vulnerabilities. However, due to faulty execution in the deployment situation, these security regulations provide insufficient protection. Because erroneous use of security features might impact entire layers of SDN architecture, Transport Layer Security implementation is critical for network operators in SDN networks. A large Severe attacks can occur, causing network performance to suffer [61]. OpenFlow facilitates the support of many vendor devices on the one hand, but it also creates significant weaknesses in data and control communication interfaces on the other. Multiple applications can produce policy conflicts and inconsistencies in the SDN environment as it facilitate dynamic flow policies, but this benefit may produce significant risks and attacks. Table 2 shows the numerous security vulnerabilities that can emerge in each layer.

Table 2: Various Security Issues in SDN Layers / Planes (Yes:✓No:✗)*

Security Concerns	Targeted SDN Layer				
	App. Layer	App -Control Interface	Control Layer	Control - Data Interface	Data Layer
1. Malicious Applications					
Fraudulent Rule Insertion	✓	✓	✓	✗	✗
2. Data Leakage					
Flow Rate Discovery	✗	✗	✗	✗	✓
Credential Management	✗	✗	✗	✗	✓
Forwarding Policy Discovery	✗	✗	✓	✓	✓
3. Unauthorized Access					
Unauthorized Controller Access	✗	✗	✓	✓	✓
Unauthorized or Unauthenticated Application	✓	✓	✓	✗	✗
4. Data Modification					
Packets modification through Flow Rule	✗	✗	✓	✓	✓
5. Configuration Issues					
Lack of TLS (Transport Layer Security) Adoption	✓	✓	✓	✓	✓
Implementation of Policy	✓	✓	✓	✗	✗
Lack of Secured Provisioning	✓	✓	✓	✓	✓
6. Denial of Service					
Controller- Switch Communication Flood	✗	✗	✓	✓	✓
Switch Flow Table Flooding	✗	✗	✗	✗	✓

5.6. Denial of Service

The placement of centralized controllers with the split of control and data planes in SDN infrastructure generated significant security flaws. An attacker can forward a flood of packets to a network device or controller, causing legitimate users to suffer. Flooding of the flow table at the infrastructure layer can occur due to a DoS attack to reduce memory resources [62]. In terms of a DoS attack, defines fake rule change and rule insertion. Fig.13 depicts the effect of a DoS attack on the controller and switch resources [63].

5.7. Infrastructure Plane Security Challenges

When a new user forwards packets, the centralized controller allows adding new flow rules in OpenFlow switches. Still, every switch has a limited capacity to install flow rules inside it. The main security difficulty is determining whether user-initiated rule insertions are valid or malicious. Second, a switch must buffer incoming flows requests from various users before the controller permits the insertion of flow rules, which causes saturation assaults because of restricted resources available to buffer TCP or UDP flows.

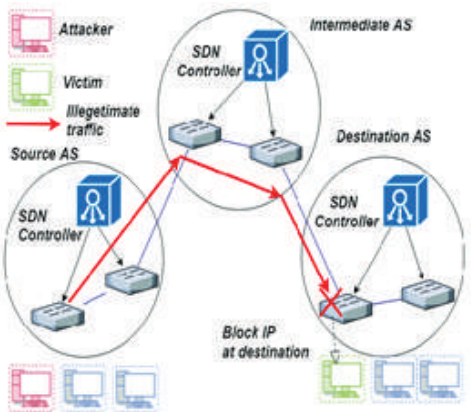

Figure 13: DoS Attack impact at Controller and Switch level resources

Entire data plane switches can be disrupted if the controller is compromised due to a black hole or man-in-the-middle attacks, so centralized controller security is essential.

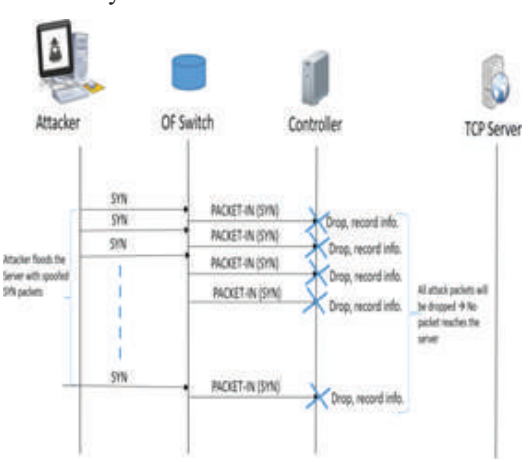

Figure 14: Attacks Ratio due to Optional Mode of Transport Layer Security

Transport layer security is used between controller and switch communication at the initial level, but later on, vendors provide it in optional mode due to higher configuration complexities, which allows for a greater number of assaults on the control channel. Various data plane security concerns, such as insertion of bogus flow rules, controller penetration, and flooding assaults, can significantly impact the data plane. Fig.14 depicts several threats caused by Transport Layer Security's optional mode [64].

5.8. Control Plane Security Challenges

The overall functionality of the network can be harmed due to vulnerable attacks on the controller. Due to a lack of authentication, auditing and permission, various attacks or threats might arise, such as applications implemented on the control plane. Before giving network resources and all information, it is necessary to classify applications according to their security elements. Below are the primary security challenges and dangers in the control plane. Various applications demand different features from the controller.

For example, a load balancing application requires all network information like packet or byte counter values to manage proper load across multiple devices. More applications, such as intrusion detection software, require each packet header field inspection. Third-party programs have varying levels of access to network resources and data. Given these constraints, a unique security method is required for different kinds of applications in the controller's north-bound API that have not been mentioned previously. Control plane saturation and flow table overloading assaults are depicted in Fig.15 [65]. To effectively handle many devices and services, employing multiple controllers is adopted, as a

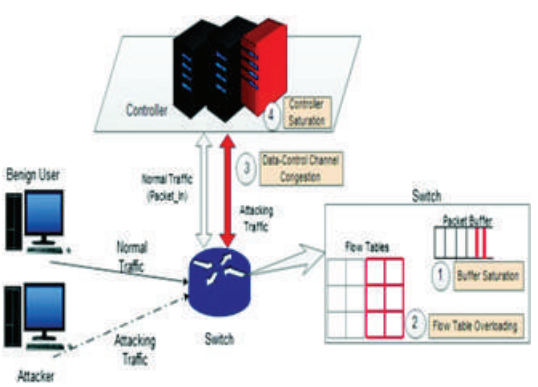

Figure 15: Control Plane Saturation and Flow Table Overloading Attacks

single controller proves inadequate. However, enforcing privacy maintenance and data aggregation standards in each sub-network becomes crucial when a software-defined network is partitioned into various sub-networks. As discussed in [65], the author found that controller implementations face challenges in managing a substantial influx of new flows in scenarios with connection limits, such as the standard 10 Gbps in high-speed networks.

The construction of a proper controller to manage simultaneous requests for flows from numerous switches while keeping the delay constraint in mind is a challenge. Multiple controllers are recommended to avoid a single point of failure, although failed controllers can increase their processing capacity, resulting in the worst-case scenario of controller failures cascading. The most dangerous security issues for controllers are DoS and Distributed DoS attacks, which render the entire network inaccessible to legitimate users. To measure flow response times between the controller and the switch, a variety of scanning instruments are available. The attacker then utilizes the flow response time value from the header field to send IP packets with intermittent headers flows requests to the controller, causing it to become unresponsive to other valid flows. DDoS attacks, such as cascading failure among numerous controllers, can compromise distributed controllers. Controller scalability while considering load and availability considerations can have a significant impact on many applications and services [65].

5.9. Application Plane Security Challenges

There is no standard to help open APIs for apps control network activities and services. Because malicious apps can significantly impact network functions, resources, and services, even security applications in the form of OpenFlow provide flow-based security detection techniques, but there is still a lack of prevention. Different autonomous development environments with diverse programming models can cause security policy collisions and interoperability constraints. Several security problems are associated with SDN applications, including the following: Application authentication is a serious concern. As various apps run on the controller to implement various functionalities with diverse vendors. Using centralized control to implement authentication for a growing number of SDN apps is a serious security problem [66].

Furthermore, there is no convincing mechanism for establishing a trustworthy relationship between distinct applications and controllers, a malicious application can cause network chaos by facilitating abstractions for underlying infrastructure access via apps. Because falsified flows can acquire credentials of authorized user and network applications. SDN layers/planes with severe security concerns are shown in Table 3. In today's fast-paced world of software engineering and evolving hacker successes, the basic difficulty in the SDN domains would be validated applications.

Table 3: Types of Security Threats with respect to SDN Layers

SDN Layer	Type of Threat	Description
Infrastructure Plane	Main in the middle attack	Complexity in configuration of TLS
	Flooding attack	Partial amount of flow rules can be stored in OpenFlow switches
	Fake flow rules insertion	More susceptible to fake flow rules
	TCP Level attacks	Transport Layer Security can be compromised
	Controller compromise	Infrastructure Layer
Control Plane	Availability and Scalability	Controller can face severe availability and scalability challenge
	DoS Attacks	Limited resource availability from control plane due to DoS attacks
	Unauthorized controller access	Malicious Applications can access controller through unauthorized way
Application Plane	Lack of accountability and access control	Accountability and access control at application plane are quite intricate to deploy
	Fraudulent flow rules insertion	Fake flow rules can be produced from malicious applications
	Lack of authentication a nd authorization	Third party applications are more disruptive due to improper authentication and authorization

Figure16 shows compromised access control policies [66]. Most authoritative plane concepts are applied by programs that run on the controller in OpenFlow. Various types of vulnerabilities and network manipulation can be occurred without proper security measures from suspicious events. Validation of applications in managing networks due to rapid growth is major security challenge with unified control layout. To ensure network applications credibility in software defined network is crucial challenge for network administrators and researchers.

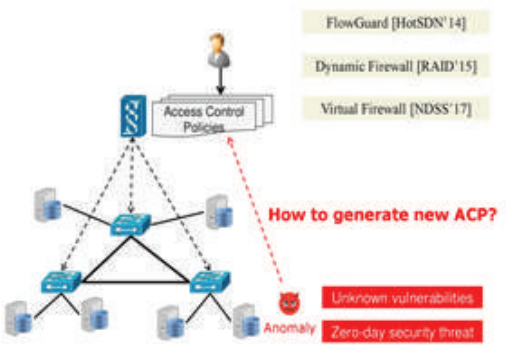

Figure 16: Vulnerable Access Control Policies

6. SOLUTIONS TO SECURITY CONCERNS IN SDN

In this section, several security solutions are discussed. For Data Leakage and Data Modification, there is no advance way of solving problem. Table 4 shows classification of research works for different security problems and the differentiation of every research work solution recognized within the types in SDN interface. Various solutions are provided for SDNs network layers through strengthening software support. A hybrid model is presented to detach the controller congestion and upgrade network regulation. Control is consolidating under standard conditions but under controller supervision installation of flow rule on other network resources in situation of massive load, network services consider the work with protective administrative control mode. The author shows signature algorithm to assure transmits flow installation request from node to node. The devices need a compact reliance manager and establish message passing and signature checking. By using Byzantine method author shows more secured SDN infrastructure for each network in [67]. Many administers designs have been suggested for adoptability as identified in [68-70]. Their qualities are emphasized as the classified mechanism inherently enhances network safety in contradiction of the consequence of illegal access. In Hyper Flow [71], a promise way is used to contemporize state between groups of controllers. Due to restricted decision-making, control plane reaction time decreases. Primary controller burden is lessened with the Soft Cell method, the authors utilize local software agents to reserve bundle classifiers and stance labels. Local deductive reasoning is distinguished from web-wide deductive reasoning in Kendo. At local controller, certain applications can be managed by event processing, minimising the main controller's load. There should be a significant deliberation during the implementation of various network resources, controllers and applications.

Table 4: Security Concerns Solutions in SDN (* Yes: ✓ No: ✗)

Solution to Security Concern	Existing Research Studies	SDN Layers				
		Application	Application – Control	Control	Control Data	Data
Unauthorized Access	SDN Distributed Control Security [68], SDN Multiple Controller [69]	✗	✗	✓	✓	✗
	Case of SDN [70	✗	✗	✓	✗	✗
	Secure Controller Platform [71]	✓	✓	✗	✗	✗
	SDN Application Control [72]	✓	✓	✓	✗	✗
	SDN Control Layer [73]	✓	✓	✓	✓	✗
	Access Control Mechanism in SDN [73]	✓	✗	✓	✓	✓

Malicious Applications	Secured Kernel in SDN [74]	✓	✓	✓	✓	✗
	Robust and Secure NOS [74]	✓	✗	✓	✗	✗
	SDN Distributed Control Security [68], SDN Multiple Controller [69]	✓	✓	✓	✗	✗
Denial of service	Scalable and Vigilant flow management in SDN [75], CP-Recovery [76]	✗	✗	✓	✓	✓
	Secure Address Validation in SDN [75]	✓	✗	✓	✓	✓
	Network Security Improvised Method [76]]	✓	✓	✓	✓	✓
	OpenFlow Application [77]	✓	✓	✗	✓	✗
	Federated OpenFlow Infrastructures [78], Invariant Security in SDN [79],Data plane debugging [80], Network wide invariant [81], Real time network Policy [82]	✓	✓	✓	✓	✗
	OpenFlow Firewall Application [82], Reliable SDN Firewall [83], Layered policy management [84]	✓	✗	✓	✓	✓
	Network Programming Language [85], Security Policy SDN [86], Change you can believe in SDN [87]	✓	✓	✓	✓	✗
	Consistent and Fault-Tolerant data store for SDN [88]	✓	✗	✓	✓	✓
	Language-Based Security for SDN [89]	✗	✓	✓	✓	✗
System Level SDN Security	SDN Debugger [91]	✓	✗	✗	✓	✗
	Enabling Secure Mobility in SDN [91], Secure Control Channel [92]	✗	✗	✗	✓	✗
	Composable Security Services for SDN [92]	✓	✓	✓	✓	✗
Data Leakage		n/a				
Data Modification		n/a				

To assure that only trusted applications linked to the internet that's why both consent and authentication of the applications are significant. To enforce permission at API entry, author proposed secure platform for controller application PermOf [93] with isolation mechanism. To prevent network from attack by application with minimum privilege. Floodlight controller is used to implement operation checkpoint. Floodlight improvement FortNOX, an extension of floodlight developed by Stanford Research Institute, commences a digitally authenticated northbound API. Granularity of approach is varying among operation check point, Perm Of

and SE-floodlight. Instead of validation, operation checkpoint and PermOF permit actions to be granted on application. A RADIUS server is deployed with an Open Flow controller in Authflow [93]. All the layers of SDN architecture control unauthorized access to SDN. The controller and application should establish a trusted connection before exchanging messages to avoid malicious applications. To probe security authorization of Open Flow application role base authentication in the FortNOX. The FortNOX enforcement manages the conflicts, whereas the acceptance and rejection depend upon the author's security control. OpenFlow applications, OF security, and OF operators are three flow rule producers. FortNOX does not solve the issue of application, which is a significant limitation. ROSEMARY [94] a high-performance NOS, is introduced to enhance reliance of a control network. This micro-NOS architecture is improved and every OpenFlow application is executed with an autonomous instance of ROSEMARY. The resolution splits the network application from trusted computing based on NOS. With LegoSDN [95] the SDN consider the application failure and consumer reliability. Separating control and data plane in SDN invited DoS attack at the controller level. By using the connection migration tools AVANT and GUARD [96], a solution to overcome this weakness is proposed. A preventive shield against IP spoofing can cause a DoS attack, as presented in VAVE [97]. The authors use SDN's dynamic rules updates and traffic analysis functionality to protect against IP spoofing. Ident++ protocol is applied to protect against DoS attacks. Configuration issues and numerous resolutions to the problem of implementing different network policies through applications are discussed. Fig.17 shows FortNOX with no proper security for applications. VeriFlow [98] is developed to identify the loops in routing tables and the network is represented as a graph. In [98], header space analysis is applied for conflict detection at application level in SDN firewall. To select proper firewall policy during updating network state by flow path is provided in most recent flow guard [99-100]. Language based resolutions is a policy conflict which is resolved by a particular northbound Application Program Interface (API) called Frenetic [101]. A summary is presented in Table 5 for unauthorized access, malicious application and denial of service concerns in SDN environment. A policy conflict resolution

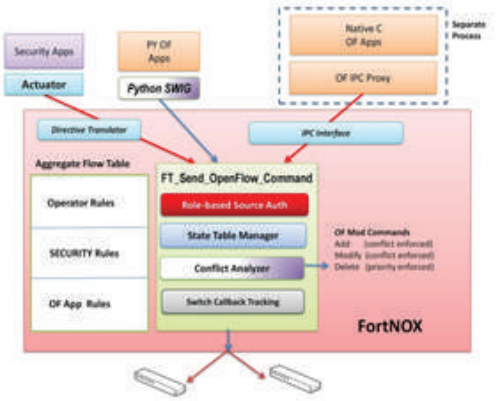

Figure 17: FortNOX with no proper Security for Applications

procedure between the control and application layer is introduced in [101], which can make network programming understandable to evade policy conflicts. The concept of per packet and per-flow invariability posits that each packet follows either an advanced or a former policy during an update operation, but not both simultaneously, as stated in [102]. To address this, an organization suggests a layered policy management solution [103], emphasizing a shared architecture over policy conflict resolution techniques. An alternative approach, presented in [104], proposes maintaining consistent network views without sacrificing activity. This involves the development of splendid isolation, ensuring the separation of program traffic in network slices. In the context of flow space detection, the flow visor introduced in [105] acts as a transparent proxy between switches and controllers, aiding administrators in identifying slices of flow space. Flow Visor doesn't implement the action isolation process, as defined in [105], which is addressed by various solutions suggested in the same reference. A formal certificate tool validates open-flow network correctness and movement-related characteristics. This tool adds specification modelling validation to ensure the integrity and accuracy of the open-flow network.

Integrating IPSec with OpenFlow, known as OFHIP, combines OpenFlow and the Host Identity Protocol (HIP). This fusion provides a solution for secure portability. HIP introduces a cryptographic name space, as detailed in [106], aligning with the IPv6 address space. FRESCO is a significant advancement in this realm, emphasizing swift and effective design and security development. FRESCO is primarily designed

to address network attacks efficiently. It stands out as a crucial contribution, establishing a library for detecting network attacks. The overarching goal is to create a robust and rapid system that ensures security and facilitates the quick identification and mitigation of potential threats.

Table 5: Problem and Proposed Solution for Unauthorized Access, Malicious Application and DoS Issues in SDN

Research Work	Problem	Proposed Solution
Unauthorized Access		
Authentication and Access Control Mechanism [97]	Avoid SDN from unauthorized users.	User verification is done through access control and authentication processes.
Case of SDN [98]	How to manage SDN infrastructure for more enhanced security.	To minimize points of severe failure in controllers
SDN Application Control [99]	Controller accessible for application	To secure the controller from malicious application
SDN Distributed Controller Security [100]	How to prevent distributed controllers from malicious utilization	Flow installation rules through the signature algorithm
SDN with Multiple Controllers [100]	Control Plane prevention from malicious attacks	Byzantine-Fault Tolerant algorithm
Secure Controller for OpenFlow Applications [101]	OpenFlow is accessible for application with complete rights	Allocate the most minor rights to applications with the new technique
Secure SDN Control Layer [101]	High-security concerns between control and application plane communication	Secure SDN Control layer through authorization.
Malicious Application		
Robust and Secure NOS [102]	Prevent application failures to avoid network control	Controller with Robust and Secure Network Operating System
Security Kernel OpenFlow networks [103]	Flow rules conflicting from applications	Security based kernel implementation for prioritizing purposes.
Tolerating SDN application failures [104]	Network and controller flexibility for application malfunction	Novel controller infrastructure to improvise network management
Network Security [104]	Eradicate network overhead	Network security policy through Ident++ protocol
Address Validation Solution with Open Flow [105]	Verification of Source Address	OpenFlow address validation through NOXController
Scalable and Vigilant switch flow management in SDN [105]	DoS attack prevention from dynamic changing flow	Scalable and Vigilant solution for switch flow management through connection migration
Replication Component SDN [106]	Secure SDN from DoS attack	Replication component for resilient OpenFlow-based networking

6.1. Infrastructure Plane Security Solutions

Ensuring the security of data and infrastructure plans against potential threats from malicious applications is paramount, especially those capable of modifying or introducing new flow rules. Implementing robust authorization and authentication mechanisms for these applications is crucial to controlling their ability to alter flow rules. FortNox is integrated into the NOX OpenFlow controller [106] to address this concern, providing authentication for flow rules to prevent any unauthorized access.

A configuration confirmation tool named FlowChecker is employed to identify and rectify potential errors in OpenFlow rules at the switch level. This tool validates OpenFlow rules, ensuring that they conform to the intended configurations and helping mitigate the risk of unauthorized modifications or installations by malicious applications. FlowChecker is employed as an OpenFlow application to uphold, examine and authorise OpenFlow end-to-end run time setup. VeriFlow, a network debugging tool discussed, detects aberrant network behaviour caused by falsified rules introduced by SDN applications. In situations where the primary controller experiences a disruption, [106] introduces a controller replication strategy to ensure the continued operation of switches. This involves switches intermittently forwarding messages to the controller, and if the controller fails to respond, the switch assumes the controller is in a down state. Subsequently, the switch promptly attempts to communicate with a secondary controller through a handshake process.

To enhance overall system efficiency, considering constraints on delay and the need for rapid restoration and instant security analysis as proposed in [107], minimising the length of paths between controllers and switches is recommended. This optimization helps achieve quicker communication, reduce delays, and facilitate faster recovery in network disruptions.

6.2. Control Plane Security Solutions

Control plane security solutions are classified using different techniques to secure from malicious applications. To secure management or control plane from malicious applications is essential for network operators as network resources are accessible though these applications. So, unauthorized applications are required to be critically monitored. The SE-Floodlight controller works as an intermediary between applications and the information plane. It allocates validation roles to OpenFlow applications to resolve rule conflicts. To achieve high scalability through a balanced load among controllers is proposed. Algorithms are implemented to estimate wildcard rules and automatically adjust to change load balancing policies. A parallelism approach through multi-core processors is implemented to acquire high availability and scalability. To provide a control plane distributed in SDN, as heterogeneous and overlay network is shown. On top of floodlight, DISCO is implemented, composed of two elements: inter and intra-domain. An event-based scalable control platform, Hyper Flow, is proposed to help network administrators minimize set-up time and maximize controller scalability. Heller et al. introduced a tradeoff to minimize latency and place controllers appropriately [106]. Different techniques are proposed to enhance control plane scalability in SDN. Using Self Organizing Maps (SOM), lightweight DDoS flooding attack detection is proposed in [106], in which outlines of data with similar statistical properties are collected for more processing. Different modules in the classifier are used to detect DDoS attacks. The controller placement problem is highlighted in terms of the topological locations of the controller in SDN for network resilience and scalability. Various algorithms are proposed by the research community related to controller placement, like generic probabilistic algorithm and Simulated Annealing (SA) algorithm. A minimum cut-based graph portioning algorithm is projected for adequate controller placement to improve network resilience. Controller placement to meet response time requirements with NP-hardness is illustrated, which improves network efficiency by minimising control path loss. Dynamic controller provisioning problem (DCPP) is described as arranging multiple controllers in WAN where network dynamics are used to adjust the controller's number and location. An integer linear program that uses traffic patterns is formulated as DCPP to reduce switch state collection costs. A single controller might have some restrictions to meet the demands of network resilience, whereas multiple controllers perform better. So, in this way, the authors suggested POCO (Pareto-based optimal controller placement) to improve network efficiency in load balancing and latency among controllers. DevoFlow is an approach that reduces control and data plane interaction, facilitating controller availability. Per-flow validation from

the controller is unnecessary, and switches can take local routing decisions using wild card OpenFlow rules. A controller can gather this information by utilizing the Link Layer Discovery Protocol (LLDP) as outlined in OpenFlow to determine the connectivity status of OpenFlow switches, including details about port connections. This approach facilitates the acquisition of network topology data, enabling the observation of links within the network, particularly for fault management scenarios. An architecture is proposed to eliminate the link monitoring function of Operation Administration Maintenance (OAM) from the controller to the switch. The work focuses on enhancing network security services to enhance the overall security of Software-Defined Networking (SDN). This involves the centralized control objective of ensuring secure cloud computing and incorporating new security services within MPLS in open-flow networks. A comprehensive breakdown of SDN security solutions is presented in Table 6, categorizing them based on their respective planes. This table highlights the positive implications associated with security solutions at different layers within the SDN architecture.

Table 6: SDN Layers / Planes with Security Solutions

SDN Layer	Target Threat	Solution Type	Security Solution
Infrastructure Plane	Availability of Controller	Controller replication framework	Replication Component in OpenFlow [93]
	Access Control	Policy enforcement and Access control framework	Dynamic Access Control [94]
	Fake flow rules	Network debugging tools	Federated OpenFlow Infrastructures [95]
	Fraudulent flow rules	Configuration verification tool	Verifying network-wide invariants [96]
	Conflicts in flow rules	Controller framework	Security Enforcement kernel for OpenFlow networks [97]
Control Plane	Controller scalability	Hybrid controller infrastructure	Comparing Openflow controller [98]
	Controller scalability	Distributed controller infrastructure	Distributed multi-domain SDN controllers [99]
	Controller availability	Distributed control plane	Controller placement Problem [100]
	Controller availability	Controller placement framework	Distributed control plane f or OpenFlow [101]
	Apps. Authorization	Secure App-Ctrl API	Security-enhanced floodlight [101]
	DDoS attack	Detection Framework	Lightweight DDoS flooding [102]
Application Plane	Programs Error	Apps. testing framework	Testing of OpenFlow applications [103]
	Conflicts in Flow rules	Apps. Debugging framework	Debugging SDN applications [104]
	Threat from Application	Security apps. Development framework	Modular, composable security services in SDN [105]
	Violation of Security Policy	Application to ensure Security Policy	Invariant security properties in OpenFlow [106]
	Access Control	Application Permission System	Secure controller platform for OpenFlow [106]

6.3. Application Plane Security Solution

Various programming languages like NetCore, Fentic, and Procera are proposed for SDN applications. A secure application, FRESCO, has been introduced for OpenFlow controllers and switches. Perm-OF's authorisation system provides a controlled approach with data representation for the OpenFlow application. VeriFlow is proposed to probe flow rules in run time. To ensure that the collection of different flow methods does not create any conflicts in network security policies and to probe flow rules in run time, Flover is proposed. The author proposes an associated gradual, automatic validation technique for characteristic bugs in open-flow programs. The NDB system provides a rectifying tool for a network computer user to probe for clarification for network errors. OFRewind is proposed and implemented with an improvisation approach to discriminate network irregularities. NDV OFRewind system will be acclimated to distinguish those applications or services that initiate security threats amid a network.

7. FUTURE DIRECTIONS

Network programmability, centralized intelligence and visibility at the global state of the network are enhanced in SDN. A mutual distribution layer gathering information regarding security prerequisites of diverse services hosts, resources and security command execution to the network essentials to apply security rules can result in overwhelming and open security execution, the same critical qualities of SDN that's centralized insights and programmable network components makes a reservation in SDN an additional curiously activity. Numerous security resolutions and platforms have been portrayed in past areas. A few grey zones still have to be caught on and legitimately adjusted, especially recently in the commercial deployment of SDNs.

7.1. Identity Location Split

One of the primary challenges the internet confronts is unauthorized user access. Jennifer Rexford, a proponent of the Clean Slate Internet Architecture, highlights the issue of security vulnerabilities stemming from unsecured identities. This same concern translates to Software-Defined Networking (SDN), where insufficient procedures exist for verifying user identities. While OpenFlow provides methodologies for understanding data flows, none of these methods adequately address identifying users. Using IP addresses can reveal a host's identity but poses limitations due to language discrepancies, reducing flexibility. Additionally, relying on IP addresses within OpenFlow necessitates frequent and rapid updates, leading to additional overhead. Hence, exploring Host Identity Protocol (HIP) in the context of SDN becomes crucial. HIP offers a potential solution by providing identities and endorsing end-to-end encryption bolstered by enhanced security measures.

7.2. Scalability and Security

Scalability is one of the imperative demanding situations confronted through coherently centralized SDN design. In SDNs, as the community develops, the sum of mange traffic ordained towards the centralized controller will increment as the flow installation time rises. Besides, it is famous that an open Flow controller's potential related operation set is conceivably confined. Subsequently, the shortage of measurability in SDN will alter targeted attacks by overwhelming communication amid the switches to cause control plane immersion. Availability may be a security estimation that emphatically connects scalability with security. Mainly, intrusion in SDN points to compromise accessibility. In this way, more than one controller is prescribed, but including one or two controllers might result in controllers' cascading failures.

7.3. Programming and Development Models

SDN enables designers to design and use novel networking architectures, applications and protocols. This ability will produce innovation but can announce security challenges with the execution of many applications in a network. Without dependence and connection, several development and control platforms will develop characteristics without applications on possibly the same but forwarding substances and produce severe security tasks. It is essential to develop standardized and proper programming models and paradigms to reduce the probabilities of incompatible modules that do security harm. Conflicting modules in a system can produce security vulnerabilities like revealing sensitive network information and producing contradictory flow statements.

7.4. Network Security Automation

Monitoring communication networks is much more challenging and tends to be inaccurate. At the same time, there are challenges in detecting

and obliging to relevant changes within the network. The control loop for the framework is to memorize and modernize itself for coming acts in traditional systems, which has decreased the chance of automation procedures. Hammed et al. showed that a manual framework of network security innovations, incorporating firewall and IPSec technologies on figurative sets of devices, tends to figurative errors, intra and inter-policy conflicts sequencing in genuine security vulnerabilities and dangers concurring to the context of network security. Improper configuration conflicts during firewall setup are one of the foremost major concerns for network security infringement. ONF (Open Networking Foundation) pronounces that SDNs propose adaptable network automation and management framework, which makes it realizable to expound tools to automate administration activities to downgrade operational overhead, decrease network insecurity displayed by administrator mistakes and bolster recently made self-services provisioning models[106].

The Procera system automates policy implementation, while the OMNI platform is employed for a responsive and automatic control framework. However, it is essential to note that there is a lack of demonstrated components for automating security in Software-Defined Networking (SDN). This situation burdens network administrators, relying on their expertise and availability to act as a safety net for the network. Consequently, there is a pressing need for automated security approaches in SDNs that minimize the reliance on administrator intervention. The goal is to ensure network security and initiate recovery processes automatically. The graphical representation in Figure 18 illustrates the current trends and advancements in SDN.

7.5. Synchronization of Network Security and Network Traffic

The essential segment of network control is network security, in which stable and study security coverage arrangements call for worldwide analysis of coverage configuration of all the network elements to circumvent conflicts in the protection techniques and decrease the possibilities of significant security cracks and network susceptibilities. Even though cooperative policies and approaches had been recommended in diverse initiatives. The fundamental motive at the backend of this loss is cooperation among safety and site visitors control, as loosely coupled control planes of sending devices, impartial security models, rules and security independent routing. Consequently, conventional IP networks have the properties of insecurity and intricacy where a minor improper configuration of a routing protocol may seriously impact networks. For example, a DoS attack at the centralized controller in OpenFlow networks would at least cause a boom delay in installing coast rules inside the open-flow switches. Comparable to distinctive networks in SDNs, visitors float caution highlights may be utilized to find designated DoS attacks. However, the worldwide network visibility inside the centralized SDN management plane and the programmability of the forwarding plane can permit the deployment of agreeable and interdependent rules.

7.6. Control Data Planes Intelligence Tradeoff

The presence of a single controller or a group of controllers results in latency between nodes or between nodes and controllers when exchanging network information. This leads to challenges in logically centralized Software-Defined Networking (SDN) architectures, as discussed in reference

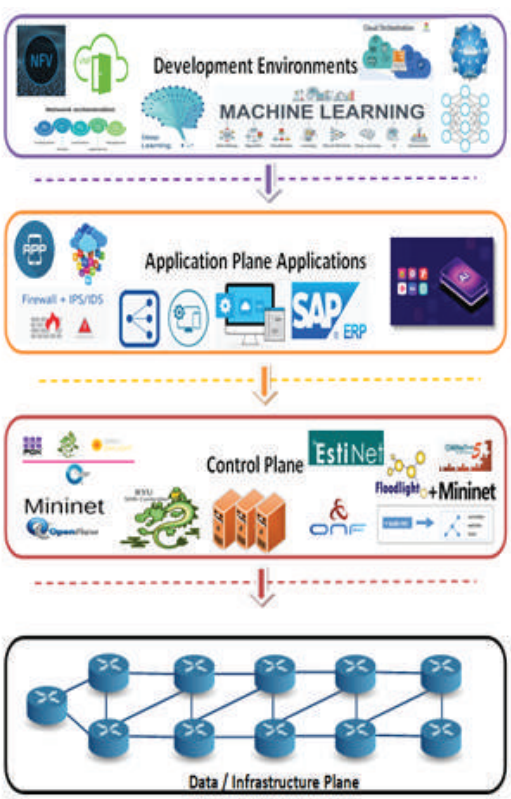

Fig.ure18: Current Trends of Development in SDN Networks

[106]. One of the issues is the increased number of flow setup requests, which, in turn, escalates the workload on a single controller and consequently extends the response time for establishing flow rules, as outlined in [106]. To address these challenges in SDNs, an intelligent tradeoff is needed. This involves enhancing the OpenFlow architecture's efficiency and reducing delays associated with local decision-making capabilities. The goal is to improve accessibility and enable swift network recovery processes simultaneously. Balancing intelligent design enhancements and minimizing delays is crucial for optimizing SDN performance.

7.7. Class-Based Application Security

Network functionalities are implemented as applications over the control plane in the SDN environment. Network statistics are observed for load balancing in some applications, whereas some applications need packet trials. Some applications require different access and functional requirements, as discussed. With the support of a control plane, applications can access network resources, packet trials and overall network statistics more appropriately. Some applications require a direct method or procedure to network resources and statistics. The control plane provides statistics to applications depending on its own information that might be disconnected from the real network. Therefore, new application trends and evolution that give an immediate approach to various applications may create legal disputes. Therefore, it is mandatory to differentiate applications in various groups according to underlying network resources and their functions. Secondly, security policies must be defined to acquire network resources. The excessive amount of incoming requests to avail network resources from each application with authentication creates massive overhead to the control plane, which is a crucial challenge for network administrators and must be focused on. Other challenges for SDN security include DDoS attacks mitigation, Multipath in BGP with route hijacking problems, Policy Forwarding, Cloud Services, Network Function Virtualization Standardized Interfaces issues, Flow Tables Exhausted Issues, QoS Issues in SDN, Distributed Controllers Security in SDN, MPLS Routing in SDN, Data Synchronization in Distributed Controller's Environment, Load Balancing and Middle Boxes Deployment Issues needs to be targeted at more intensive level before SDN deployment in organizations.

8. CONCLUSION

The advent of SDN, an emerging architecture with unprecedented control in a network environment and reprogrammable technology, brings resilience to network management at a large scale. Still, several numbers of queries are disseminated among the research community about its three planes or layers of security with ambiguous and indistinct responses. With several advancements and features, SDN infrastructure also invited vulnerabilities in data, control and application planes. By using synthetic, existing standard available datasets and real-time datasets, various numbers of researchers are focused on authentication mechanisms in terms of digital signature, encryption techniques and feature selections by using artificial intelligence, machine learning, convolutional neural network, long-short-term memory, deep learning, recurrent neural network and auto-encoders approaches which have shown efficient results in this domain. This article presented early programmable networks before the SDN invention and why all these improvisations were essential in computer networks. The properties of software-defined network architecture are shown. Several potential attacks at SDN layers, inconsistent policies, and security analysis are discussed, along with preventive measures against these prevailing attacks. They kept critical security controls like middle-boxes, spam detectors, and IDS in view. IPS, Firewall and policy management, and several numbers of security vulnerabilities or challenges in SDN planes are addressed in this paper after having intensive articles analysis on SDN security in accordance with research groups dimensions and then we showed various security solutions for each plane of SDN. Lastly, we illustrated various future research directions and challenges for SDN research community to observe all these highlighted and discussed aspects before SDN deployment with effective security solutions.

REFERENCES

[1] S. Natarajan et al., "A Software defined Cloud- Gateway automation system using OpenFlow," *in 2013 IEEE 2nd International Conference on Cloud Networking (CloudNet),* pp. 219–226, Nov 2013.

[2] S. Jain et al., "B4: Experience with a globally-deployed software defined wan," *in*

Proceedings of the ACMSIGCOMM 2013 conference. ACM, vol. 43, no. 4, pp. 3–14, 2013.

[3] P. Patel et al., "Ananta: Cloud Scale Load Balancing," *in Proceedings of the ACM SIGCOMM2013 Conference on SIGCOMM. ACM,* vol. 43, no. 4, pp. 207–218, 2013.

[4] D. Clark et al., "New Arch: Future generation Internet architecture," *Def. Tech. Inf. Center (DTIC), Fort Velvoir, VA, USA, Tech. Rep. AFRL-IF-RS-TR-2004-235,* 2004.

[5] Z. Cai et al., "Maestro: A system for scalable OpenFlow control," *RiceUniv., Houston, TX,USA, Tech. Rep. TR10-08,* 2010.

[6] S. Murphy et al., "Strong security for active networks," *in Proc. IEEE Open Archit. Netw. Programm.,* pp. 63–70, 2001.

[7] M. Casado et al., "Ethane: Taking control of the enterprise," *ACM SIGCOMM Comput. Commun. Rev.,* vol. 37, no. 4, pp. 1–12, Oct. 2007.

[8] M. Liyanage et al., "Software Defined Mobile Networks (SDMN): Beyond LTE Network Architecture," *Hoboken, NJ, USA: Wiley,* 2015.

[9] T. Nadeau, "Software driven networks problem statement," *Network Working Group Internet-Draft,* 30 Sep 2011.

[10] H. Xie, T. Tsou, D. Lopez, H. Yin, and V. Gurbani, "Use cases for ALTO with software defined networks," *Working Draft, IETF Secretariat, Internet-Draft,* 2012.

[11] N. Gude et al., "NOX: towards an operating system for networks," *ACM SIGCOMM Computer Communication Review,* vol. 38, no. 3, pp. 105–110, 2008.

[12] D. Erickson, "The beacon openflow controller," *in Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, ACM,* pp. 13–18, 2013.

[13] X. Jin et al., "Softcell: Scalable and flexible cellular core network architecture," *inProceedings of the ninth ACM conference on Emerging networking experiments and technologies, ACM,* pp. 163–174, 2013.

[14] M. Paliwal et al., "Controllers in SDN: A Review Report," *IEEE Access,* pp.36256-36270, 10.1109/ACCESS.2018.2846236, 2018.

[15] N. McKeown et al., "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review,* vol. 38, no. 2, pp. 69–74, 2008.

[16] W. Hassan and T. Omar, "Future Controller Design and Implementation Trends in Software Defined Networking," *Journal of Communications.* vol. 13, no. 5, pp. 209-217, 10.12720/jcm.13.5.209-217, 2018.

[17] D. S. Rana et al., "Software Defined Networking (SDN) Challenges, issues and Solution," *International journal of computer sciences and engineering,* vol. 7, pp. 884-889. 10.26438/ijcse/v7i1.884889, 2019.

[18] S. H. Yeganeh and Y. Ganjali, "Kandoo: a framework for efficient andscalable offloading of control applications," i*n Proceedings of the first workshop on Hot topics in software defined networks, ACM,* pp. 19–24, 2012.

[19] B. Goswami, Bhargavi et al., "Implementation of SDN using OpenDayLight Controller," 2017.

[20] P. Berde et al., "ONOS: towards an open, distributed SDN OS," *in Proceedings of the third workshop on Hot topics in software defined networking, ACM,* pp. 1–6, 2014.

[21] D. Kreutz et al., "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE,* vol. 103, no. 1, pp.14-76, 2014.

[22] S. Namal et al., "Implementation of OpenFlow based cognitive radio network architecture: SDN&R," *in Wireless Networks. New York, NY, USA: Springer,* pp. 1–15, 2015.

[23] M. Liyanage et al., "Software Defined Mobile Networks (SDMN): Beyond LTE Network Architecture," *Hoboken, NJ, USA: Wiley,* 2015.

[24] S. J. Vaughan-Nichols, "OpenFlow: The next generation of the network?" *Computer*, vol.

44, no. 8, pp. 13–15, Aug. 2011.

[25] T. Nadeau, "Software driven networks problem statement," *Network Working Group Internet-Draft,* 30 Sep 2011.

[26] S. Shin and G. Gu, "Attacking Software-Defined Networks: The First Feasibility Study," *in Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, ACM,* pp. 165–166, 2013.

[27] S. Scott-Hayward et al., "SDN Security: A Survey," *in IEEE SDN for Future Networks and Services (SDN4FNS)*, pp. 1–7, 2013.

[28] R. Kloeti et al., "OpenFlow: A Security Analysis," *In 2013 21st IEEE International Conference on Network Protocols (ICNP),* pp. 1-6, IEEE, April 2013.

[29] K. Benton et al., "OpenFlow Vulnerability Assessment," *in Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, ACM,* pp. 151–152, 20139.

[30] D. Kreutz et al., "Towards secure and dependable software-defined networks," *in Proceedings of the second ACMSIGCOMM workshop on Hot topics in software defined networking, ACM,* pp. 55–60, 2013.

[31] D. Li et al., "Evaluation of Security Vulnerabilities by Using ProtoGENI as a Launch pad," *in Global Telecommunications Conference (GLOBECOM 2011), IEEE,* pp. 1–6, 2011.

[32] R. L. Smeliansky, "SDN for network security," *in Science and Technology Conference (Modern Networking Technologies) (MoNeTeC), 2014 First International, IEEE,* pp. 1–5, 2014.

[33] L. Schehlmann et al., "Blessing or curse? Revisiting security aspects of Software-Defined Networking," *in Network and Service Management (CNSM), 2014 10th International Conference on, IEEE,* pp. 382–387, 2014.

[34] A. A. Barakabitze et al., "5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges," *Computer Networks,* vol. 167, pp. 106984, Feb 2020.

[35] A. Y. Ding et al., "Software defined networking for security enhancement in wireless mobile networks," *Computer Networks,* vol. 66, pp. 94–101, 2014.

[36] M. Tsugawa et al., "Cloud Computing Security: What Changes with Software-Defined Networking, ser. Secure Cloud Computing," *Springer,* pp. 77–93, 2014.

[37] S. Hernan et al., "Threat modeling uncover security design flaws using the stride approach," *MSDN Magazine-Louisville,* pp. 68–75, 2006.

[38] R. Zhao et al., "Deep reinforcement learning based mobile edge computing for intelligent Internet of Things," *Physical Communication,* vol. 43, pp. 101184, Dec 2020.

[39] K. Benton et al., "OpenFlow Vulnerability Assessment," *in Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, ACM,* pp. 151–152, 2013.

[40] D. Kreutz et al., "Towards secure and dependable software-defined networks," *in Proceedings of the second ACMSIGCOMM workshop on Hot topics in software defined networking, ACM*, pp. 55–60, 2013.

[41] D. Li et al., "Evaluation of Security Vulnerabilities by Using ProtoGENI as a Launch pad," *in Global Telecommunications Conference (GLOBECOM 2011), IEEE,* pp. 1–6, 2011.

[42] S. Shin and G. Gu, "Attacking Software-Defined Networks: The First Feasibility Study," *in Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, ACM,* pp. 165–166, 2013.

[43] R. L. Smeliansky, "SDN for network security," i*n Science and Technology Conference (Modern Networking Technologies) (MoNeTeC), 2014 First International, IEEE,* pp. 1–5, 2014.

[44] L. Schehlmann et al., "Blessing or curse? Revisiting security aspects of Software-Defined Networking," *in Network and Service Management (CNSM), 2014 10th International Conference on, IEEE,* pp. 382–387, 2014.

[45] V. T. Costa and L. H. M. K. Costa, "Vulnerability Study of FlowVisor based Virtualized Network Environments," *in 2nd Workshop on Network Virtualization and Intelligence for the Future Internet,* 2013.

[46] N. Panwar et al., "A survey on 5G: The next generation of mobile communication," *Physical Communication*, vol. 18, pp. 64–84, Mar 2016.

[47] M. Tsugawa et al., "Cloud Computing Security: What Changes with Software-Defined Networking," s*er. SecureCloud Computing, Springer*, pp. 77–93, 2014.

[48] R. Ducato, "Data protection, scientific research, and the role of information," *Computer Law & Security Review,* vol. 37, pp. 105412, July 2020.

[49] R. H. Weber, "Internet of Things – New security and privacy challenges," *Computer Law & Security Review,* vol. 26, no. 1, pp. 23–30, January 2010.
.

[50] G. Yao et al., "Source address validation solution with OpenFlow/NOX architecture," *In: Proceedings of the IEEE International Conference on Network Protocols (ICNP),* V*ancouver, BC, Canada,* pp. 7-12, 17-20 October 2011.

[51] A. Kalyaevet al., " A hardware approach for organisation of software defined network switches based on FPGA," *In 2015 International Conference on Informatics, Electronics & Vision (ICIEV),* pp. 1-4, IEEE, 2015.

[52] A. Zaalouk et al., "OrchSec: an orchestrator-based architecture for enhancing network-security using Network Monitoring and SDN control functions," *In NOMS,* pp. 1-9, 2014.

[53] F. Salahdine et al., "A survey on compressive sensing techniques for cognitive radio networks," *Physical Communication,* vol. 20, pp. 61–73, Sep 2016

[54] R. Sherwood et al., "Flowvisor: A network virtualization layer," *OpenFlow Switch Consortium, Tech.Rep,* pp.132, 2009.

[55] R. W. Skowyra et al., "Verifiably safe software defined networks for CPS," *in Proceedings of the 2ndACM international conference on High confidence networked systems*, *ACM,* pp. 101–110, 2013.

[56] J. McCauley et al., "Extending SDN to large-scale networks," *Open Networking Summit,* pp. 1–2, 2013.

[57] A. Al-Shabibi et al., "OpenVirteX: make your virtual SDNs programmable," i*n Proceedings of the third workshop on Hot topics in software defined networking. ACM,* pp. 25–30, 2014.

[58] F. A. Botelho et al., "On the feasibility of a consistent and fault-tolerant data store for SDNs," *in Software Defined Networks (EWSDN), 2013 Second European Workshop on. IEEE,* pp. 38–43, 2013.

[59] T. Hu et al., "SAIDE: Efficient application interference detection and elimination in SDN," *Computer Networks,* vol. 183, pp. 107619, Dec 2020.

[60] V. T. Costa and L. H. M. K. Costa, "Vulnerability Study of FlowVisor based Virtualized Network Environments," *in 2nd Workshop on Network Virtualization and Intelligence for the Future Internet,* 2013.

[61] S. Sezer et al., "Are we ready for SDN? Implementation challenges for software-defined networks," *Communications Magazine, IEEE,* vol. 51, no. 7, 2013.

[62] S. E. Quincozes et al., "A survey on intrusion detection and prevention systems in digital substations," *Computer Networks,* vol. 184, pp. 107679, Jan 2021.

[63] H. Li et al., "Byzantine-resilient secure software defined networks with multiple controllers," *in Communications (ICC), 2014 IEEE International Conference on IEEE,* pp. 695–700, 2014.

[64] P. Fonseca et al., "A replication component for resilient OpenFlow-based networking," *in Proc. IEEE NOMS,* pp. 933–939, Apr 2012.

[65] X. Wen et al., "Towards a secure control-

ler platform for OpenFlow applications," *in Proc. 2ndACM SIGCOMM Workshop Hot Topics Softw. Defined Netw.,* pp. 171–172, 2013.

[66] K. Giotis et al., "Combining OpenFlow and sFlow for an effective and scalable Anomaly Detection and Mitigation mechanism on SDN Environments," *Computer Networks,* 2013.

[67] G. Malgieri and J. Niklas, "Vulnerable data subjects," C*omputer Law & Security Review,* vol. 37, pp. 105415, Jul 2020.

[68] X. Jin et al., "Softcell: Scalable and flexible cellular core network architecture," *in Proceedings of the ninth ACM conference on Emerging networking experiments and technologies, ACM,* pp. 163–174, 2013.

[69] R. Sherwood et al., "Flowvisor: A network virtualization layer," *OpenFlow Switch Consortium, Tech.Rep,* pp.132, 2009.

[70] A. Y. Ding et al., "Software defined networking for security enhancement in wireless mobile networks," *Computer Networks,* vol. 66, pp. 94–101, 2014.

[71] A. Guha et al., "Machine-verified network controllers," *in ACM SIGPLAN Notices,* vol. 48. ACM, pp. 483–494, 2013.

[72] A. Zaalouk et al., "OrchSec: An orchestrator-based architecture for enhancing network-security using Network Monitoring and SDN Control functions," *in Network Operations and Management Symposium (NOMS),* pp. 1–9, IEEE, 2014.

[73] D. M. F. Mattos et al., "AuthFlow: Authentication and Access Control Mechanism for Software Defined Networking," pp.607-615.

[74] M. Reitblatt et al., "Consistent updates for software-defined networks: Change you can believe in!" *in Proceedings of the 10th ACM Workshop on Hot Topics in Networks, ACM,* p. 7, 2011.

[75] G. Yao et al., "Source address validation solution with OpenFlow/NOX architecture," *in Proc. 19th IEEE ICNP,* pp. 7–12, 2011.

[76] J. Naous et al., "Delegating network security with more information," *in Proc. 1st ACM Workshop Res. Enterprise Netw.,* pp. 19–26, 2009.

[77] A. Khurshid et al., "Veriflow: Verifying network-wide invariants in real time," *SIGCOMM Comput. Commun. Rev.,* vol. 42, no. 4, pp. 467–472, Sep 2012.

[78] E. Al-Shaer and S. Al-Haj, "FlowChecker: Configuration analysis and verification of federated openflow infrastructures," *in Proc. 3rd ACM Workshop Safe Config,* pp. 37–44, 2010.

[79] S. Son et al., "Model checking invariant security properties in openflow," *in Proc. IEEE ICC,* pp. 1974–1979, 2013.

[80] A. Khurshid et al., "Veriflow: Verifying network-wide invariants in real time," *SIGCOMM Comput. Commun. Rev.,* vol. 42, no. 4, pp. 467–472, Sep 2012.

[81] S. Gutz et al., "Splendid isolation: A slice abstraction for software-defined networks," *in Proc. 1st Workshop HotSDN,* pp. 79–84, 2012.

[82] L. Vanbever et al., "HotSwap: Correct and efficient controller upgrades for software defined networks," *in Proc. 2nd ACM SIGCOMM HotSDN,* pp. 133–138, 2013.

[83] E. Tantar et al., "Cognition: A Tool for Reinforcing Security in Software Defined Networks," *ser. EVOLVE-A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V. Springer,* pp. 61–78, 2014.

[84] W. Han et al., "LPM: Layered Policy Management for Software-Defined Networks," *ser. Data and Applications Security and Privacy XXVIII. Springer,* pp. 356–363, 2014.

[85] N. Foster et al., "Frenetic: A network programming language," *SIGPLAN Notices,* vol. 46, no. 9, pp. 279–291, Sep. 2011.

[86] M. Polverini et al., "Improving dynamic service function chaining classification in NFV/SDN networks through the offloading concept," *Computer Networks,* vol. 182, pp. 107480, Dec 2020.

[87] R. Braga et al., "Lightweight DDoS flooding attack detection using NOX/OpenFlow," *in IEEE 35th Conference on Local Computer Networks (LCN), IEEE,* pp. 408–415, 2010.

[88] C. Schlesinger et al., "Splendid Isolation: Language-Based Security for Software-Defined Networks," *in Proceedings of the first workshop on Hot topics in software defined networks, ACM,* pp. 79–84, 2012.

[89] C. Schlesinger et al., "Splendid Isolation: Language-Based Security for Software-Defined Networks," *in Proceedings of the first workshop on Hot topics in software defined networks, ACM,* pp. 79–84, 2012.

[90] S. Sinha, "Building an Nmap Network Scanner," 10.1007/978-1-4842-2541-7_24, 2017.

[91] K. Wang et al., "LiveSec: Towards effective security management in large-scale production networks," *in Distributed Computing Systems Workshops (ICDCSW), 2012 32ndInternational Conference on IEEE,* pp. 451–460, 2012.

[92] S. Shin et al., "FRESCO: Modular composable security services for software-defined networks," *in Proceedings of Network and Distributed Security Symposium,* 2013.

[93] S. Seeber and G. D. Rodosek, "Improving Network Security Through SDN in Cloud Scenarios," *In 10th International Conference on Network and Service Management (CNSM) and Workshop,* pp. 376–381, 2014.

[94] S. Shin et al., "Rosemary: A Robust, Secure, and High- Performance Network Operating System," *in Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. ACM,* pp. 78–89, 2014.

[95] B. Chandrasekaran and T. Benson, "Tolerating SDN application failures with LegoSDN," *in Proceedings of the 13th ACM Workshop on Hot Topics in Networks. ACM,* pp. 22, 2014.

[96] Y. Zhang, "An adaptive flow counting method for anomaly detection in SDN," *in Proceedings of the ninth ACM conference on Emerging networking experiments and technologies, ACM,* pp. 25–30, 2013.

[97] A. Sharma, Software Defined Networking Market Size, Growth, Forecast Research Report 2018-2025.

[98] "GENI: Global Environment for Network Innovation," *[Online].Available: http://www.geni.net*

[99] H. Hu et al., "Towards a Reliable SDN Firewall," *Presented as part of the Open Networking Summit, (ONS 2014),* 2014.

[100] S. Dotcenko et al., "A fuzzy logic-based information security management for software-defined networks," *in Advanced Communication Technology (ICACT), 2014 16th International Conference on.IEEE,* pp. 167–171, 2014.

[101] N. Foster et al., "Frenetic: A network programming language," *ACM SIGPLAN Notices,* vol. 46, no. 9, pp. 279–291, 2021.

[102] R. Deb and S. Roy, "A comprehensive survey of vulnerability and information security in SDN," *Computer Networks,* vol. 206, pp. 108802, 2022.

[103] B. Ayodele and V. Buttigieg, "SDN as a defence mechanism: a comprehensive survey," *International Journal of Information Security,* pp. 1-45, 2023.

[104] V. Prabhu, and P. Balamurugan, "Survey on Security Based Approach to Prevent Attacks in SDN Using IoT Framework," *2023 2nd International Conference on Edge Computing and Applications (ICECAA), IEEE,* 2022.

[105] M. S. Farooq et al., "Security and Privacy Issues in Software-Defined Networking (SDN): A Systematic Literature Review," *Electronics,* vol. 12, no. 14, pp. 3077, 2023.

[106] W. Zhang et al., "A Survey of SDN Data Plane Attacks and Defense Strategies," *Proceedings of the 2023 2nd International Conference on Networks, Communications and Information Technology*, 2023.

Editorial Policy and Guidelines for Authors

LGURJCSIT is an open access, peer reviewed quarterly Journal published by LGU Society of Computer Sciences. The Journal publishes original research articles and high quality review papers covering all aspects of computer science and information technology.

The following note set out some general editorial principles. A more detailed style document can be download at www.research.lgu.edu.pk is available. All queries regarding publications should be addressed to editor at email Igurjcsit@lgu.edu.pk. The document must be in word format, other format like pdf or any other shall not be accepted.

Heading and sub-heading should be differentiated by numbering sequences like, **1. HEADING** (Bold, Capitals) 1.1. ***Subheading*** (Italic, bold) etc. The article must be typed in Times New Roman with 11 font size 1.5 space, and should have margin 1 inches on the left and right.
Length of paper should not be longer than 15 pages, including figures, tables, exhibits and bibliography. Table must have standard caption at the top while figures below with. Figure and table should be in continues numbering. Citation must be in according to the IEEE 2006 style.

Editorial Policy and Guidelines for Authors

LGURJCSIT is an open access, peer reviewed quarterly Journal published by LGU Society of Computer Sciences. The Journal publishes original research articles and high quality review papers covering all aspects of computer science and information technology.

The following note set out some general editorial principles. A more detailed style document can be download at www.research.lgu.edu.pk is available. All queries regarding publications should be addressed to editor at email Igurjcsit@lgu.edu.pk. The document must be in word format, other format like pdf or any other shall not be accepted.

Heading and sub-heading should be differentiated by numbering sequences like, **1. HEADING** (Bold, Capitals) 1.1. ***Subheading*** (Italic, bold) etc. The article must be typed in Times New Roman with 11 font size 1.5 space, and should have margin 1 inches on the left and right.
Length of paper should not be longer than 15 pages, including figures, tables, exhibits and bibliography. Table must have standard caption at the top while figures below with. Figure and table should be in continues numbering. Citation must be in according to the IEEE 2006 style.

Lahore Garrison University

Lahore Garrison University has been established to achieve the goal of Excellence and quality education in minimum possible time. LGU is situated in the Punjab metropolis city of Lahore is an important milestone in the history of higher education in Pakistan. In order to meet global challenges, it is necessary to touch the highest literacy rate while producing skillful and productive graduates in all fields of knowledge.

VISION

To be a renowned University in Teaching, Research, Innovation, and Commercialization, providing a conducive environment for the acquisition of latest knowledge so that students may contribute to community support, technical and socioeconomic development.

MISSION

To play a leading role in technical and socioeconomic development through academic and research excellence while adhering to international quality standards. The University would develop leaders who are multi-disciplinary, value-oriented, creative, and entrepreneurial.

Contact us:
Phone: +92-042-37181823
Email: lgurjcsit@lgu.edu.pk

Published by: Faculty of Computer Sciences, Lahore Garrison University, Lahore, Pakistan.