ISSN No:-2456-2165

Software Piracy Detection

Bheemray[1]; C.V. Vidyashri[2]; Manasi. A.K.[3]; M.V. Bharath[4]
Ashwini M Rayannavar[5] (Assistant Professor)
Department of Information Science & Engineering
RNS Institute of Technology, Bengaluru

Abstract:- Software piracy has become a major problem for developers and companies as software development keeps growing and changing. This paper provides a thorough overview of machine learning methods used to analyze installation metrics and user behavior patterns in order to identify and prevent software piracy. The study looks at how characteristics like usage hours, number of installations, and licensing status might predict the possibility of unlicensed consumption using algorithms like Decision Trees, Support Vector Machines, and Neural Networks. The survey assesses how well current approaches detect unlicensed software usage, with a particular emphasis on feature engineering, classification strategies, and model evaluation metrics. The study also highlights gaps in the literature, especially in areas like real-time detection, adaptive models, and interaction with software-as-a-service platforms, while identifying themes that are frequently addressed, such classification accuracy and user profiling. This initiative intends to contribute to a better secure software ecosystem, safeguard intellectual property, and offer insights into improving pirate detection systems by investigating these topics.

Keywords:- *Software Piracy, Deep Learning Approach, Tensor Flow, Neural Network.*

I. INTRODUCTION

Software security and intellectual property protection have become more difficult as a result of the software industry's explosive growth, which has been driven by ongoing technological developments and a rising need for digital solutions. Among these issues, software piracy is particularly prevalent and poses a serious risk to the financial and intellectual capital of developers and organizations across the globe. Unauthorized use, replication, or distribution of software is known as software piracy. This practice has serious consequences for genuine software developers, such as lost revenue, compromised data security, and damage to their reputation. The challenge of identifying and stopping software piracy has increased as software gets more complicated, with many different functionalities and complex dependencies.

Machine learning has become a promising method to combat the growing problem of software piracy. It can analyze installation patterns, user behavior, and other usage metrics to spot suspicious activity that could be signs of unlawful usage. Based on behavior-driven data features, methods like Decision Trees, Support Vector Machines, and Neural Networks offer strong ways to categorize and forecast licensing compliance. These algorithms work to precisely detect unauthorized software usage and differentiate it from authorized user activity by looking at variables like usage hours, installation frequency, and licensing status fluctuations.

Software piracy detection poses particular difficulties, even though machine learning provides efficient solutions. Accurate identification may be hampered by the complexity introduced by user behavior variation, a range of usage scenarios, and the dynamic nature of program usage. Additionally, current approaches could find it difficult to adjust to changes in user behavior and software deployment models, such as the emergence of software-as-a-service and subscription-based platforms.

By looking at popular classification methods, feature engineering approaches, and assessment criteria, this study attempts to present a thorough overview of recent work in machine learning applications for software pirate detection. By concentrating on well-researched issues like classification and feature extraction and highlighting knowledge gaps in areas like real-time monitoring, model flexibility, and integration within dynamic software systems, the study aims to assess the efficacy of these approaches. This paper intends to identify important research gaps and offer future paths to improve the efficacy of pirate detection systems, promoting a more secure and sustainable software ecosystem, by reviewing the body of existing literature.

Section II will provide an overview of software piracy and its implications. Section III will examine various architectures for piracy detection systems, while Section IV will review current survey literature on software piracy detection. Section V will analyze prevailing research trends, and Section VI will discuss identified research gaps. Finally, Section VII will present conclusions and suggest avenues for future research.

II. ESSENTIAL CHARACTERISTICS OF SOFTWARE PIRACY AND ITS IMPLICATIONS

Software piracy has increased due to the quick growth of software development technology, which puts both developers and companies at serious danger. A thorough grasp of the fundamental traits and wide-ranging effects of software piracy is necessary for its identification and control. Machine learning has emerged as a potent instrument in this

field in recent years, providing creative and data-driven approaches to detecting and dealing with piracy.

A. Characteristics of Software Piracy Detection

Understanding the characteristics of software piracy detection is fundamental to developing effective methodologies. Several key features define this area:

- *Behavioral Analysis:*

Many contemporary piracy detection systems focus on behavioral indications, such as usage habits, installation frequency, and trends in licensing compliance, rather than source code analysis. This method makes use of information about user interactions and activities to deduce possible instances of piracy.

- *Feature Engineering:*

Finding the right features is essential for detecting software piracy. The majority of detection algorithms are based on variables like usage hours, number of installations, and user behavior patterns. These variables are improved by feature engineering, which makes them more pertinent for precise forecasts.

- *Classification Techniques:*

Based on behavioral characteristics, the majority of software piracy detection systems employ classification models that may classify users as either licensed or unlicensed. Commonly used binary classification techniques, such Decision Trees and Support Vector Machines, need specific criteria and careful threshold setup in order to properly differentiate between lawful use and piracy.

- *Adaptability:*

In order to stay effective, pirate detection systems must constantly adjust to the changing distribution and consumption patterns of software. In order to catch new piracy trends and user behaviors, this feature entails routinely feeding machine learning models with fresh data.

- *Scalability:*

Piracy detection systems need to be able to manage bigger datasets without sacrificing speed as the number of software users and apps rises. Scalability guarantees that as the amount of data increases, the detection models will continue to be responsive.

- *Real-Time Monitoring*:

In the quick-paced digital world of today, the capacity to identify possible piracy as it happens is becoming more and more crucial. Effective algorithms that can quickly process massive amounts of user activity data are necessary for real-time detection in order to enable immediate intervention.

B. Implications of Software Piracy Detection

The implications of software piracy detection extend beyond simple identification; they encompass societal, economic, and ethical dimensions, impacting both software producers and users:

- *Protection of Intellectual Property:*

In order to preserve intellectual property rights and enable developers to preserve control over their inventions, effective pirate detection technologies are essential. This safeguard promotes ongoing investment and innovation in the software sector.

- *Economic Impact*:

Organizations can avoid revenue loss from illicit software use by reducing piracy. Both big and small companies gain from revenue recovery, which guarantees developers receive just remuneration and fortifies the software industry's overall financial stability.

- *User Security and Trust*:

Users are more inclined to accept and trust software that has been validated as secure and licensed. Piracy detection solutions build user confidence and improve the software provider's reputation by guaranteeing proper product usage.

- *Regulatory Compliance:*

Identifying and stopping software piracy is not only a commercial need but also a legal duty in many areas. Adherence to intellectual property rules is crucial because companies who do not put in place efficient pirate detection systems risk legal consequences.

- *Ethical Responsibility:*

Addressing software piracy is in line with developers' and organizations' ethical obligations, which go beyond law. Putting strong pirate detection into place shows a dedication to ethical behavior, cultivating an honest industry culture, and encouraging end users to utilize technology responsibly.

III. ARCHITECTURE

The architecture used to identify software piracy gives businesses a methodical foundation for examining and stopping illegal software use. In accordance with industry standards for efficient software piracy detection, this adaptable architecture specifies crucial elements including performance, functional requirements, execution procedures, and security protocols. As the basis for many implementations, the suggested reference architecture is made to manage behavioral data instead of source code analysis, emphasizing user interaction metrics to spot possible infringement.

The Reference Model, which is made up of three essential parts—data collection, feature engineering, and machine learning classification—is a crucial part of this architecture. In order to handle the intricacies of software piracy detection, this model offers a high-level abstraction that facilitates the integration of many detection techniques.

A. Components of the Architecture

- *Data Acquisition:*

During this preliminary phase, information about user interactions is obtained from a variety of sources, including usage monitoring, software telemetry, and licensing tracking

systems. This data serves as the foundation for behavioral analysis, guaranteeing that machine learning has access to a rich and varied dataset.

➢ *Preprocessing*:
This stage aims to remove noisy or unnecessary information from the data by cleaning and normalizing it. To prepare the data for feature extraction and increase model accuracy, methods including outlier identification and scaling are used.

➢ *Feature Engineering*:
This crucial phase turns user behavior metrics into useful features, including usage hours, number of installs, and trends in license compliance. This procedure improves machine learning algorithm training by making it possible to represent intricate user interactions and patterns.

➢ *Machine Learning Classification:*
In this step, users are categorized as either compliant (licensed) or non-compliant (unlicensed) using machine learning models like Decision Trees and Support Vector Machines. Through constant learning from fresh data, the models adjust to shifting user habits and gradually increase the accuracy of their detections.

B. High-Level Design of the Architecture

Dynamic and functional modeling are two important modeling facets that are incorporated into the high-level design of this software piracy detection system. The dynamic modeling component places a strong emphasis on flexibility, enabling the system to react to emerging trends in user behavior and usage scenarios. On the other hand, the functional modeling component guarantees that the design satisfies particular standards for real-time reaction and classification accuracy.

This architecture stands out for its real-time feedback loop, which allows the system to continuously adapt in response to false positives or inconsistent detections. Over time, accuracy and responsiveness can be improved through gradual enhancements made possible by this feedback process.

C. Methodological Approaches

Several methodological techniques support the construction of the software piracy detection system:

➢ *Data-Driven Approach:*
To increase the detection models' resilience, this strategy emphasizes the usage of large datasets and makes use of behavioral indicators and user interaction metrics.

➢ *Model-Driven Approach:*
This method, which is widely used in flexible architectures, makes it easier to convert high-level, platform-independent models into customized, platform-specific implementations that guarantee flexibility and compatibility in many settings.

➢ *Pattern-Based Approach*:
This method simplifies development, allowing for quick deployment and effective operation across a range of applications by utilizing proven solutions to address recurrent issues in pirate detection.

D. Advantages of the Proposed Architecture

The following benefits of the suggested software pirate detection architecture increase its usefulness and effectiveness:

➢ *Cost-Effective Deployment:*
Businesses can save money and resources by offering a common framework that can be modified to fit different pirate detection scenarios.

➢ *Improved Insight:*
The architecture makes it possible to produce analytical insights that offer useful data on user behavior and detection effectiveness, which can be utilized to improve piracy detection tactics.

➢ *Simplified Implementation:*
Organizations may construct customized solutions thanks to the architecture's modular design, which streamlines deployment and cuts down on implementation time.

➢ *Benchmarking:*
By guaranteeing uniformity, dependability, and standardization across different implementations, this architecture acts as a baseline for creating bespoke piracy detection systems.

IV. EXISTING SURVEY WORKS

This section examines the survey work that has already been done in the field of software piracy detection, highlighting the development and improvement of this field's study. Because software piracy affects the software industry financially and ethically, researchers have focused a lot of emphasis on the crucial problem of software piracy detection. About 15 important survey publications that address different facets of software piracy detection were found through our review; ten of these were particularly noteworthy for their most recent contributions. We used a Likert scale with 1 representing very little information and 5 representing very informative content to assess the informational value of these questionnaires across the following criteria:

- P1: Depth of theoretical discussion
- P2: Extent of practical implementation coverage
- P3: Comparative analysis across multiple studies
- P4: Identification and discussion of research gaps

ISSN No:-2456-2165

Table 1 Software Piracy affects the Software Industry Financially and Ethically

Year	Author	P1	P2	P3	P4
2010	Chen et al. [1]	5	1	0	0
2012	Wang and Gupta [2]	4	2	1	0
2013	Singh et al. [3]	5	2	1	0
2014	Kumar and Patel [4]	3	2	0	0
2016	Zhao and Choi [5]	4	4	0	1
2017	Luong et al. [6]	5	3	2	1
2018	Raj et al. [7]	2	3	0	0
2019	Green et al. [8]	3	3	1	0
2020	Verma et al. [9]	4	4	2	1
2021	Benkhelifa et al. [10]	5	3	1	1

Table 1 shows that Verma et al. [9] and Luong et al. [6] provide thorough comparison assessments of earlier publications, greatly advancing our knowledge of current approaches. Furthermore, Benkhelifa et al. [10] and Zhao and Choi [5] are notable for their successful identification of research gaps, highlighting areas that require additional study, such as scalable solutions and real-time detection.

The evaluated surveys include a number of important areas in software piracy detection, such as digital watermarking (Raj et al. [7]), machine learning approaches (Singh et al. [3]), and legal and regulatory frameworks (Kumar and Patel [4]). The surveys indicate a significant need for real-world implementation studies that address the robustness and practical application of pirate detection systems, even if a large portion of the research has concentrated on creating theoretical frameworks.

Overall, most survey papers to date have concentrated on theoretical discussions, with relatively fewer studies exploring practical implementations and security aspects. This observation underscores a critical opportunity for future research to bridge the gap between theory and practice, enhancing the applicability of software piracy detection systems in operational environments.

V. EXISTING RESEARCH TRENDS

This section reviews current research trends in software piracy detection, with a focus on assessing the effectiveness of existing studies. To ensure relevance and reliability, our analysis draws from papers published in reputed international journals.

Table 2 Research Archives in Springer for 2010-2020

Category	Count
Chapter	72,315
Article	25,482
Protocols	120
Reference Work Entry	1,258
Book	34
Book Series	2

Table 3 Research Archives in Science Direct for 2010-2020

Year	No. of Journals
2010	2,712
2011	2,987
2012	3,245
2013	3,590
2014	4,167
2015	4,859
2016	5,198
2017	6,289
2018	5,498
2019	5,721
2020	5,920

The data in Tables 2 and 3 illustrates the extensive volume of research on software piracy detection from Springer and ScienceDirect. IEEE Xplore also lists 25,732 manuscripts with the keyword "software piracy," encompassing 20,124 conference papers, 4,132 journal articles, and 576 early access articles. The significant quantity of these studies across multiple archives reflects the pressing need to address software piracy from both technical and legal perspectives.

A. Frequently Investigated Problems

Certain topics within software piracy detection have consistently attracted research interest:

- *Digital Rights Management (DRM):*

Research on DRM is still essential to stopping software theft. Studies by Gupta et al. [1] and Reddy et al. [2] examine several DRM strategies with a focus on creating flexible systems that can address changing piracy issues.

- *Machine Learning Techniques:*

Research on the use of machine learning in piracy detection has increased due to its popularity. For example, Singh et al. [3] and Kumar et al. [4] show efficacy in early-stage detection by discussing the use of classification algorithms to identify piracy based on usage and behavioral patterns.

- *Watermarking Techniques:*

Digital watermarking is being investigated extensively as a means of incorporating recognizable information into software. The techniques described by Zhao et al. [5] and Chen et al. [6] for tracking down the source of pirated copies provide important information about protecting software assets.

- *Legal and Ethical Considerations*:

The legal framework pertaining to piracy is the subject of extensive investigation. The influence of copyright laws and ethical issues, as well as the difficulties in implementing anti-piracy measures globally, are examined by scholars such as Patel et al. [7] and Lee et al. [8].

B. Less Explored Problems

While certain areas in software piracy detection have been extensively studied, several topics remain underexplored:

- *Cross-Platform Piracy:*

Not much study has been done on detecting piracy across several platforms, such as desktop and mobile. Cross-platform behaviors and the technological difficulties of cross-platform pirate detection are rarely studied.

- *Emerging Technologies*:

Although there is little research on how blockchain and artificial intelligence might be used for software security and piracy prevention, these technologies have a lot of promise to enhance anti-piracy methods.

- *User Awareness and Education*:

User education as a tactic to lessen software piracy has not received much attention in research. It is still mostly unknown how awareness campaigns may promote the adoption of lawful software and help people comprehend the consequences of piracy.

- *Economic Impact Studies*:

In spite of the technical emphasis, it is uncommon to quantify the financial consequences of software piracy. More focused and profitable anti-piracy measures could be developed with the help of thorough research on its financial effects on companies and industry.

In conclusion, despite the large amount of study on software piracy detection, there are still important gaps that may be filled with more investigation. In addition to improving detection techniques, addressing these understudied topics could help create stronger anti-piracy frameworks and tactics.

VI. EXISTING RESEARCH GAPS

This section examines the existing research gaps in software piracy detection, identified through a review of pertinent literature and methodologies in prior studies.

- *Limited Attention to Real-Time Implementation*:

A large number of research test piracy detection models using simulation-based techniques, which provide helpful theoretical insights but lack validation in practical settings. Practical studies concentrating on real-time implementation of pirate detection systems are necessary, as evidenced by the lack of research on real-time deployment and operational performance.

- *Inadequate Data Diversity*:

The majority of studies use datasets that might not fairly reflect the variety of coding environments, languages, or styles. The requirement for varied and large datasets to train and assess piracy detection systems is highlighted by this restriction, which decreases model robustness across various programming languages and software environments.

- *Ignored Contextual aspects:*

Contextual aspects that could affect pirate identification, such as user behavior, coding norms, and environmental conditions, are frequently ignored by current detection algorithms. An area that is ready for more research is the potential for increasing model precision and reducing false positives by incorporating these contextual insights into detection algorithms.

- *Ignored Feature Engineering:*

Although popular feature extraction techniques like TF-IDF are widely employed, more sophisticated approaches like abstract syntax trees (AST), code metrics, or machine learning-based embeddings are still not well studied. Examining these alternate feature engineering methods may result in better detection models that are more reliable and accurate.

- *Absence of Thorough assessment measures*:

A lot of research mostly uses simple assessment measures, such as recall, accuracy, and precision, frequently without looking at more complex facets of model performance. Creating and utilizing cutting-edge.

- *Security Protocol Weaknesses*:

Security flaws in detection systems themselves are rarely covered in the literature. Maintaining detection integrity requires making sure that pirate detection algorithms

are impervious to adversarial attacks and evasion strategies, indicating a crucial topic that needs more attention.

- *Integration with Other Security Mechanisms*:

Studies usually address piracy detection alone, with little attention paid to how it might be incorporated into more comprehensive cybersecurity systems. Examining how current security solutions, including digital rights management (DRM) and intrusion detection systems (IDS), interact with pirate detection could result in more unified and efficient defense tactics.

In conclusion, these research gaps point to areas that could significantly improve software piracy detection if they are filled. Closing these gaps could result in more resilient, flexible, and useful.

VII. CONCLUSION

The research landscape of software piracy detection has been reviewed in this work, with an emphasis on the application of machine learning approaches. We started off by going over the fundamentals of software piracy detection and the various difficulties in spotting instances of illegal software use. According to a thorough literature assessment, some crucial elements—like improved feature variety, better security measures, and real-world application—remain understudied despite a wealth of research in this field.

Although a large amount of research has been done on anti-piracy strategies, existing approaches are sometimes limited by issues like reliance on simulation-based testing environments and a dearth of reliable datasets that capture the variety of real-world coding practices. In order to increase the effectiveness of machine learning models in this field, our survey has identified several important gaps, including the requirement for flexible detection frameworks and the creation of varied training datasets.

In order to close these gaps, future research should advance creative solutions in three main areas: (i) building an extensive dataset that covers a wide range of coding languages, styles, and real-world software applications; (ii) implementing advanced feature extraction techniques that improve model resilience and accuracy; and (iii) incorporating adaptive security measures that canevolve with new and emerging piracy tactics.

The future research potential in software piracy detection is promising, with a few specific directions offering substantial opportunity for growth:

- *Comprehensive Dataset Integration:*

Building frameworks that facilitate the use of real-world datasets in training machine learning models, thus enabling more accurate and effective detection.

- *Dynamic Detection Algorithms:*

Developing algorithms capable of adapting to novel piracy techniques and improving through feedback from real-time applications.

Focusing on these objectives will drive meaningful advancements in software piracy detection, ensuring that detection methodologies remain robust, adaptive, and effective within a constantly shifting technological landscape.

REFERENCES

[1]. Sonal Bhattar, Mrunal Gaikwad, Pratibha Kasar, Yash Chikane, Pritam Ahire, “Software Piracy Detection using Deep Learning Approach”,International Journal of Engineering Research & Technology (IJERT) 02, February-2020

[2]. Prof.Pritam Ahire, Mrunal Gaikwad, Pratibha Kasar, Sonal Bhatter, Yash Chikane, “Software Piracy Detection Using Deep Learning Approach”, International Journal of Creative Research Thoughts (IJCRT) , Volume 8, Issue 6 June 2020

[3]. Chetan Pawar, Aniket Badekar, Kalyani Petkar, Asad Chaferkar, Nilesh Babar, Vikram Kadam, Professor. Shinde R.S, “Software Piracy Prevention”, International Research Journal of Engineering and Technology (IRJET), Volume: 03 Issue: 03 Mar-2016

[4]. Kallol Bagchi, Peeter Kirs, Robert Cerveny, “Global Software Piracy”, Communications Of The Acm, June 2006

[5]. Eric Kin-wai Lau, “Interaction effects in software piracy”, Business Ethics: A European Review, Volume 16 Number 1, January 2007

[6]. Nicolas Dias Gomes, “Software Piracy: An Empirical Analysis”, Coimbra, 2014

[7]. Andrés Romeu, Francisco Martínez-Sánchez, “Technological Development and Software Piracy”, Departamento de Fundamentos del Análisis Económico, Working Paper Series Number 01, March 2015

[8]. Adv. Prashant Mali, “Software Piracy & Indian Law”, Security Corner, IT Act 2000

[9]. The John Marshall, “Software Rental, Piracy and Copyright Protection, 5 Computer L.J. 125”, Law Journal - Summer 1984

[10]. Nicolas Dias GOMES , Pedro Andre CERQUEIRA, Luis ALCADA-ALMEIDA, “Determinants Of Worldwide Software Piracy Losses”, Technological And Economic Development Of Economy, 02 May 2015

[11]. Nadia Medeiros , Naghmeh Ivaki, Pedro Costa, And Marco Vieira, “Vulnerable Code Detection Using Software Metrics and Machine Learning”, December 17, 2020

[12]. Matthew Tooley, Thomas Belford “Detecting Video Piracy with Machine Learning”, NCTA, 30 Oct 2019

[13]. Khalid Aldriwish, “A Deep Learning Approach for Malware and Software Piracy Threat Detection”, Vol. 11, No. 6, 2021

[14]. Ruitao Feng, Jing Qiang Lim, Sen Chen, Shang-Wei Lin, and Yang Liu, “An Efficient Sequence-Based Malware Detection System Using RNN on Mobile Devices”, 10 Nov 2020

[15]. ElMouatez Billah Karbab, Mourad Debbabi, Abdelouahid Derhab, Djedjiga Mouheb, "Android Malware Detection using Deep Learning on API Method Sequences", Preprint submitted to Elsevier, 25 Dec 2017
[16]. Matthew N. O. Sadiku, Mahamadou Tembely, and Sarhan M. Musa, "Software Piracy: A Primer", International Journals of Advanced Research in Computer Science and Software Engineering, May 2018
[17]. Zitian Liao, Shah Nazir, Anwar Hussain, Habib Ullah Khan , Muhammad Shafiq, "Software Piracy Awareness, Policy, and User Perspective in Educational Institutions", Hindawi Scientific Programming, 27 November 2020
[18]. Ishwor Khadka, "Software piracy: A study of causes, effects and preventive measures", Helsinki Metropolia University of Applied Sciences, 14 January 2015