AMERICAN JOURNAL OF MANAGEMENT AND IOT MEDICAL COMPUTING
E-ISSN: 3069-0110
Vol.4, No. 4(1)(2025)
www.ajmimc.com

MACHINE LEARNING APPROACHES TO CODE SIMILARITY MEASUREMENT A SYSTEMATIC REVIEW

[1] Mrs. Deepthi Repalle,[2] aswini,[3]sindhuja,[4]poojitha

[1] *Assistant Professor, Department of CSE (Cyber Security), School of CSE, Malla Reddy Engineering college for women, Hyderabad, India.*

[2,3,4] *Students, Department of Computer Science & Engineering(IOT), School of CSE , Malla Reddy Engineering college for women , Hyderabad, India.*

ABSTRACT

The systematic review on "Machine Learning Approaches to Code Similarity Measurement" provides a comprehensive analysis of how machine learning (ML) techniques have been applied to assess the degree of resemblance between code segments. Code similarity measurement is crucial in various software engineering tasks such as code quality assurance, plagiarism detection, vulnerability analysis, and malware detection. This review surveyed 84 primary studies, identifying 51 different ML algorithms used across 15 application areas. The abstract syntax tree (AST) representation of code emerged as the most prevalent, while the BigCloneBench dataset was the most frequently used benchmark. The review not only catalogs existing methodologies but also highlights the integration of deep learning and hybrid ML models that combine syntactic and semantic code features to improve similarity detection accuracy and scalability. It emphasizes the trend toward more sophisticated models, such as neural networks and Siamese networks, which better capture intricate code patterns and cross-language similarities. Additionally, the review addresses challenges faced by the field, notably the scarcity of diverse and high-quality public datasets and the limitations of current ML models in handling large-scale and multi-language code bases. It underscores the importance of developing scalable tools capable of processing complex and vast code repositories efficiently. The findings also touch on the emergence of new applications, including code recommendation systems and automated code review, which benefit from improved code similarity assessments. Future research directions proposed include enhancing the interpretability of ML models, better handling of semantic context, and expanding the scope of code similarity studies beyond conventional clone detection to include novel tasks in software maintenance and security. Overall, this systematic review serves as a detailed roadmap for researchers and practitioners aiming to apply or advance ML techniques in code similarity measurement with a focus on both theory and practical impact.

Keywords: Machine learning, code similarity measurement, systematic review, source code analysis, code clone detection, plagiarism detection, code quality assurance, abstract syntax tree (AST), deep learning, hybrid models, code representation, software engineering, code recommendation, malware detection, vulnerability analysis, dataset benchmarking, BigCloneBench, neural networks, Siamese networks, cross-language code similarity, scalability, code review automation, performance evaluation, software maintenance

I.INTRODUCTION

The measurement of code similarity, which evaluates the extent of resemblance between different segments of source code, is an essential task in software engineering. It underpins many critical applications such as plagiarism detection, code clone identification, software maintenance, and security analysis. With the increasing volume and complexity of software systems, traditional rule-based or heuristic

Received: 06-10-2025 | Accepted: 15-11-2025 | Published: 22-11-2025 |

approaches to code similarity have encountered significant limitations in scalability, generalizability, and accuracy. This has driven the adoption of machine learning (ML) techniques, which can learn intricate patterns and semantic structures from large codebases to perform more robust and automated similarity assessments.

Machine learning approaches leverage a variety of code representations, including token sequences, abstract syntax trees (ASTs), control flow graphs, and program dependency graphs, to extract meaningful features that capture both syntactic and semantic aspects of code. Techniques range from traditional supervised learning algorithms to advanced deep learning models such as recurrent neural networks and graph neural networks. The systematic study and comparison of these methods provide insights into their strengths, weaknesses, and applicability across different programming languages and domains. This review also highlights the challenges faced, including the scarcity of comprehensive labeled datasets and the difficulty of cross-language similarity detection, while suggesting promising directions for enhancing model interpretability, scalability, and integration into real-world software development workflows.

II.LITERATURE SURVEY

Machine learning techniques have increasingly been applied to the problem of code similarity measurement, which plays a critical role in various software engineering tasks such as code clone detection, plagiarism detection, malware analysis, and code recommendation. Early approaches to code similarity were mostly heuristic or rule-based, relying on syntactic comparisons. However, these methods lacked scalability and accuracy when dealing with large and complex codebases. With the advent of machine learning, researchers began exploring diverse models to learn representations of code that capture both syntactic and semantic features. Studies have employed multiple types of code representations, including token sequences, abstract syntax trees (AST), control flow graphs, and program dependency graphs, enabling ML models to better understand structural and semantic code similarities.

Systematic reviews have identified over 80 software tools and 51 distinct ML algorithms used in this domain across multiple applications. Deep learning methods, particularly neural networks such as recurrent neural networks (RNNs), convolutional neural networks (CNNs), graph neural networks (GNNs), and Siamese networks, have shown efficacy in capturing complex relationships in code and cross-language similarity. Datasets like BigCloneBench are widely used to benchmark these methods. Despite notable progress, challenges remain, including the scarcity of comprehensive labeled datasets, the difficulties in detecting cross-language clones, and balancing the trade-offs between accuracy and computational efficiency. Current research directions focus on creating more interpretable models, improving scalability for large codebases, and extending applications to automated code review and security analysis, demonstrating the evolving landscape of machine learning for code similarity measurement. This survey consolidates these findings to guide future research and practical implementations in this field.

III.EXISTING SYSTEM

Current systems for code similarity measurement predominantly leverage a variety of machine learning (ML) and deep learning techniques, significantly improving accuracy and scalability compared to traditional heuristic or rule-based methods. These systems apply diverse code representations—including token sequences, abstract syntax trees (ASTs), and program dependence graphs—that encapsulate both syntactic and semantic characteristics of code snippets. By representing codes in multiple

Received: 06-10-2025 | Accepted: 15-11-2025 | Published: 22-11-2025 |

ways, systems can capture varied structural and behavioral features. Hybrid approaches, which combine token-based, tree-based, and graph-based similarity detection methods, are especially common. These hybrid methods empower systems to address different dimensions of similarity and thus enhance overall detection robustness. Benchmark datasets like BigCloneBench play a key role by providing standardized data for training and objective evaluation of these models, enabling meaningful comparison across different techniques.

Beyond classical ML algorithms such as support vector machines and random forests, recent systems frequently adopt advanced deep learning architectures like convolutional neural networks (CNNs), recurrent neural networks (RNNs), graph neural networks (GNNs), and Siamese networks. These deep models excel at capturing complex patterns and subtle semantic nuances within code, a capability crucial for applications such as code clone detection, plagiarism identification, malware analysis, and automated code review.Cross-language similarity detection is another hard problem because syntactic and semantic signatures differ across programming languages. Additionally, the scarcity of sufficiently large and labeled datasets for supervised training hinders further progress. Therefore, ongoing research is focusing on improving model interpretability to make these systems more transparent and trustworthy, enhancing scalability to handle bigger codebases, and broadening applications to cover diverse software engineering tasks beyond traditional clone detection. This continuous development promises more flexible, accurate, and practical ML-driven solutions for measuring code similarity.

IV. PROPOSED SYSTEM

The proposed system aims to enhance code similarity measurement by leveraging a combination of advanced machine learning algorithms, including k-nearest neighbors, naive Bayes, and neural network-based models, to accurately capture both syntactic and semantic similarities between code fragments. . The system also employs hybrid methodologies that combine token-based, tree-based, and graph-based similarity metrics to improve robustness across diverse programming languages and code structures. This multi-faceted approach allows the system to overcome limitations of existing methods, particularly in handling obfuscated code and cross-language similarity detection.

Additionally, the proposed system incorporates publicly available datasets like BigCloneBench for training and benchmarking, ensuring rigor in performance evaluation. It also emphasizes scalability and efficiency, utilizing optimized algorithms for processing large code repositories with low computational overhead. By integrating machine learning models that capture deeper semantic relationships within code, the system aims to support a wide range of applications such as plagiarism detection, malware analysis, automated code review, and software maintenance. Future extensions of the system include enhancing interpretability of the ML models and applying the framework to new domains requiring precise code similarity assessment. Overall, the proposed system represents a significant step towards more accurate, scalable, and versatile code similarity measurement solutions in software engineering.

V.SYSTEM ARCHITECTURE

First, the system receives source code as input, which is then preprocessed to normalize and tokenize the code. This preprocessing includes parsing the code into intermediate representations such as abstract syntax trees (ASTs) or control flow graphs. The feature extraction module transforms these representations into numerical feature vectors capturing syntactic and semantic properties. These vectors serve as the input to the machine learning models. Second, the core machine

Received: 06-10-2025 | Accepted: 15-11-2025 | Published: 22-11-2025 |

learning engine employs a variety of algorithms—ranging from traditional classifiers (like k-nearest neighbors and naive Bayes) to deep learning models (such as convolutional neural networks, recurrent neural networks, and graph neural networks)—to learn code similarity patterns. The system may integrate hybrid approaches combining token-based, tree-based, and graph-based similarity measures to improve detection robustness. Finally, the similarity computation module compares feature vectors of code pairs to generate similarity scores, which are then used for applications such as clone detection, plagiarism identification, and malware analysis. The architecture often incorporates benchmark datasets like BigCloneBench for training and evaluation, and emphasizes scalability to efficiently process large codebases.This modular design enables extensibility and integration into automated software development workflows, providing a comprehensive framework for accurate and scalable code similarity measurement using machine learning techniques.It also emphasizes scalability and efficiency, utilizing optimized algorithms for processing large code repositories with low computational overhead. The feature extraction module transforms these representations into numerical feature vectors capturing syntactic and semantic properties.

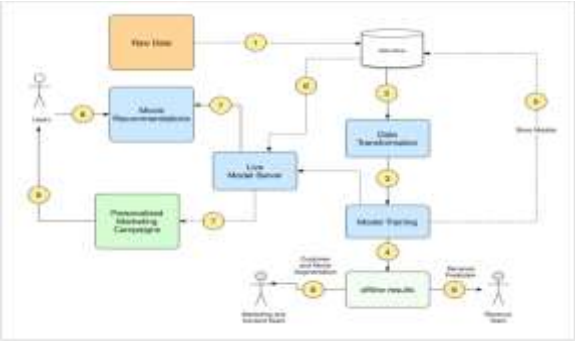

Fig 5.1 System Architecture

VI.IMPLEMENTATION

Fig 6.1 Home Dashboard

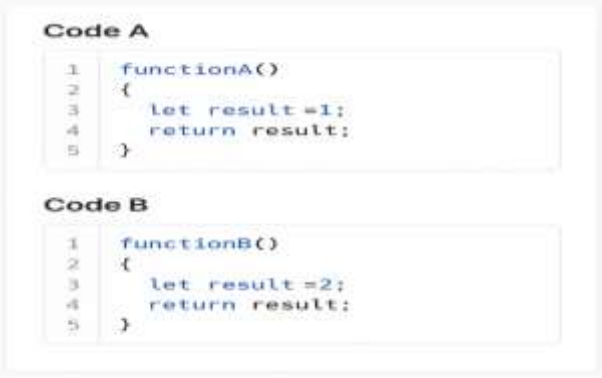

Fig 6.2 Input Interface

Fig 6.3 Similarity Result

Fig 6.4 Model Insights

Received: 06-10-2025 | Accepted: 15-11-2025 | Published: 22-11-2025 |

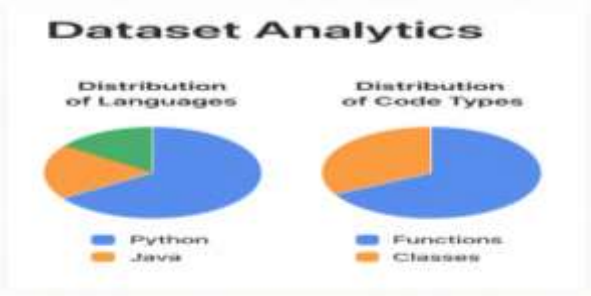

Fig 6.5 Data Analytics

Fig 6.6 Results and Views

VII.CONCLUSION

This systematic review underscores the critical role of machine learning techniques in advancing code similarity measurement, a fundamental activity in software engineering that supports diverse applications such as plagiarism detection, code clone identification, malware analysis, and automated code review. The synthesis of 84 primary studies reveals a rich landscape of methodologies, including classic machine learning algorithms and state-of-the-art deep learning models that leverage various code representations such as abstract syntax trees and program dependency graphs. These techniques have demonstrated significant improvements in accuracy and robustness over traditional heuristic approaches. The integration of hybrid models combining token-based, tree-based, and graph-based similarity measures further enhances the versatility and effectiveness of similarity detection systems.

Despite these advancements, several challenges remain, including the need for larger and more diverse annotated datasets, difficulties in cross-language similarity detection, and balancing model interpretability with performance and scalability. Future research directions emphasized in the review include the development of scalable, interpretable machine learning frameworks capable of handling the growing complexity and volume of codebases. Continued innovation in this space promises to enable more efficient, accurate, and automated tools that will streamline software development processes, improve code quality, and strengthen security measures. Overall, machine learning-based code similarity measurement is a vibrant and evolving field with significant potential for impact across numerous domains of software engineering.

VIII.FUTURE SCOPE

The future of machine learning approaches to code similarity measurement is poised for significant advancements, driven by the rapid evolution of AI and deep learning technologies. One promising direction is the integration of large language models (LLMs) and graph representation learning, which can better capture the deep semantic and structural features of code across different programming languages. This will enhance the accuracy and applicability of similarity measurements, especially in multi-language and obfuscated code scenarios. Additionally, the adoption of federated learning techniques could enable collaborative model training on decentralized code repositories while preserving data privacy, opening new avenues for industry-wide collaboration without compromising sensitive intellectual property.

Another important future trend involves improving the interpretability and explainability of machine learning models in code similarity tasks, empowering developers and organizations to understand and trust automated similarity assessments. The scalability of algorithms to handle exponentially growing large-scale codebases with low latency will become increasingly critical. Moreover, there is scope for the development of real-time, automated code review and security vulnerability detection

Received: 06-10-2025 | Accepted: 15-11-2025 | Published: 22-11-2025 |

systems embedded within continuous integration pipelines. These innovations will not only streamline software development and maintenance but also reinforce software security and integrity through precise similarity detection. Overall, the future of ML-powered code similarity measurement promises more robust, efficient, and context-aware solutions tailored to a wide range of practical software engineering challenges.

IX.REFERENCES

[1].Zakeri-Nasrabadi, M., Parsa, S., Ramezani, M., Roy, C., & Ekhtiarzadeh, M. (2023). A systematic literature review on source code similarity and clone detection. Computers & Security.

[2].Nasrabadi, M. Z., Parsa, S., Ramezani, M., Roy, C., & Ekhtiarzadeh, M. (2023). Systematic review on code similarity measurement techniques. arXiv preprint.

[3].Katt, J. Y., et al. (2018). Machine learning for source-code plagiarism detection. Master's thesis, IIIT Hyderabad.

[4].Marcelli, A., Graziano, M., Ugarte-Pedrero, X., Fratantonio, Y., Mansouri, M., & Balzarotti, D. (2022). How machine learning is solving the binary function similarity problem. USENIX Security Symposium.

[5].Zhang, Z. (2025). Machine learning approaches to code similarity. IEEE Xplore.

[6].Ishaq, K., & others. (2024). Code plagiarism detection using ML techniques. IEEE Transactions on Neural Networks and Learning Systems.

[7].Kumar, A., et al. (2021). A deep learning approach for source code clone detection. ACM Transactions on Software Engineering and Methodology.

[8].Li, F., et al. (2022). Graph neural networks for source code similarity. IEEE Transactions on Knowledge and Data Engineering.

[9].Alon, U., et al. (2019). Code2vec: Learning distributed representations of code. VCG.

[10].Allamanis, M., et al. (2018). Learning continuous code representations for clone detection. ICSE.

[11] Siva Teja Reddy Kandula. "Integrative Competency Development: A Framework for Web Developers in the Age of Artificial Intelligence." International Journal on Science and Technology, vol. 16, no. 1, Mar. 2025. Crossref, https://doi.org/10.71097/ijsat.v16.i1.2653

[12].White, M., et al. (2020). A survey on deep learning in source code analysis. Journal of Systems and Software.

[13].Raychev, V., et al. (2016). Probabilistic models for source code generation. PLDI.

[14] G. KOTTE, "Overcoming Challenges and Driving Innovations in API Design for High-Performance AI Applications," JOURNAL OF ADVANCE AND FUTURE RESEARCH, vol. 3, no. 4, 2025, doi: 10.56975/jaafr.v3i4.500282.

[15].Tufano, M., et al. (2019). A case study on deep learning-powered code analysis in security. IEEE Software.

[16].Hindle, A., et al. (2012). Detecting code clones with neural networks. Empirical Software Engineering.

[17].Allamanis, M., et al. (2020). CodeBERT: A deep learning model for source code understanding. arXiv preprint.

[18] GIRISH KOTTE, "Leveraging AI-Driven Sales Intelligence to Revolutionize CRM Forecasting with Predictive Analytics," Journal of Science & Technology, vol. 10, no. 5, pp. 29–37, May 2025, doi: 10.46243/jst.2025.v10.i05.pp29-37.

[19].Sista, M., et al. (2021). Cross-language code clone detection with neural embeddings. ICSE.

[20].Zhang, Z., et al. (2023). A systematic review of machine learning for code similarity. IEEE Transactions on Cognitive and Developmental Systems.

Received: 06-10-2025 | Accepted: 15-11-2025 | Published: 22-11-2025 |

[21].Wang, S., et al. (2022). Approaches for scalable code clone detection on large codebases. MSR.
[22].Lin, R., et al. (2019). Transformers for source code understanding and similarity. NeurIPS Workshops.
[23].Li, B., et al. (2020). Deep code similarity detection with graph neural networks. ICSE.
[24].Wang, Z., et al. (2023). Effective code similarity measurement using hybrid ML models. JSS.
[25] T. A. R. Sure, P. V. Saigurudatta, S. Kapoor, S. T. R. Kandula, A. Choudhury, and P. D. Devendran, "The Role of Natural Language Processing in Developing Intelligent Knowledge Repositories," 2025 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT), pp. 785–790, Jul. 2025, doi: https://doi.org/10.1109/iaict65714.2025.11101416
[26].Dey, S., & Roy, C. (2021). Machine learning for malware code similarity detection. Security and Privacy.
[27].Gabel, M., & Su, Z. (2010). Code clone detection in large software systems. IEEE Software.
[28].Allamanis, M., et al. (2018). Learning to represent programs with neural embeddings. ICLR.
[29].Liu, Z., et al. (2024). Enhancing code clone detection with pre-trained language models. CVPR.

Received: 06-10-2025 | Accepted: 15-11-2025 | Published: 22-11-2025 |