iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.79537

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Intelligent Malware Detection and Alert Systems Using Machine Learning and Deep Learning

Mrs.E.Padmavathi[1], P.Gopi Raju[2], M.Suryaprakash[3], P.Mahaboob Subhani[4], N.Taraka Ram[5]
[1]*Asst. Professor,* [2, 3, 4, 5]*Student, Department of CAI, KKR & KSR Institute of Technology and Sciences, Guntur, Andhra Pradesh*

Abstract: ***Theexpansionofinterconnectedcomputinginfrastructureshasintensifiedthescaleand sophistication of malicious software attacks. Detection mechanisms that depend solely on stored signaturedatabasesareincreasinglyineffectiveagainst newlyemergingandrapidlymutating threats.Thisstudyintroducesisandbox, capturing indicators such as process invocation sequences, system configuration changes, file accessbehavior,interface-levelinteractions,andcommunicationflows.***
To enhance predictive capability, a multi-model classificationapproachcombining Random Forest, Support VectorMachine, andConvolutionalNeuralNetworkarchitecturesisimplemented. Eachmodelcontributescomplementaryanalytical strengths, and their outputs are aggregated through an ensemble decision mechanism to improveclassification stability. The framework also incorporatesan automated alert module and structured logging facilitytosupporttimely threatresponseandtraceability.Experimentalvalidationdemonstratesthattheintegratedmodelattains 97% overall detection accuracy, alongside strong precision and recall, with a minimal rate of false alarms.
The resultsconfirm thata runtime-focusedhybridlearningarchitecture canprovide aresilientandscalable defense strategy against adaptive and previously unseen malware variants.
Keywords: MalwareDetection,MachineLearning,DeepLearning, Behavioural Analysis,RandomForest,SVM,CNN, Real-Time Alert System.

I. INTRODUCTION

In The growing reliance on internet-connected systemshassubstantiallyincreasedtheattacksurface of contemporary computing systems. Although internet connectivity facilitates efficient communication and data transfer, it also provides opportunitiesformaliciousssoftwaretoentersystems. Malicious software, such as ransomware, trojans, spyware,worms,andviruses,continuestoincreasein complexity and sophistication.
Traditional antivirus software is based on signature matching. Inthismethod, filesarematchedagainst a database of previously known malware signatures. Althoughsuccessfulagainstknownmalware,this method is ineffective against newly developed or altered malware variants. Attackers often modify code structures, employ obfuscation techniques, and utilizeencryptiontoolstobypasstraditionaldetection software.
To overcome these issues, intelligent detection methodsbasedon MachineLearning(ML)andDeep Learning (DL) have emerged. Unlike traditional signature-matching methods, learning-based detection systems focus on behavioral patterns produced during program execution. By analyzing how an application behaves when interacting with system resources during runtime, it is possible to detect malicious activity even when encountering unfamiliar malware.
Theproposedsystempresentsacompletelybehavior- driven malware detection system that incorporates multiple learning algorithms and provides instantaneous notifications upon detecting unusual runtime patterns.

II. RELATED WORK

Malware classification methodologies have evolved from simple pattern matching tools to intelligent classification systems. In the past, security software almostentirelydependedonsignature-basedsystems. Although computationally inexpensive, these systems werenotadaptableandperformedpoorlyon zero-dayattacks.
With the rise of data-intensive approaches, researchersstartedtoincorporatesupervisedMachine Learning algorithms such as Support Vector Machines and Random Forest classifiers. These approaches enhanced detection accuracy by identifying characteristic patterns from labeled datasets. However, most early research works relied on static analysis for feature extraction, which involves analyzing executable files without actually running them. Although computationally inexpensive, static analysis is prone to obfuscation and packing techniques employed by contemporary attackers.Dynamic analysis has since emerged as a more effective approach.

Rather than analyzing code patterns, dynamic analysis involves analyzing program behavior in sandboxed execution environments. Byanalyzingruntimepatternssuchas memory consumption patterns, process creation patterns, and network traffic patterns, more accurate informationaboutmaliciousactivitycanbeobtained.

More recently, Deep Learning models, especially ConvolutionalNeuralNetworks,havebeenproposed to automatically identify complex patterns from structured behavioral data. Hybrid models that combine traditional Machine Learning models with Deep Learning models have shown better detection accuracy. However, most existing works either combinedstaticanddynamicanalysisorusedasingle primary model, which is not very adaptable.

This work, therefore, proposes the design of a completely behavior-driven hybrid model that combines multiple learning paradigms to improve detection accuracy and robustness.

A. Existing System

In today's state of cybersecurity, most malware detection systems are still relying on traditional methods. Most of the existing systems are using signature-based detection methods. These systems compare files ornetwork traffic data with a database ofknownmalwaresignatures.Ifamatchisfound,the file is identified as malicious. Although this method is effective for known malware, it is not capable of detecting unknown malware attacks.

Most of the existing research models are also using static analysis methods. In static analysis, the executablefilesareanalyzedwithoutactuallyrunning them. File structure, binary code, and other code- related information are extracted and used for classification.Althoughthismethodisfasteranddoes notrequirerunning thefile, it isnot effective against advanced malware attacks. Modern attackers are using methods such as encryption, packing, and obfuscation toconcealmaliciouscode, making static analysis less effective.

To makethe system more efficient, researchershave developed Machine Learning models such as RandomForestandSupportVectorMachine(SVM). These models analyze patterns from extracted features and classify them as either malicious or benign. They showed improved accuracy compared to traditional signature-based detection systems. However, most of these systems are still relying on static features, which makes them less capable of detecting advanced and behavior-based malware attacks.

Some of the existing systems have recently started using dynamic analysis techniques. In dynamic analysis, the program is actually run in a controlled environment, and its behavior is analyzed. Features such as memory usage patterns, process activities, system calls, and network traffic patterns are extracted. These systems are more effective in detecting unknown malware compared to static analysis techniques. However, most of the existing dynamic analysis-based systems are requiring high computational power and may not be including real- time alerting capabilities.

III. METHODOLOGY

The proposed framework operates entirely on behaviouralinformationcollectedduringprogram execution. Nostaticfileattributesor signature-based indicatorsareutilized.Unliketraditionalsystems,this methodologydoesnot usestaticfeaturesor APIcall- based features. Applications are executed within a controlled environment to record runtime activities. The collected dataset contains both legitimate and malicioussamples,eachlabelledaccordingly.

A. DataCollection

The first step in the system is collecting dynamic behavioural data. The dataset contains runtime information generated while applications are executedinacontrolledenvironment.Thesedynamic features include system behaviour patterns, process activities, memory usage behaviour, and network traffic characteristics. Since the system focuses only on behaviour-based analysis, nostatic fileproperties are considered. The collected dataset contains both benign and malicious samples to train the classification models effectively.

B. DataPreprocessing

Before training the models, the collected data is cleaned and prepared. In this stage: Missing or inconsistent values are removed. Feature values are normalized to maintain uniform scale. Irrelevant or redundant behavioural features are eliminated. This step improves model performance and ensures that the data is suitable for training.

C. FeatureExtraction(DynamicBehavioural Features Only)

In this phase, important behavioural features are selected from the runtime dataset. These features representhowa program behaves while executingin the system. Since the proposed framework strictly avoids static and API call-based features, only behaviouralcharacteristicssuchasexecutionpatterns and system activity indicators are used. These features help the models understand the difference between normal and malicious behaviour.

D. ModelTraining

Thesystemusesthreeclassificationmodels:

1) *Random Forest:* Random Forest is an ensemble-based algorithm that creates multiple decision trees and combines their outputs.Itiseffectiveinhandlingcomplexandhigh- dimensional behavioural data. It also reduces overfitting and improves classification stability.
2) *SupportVectorMachine(SVM):*

SVM is used to create optimal decision boundaries between malicious and benign samples. It performs well in binary classification problems and helps improve detection accuracy.

3) *ConvolutionalNeuralNetwork(CNN):*

CNNisaDeepLearningmodelusedtoautomatically extract hidden patterns from structured behavioural data. It identifies complex relationships between features and improves detection of advanced malware.

Each model is trained separately using the prepared dynamic dataset.

E. EnsembleDecisionMechanism

After individual training, the outputs of Random Forest, SVM, and CNN are combined using an ensemble strategy. The final prediction is generated based on majority voting or probability averaging. Thishybridapproachimprovesaccuracyandreduces false positive rates compared to single-model systems.

F. AlertGeneration

If the final prediction indicates malicious behaviour, the system immediately generates a real-time alert. This alert mechanism helps in taking quick action to prevent further damage. The alert system makes the framework practical for real-world security applications.

IV. MODELLING AND ANALYSIS

This section explains how the proposed Intelligent MalwareDetectionandAlertSystemismodelledand evaluated. Thecomplete modellingprocessiscarried out using only dynamic behavioural features collectedduringprogramexecution.Thesystemdoes not use any static features or API call-based information. The main objective of this modelling phase is to train intelligent models that can clearly distinguish between normal and malicious runtime behaviour.

A. DataPreparationandExperimentalSetup

The collected dynamic behavioural dataset contains runtime characteristics such as execution behaviour, process-related activities, memory usage patterns, andnetworkbehaviourindicators.Eachsampleinthe dataset is labelled as either benign or malicious.

Before training the models, the dataset is pre- processed to improve quality and consistency. Missing values are handled, irrelevant behavioural attributes areremoved, and feature scalingisapplied where necessary. This ensures that all features contribute equally during model training.

The dataset is divided into two main parts: training data and testing data. The training dataset is used to allowthemodelstolearn behavioural patterns,while the testing dataset is used to evaluate how well the models perform on unseen data. This approach ensures fair and realistic performance evaluation.

B. RandomForestModelling

RandomForestisusedasoneoftheprimarymachine learning models in this framework. It isan ensemble algorithm that builds multiple decision trees using different subsets of the dataset. Each tree makes an individual prediction, and the final output is determined using majority voting. Random Forest is suitable for dynamic behavioural data because it handles high-dimensional features effectively. It also reduces overfitting by averaging the predictions of multiple trees. In this system, Random Forest learns structured runtime behaviour patterns that differentiate malicious activities from normal system operations. During testing, it compares new behavioural inputs with previously learned patterns to generate predictions.

C. SupportVectorMachineModelling

Support Vector Machine is used as a binary classification algorithm to separate benign and malicious samples. SVM works by constructing an optimal decision boundary, also known as a hyperplane, that maximizes the margin between two classes.

SVM performs well in situations where the dataset containscomplexrelationshipsamongfeatures.Since dynamic behavioural data often includes multiple correlated attributes, SVM helps in creating a clear separation between normal and abnormal behaviour. When new runtime data is provided, SVM analyses its position relative to the learned decision boundary and classifies it accordingly.

D. ConvolutionalNeuralNetworkModelling

Convolutional Neural Network is used as the deep learning component of the system. CNN automatically extracts deeper and hidden patterns fromstructureddynamicbehaviouralfeatures.Unlike traditional machine learning algorithms, CNN does notrelyentirelyon manualfeatureselection.Instead, it learns hierarchical feature representations during training.

The CNN model consists of an input layer that receives dynamic features, convolution layers that perform feature extraction, activation functions that introduce non-linearity, and fully connected layers that produce the final classification output. By analysing complexruntimebehaviour patterns,CNN improves thesystem's abilitytodetect advancedand evolving malware.

E. HybridEnsembleStrategy

After training Random Forest, SVM, and CNN independently, their outputs are combined using an ensemblestrategy.Thefinalclassificationdecisionis generated usingmajorityvoting or probability-based aggregation.

This hybrid approach increases reliability because each model contributes its strengths to the final prediction.

Random Forest captures structured behavioural trends,SVMcreatesstrongclassificationboundaries, andCNNidentifiesdeephiddenruntimepatterns.By combining these models, the system reduces false positive and false negative rates. The ensemble method improves stability and enhances overall detection performance.

F. PerformanceEvaluationandAnalysis

Toevaluatetheeffectivenessoftheproposedsystem, standard performance metrics are used. Accuracy measures the overall correctness of predictions. Precisionindicateshowmanydetectedmalware samplesare malicious. Recall measures thesystem's ability to detect real malware instances. F1-score provides a balanced measure of precision and recall. The confusion matrix presentsa detailed breakdown of true positives, true negatives, false positives, and false negatives.

Experimental analysis shows that individual models achieve strong classification performance when trained on dynamic behavioural features. However, the hybrid ensemble model provides better results comparedtosingle-modelapproaches.Thecombined system improves detection rate, reduces misclassification, anddemonstratesstrongcapability in identifying unknown and evolving malware threats.

By focusing completely on runtime behavioural analysis and integrating both machine learning and deep learning techniques, the proposed modelling framework becomes more robust, adaptive, and suitable for real-world cybersecurity applications

V. RESULTS AND DISCUSSIONS

This section presents the experimental results of the proposed Intelligent Malware Detection and Alert System and discusses its overall performance. The evaluation is carried out using only dynamic behavioural features collected during runtime execution. The system is tested using three models: RandomForest,SupportVectorMachine(SVM),and Convolutional Neural Network (CNN), along with their hybrid ensemble combination.

A. Performance of Individual Models

After training and testing the models using the prepared dataset, each classifier produced strong detection performance.

Random Forest showed stableandconsistentresults. It handled high-dimensional behavioural data effectivelyandprovidedgoodaccuracywithreduced overfitting.Themodel performed well in identifying structured runtime behaviour differences between malicious and benign samples.

Support Vector Machine achieved strong classification performance by creating a clear separation between normal and malicious behavioural patterns. It showed good precision and helped reduce false positive rates.

Convolutional Neural Network demonstrated the ability to detect deeper and more complex behavioural relationships. CNN performed better in identifying subtle malware patterns that may not be easily captured by traditional machine learning algorithms.
Although each model performed well individually, smallvariationswereobservedinprecisionandrecall values depending on the complexity of behaviouralpatterns.

B. Hybrid Model Performance

WhentheoutputsofRandomForest,SVM,andCNN were combined using an ensemble strategy, the overall system performance improved significantly. The hybrid model produced higher accuracy compared to individual models.
The ensemble approach reduced misclassification because the final prediction was based on combined model decisions. If one model produced a weaker prediction,theothermodelshelpedbalancetheresult. This improved detection stability and reduced both false positives and false negatives.
Thehybridsystemdemonstratedbettergeneralization capability when tested on unseen runtime data.

C. EvaluationMetricsAnalysis

The system performance was evaluated using standard classification metrics. Accuracy measured the overall correctness of predictions and showed improvement in the hybrid model compared to standalonemodels.
Precision indicated how many detected malware samples were trulymalicious. High precision values showed that the system effectively minimized false alarms.
Recall measuredthedetectionrateofactual malware samples. The system achieved strong recall, which means most malicious activities were successfully identified.
F1-score provided a balanced performance measure between precision and recall. The hybrid model achieved the highest F1-score, indicating stable and reliable classification.
The confusion matrix analysis showed that true positiveandtruenegativevalueswerehigh,while false positive and false negative values were comparatively low. This confirms that the system performs effectively in distinguishing malicious and benign runtime behaviour.

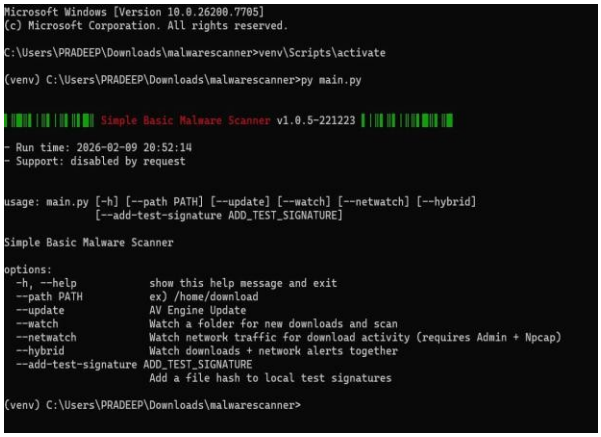
D. Discussion

From the experimental results, dynamic behavioural features provide strong capability in identifying malwareactivities.Sincethesystemdoesnotdepend onstaticorAPIcall-basedfeatures,itismoreresilient againstobfuscationandcodemodificationtechniques used by attackers.
The combination of Machine Learning and Deep Learning models enhances detection robustness. Random Forest captures structured behaviour patterns, SVM strengthens decision boundaries, and CNN extracts deep hidden relationships. Together, theycreateareliableandadaptivemalwaredetection framework.
The results indicate that the proposed hybrid system is suitable for real-time malware detection environments. It improves detection accuracy, reduces false alarms, and adapts better to evolving cyber threats compared to traditional approaches.
Overall, theexperimental analysisconfirmsthatusing only dynamic behavioural features along with a hybrid ML–DL approach leads to an efficient and intelligent malware detection system.

VI. OUTPUT SCREENS

This section presents the output screens of the developed malware detection system. The screenshots show the execution of the system in a Windows command-line environment and the generated log records after scanning. These outputs demonstrate the practical implementation and working of the proposed framework.

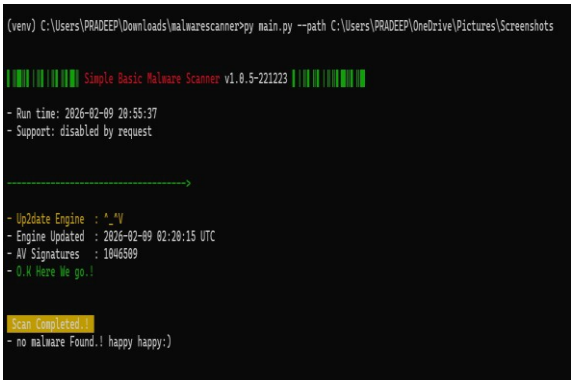
A. System Initialization



The first output screen shows the successful activationofthevirtualenvironmentandexecutionof themainPythonfile.Afterrunningthecommand,the system displays the title “Simple Basic Malware Scanner” along with the version information. The console also shows the runtime details and engine status. This confirms that the malware detection application is properly installed and ready for execution.

Thehelpmenudisplayedintheconsolelistsavailable optionssuchasscanningaspecificpath,updatingthe antivirus engine, monitoring folders, and enabling hybrid scanning. This indicates that the system supports multiple operational modes and flexible execution.

B. ScanExecutionandEngineUpdate



The second output screen shows the scanning process.Thesystemupdatestheantivirusengine beforestartingthescan. Itdisplaystheengineupdate time and the total number of available signatures. This step ensures that the detection mechanism uses the latest available threat intelligence.

After updating, thesystem performsscanning on the specifieddirectorypath.Oncethescanningprocessis completed, the message “Scan Completed” is displayed. In this particular execution, the result showsthatnomalwarewasfound.Thisconfirmsthat the system successfully scanned the selected directory and generated a detection result.

C. LogFileGeneration

Thethirdoutputscreendisplaysthegeneratedlogfile entries. Each log record contains important information such as date and time of scan, scan ID, operatingsystem,hostname,IPaddress,andinfection status. The infected file field shows "None," indicating that no malicious files were detected during these scans.

The log generation feature is important because it provides traceability and maintains a history of scanning activities. This helps in monitoring system securityovertimeandsupportsfurtheranalysisifany suspicious activity is detected in future.

D. OverallObservation

The output screens clearly demonstrate that the malware detection system executes successfully, updates the detection engine, performs scanning operations,andgeneratesstructuredlogs.Theresults confirm that the system is functional and capable of scanning directories in a controlled environment.

These implementation results support the effectiveness of the proposed malware detection framework and validate its practical applicability in real-time security monitoring scenarios.

VII. CONCLUSIONS

This study presented a behavior-oriented malware detection framework that integrates Machine Learning and Deep Learning techniques within a unified architecture. By focusing exclusively on runtime characteristics, the system avoids the weaknesses associated with static analysis and signature dependence.

Experimental evaluation demonstrates that each individual classifier performs effectively when trained on behavioral data. However, combining RandomForest,SVM,andCNNthroughanensemble mechanismproduces superior accuracyandstability. The hybrid configuration minimizes both missed detections and false alarms.

Thefindingsconfirmthatruntimebehavioralanalysis offers stronger resilience against evolving malware variants, including obfuscated and zero-day attacks. The proposed approach provides a scalable and practical foundation for intelligent security monitoring systems..

VIII. FUTURE SCOPE

Thefutureimprovementsmayinvolveexpandingthe behavioral dataset to include more diverse and real-worldmalwarefamilies.Largerdatasetscanenhance generalization and improve resistance to emerging threats.

Optimization of the deep learning component is anotherpotentialdirection.Lightweightarchitectures or pruning techniques may reduce computational requirements and enable deployment in resource- constrained environments.

Integration into live enterprise networks would furthervalidatereal-timeeffectiveness.Additionally, incorporating explainable AI mechanisms could improve transparency by providing interpretable justifications for classification decisions. The computational efficiency of the deep learning model can also be optimized. Since CNN models may require higher processing power, lightweight architecturesormodeloptimizationtechniquescanbe applied to reduce training and prediction time. This will make the system more suitable for real-time applications with limited hardware resources.

Further research can explore advanced ensemble techniques or adaptive learning mechanisms. For example, models can be designed to update themselves automatically when new malware patterns are detected.

In addition, the system can be extended by incorporating explainable AI techniques. Providing explanations for why a sample is classified as malicious will increase transparency and trust in securityenvironments.

Overall, the future scope of this research lies in improving scalability, adaptability, computational efficiency, and real-time deployment. With further enhancements,theproposeddynamicbehavior-based hybridframeworkcan becomeahighlyeffectiveand practical solution for modern cybersecurity challenges.

REFERENCES

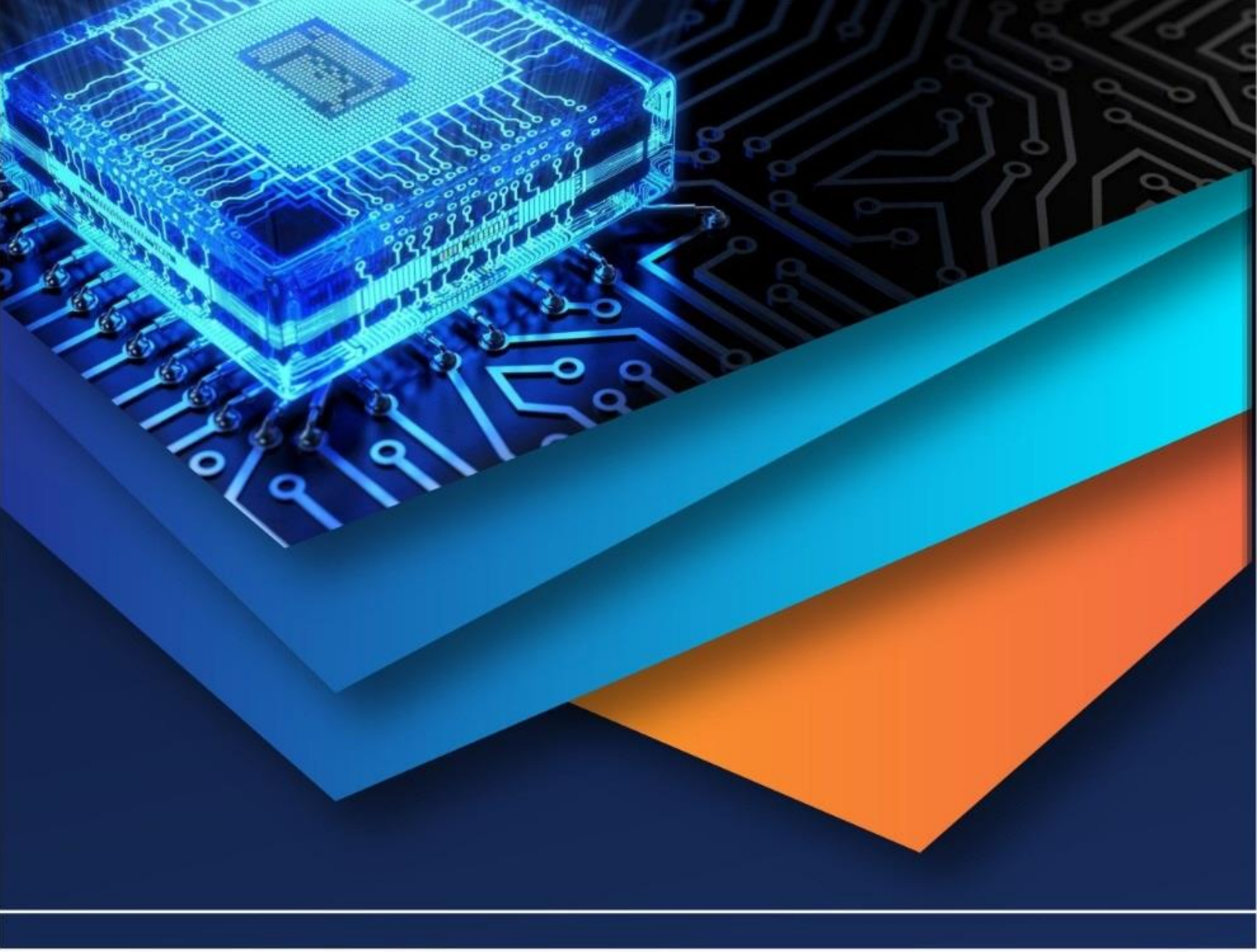
[1] J.F.Cantone,G.DeGaspari,M.G.Pizzuti,and A. Saccà, "Machine Learning in Network Intrusion Detection: A Cross-Dataset Generalization Study," IEEE Access, vol. 12, 2024, Doi: 10.1109/ACCESS.2024.3472907.

[2] J.Alex,R.Kumar,andS.Mathew,"AMachine Learning and Deep Learning Approach to Network Intrusion Detection System," in Proceedings of the IEEE International Conference on Electrical, Computer and Communication Engineering (ECCE), 2025, Doi: 10.1109/ECCE64574.2025.11013840.

[3] A. Elsersy and M. Abdelwahab, "Towards Transparent IoT Malware Detection: A ML/DL and XAI-Based Multi-Class Classification Approach," in Proceedings of IEEE WINCOM, 2025, Doi:10.1109/WINCOM65874.2025.11313366.

[4] K. Anuradha, P. R. Kumar, and S. S. Kumar, "Improving Malware Detection Performance Using Hybrid Deep Representation Learning with Heuristic Search Algorithms," Scientific Reports,2026,Doi:10.1038/s41598-026-35481-x.

[5] A.AbuAlhassan,M.Alkasassbeh,andA.Almomani,"MalwareRecognition Using Novel Convolutional Neural Network with Residual Connections," International Journal of Machine Learning and Cybernetics, 2026, Doi: 10.1007/s13042-025-02815-6.
[6] J. Park, H. Kim, and Y. Lee, "Smart Deep Learning Model for Enhanced IoT Intrusion Detection," Scientific Reports, 2025, Doi: 10.1038/s41598-025-06363-5.
[7] M. Rashid, T. Ahmed, and A. Mahmood, "Hybrid Android Malware Detection and Classification Using Deep Neural Networks," International Journal of Computational Intelligence Systems, 2025, Doi: 10.1007/s44196-025-00783-x.
[8] I. Sharafuddin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in Proceedings of the International Conference on Information Systems Security and Privacy (ICISSP), 2018.
[9] M.Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," in Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009.
[10] N.Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems," in Proceedings of the Military Communications and Information Systems Conference, 2015.
[11] J.Z. Kolter and M. A. Maloof, "Learning to DetectandClassifyMaliciousExecutablesinthe Wild," Journal of Machine Learning Research, vol. 7, pp. 2721–2744, 2006.
[12] A.Saxe and K. Berlin, "Deep Neural Network Based Malware Detection Using Two- Dimensional Binary Program Features," in Proceedings of the IEEE International Conference on Malicious and Unwanted Software (MALWARE), 2015.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY