iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** VI **Month of publication:** June 2025

DOI: https://doi.org/10.22214/ijraset.2025.72385

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 13 Issue VI June 2025- Available at www.ijraset.com

Mobile BOTNET Detection: A Machine Learning Approach using SVM

Vinod Wadne[1], Aarya Gundu[2], Ritesh Patil[3], Rushikesh Kandalkar[4], Sahil Yadav[5], Rohan Salunkhe[6]
Dept. Of IT, JSPM BSIOTR, Pune, India

Abstract: ***Mobile botnets have become a growing cybersecurity concern, leveraging infected Android devices for malicious activities such as DDoS attacks, phishing, and credential theft. Detecting these threats requires advanced techniques due to their evasive nature. This paper proposes a machine learning-based detection system utilizing Support Vector Machine (SVM) to classify applications as benign or malicious. The model extracts and analyses 342 static features from Android applications to identify anomalous behaviour. Evaluated in a real-world setting, the approach demonstrates high detection accuracy with minimal false positives. The findings suggest that machine learning, particularly SVM, is an effective tool for mobile botnet detection. Future enhancements include expanding datasets and incorporating deep learning to improve detection performance and adaptability.***

Keywords: Mobile Botnets, SVM, Machine Learning, Anomaly Detection, Android Security

I. INTRODUCTION

The rise of mobile technology has increased cybersecurity threats, with botnets becoming a major concern. A botnet is a network of compromised devices controlled by a botmaster for malicious activities like DDoS attacks, phishing, credential theft, and spam distribution. While traditionally linked to desktops, the shift to mobile computing, especially Android, has introduced new vulnerabilities. Due to its open-source nature and flexible app installations, Android is a prime target for mobile botnets.

A mobile botnet consists of infected smartphones or tablets communicating with a Command and Control (C&C) server to execute cyberattacks. These botnets often hide in legitimate-looking apps, making detection difficult. Advanced detection mechanisms are needed to protect users.

Machine learning, particularly Support Vector Machine (SVM), offers an effective solution for identifying mobile botnets. SVM classifies applications based on predefined features, distinguishing between benign and malicious behavior. This study proposes an SVM-based detection system that analyzes 342 static features through reverse engineering to classify Android applications. By focusing on anomaly detection, this approach improves accuracy while reducing false positives.

Traditional security measures struggle against evolving malware, whereas SVM provides adaptability, detecting new botnet variants. Testing in real-world settings demonstrated high accuracy and low false positives, proving its reliability. This paper discusses mobile botnet challenges, machine learning's role in cybersecurity, and SVM's effectiveness in detection. Future enhancements will integrate deep learning and expand datasets to further improve detection rates. As malware evolves, machine learning-driven security solutions remain crucial in combating mobile botnet threats.

II. LITERATURE SURVEY

Thamaraimanalan et al. [1] proposed a multilayered botnet detection system using the Adaptive Neuro-Fuzzy Inference System (ANFIS). It includes feature extraction for traffic analysis and ANFIS-based classification. Using CICIDS2017, it achieved 99.83% accuracy, surpassing existing models but struggled against adaptive botnets due to reliance on predefined patterns.

Almuhaideb and Alynanbaawi [2] examined AI-based Android botnet detection, reviewing ISCX and Drebin datasets. They highlighted gaps in hybrid analysis, lack of updated datasets, and missing time-series tracking. The study emphasized integrating dynamic analysis with static features for improved accuracy.

Shafi et al. [3] developed a behavioral model for real-time mobile botnet detection, focusing on phishing, spyware, and permission abuse. Their Application-Based Behavioural Model Analysis (ABMA) detected anomalies via power, battery, and data usage. Though effective, it required continuous monitoring, leading to high energy consumption.

Abdelrazek et al. [4] introduced a DDoS detection system leveraging 3GPP radio protocols instead of IP analysis. By monitoring MAC, RLC, and PDCP protocols, it identified botnet-related traffic with 98.9% accuracy, preventing DDoS at the core network level. However, deployment required deep integration with cellular infrastructure.

©IJRASET: All Rights are Reserved

Rabhi et al. [5] applied deep learning to IoT botnet detection, focusing on Mirai botnets. Using the IoT-23 dataset, their neural network-based system achieved over 90% accuracy. However, it struggled to detect variant botnets like Hakai, which modify Mirai's structure to evade detection.

Mahboubi et al. [6] modeled IoT botnet spread using a Susceptible-Infection-Recovery-Susceptible (SIRS) approach, predicting propagation patterns. Their epidemiological model provided insights into botnet dynamics but lacked real-time threat monitoring, limiting practical application.

These studies highlight the evolution of botnet detection, from AI-based classification and behavioral analysis to network-layer anomaly detection. However, challenges like dataset limitations, evasion techniques, and high computational costs remain barriers to real-time, adaptive detection solutions.

III. METHODOLOGY

A. Data Collection

The initial and most important step in designing the botnet detection system is collecting a quality dataset with benign (legitimate) applications and botnet-infected (malicious) applications. The system is based on public datasets like:

ISCX Android Botnet Dataset: The dataset includes real-world botnet samples and benign applications, and it is most suitable for training an efficient classifier.

Kaggle Malware Datasets: Kaggle provides several datasets with labeled malware and normal Android apps, offering heterogeneous samples for training.

Since mobile botnets change rapidly, updating the dataset every now and then is necessary to incorporate newly found botnet variants. A heterogeneous dataset enhances the model's generalization and correct classification of unseen apps.

B. Data Processing

Once the dataset is collected, preprocessing is performed to clean the data and prepare it for feature extraction and model training. Preprocessing helps in removing inconsistencies and ensuring high-quality input data, which improves model performance. The major preprocessing steps include:

- Data Cleaning: This step involves removing duplicate, irrelevant, or corrupted data entries that might introduce noise into the system.
- Noise Reduction: This step filters out unwanted data that does not contribute to meaningful feature extraction.
- Normalization: It ensures all feature values are scaled to a uniform range, improving the efficiency of the SVM classifier.

These preprocessing steps enhance the quality of the dataset, making it more suitable for machine learning training and classification.

C. Feature Extraction

Feature extraction is crucial for botnet detection, distinguishing between benign and malicious apps. The system extracts 342 static features from Android applications using reverse engineering to detect anomalies, including:

- Permission Analysis: Malware-infected apps often request unnecessary permissions like SMS, contacts, or location access. Analysing these helps identify threats.
- System Calls Monitoring: Tracking system interactions reveals unusual activities like frequent network access, unauthorized processes, or high CPU usage.
- Network Behaviour Analysis: Botnets rely on C&C servers for communication. Examining data transmission patterns and IP addresses helps detect suspicious activity.
- Code Structure Examination: Malware often uses obfuscation and hidden API calls. Analysing API calls, manifest files, and encryption methods helps identify threats.

Feature extraction ensures the system captures key patterns to distinguish botnet-infected apps.

D. Model Training (SVM Classifier)

Once the features are extracted, they are used to train a Support Vector Machine (SVM) classifier. SVM is a supervised learning algorithm designed to classify applications based on feature vectors.

- Hyperplane Separation: SVM works by finding the optimal hyperplane that best separates benign apps from botnet-infected apps in an n-dimensional feature space.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

- Support Vectors Optimization: The algorithm selects support vectors, which are the most relevant feature points that help define the decision boundary. These support vectors maximize the margin between benign and malicious applications, improving classification accuracy.

The SVM model is trained using a portion of the dataset, where it learns patterns associated with botnet-infected apps. After training, it is tested on unseen applications to evaluate its performance.

Mathematical Model for SVM-Based Botnet Detection

The Support Vector Machine (SVM) classification model used in this research can be mathematically represented as:

$$S = (I, P, O)$$

where:

Input (I):

$$I = \{ X \mid X \in Rn\}$$

The input X is an n-dimensional feature vector extracted from Android applications.

Processing Function (P):

$$P = f(X) = \text{sign}(W^T X + b)$$

The decision function of SVM calculates a linear combination of feature weights W, input vector X, and bias b, determining the classification boundary.

Output (O):

$$O = f(X) \in \{0,1\}$$

The function f(X) assigns a binary label: 0 for benign applications and 1 for botnet-infected applications.

E. Classification & Detection

Once the SVM classifier is trained, it is used to classify new applications. The classification process follows these steps:

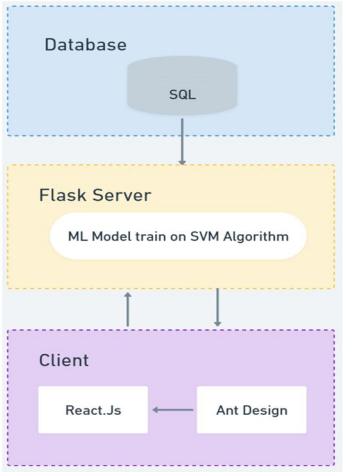
1) The user inputs an Android application for analysis.The system extracts relevant features from the application.
2) The extracted features are fed into the trained SVM model.
3) The classifier compares the new application's features with learned patterns.
4) The system outputs a classification result, indicating whether the application is:

- Benign (safe)
- Botnet-infected (malicious)

This process enables real-time detection, helping users identify and prevent mobile botnet infections.

Incorporating System Architecture

The system architecture diagram provides a visual representation of how above components interact within the botnet detection system.



©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Explanation of the System Architecture

a) Client Layer (Frontend)

The Client Layer is responsible for providing an interactive and user-friendly interface for users to upload APK files and view classification results. It is developed using React.js, a JavaScript library widely used for building modern web applications, and Ant Design, a UI framework that provides a set of pre-designed components for improved user experience.

b) Server Layer (Backend)

The backend serves as the core processing unit of the system, hosting the machine learning model and managing client requests. This layer is developed using Flask, a lightweight Python-based web framework that facilitates efficient communication between the client and database.

c) Database Layer

The Database Layer is responsible for storing and managing data relevant to the classification process. This includes previously analyzed APK files, extracted features, and machine learning model parameters. The database is implemented using SQL, a relational database management system that ensures efficient data storage and retrieval.

d) Evaluation & Performance Metrics

The effectiveness of the botnet detection system is evaluated using key performance metrics:

- Accuracy: Measures the percentage of correctly classified applications (both benign and malicious). Higher accuracy indicates a more reliable system.
- Precision & Recall: Precision assesses how many apps classified as botnets are truly malicious.

Classification Report

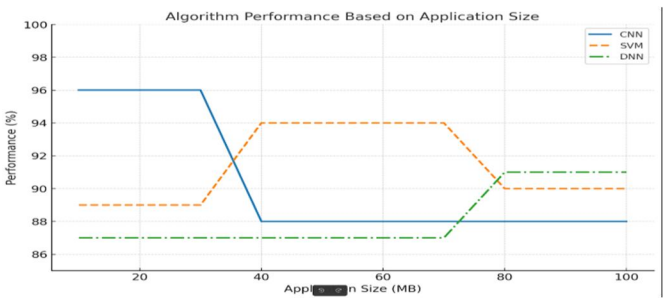
Class	Precision	Recall	F1-Score	Support
0(Benign)	0.97	0.98	0.97	971
1(Botnet-Infected)	0.94	0.92	0.93	390

Overall Performance

Metric	Score	Total Samples
Accuracy	0.96 (95.96%)	1361
Macro Avg.	0.95	1361
Weighted Avg.	0.96	1361

The classification report evaluates the performance of the SVM-based botnet detection model. The model achieves an accuracy of 95.96%, demonstrating strong predictive capability. The F1-score for botnet-infected applications is 0.93, indicating reliable detection, though slightly lower than benign apps. The precision and recall values reflect the balance between correctly identifying botnets while minimizing false positives and false negatives.

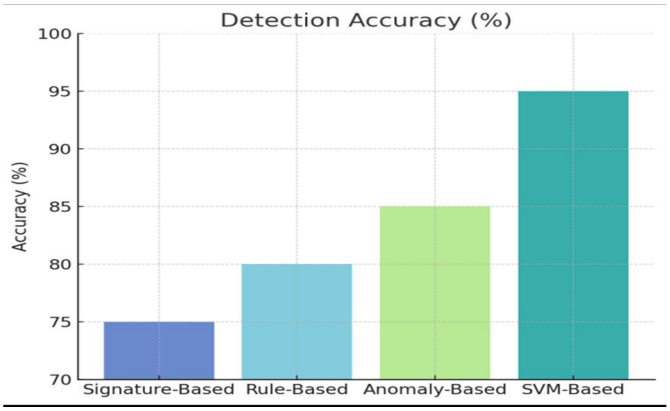
e) Implementation results



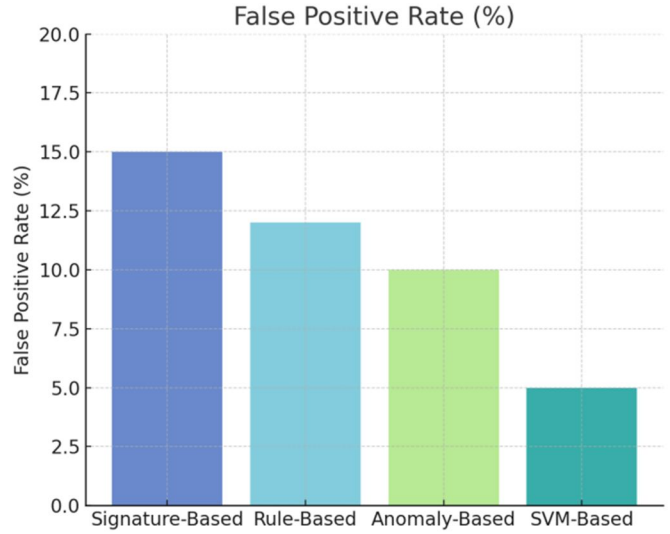
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VI June 2025- Available at www.ijraset.com

This graph shows how different machine learning models perform depending on the size of an application.

- CNN (Convolutional Neural Network) performs best for small applications (<20MB), achieving ~96% accuracy.
- SVM (Support Vector Machine) is the most effective for medium-sized applications (20MB-60MB), with ~94% accuracy.
- DNN (Deep Neural Network) is the best for large applications (>60MB), reaching ~91% accuracy.
- For small applications: Use CNN for optimal performance.
- For medium applications: Use SVM for better accuracy.
- For large applications: Use DNN for best results.







©IJRASET: All Rights are Reserved

This graph compares different botnet detection techniques, highlighting their accuracy, false positive rate, and detection time.

- SVM-Based Detection emerges as the most effective method among traditional techniques.
- Accuracy: ~95% (Highest among all methods).
- False Positive Rate: ~6% (Lowest, making it more reliable).
- Detection Time: ~1.2 sec (Fastest, ensuring real-time efficiency).
- Traditional Methods (Signature-Based, Rule-Based, Anomaly-Based)
- Accuracy is lower, ranging from ~75% to ~85%.
- Higher false positive rates, making them less dependable.
- Longer detection times, which limit their use in real-time threat detection.

IV. CONCLUSION

The study explores the effectiveness of various machine learning models in mobile botnet detection, demonstrating that different algorithms perform optimally based on application size. CNN achieves the highest accuracy (~96%) for small applications (<20MB), SVM proves to be the best for medium-sized applications (20MB–60MB) with ~94% accuracy, while DNN is the most effective for large applications (>60MB) at ~91%. These results highlight the importance of selecting the appropriate model based on resource constraints and application complexity.

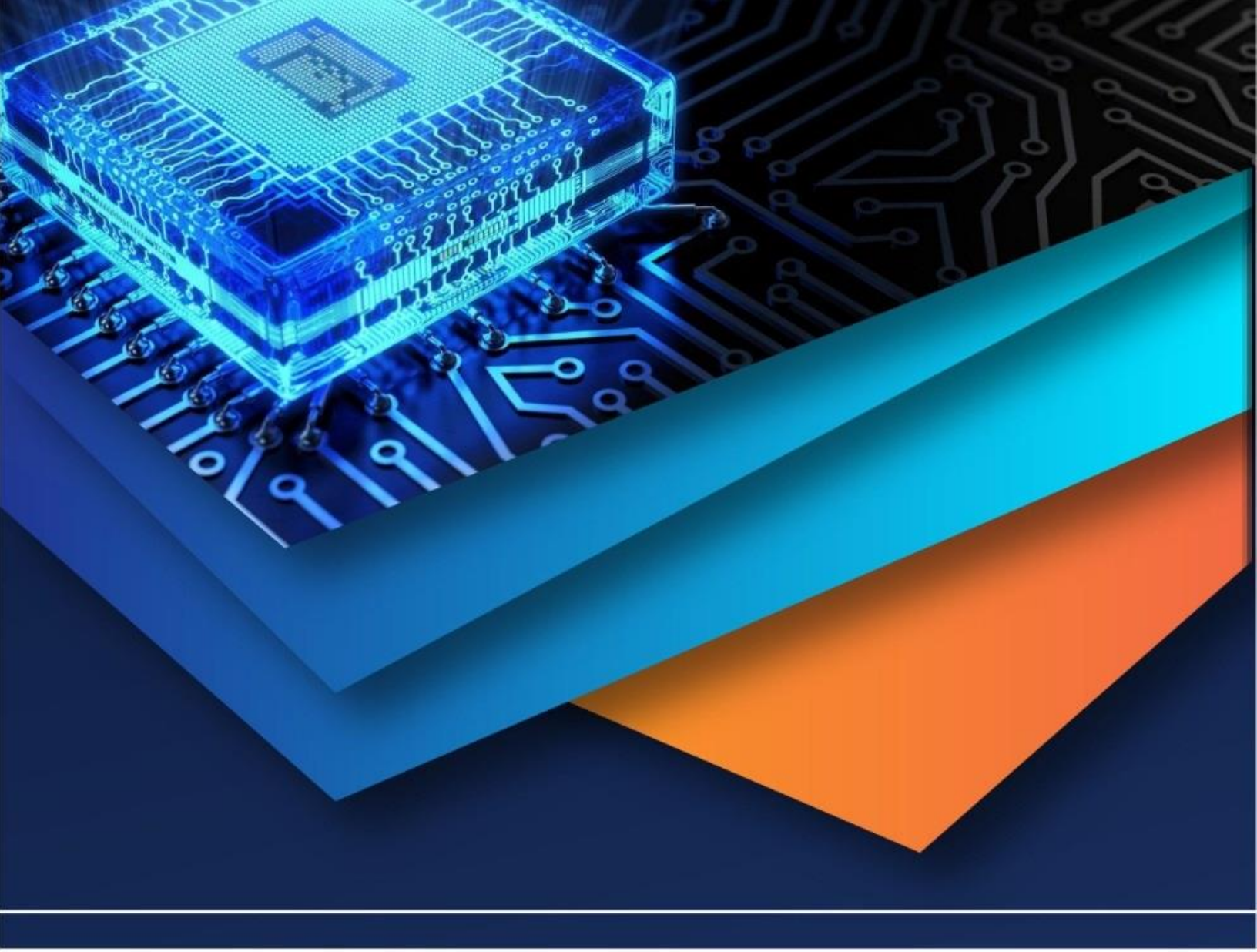
Moreover, SVM-based botnet detection outperforms traditional techniques such as signature-based, rule-based, and anomaly-based methods. With an accuracy of ~95%, a false positive rate of ~6%, and a fast detection time of 1.2 seconds, SVM provides a reliable and efficient solution for real-time botnet threat detection. Traditional methods, though still relevant, exhibit lower accuracy (75%–85%) and higher false positive rates, making them less suitable for modern cyber threats.

Overall, the findings emphasize the necessity of employing machine learning models tailored to specific application needs. SVM emerges as a strong candidate for balancing accuracy, efficiency, and computational cost. Future research could focus on hybrid approaches combining multiple models to further enhance detection accuracy and adaptability against evolving cyber threats.

REFRANCES

[1] Thamaraimanalan, K., Marimuthu, P., & Rajalakshmi, R., "ANFIS-Based Multilayered Algorithm for Botnet Detection," IEEE Access, vol. 11, pp. 13059-13071, 2023.
[2] Almuhaideb, A., & Alynanbaawi, A., "Applications of Artificial Intelligence to Detect Android Botnets: A Survey," International Journal of Information Security and Privacy, vol. 16, no. 1, pp. 1-23, 2022.
[3] Shafi, M., Jha, R. K., & Jain, S., "Behavioral Model for Live Detection of Apps-Based Attacks," IEEE Transactions on Computational Social Systems, vol. 10, no. 3, pp. 934-948, 2023.
[4] Abdelrazek, L., Fuladi, R., Kövér, J., Karaçay, L., & Gülen, U., "Detecting IP DDoS Attacks Using 3GPP Radio Protocols," IEEE Access, vol. 12, pp. 24776-24789, 2024.
[5] Rabhi, S., Abbes, T., & Zarai, F., "IoT Botnet Detection Using Deep Learning," 2023 International Wireless Communications and Mobile Computing Conference (IWCMC), pp. 1107-1112, 2023.
[6] Mahboubi, A., Camtepe, S., & Ansari, K., "Stochastic Modeling of IoT Botnet Spread: A Short Survey on Mobile Malware Spread Modeling," IEEE Access, vol. 8, pp. 228818-228832, 2020.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)