ENHANCED BOTNET DETECTION AND SECURITY USING FUZZY LOGIC AND THE J48 DECISION TREE ALGORITHM

Supported by tetfund
TERTIARY EDUCATION TRUST FUND

[1]Agu, Edward O., [2]Bala, Tsenfa, [3]Ishaya, Nubunga
[1,2]Department of Computer Science, Federal University Wukari, Taraba, Nigeria
[3]Department of Computer Science, Federal University of Education, Zaria, Nigeria
Corresponding author: aguedward@fuwukari.edu.ng

Received: September 5, 2025, Accepted: November 28, 2025

Abstract

Botnets are a big and constant problem for computer security worldwide. They take advantage of weaknesses in devices that are online to start huge attacks, like preventing people from using websites (DDoS), stealing info, and spreading bad software. This research shows a mix of ways to find botnets. It puts together Fuzzy Logic and the J48 Decision Tree to get better results and fewer mistakes. Fuzzy logic is good at picking out what's important because it can deal with data that's not clear. The J48 thing helps make solid decisions based on what's going on with network traffic. The system was tested using four datasets that anyone can get: CTU-13 (Scenario 1), Obfuscated-MalMem2022, log-CTI, and CIC Botnet. The system was right 95%, 100%, 100%, and 64% of the time on each one. This means it can find botnets in lots of situations. This method is easy to expand and understand, and it can help spot botnets early. It could even be used in real time to keep computer systems safe.

Keywords: Botnet Detection, J48 Algorithm, Fuzzy Logic, Machine Learning, and Decision Tree.

Introduction

In the last few years, botnets have become one of the biggest threats to cybersecurity because they can hack millions of devices and plan big attacks. Attackers sometimes use these networks of hacked devices that are spread out and managed from afar to carry out Distributed Denial of Service (DDoS) assaults, steal private information, spread malware, and commit click fraud (Zahraa, Eman, Dalia, & Radhwan, 2020). Botnets are hard to find because they hide and change all the time to get around normal security measures.

The Internet of Things (IoT) and the Internet of Everything (IoE) have made this problem worse. In these situations, a lot of smart devices don't have good security features, so they are easy targets for botnet recruiting (Chirag, Ranjeet, & Vishal, 2021). Cybercriminals are still using these flaws to their advantage, and botnets are being used more to carry out targeted, long-lasting, and financially damaging attacks all over the world.

There are many ways to find botnets, but a lot of the old ones aren't very flexible or accurate. Researchers have turned to Machine Learning (ML) and Artificial Intelligence (AI) to make detection methods that are more intelligent and automatic. One of them, the J48 decision tree, has been very good at finding bad patterns in network traffic (Mathur, Raheja, & Ahlawat, 2018). J48 is a Java version of the C4.5 algorithm. People like it because it's easy to understand, powerful, and accurate when used for supervised learning tasks.

But decision trees alone may not be able to show the uncertainty and ambiguity that are common in real-world network traffic data. As a second strategy to solve this problem, fuzzy logic has been added. It works with unclear or hazy information in a way that is similar to how people think (Gardner & Nagaraja, 2016). Fuzzy logic helps decision tree algorithms handle input that isn't clear. This makes it easier to find things and cuts down on the number of false positives.

This study suggests a better way to find things by combining fuzzy logic with the J48 decision tree method. This is because botnets are getting more complex and have a bigger impact on digital infrastructure. The goal is to make it easier to spot botnet attacks by improving categorization and making security more responsive, especially in large, noisy datasets.

This paper adds a hybrid detection model that can find botnet activity by using smart feature selection, training on publicly available botnet datasets, and checking how well the model works with confusion matrix metrics. Adding fuzzy logic to the decision tree structure is meant to make it easier to find botnets and make it possible to monitor security in real time.

Literature Review

Since botnets are always changing, we need to use smart, flexible methods to make sure we can find and stop them. This review of the literature looks at current methods for finding botnets, with a focus on machine learning, fuzzy logic, and hybrid models. The J48 decision tree algorithm is given special attention.

Botnet Architecture and Detection Challenges

A botmaster controls a botnet through command and control (C&C) servers. These servers connect compromised devices in an organized way. Modern botnets can change their IP addresses, encrypt their data, and talk to each other directly, which makes old-fashioned static detection methods useless (Silva, Silva, & Pinto, 2013). Rule-based intrusion detection systems (IDS) can't keep up with new and sneaky attack patterns, which makes them more likely to miss real attacks.

Machine Learning in Botnet Detection

Advanced IDS now rely heavily on machine learning. Supervised learning methods let models learn from labeled datasets and tell the difference between good and bad network behavior. Support vector machines (SVM), k-nearest neighbors (KNN), random forests, and decision trees are some of the algorithms that have shown promise in finding anomalies (Moustafa et al., 2019). Decision trees are the best choice for network security applications because they are easy to understand and don't require a lot of computing power.

J48 Decision Tree Algorithm

The J48 decision tree is a Java version of the C4.5 algorithm. It can work with both categorical and continuous data. It makes a decision tree by picking the attributes at each node that give the most information (Quinlan, 1993). J48 is a popular choice for botnet detection because it is fast and reliable. For example, Mathur, Raheja, and Ahlawat (2018) showed that J48 was better than a number of other classifiers at finding malicious botnet traffic in the CTU-13 dataset in terms of precision and recall. But decision trees like J48 are naturally crisp classifiers, which means they have trouble when data has noise, ambiguity, or missing labels, which is common in real-world traffic datasets.

Fuzzy Logic in Security Systems

Fuzzy logic lets us think logically when we don't know for sure what is true by giving things degrees of membership instead of binary classifications (Zadeh, 1965). Because they are so flexible, fuzzy systems can be used to model vague ideas like "suspicious behavior" in network traffic. Gardner and Nagaraja (2016) said that fuzzy-based detection systems were better at adapting to quickly changing threat environments, especially in IoT networks that were not all the same.

Fuzzy logic has been used to sort traffic into groups based on fuzzy rules and language variables in botnet detection. But it can be hard to design standalone fuzzy systems in the best way possible without the help of data-driven learning algorithms.

Hybrid Models: Fuzzy Logic + Machine Learning

Recent research has looked into hybrid models that use both fuzzy logic and machine learning to get the best of both worlds. Sahu and Ghosh (2020) came up with a fuzzy-based ensemble learning system for finding network anomalies that works better than standard methods. A fuzzy-enhanced decision tree classifier can also add human-like reasoning to the tree-building process. This makes it more tolerant of imprecise input and less likely to overfit.

Singh and Sharma (2021) made a fuzzy decision tree to find botnets in big networks as part of a well-known study. Their model was able to find 95% of the cases, with a lot fewer false positives, which proved that this hybrid approach works.

Benchmark Datasets and Evaluation Metrics

Datasets such as CTU-13, UNSW-NB15, and CICIDS2017 have become standard for evaluating botnet detection systems. These datasets offer labeled traffic flows with detailed botnet behaviors and attack vectors (García et al., 2014). Performance metrics such as accuracy, precision, recall, F1-score, and false positive rate are commonly used to benchmark detection models.

Research Gap and Motivation

The literature talks a lot about how J48 and fuzzy logic could be useful for network security, but not much has been done to look into how they could be used together to find botnets. Most of the systems that are already out there use either crisp machine learning classifiers or fuzzy expert systems that don't learn new things on their own. Also, as botnets become more complicated, especially in cloud and IoT settings, we need models that are easy to understand and can change.

This study fills in the gaps by suggesting a hybrid fuzzy-J48 model for better botnet detection. The goal is to make the model more accurate, generalizable, and resistant to data uncertainty.

Research Methodology

Information gathering, data format conversion, feature extraction, classification, model assessment, and interpretation of results are the stages involved. The J48 decision tree is used for classification in this hybrid architecture, while fuzzy logic is used for feature selection. This allows for accurate and versatile botnet identification.

Datasets Used

Four publicly accessible datasets were use to make sure the assessment was strong and varied:

i. CTU-13 (Scenario 1): Includes.pcap files with botnet and benign traffic labels. Information taken from stratosphereips.org.
ii. Obfuscated-MalMem2022: This malware variant is designed to hide its true nature. Taken from unb.ca.
iii. The Log-CTI Dataset records every botnet activity in Malaysia over the years. Downloadable from Kaggle.
iv. CIC Botnet Dataset: A repository on Kaggle that contains generic datasets on botnet traffic.

Packet Capture (PCAP) to CSV Conversion

Most of the datasets were originally in .pcap format. To facilitate machine learning tasks, these files were converted into .csv format using Wireshark, a packet analysis tool. This conversion enabled structured data representation necessary for feature analysis.

Feature Extraction

Once in CSV format, the datasets were parsed to extract relevant traffic attributes such as:

i. Source/Destination IPs
ii. Port numbers
iii. Packet lengths
iv. Protocol types
v. Flow durations

This phase exposed all available features for further analysis.

Feature Selection Using Fuzzy Logic

Fuzzy logic was used to address uncertainty and imprecision in the significance of features. Each characteristic was categorized as "relevant" or "not relevant" using a fuzzy-based filter mechanism, based on linguistic values ("Yes" or "No").

Algorithm for pcap file format Conversion

Algorithm: Conversion of pcap file format to csv using Wireshark.
Phase 1 Initialization: Import pcap;
Phase 2 Read **pcap** Features as F1, F2, F3------Fn;
Phase 3 **While** All Features are processed **do**
end
Phase 4 **If** features are required features, **then**
| save the features to csv file format;
End
Phase 5 **If** not all features are required, **then**
| drop unnecessary features;
End
Phase 6 **If** step 5 is successful completed, **then**
| save the dataset to csv file format;
Else go to Phase 1;
EndIf

Algorithm for Fuzzy logic features selection

Algorithm: Features selection using fuzzy logic (yes/no)
Phase 1 Initialization: Import csv;
Phase 2 Read **csv**;
Phase 3 Feature(s); Relevant = yes, Not relevant = no
Phase 4 **If** All Features are displayed **AND** Feature = = Relevant, **then**
| Select most relevant feature(s),
Phase 5 **Else If** Feature(s) = = Not Relevant **then**
| go to Step 1,
Phase 6 **EndIf**

Model Training and Testing

After picking the important parts, The data was split into training and testing groups. Then a method where I made sure each group had a good mix of everything was utilize. The data was trained to model on bigger sets, and then checked how well it did on 100 random examples. When the data was too big to easily work with, smaller data were used, but still good, pieces to train the model.

Classification with J48 Decision Tree

The dataset was divided into training and testing sets following feature selection. Stratified sampling was employed, which usually involves testing on 100 randomly chosen cases for validation after training on larger samples. Representative subsets were used for training when the size of the dataset made computation difficult.

Hardware and Software Environment

Hardware

i. Processor: Intel Core i3, 2.00GHz

ii. RAM: 4 GB

iii. Storage: 999 GB HDD

Software

i. Operating System: Windows 10

ii. Programming Language: R

iii. Tools Used:

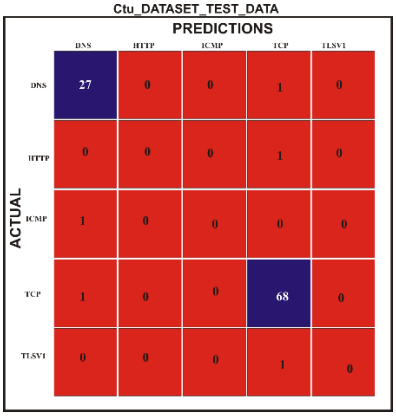
a) rpart, rpart.plot (for decision tree modeling)

b) Wireshark (for pcap to CSV conversion)

c) Microsoft Excel (for data handling)

Findings and Discussion

Table 1: *Confusion Matrix Visualization of ctu13dataset*



In the available features, this research has used; (57) Class and (1) Category, which are more relevance to this research. The figure 1 shows the Decision Tree output of the prediction of the dadaset.

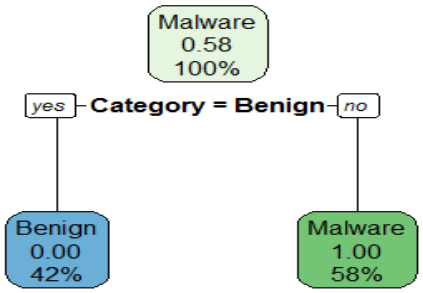


Figure 1: Decision Tree Visualization of Obfuscated-MalMem2022 Test Data

Table 2: *Classification of Obfuscated-MalMem2022 Dataset Test Data*

		PREDICTION	
		Benign	Malware
ACTUAL	Benign	42	0
	Malware	0	58

Table 2 shows that, in the 100 randomly selected sampled in the dataset, the model correctly predicted, Benign has 42 scenarios and were correctly predicted as True Positive, and Malware has 58 scenarios and where also correctly predicted as True Negative. This shows that the model performance rate is 100% accurately classification of the objects in the dataset using the test data.

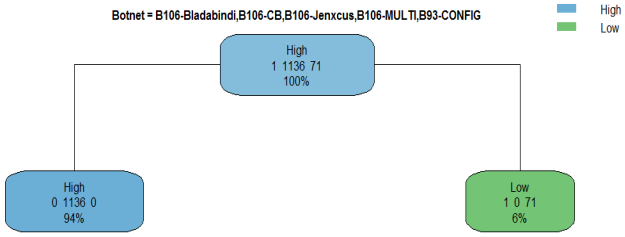


Figure 2 shows the decision tree output of the log-CTI Dataset.

Figure 2: Decision Tree log-CTI Dataset of Trained Data

Table 3: shows that CTU13 (Scenario 1) performance is 95%, Obfuscated-MalMem2022 performance is 100%, log-CTI Dataset performance is 100% and Dataset performance is 64% respectively. This shows that, three (3) datasets yielded good results base on their accuracy and result.

Table 3: shows that CTU13 (Scenario 1)

Dataset	**False Positive Rate (FAR)**	**Accuracy (ACC)**	**Result (%)**
CTU13 (Scenario 1)	0.04	0.95	95%
Obfuscated-MalMem2022	0	1	100%
log-CTI Dataset	0	1	100%
Dataset	0	0.64	64%

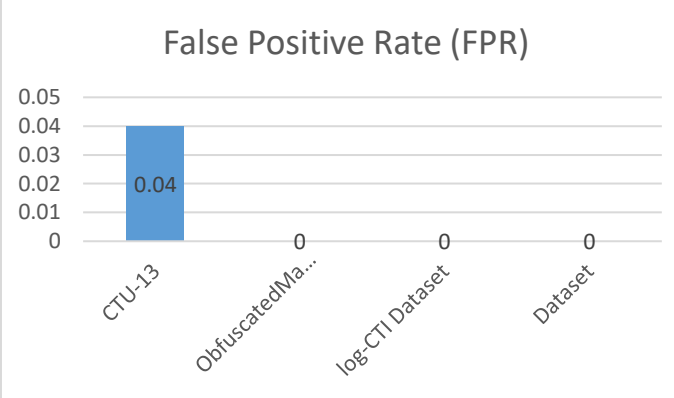


Figure 3: Presentation of False Positive Rate (FPR).

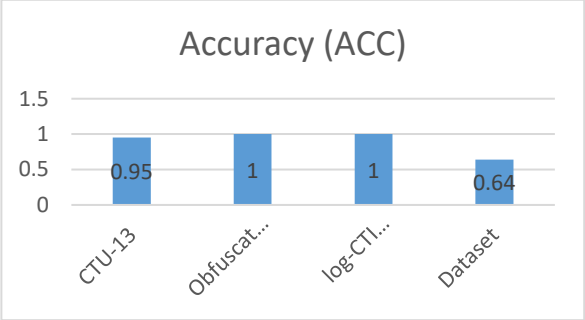


Figure 4: Presentation of Accuracy (ACC)

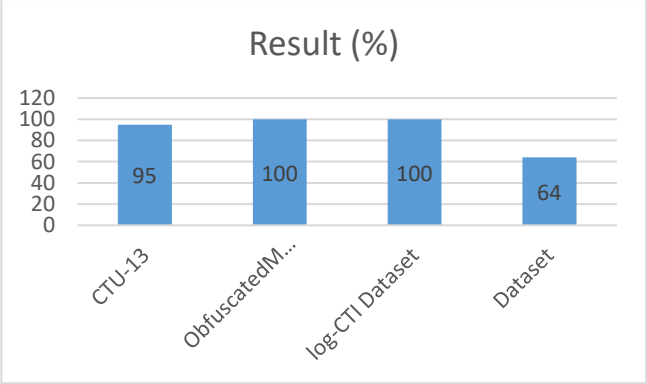


Figure 5: Presentation of Final Results.

Conclusion and Recommendation

By combining fuzzy logic with the J48 decision tree algorithm, this study improved the detection of botnets. By identifying important characteristics, this technique was able to identify compromised computers and botnet activity, making the detection more accurate and less expensive to execute. We tested it on several datasets, including log-CTI, Obfuscated-MalMem2022, and CTU13. With accuracy levels of 95%, 100%, 100%, and 64%, the model performed admirably, demonstrating the potential of fuzzy logic and J48 in the early detection and classification of botnets.

Here are some recommendations for the future:

i. To see how well the model performs, future experiments should use mixed datasets that display a wide variety of botnet actions. To get the best model results, cybersecurity professionals should consider feature selection techniques that prioritize the most important factors.

ii. Especially when performing intensive number crunching, make sure the data you're working with isn't too large to avoid crashing the system. The best method for identifying botnets as they appear on active networks may involve combining fuzzy systems with decision tree classifiers. To strengthen their security, network gatekeepers and those in charge would be well advised to employ intelligent detection systems such as this one.

References

Business Dictionary (online). Assessed by Coker, 2020 via www.google.com on 26th November, 2018.

Chirag, J., Ranjeet, K. R., & Vishal, B. (2021). A fuzzy logic-based feature engineering approach for botnet detection using ANN. *Journal of King Saud University – Computer and Information Sciences*. https://doi.org/10.1016/j.jksuci.2021.01.003

Coker, P. R. (2020). Understanding the Concept Of Knowledge Gap And Knowledge Expansion: A Theoretical Perspective. *Researchjournali's Journal of Management*, 7(3). ISSN 2347-8217.

Cope, J. (2002). What's a Peer-to-Peer (P2P) Network? Available online: https://www.computerworld.com/article/2588287/networking-peer-to-peer-network.html (accessed on 30 December 2021).

Cyberattacks in India in Q1 (2020): Report. (2020). https://ciso.economictimes.indiatimes.com/news/37-increase-in-cyberattacks-in-india-in-q1-2020-report/75962696.

Dissanayake, D. M. N. S. W. (2013). Research, Research Gap and the Research Problem. *MPRA Paper*, No. 47519.

Food Major Haldiram's Attacked by Ransomware. (2020). https://economictimes.indiatimes.com/industry/cons-products/food/food-major-haldirams-attackedby-ransomware-hackers-demanded-usd-750000-for-decryption/articleshow/78714817.cms

Gardner, J., & Nagaraja, S. (2016). On the security of machine learning in malware C&C detection: A survey. *ACM Computing Surveys*, 49(3), 1–38. https://doi.org/10.1145/3003816

Haq, S., & Singh, Y. (2018). Botnet detection using machine learning. In *Proceedings of the 5th International Conference on Parallel, Distributed and Grid Computing (PDGC)*, pp. 240–245. https://doi.org/10.1109/PDGC.20118.8745912

Khlood, S., Khalid, A., Ahmed, A., & Muhammad, U. A. (2021). Machine learning-based botnet detection in software-defined networks: A systematic review. *Symmetry*, 13(5), 866. https://doi.org/10.3390/sym13050866

Kuochen, W., Chun-Ying, H., Shang-Jyh, L., & Ying-Dar, L. (2011). A fuzzy pattern-based filtering algorithm for botnet detection. *Computer Networks*, 55(15), 3275–3286. https://doi.org/10.1016/j.comnet.2011.05.003

Wainwright, P., & Kettani, H. (2019). An analysis of botnet models. In *Proceedings of the International Conference on Compute and Data Analysis (ICCDA 2019)* (pp. 116–121). ACM. https://doi.org/10.1145/3314545.3314562