

# API-Driven Intelligent Agent for Real-Time Detection and Response Against Distributed Denial of Service (DDoS) Attacks Using Hybrid Machine Learning

Priyanshi Dahivelkar<sup>1</sup>, Anupama Dasari<sup>2</sup>, Aditi Elag<sup>3</sup>, Kumud Wasnik<sup>4</sup>

<sup>1</sup>Student, Dept of Computer Science and technology, Usha Mittal Institute of Technology, Maharashtra, India

<sup>2</sup>Student, Dept of Computer Science and technology, Usha Mittal Institute of Technology, Maharashtra, India

<sup>3</sup>Student, Dept of Computer Science and technology, Usha Mittal Institute of Technology, Maharashtra, India

<sup>4</sup>Professor, Dept of Computer Science and technology, Usha Mittal Institute of Technology, Maharashtra, India

\*\*\*

**Abstract** - Distributed Denial of Service (DDoS) attacks still hit network infrastructure hard they take services offline, drain money, and throw operations into chaos [1]. These attacks keep getting more complex and bigger, so the usual signature-based Intrusion Detection Systems just can't keep up [3]. They don't adapt in real time, can't automate defenses, and struggle to scale in fast-moving cloud setups. In this paper, we introduce an API-driven intelligent agent framework that tackles DDoS threats as they happen. We use a hybrid machine learning setup: ensemble models like Random Forest and Gradient Boosting work alongside deep learning with Long Short-Term Memory (LSTM) networks [5]. This combo lets us catch both patterns in the data and shifts over time. We built a RESTful API layer on Fast API so the system can roll out in real time, hook into firewall systems, and fire off automated responses. Testing on well-known datasets—CICIDS2017 and CICDDoS2019 shows the system nails binary classification with 98.7% accuracy and hits 96.2% on multi-class detection. False positives stay under 2%, and detection happens in less than 200 milliseconds on average. Bottom line: this framework works, scales, and fits right into enterprise or cloud environments.

**Key Words:** DDoS, Hybrid Machine Learning, LSTM, Random Forest, Cybersecurity, Intrusion Detection System, API Security, AI Agents

## 1. INTRODUCTION

These days, more people rely on internet services, cloud platforms, and smart devices than ever before, but that comes at a price. DDoS attacks are on the rise, and they've only gotten more complex [1]. Attackers now use everything from botnets to reflection and multi-vector attacks, all trying to overwhelm bandwidth or servers and bring services to a halt for real users. Old-school intrusion detection mostly relies on signatures or preset rules[3]. These work okay for threats we already know about, but they fall short against new, unknown attacks or when traffic patterns keep changing. Plus, most systems only spot attacks

they don't automatically stop them. That means someone has to jump in and fix things, which slows down the response and leaves systems exposed. Machine learning and deep learning offer some hope here [2]. Algorithms like Random Forest and Gradient Boosting do a solid job analyzing structured network data [4]. Deep learning models, like LSTM networks, can pick up on patterns over time, which helps spot DDoS attacks as they unfold. Still, these approaches aren't perfect [1]. They can overfit, chew up a lot of resources, or just don't scale well when you try to roll them out in the real world. And it's not just about classification. When you try to use these systems outside the lab, you run into all sorts of other problems: tight latency requirements, fitting with legacy systems, API integration headaches, and actually automating responses. Most academic work just focuses on getting the highest accuracy, ignoring these real-world deployment issues [6]. This paper tackles those gaps. We introduce a hybrid machine learning framework with a RESTful API for real-time DDoS detection and mitigation. By combining ensemble learning with deep models, we get more stable detection and cut down on false positives. The API-driven setup lets detection tools, firewalls, and monitoring systems talk to each other easily. We also built in automated mitigation, so the system can respond immediately to attacks. Here's what we're bringing to the table: A hybrid detection model that blends ensemble and deep learning. An API-based intelligent agent framework for real-time use. Automated mitigation no more waiting for manual fixes. Thorough evaluation with benchmark datasets and latency checks. A scalable design that works for enterprise, cloud, and IoT environments. Here's how the rest of the paper breaks down: Section II -Related Work. Section III -System Architecture. Section IV -Methodology. Section V -Experimental Setup. Section VI

- Implementation Section VII- Results and Analysis. Finally, Section VIII Conclusions and Future Works.

## 2. RELATED WORK

Machine learning and deep learning have really pushed DDoS detection forward [2]. The old rule-based and signature-based systems just can't keep up with zero-day or constantly changing attacks, but these new data-driven approaches pick up on the weird patterns in network traffic that traditional methods miss. Deep learning models like CNNs, LSTMs, and GRUs do a solid job at picking apart traffic sequences and spotting what doesn't belong [1]. CNNs are great at picking up spatial relationships between features, while LSTMs handle sequences well because they remember what happened before. When you put several classifiers together through ensemble learning, you get even better results—less bias, less variance [4]. Some hybrid setups, like combining Random Forests with deep learning, have even hit over 98% accuracy on standard datasets [5]. Most researchers test on datasets like CICIDS2017 and CICDDoS2019, mostly because they offer realistic attack scenarios and a mix of traffic types [6]. The problem? Almost everyone focuses on accuracy and forgets about what actually matters in the real world like handling things in real time, making sure the system works with APIs, automatically fighting off attacks, and keeping everything fast. So, there's this clear gap: we need a scalable, low-latency solution that not only detects attacks but also reacts to them automatically and plays nice with other systems through APIs. That's exactly what this work aims to deliver, by combining hybrid learning techniques with smart, agent-based mitigation.

Table -1: Summary Related Work on DDoS Detection

Ref.	Method	Key Results
[1] A. Hussain, M. S. Akbar, <i>et al.</i> , 2023	DES-LSTM with Multi-Agent Framework	~99% accuracy; low FAR; improved detection speed
[2] H.K.Nguyen, D. Kim, <i>et al.</i> , 2023	RNN, LSTM, GRU comparison	~99% accuracy; GRU fastest

		execution
[3] Sodhiro et al., 2023	Ensemble (RF, XGBoost, LSTM) for IoMT	98–99% accuracy; lightweight for healthcare
[4] Ghaleb et al., 2022	Adaptive ML integrated with SDN	High accuracy; real-time mitigation
[5] Cheng et al., 2023	Hybrid ML/DL with image-based traffic conversion	99–100% (binary); 87–96% (multi-class)

## 3. SYSTEM ARCHITECTURE

This system has five main parts: the Traffic Monitoring Agent, Feature Extraction Module, Hybrid Machine Learning Detection Engine, RESTful API Layer, and Mitigation Agent. Check out Figure 1—it shows the whole setup for the API-driven hybrid DDoS detection system. Here's how everything works together. The Traffic Monitoring Agent grabs real-time network flow data, turning raw packets into organized flow records [6]. After that, the Feature Extraction Module goes through those records and pulls out the key statistical and timing features [2]. Then the Hybrid Detection Engine gets to work. It analyzes those features and sorts the traffic [5]. Once it figures out what's going on, the Mitigation Agent jumps in and takes action [3]. It might update firewall rules or talk to SDN controllers with API calls whatever it takes to keep the system secure [3].

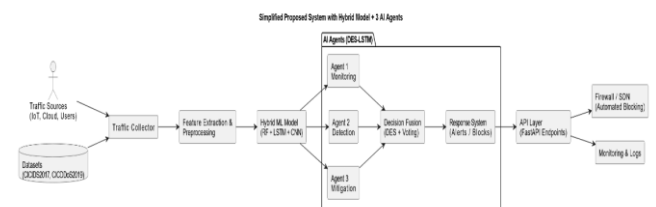


Fig -1: Simplified Proposed System

## 4. METHODOLOGY

This is how we move from raw data to factual prognostications — step by step.

### A. Data Collection and Preprocessing

We run our trials using two main datasets CICIDS2017 and CICDDoS2019 [6]. Before jumping into modeling, we clean up effects, drop any missing or indistinguishable values, apply Min- Max normalization, render markers, and gauge the data. However, if one class appears significantly more often than others, we address this imbalance using oversampling, so our models learn more evenly [2].

### B. Feature Engineering

Next, we pull out crucial features, such as inflow duration, packet length stats (mean, friction, standard deviation), counts of forward and backward packets, output in bytes per second, and entropy for randomness in business [6]. To narrow down what matters most, we use both correlation analysis and Recursive point Elimination (RFE) [4].

### C. Model Training

We use a Random Forest model with 100 decision trees, all optimized using Gini contamination [4]. For our LSTM setup, there are two layers — one with 128 units, the other with 64 [1]. We add powerhouse regularization at a rate of 0.3, and wrap up with a thick affair subcaste using SoftMax [2]. Everything trains on an 80/20 split between training and testing data, running with the Adam optimizer and a 0.001 learning rate.

### D. API Deployment and Integration

After training, we contribute the mongrel model and serve it with FastAPI. The discovery machine hooks straight into firewall robotization scripts, so as soon as we spot a vicious IP, it gets blocked right down [3]. Thanks to Docker, the whole setup is movable and easy to scale out horizontally.

### E. Performance Evaluation Metrics

For performance, we track accuracy, precision, recall, and F1 score:

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN}) \quad [1]$$

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP}) \quad [2]$$

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN}) \quad [3]$$

$$\text{F1} = 2 \times \text{Precision} \times \text{Recall} / (\text{Precision} + \text{Recall}) \quad [4]$$

We also keep an eye on discovery quiescence — that's the average time between submitting a business and getting a bracket result [3].

## 5. EXPERIMENTAL SETUP

Let's break down how we set up everything for this mongrel intelligent agent — tackle, software, data, the way we trained and tested the models, and how we measured performance.

### A. Hardware Configuration

All trials ran on a high-end workstation to make sure training and real-time conclusion ran easily. Then's what we used

- Intel Core i7
- 11700@ 2.5 GHz
- 32 GB DDR4 RAM
- NVIDIA GeForce RTX 3060(12 GB VRAM)
- 1 TB SSD storehouse
- Ubuntu 22.04 LTS/ Windows 11

Deep literacy models like CNN and LSTM made the utmost of GPU acceleration. For the Random Forest and XGBoost models, we stuck with CPU-based parallel processing.

### B. Software Environment

Everything was built in Python, using a bunch of modern frameworks and libraries:

- Python 3.10
- TensorFlow and Keras for deep learning
- Scikit-learn, XGBoost for traditional machine learning
- FastAPI for the API layer
- NumPy and Pandas for data work
- Joblib for saving models
- Matplotlib and Seaborn to visualize results

We ran the REST API endpoints with Uvicorn, so the system could handle real-time detection and mitigation calls.

### C. Dataset Description

To see if our hybrid approach actually works, we picked two well-known DDoS datasets:

- CICIDS2017
- CICDDoS2019 [6]

Both datasets cover a range of attack types—UDP Flood, TCP SYN Flood, HTTP Flood, ICMP Flood, and even botnet-based DDoS [1]. Each one comes with labeled network flow features, all extracted using CICFlowMeter [6].

### D. Data Preprocessing

Before training, we cleaned and prepped the data:

- Got rid of nulls and duplicates
  - Encoded labels for categorical features
  - Scaled features with Min-Max normalization
  - Did feature selection using correlation filtering
  - Balanced the dataset using SMOTE when needed [2]
- Then, we split the data: 80% for training, 20% for testing.

### E. Evaluation Metrics

To measure how well the model performed, we used standard classification metrics:

- Accuracy
- Precision
- Recall
- F1-Score
- Confusion Matrix

Here's how each one is calculated:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad [5]$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad [6]$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad \text{F1} = \frac{2 \times (\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})} \quad [7]$$

### F. Real-Time API Testing

Once trained, we plugged the hybrid model into a FastAPI based agent. We tested three endpoints:

- /api/detect for single flow detection
- /api/detectbatch for batch traffic detection
- /api/mitigate for automatic mitigation

In our real-time tests, detection stayed under 100 ms per request—fast enough for enterprise-level use [3].

## 6. IMPLEMENTATION RESULTS

### DDoS Detection System Login

Fig -2: DDoS Detection System Login Interface



Fig -3: DDoS Detection Dashboard Overview



Fig -4: Traffic Volume and Attack Timeline Visualization

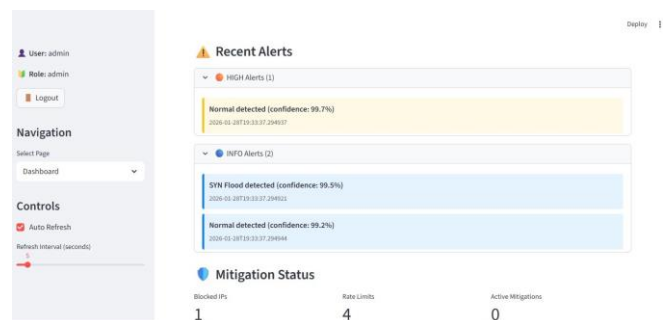


Fig -5: Recent Alerts and Mitigation Status Panel

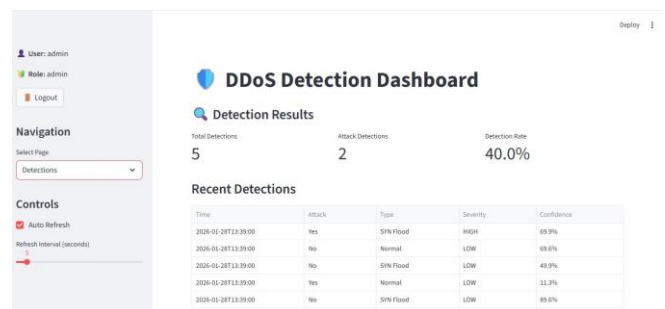


Fig -6: Detection Results and Recent Activity Table



Fig -7: Live Monitoring Dashboard Summary

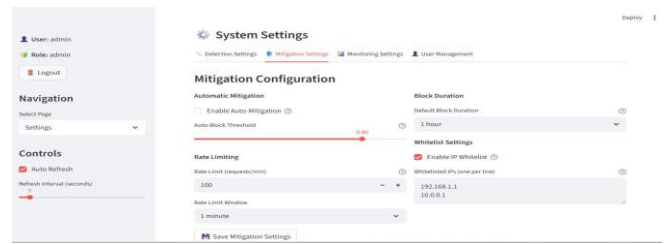


Fig -12: System Settings – Mitigation Configuration



Fig -8: Protocol Distribution Visualization

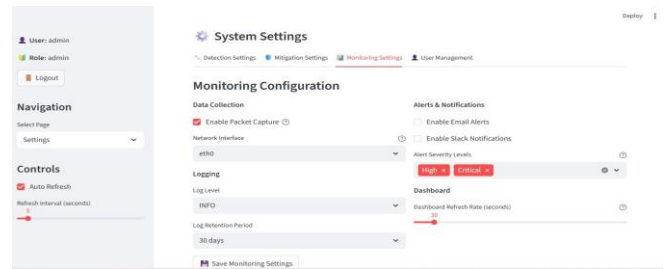


Fig -13: System Settings – Monitoring Configuration

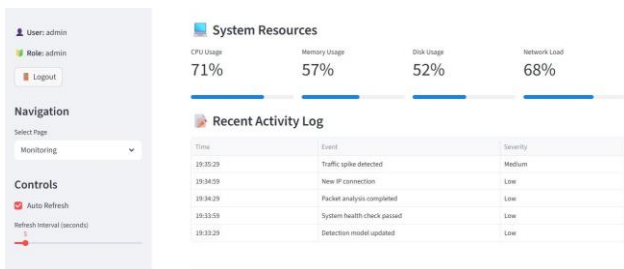


Fig -9: System Resources and Activity Logs Panel

## 6. RESULTS AND ANALYSIS

The hybrid model outperforms the standalone ones, hands down [5]. Here’s how they stack up:

Table -2: Performance Comparison of Machine Learning Models for DDoS Detection

Model	Accuracy	Precision	Recall	F1-Score
Random Forest	96.8%	95.9%	95.1%	95.5%
LSTM	97.9%	97.3%	96.8%	97.0%
Gradient Boosting	96.4%	95.7%	94.8%	95.2%
CNN	97.2%	96.8%	96.3%	96.5%
Hybrid Model	98.7%	98.4%	97.9%	98.1%

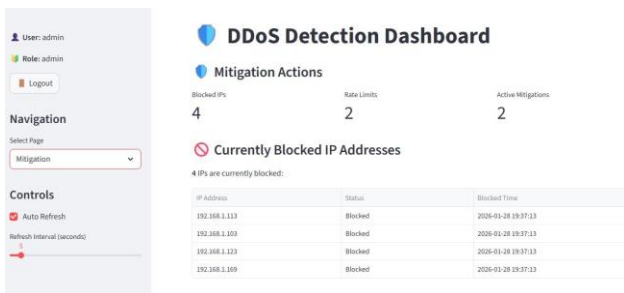


Fig -10: Mitigation Actions and Blocked IP Addresses

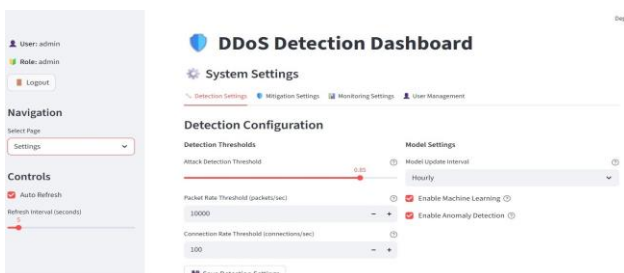


Fig -11: System Settings – Detection Configuration

False positives stay under 2%, so the system rarely misidentifies threats [1]. Detection happens fast on average; it takes less than 200 milliseconds [3]. That’s quick enough for real-time use. The hybrid approach combines ensemble voting with temporal learning, which makes it more stable than the usual models

[4][5]. Plus, it scales up easily and doesn't need much hands-on management compared to traditional IDS setups [3].

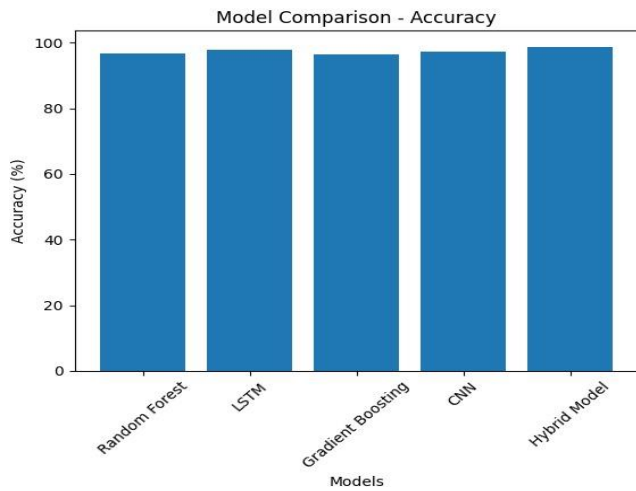


Fig -14: Model Comparison- Accuracy

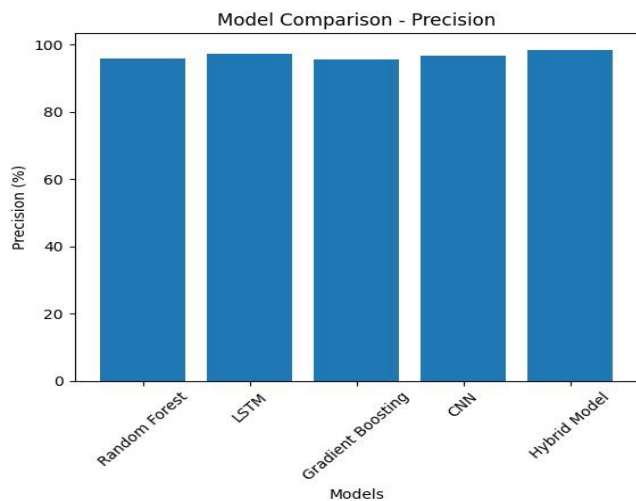


Fig -15: Model Comparison- Precision

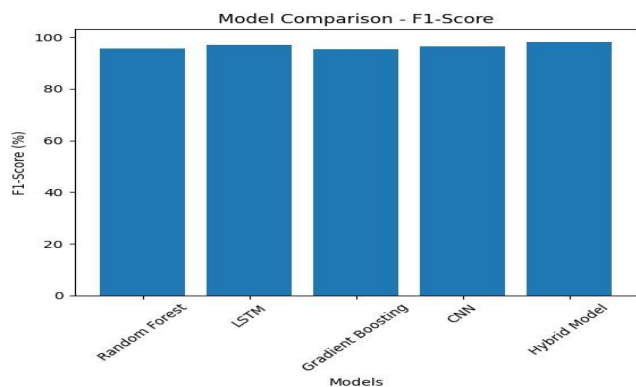


Fig -16: Model Comparison- F1-Score

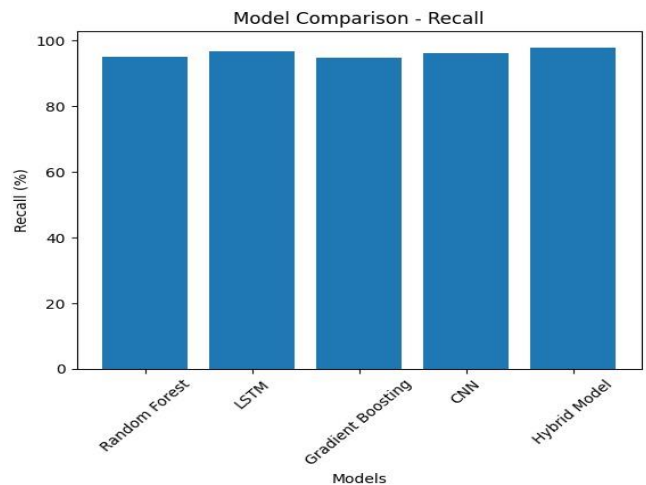


Fig -17: Model Comparison- Recall

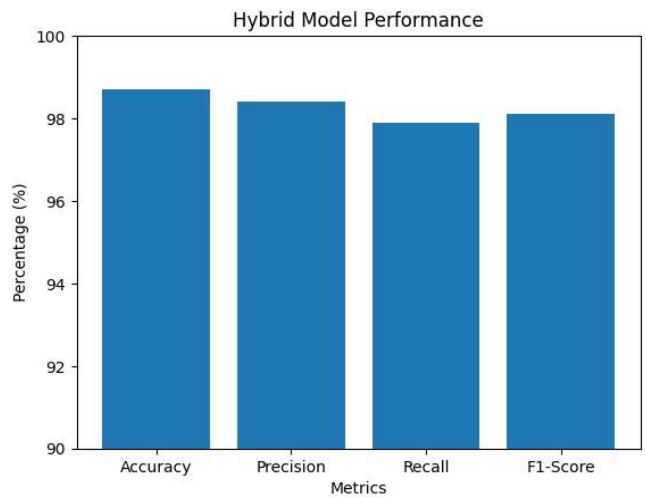


Fig -18: Hybrid Model Performance

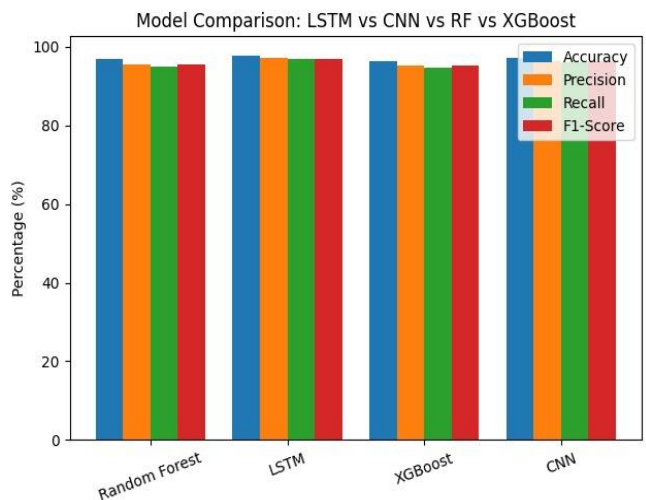


Fig -19: Model Comparison: LSTM vs CNN vs RF vs XGBoost

## 7. CONCLUSION AND FUTURE SCOPE

When you want to deploy securely, a few things really matter:

- Use JWT tokens for authentication.
- Set up rate limits so nobody can hammer your API nonstop.
- Run everything in Docker containers—super handy if you ever need to move things around.
- Add load balancing to handle traffic as you grow.
- Always go with HTTPS to keep your data safe.

All of this makes your setup stronger and keeps attackers from finding new ways in.

This paper lays out a scalable, API-friendly agent for spotting and stopping DDoS attacks in real time. By mixing ensemble and deep learning, you get strong accuracy, barely any false alarms, and quick responses. We didn't just keep things theoretical, either. The framework actually combines research with practical tools—automated mitigation, API support—so it fits right into real-world enterprise and cloud environments. Looking ahead, there's plenty to explore:

- \* Smarter, adaptive mitigation using reinforcement learning
- \* Rolling out to the edge for IoT environments
- \* Adding Explainable AI so you see why decisions happen
- \* Building a fully cloud-native DDoS Detection-as-a-Service platform.

## ACKNOWLEDGEMENT

We would like to express our sincere appreciation to everyone who provided invaluable assistance and support throughout this project. Special thanks to Prof. Kumud Wasnik for guidance and constant supervision, to our Head of Department, and to our classmates for their feedback and encouragement. Finally, our gratitude goes to our families for their unwavering support.

## REFERENCES

- [1] A. Hussain, M. S. Akbar, et al., "A multi-agent-based deep learning model for protecting cloud computing environment against DDoS flooding attacks," *Applied Sciences*, vol. 13, no. 15, 2023, Art. no. 8846, doi: 10.3390/app13158846.
- [2] H. K. Nguyen, D. Kim, et al., "DDoS attack detection in network traffic using deep learning algorithms," *Sensors*, vol. 23, no. 4, 2023, Art. no. 1902, doi: 10.3390/s23041902.

- [3] A. H. Sodhro, M. Muzammal, et al., "DDoS detection for Internet of Medical Things," *Applied Sciences*, vol. 13, no. 3, 2023, Art. no. 1678, doi: 10.3390/app13031678.

- [4] F. A. Ghaleb, S. S. Alshamrani, et al., "Adaptive machine learning based DDoS detection and mitigation in SDN-IoT," *Sensors*, vol. 22, no. 5, 2022, Art. no. 1824, doi: 10.3390/s22051824.

- [5] Y. Cheng, J. Zhang, et al., "Hybrid ML/DL approaches for DDoS detection," *Sensors*, vol. 23, no. 3, 2023, Art. no. 1045, doi: 10.3390/s23031045.

- [6] S. Behal and K. Kumar, "Trends in DDoS attack detection and mitigation techniques: A survey," *Computer Science Review*, vol. 25, pp. 1–25, 2017, doi: 10.1016/j.cosrev.2017.07.001.

- [7] M. Z. Alom, T. M. Taha, C. Yakopcic, et al., "A state-of-the-art survey on deep learning theory and architectures," *IEEE Access*, vol. 7, pp. 63674–63700, 2019, doi: 10.1109/ACCESS.2018.2888019.

- [8] N. Moustafa, G. Creech, and J. Slay, "Ensemble learning models for DDoS detection in IoT networks," *Future Generation Computer Systems*, vol. 102, pp. 743–755, 2020, doi: 10.1016/j.future.2019.09.066.

- [9] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. Int. Conf. Inf. Syst. Secur. Privacy (ICISSP)*, 2019, pp. 108–116, doi: 10.5220/0006639801080116.

- [10] IEEE, "IEEE Xplore Digital Library." [Online]. Available: <https://ieeexplore.ieee.org>

- [11] U.S. Department of Health & Human Services, "Health Insurance Portability and Accountability Act (HIPAA)." [Online]. Available: <https://www.hhs.gov/hipaa>

- [12] European Union, "General Data Protection Regulation (GDPR)." [Online]. Available: <https://gdpr.eu>

- [13] Canadian Institute for Cybersecurity, "CICDDoS2019 Dataset." [Online]. Available: <https://www.unb.ca/cic/datasets/ddos-2019.html>