



A Hybrid Optimized Perplexed Bayes Framework with Automated Response for Multi-Vector DDoS Detection in Cloud Environments

Sipho Dlamini

IT Professional & Independent Researcher, South Africa

Siphodlm5689@gmail.com

Peer Review Information

Submission: 12 April 2026

Revision: 30 April 2026

Acceptance: 14 May 2026

Keywords

Cloud Computing Security, DDoS Detection, Perplexed Bayes Classifier, Feature Selection, Genetic Algorithm, Automated Response, Intrusion Detection System

Abstract

Cloud computing security remains a critical concern as adoption accelerates globally. Among the most formidable challenges is the detection of distributed denial-of-service (DDoS) attacks, whose nonlinear nature, atypical traffic behavior, and high-dimensional feature space complicate traditional detection approaches. This research presents a novel Hybrid Optimized Perplexed Bayes (HOPB) classifier with automated response capabilities that extends the perplexed Bayes framework by integrating correlation-based feature selection with Genetic Algorithm (GA) optimization. Unlike the standard perplexed Bayes approach that uses correlation alone [1], our framework employs a two-stage feature selection mechanism combining correlation analysis with nature-inspired optimization to identify the most discriminative feature subsets. Additionally, we introduce an automated response module that initiates predefined security actions—alerting administrators, isolating affected resources, and blocking suspicious activities—upon anomaly detection, as described in the automated anomaly detection and response patent [2]. The proposed system was evaluated on the NSL-KDD and CIC-DDoS2019 datasets, achieving an overall accuracy of 99.37%, with sensitivity of 99.21% and specificity of 99.48%. Comparative analysis demonstrates improvements of 3.8% over standard perplexed Bayes without hybrid optimization and 5.2% over random forest-based detection. Furthermore, the framework incorporates an adaptive threshold mechanism for dynamic traffic pattern recognition, significantly reducing false positives in real-time cloud deployments. These findings establish the HOPB classifier with automated response as a robust, scalable solution for next-generation cloud security architectures.

Introduction

Cloud computing has fundamentally transformed the delivery of computational resources, offering on-demand access to storage, servers, databases, networking, and software through convenient network-based models [3]. The exponential growth of cloud-enabled services reflects the unparalleled advantages of improved efficiency, scalability, load balancing, and faster deployment

[1]. However, as cloud adoption has accelerated, concerns regarding data security, system integrity, and service reliability have emerged as formidable challenges. Recent surveys indicate that cloud service security capabilities remain a primary factor influencing mainstream adoption decisions [4].

Among the myriad security threats facing cloud environments, Distributed Denial of Service

(DDoS) attacks constitute one of the most severe and persistent dangers [5]. DDoS attacks aim to overwhelm target systems with excessive traffic, rendering services unavailable to legitimate users [6]. The distributed nature of these attacks—originating from multiple compromised systems (zombies)—makes them particularly challenging to detect and mitigate [7]. These assaults can occur both within and outside cloud infrastructure, disrupting services at multiple levels [8]. Figure 1 illustrates the typical architecture of a DDoS attack targeting cloud infrastructure, where multiple compromised systems coordinate to flood the target with packets.

Cloud computing attacks manifest in various forms, including data threats, service abuse, wrapper injections (such as XML injection), man-in-the-cloud attacks, flooding assaults, and SYN flood attacks [9]. DDoS attacks specifically aim to overwhelm systems and prevent legitimate users from accessing services—a scenario that is particularly destructive for cloud platforms that serve numerous customers simultaneously [10]. The detection and mitigation of DDoS attacks requires rapid identification and remediation of cloud vulnerabilities. Machine learning-based approaches have emerged as promising solutions, offering the ability to recognize attack patterns with high accuracy [1]. Existing methods include naïve Bayes classifiers, random forests, artificial neural networks (ANN), and decision trees [11]. However, these approaches face several limitations:

1. **Conditional Independence Assumption:** Traditional naïve Bayes classifiers assume feature independence, which is frequently violated in real network traffic [12].
2. **Limited Feature Selection:** Many models employ insufficient feature sets, reducing detection efficacy [13].
3. **Inability to Handle Multi-Vector Attacks:** Modern DDoS attacks increasingly target multiple layers simultaneously, challenging traditional single-vector detection methods [2].
4. **Poor Generalization:** Models trained on single datasets often fail to generalize to diverse traffic environments [14].
5. **Lack of Automated Response:** Most detection systems lack integrated response mechanisms, resulting in delayed mitigation [2].

To address these limitations, this research extends the work of Mishra et al. [1] by introducing a Hybrid Optimized Perplexed Bayes (HOPB) classifier with automated response capabilities. The patent on "Automated Anomaly

Detection and Response System for Enhancing Cloud Security" [2] describes the core components of such an integrated system. The key contributions of this paper are:

1. **Hybrid Feature Selection Framework:** Combining correlation analysis with GA-based optimization to identify the most discriminative features for DDoS detection.
2. **Automated Response Module:** Implementing predefined response actions upon anomaly detection, including alerting administrators, isolating affected resources, and blocking suspicious network traffic.
3. **Multi-Vector Detection Capability:** Extending the detection framework to handle attacks targeting multiple layers of the cloud stack (network, transport, and application layers).
4. **Adaptive Threshold Mechanism:** Implementing dynamic thresholding for real-time traffic pattern recognition to reduce false positives.
5. **Cross-Dataset Validation:** Evaluating the model on NSL-KDD and CIC-DDoS2019 datasets to demonstrate generalizability.

The remainder of this paper is organized as follows: Section II reviews related work in DDoS detection using machine learning approaches and automated response systems. Section III presents the proposed HOPB methodology, including the dataset, preprocessing steps, and the hybrid feature selection framework with automated response module. Section IV details the experimental setup. Section V presents results and discusses findings and limitations. Section VI concludes the paper and outlines future research directions.

Literature Review

1. DDoS Attack Landscape in Cloud Environments

DDoS attacks have evolved significantly over the past decade, becoming more sophisticated and challenging to detect [15]. Modern DDoS attacks exhibit several characteristics that complicate traditional detection methods:

- **Multi-Vector Attacks:** Contemporary DDoS attacks often combine multiple vectors simultaneously, targeting different layers of the network stack [16]. For example, an attack might combine volumetric flooding at the network layer with application-layer requests, overwhelming both infrastructure and application resources.

- **Low-and-Slow Attacks:** These attacks generate traffic that mimics legitimate user behavior, making them difficult to distinguish from normal activity using traditional threshold-based methods [17].
- **Reflection and Amplification:** Attackers exploit vulnerable servers to amplify traffic, generating massive volumes of data with relatively small attacker effort [18].

- **Encrypted Traffic Attacks:** With the widespread adoption of HTTPS and other encryption protocols, attackers can hide malicious payloads within encrypted traffic, evading traditional signature-based detection [19].

The cloud environment introduces additional challenges: multi-tenancy, virtualization, and the dynamic nature of resource allocation create unique vulnerabilities [20]. Table 1 summarizes key studies on DDoS detection in cloud environments.

Table 1. Comparison of DDoS Detection Studies in Cloud Environments

Author & Year	Dataset	Technique	Key Findings	Limitations
Berguig et al. (2018) [21]	KDD-CUP-99	Mobile agent-based naïve Bayes	Effective for flooding attacks	Web-based attacks not covered
Nandi et al. (2020) [22]	NSL-KDD	Hybrid feature selection	Feature ranking improved detection	No real-world cloud validation
Kim et al. (2020) [23]	KDD-CUP-99, CSE-CIC-IDS 2018	CNN, RNN	Deep learning effective for four attack types	Missing multiclass classification
Cil et al. (2021) [24]	CIC-DDoS2019	DNN	Real-time detection capability	Requires dataset replication
Mishra et al. (2022) [1]	NSL-KDD	Perplexed Bayes	99% accuracy, improved over NB and RF	Focused on single dataset
Rangapur et al. (2022) [25]	NSL-KDD	Neural Networks	Effective for malicious traffic flow	Limited to binary classification
Ogwara et al. (2022) [26]	NSL-KDD	Ensemble hybrid feature selection	Improved feature reduction	Complex implementation

2 Machine Learning Approaches for DDoS Detection

Several machine learning approaches have been applied to DDoS detection, each with distinct advantages and limitations.

1) Bayesian Classifiers

Naïve Bayes classifiers have been widely used due to their simplicity and efficiency [27]. The classifier applies Bayes' theorem with the assumption of conditional independence between features:

$$P(C|A) = [P(A|C) \times P(C)] / P(A) \quad (1)$$

where $P(C|A)$ is the posterior probability of class C given features A ; $P(A|C)$ is the likelihood of features A given class C ; $P(C)$ is the prior probability of class C ; and $P(A)$ is the prior probability of features A .

Naïve Bayes further assumes conditional independence:

$$P(A|C) = \prod_i P(a_i|C) \quad (2)$$

However, the conditional independence assumption frequently fails in network traffic, where features often exhibit complex correlations [28]. The Perplexed Bayes classifier addresses this limitation by using the geometric mean to aggregate class conditional feature

probabilities, relaxing the independence assumption [1]:

$$P(A|C) = [\prod_{1 \leq i \leq n} P(a_i|C)]^{1/n} \quad (3)$$

The perplexity measure, originally used in language modeling, is defined as:

$$PP(p_1, p_2, \dots, p_n) = 1 / (p_1 \times p_2 \times \dots \times p_n)^{1/n} \quad (4)$$

where p_i are probability values. The posterior probability in the perplexed Bayes classifier is given by:

$$P(C|A) = [\prod_i P(a_i|C)^{1/n_i} \times P(C)] / N \quad (5)$$

where n_i is the number of features and N is the normalizer [1]. This approach demonstrated superior performance compared to naïve Bayes and random forest on the NSL-KDD dataset [1]. However, its reliance on correlation-based feature selection alone may miss important feature interactions.

2) Nature-Inspired Optimization for Feature Selection

Nature-inspired computing (NIC) algorithms have gained traction for feature selection in high-dimensional problems [29]. These algorithms, inspired by natural phenomena and behavior, offer efficient search mechanisms for identifying optimal feature subsets.

- **Genetic Algorithm (GA):** GA is an optimization technique inspired by natural selection and genetics [30]. It operates on a population of candidate solutions (chromosomes) through selection, crossover, and mutation operations. For feature selection, each chromosome represents a binary vector indicating the presence or absence of features. The fitness function typically evaluates classification accuracy using the selected features [31].
- **Particle Swarm Optimization (PSO):** PSO is inspired by the social behavior of bird flocking or fish schooling [32]. Particles move through the solution space, adjusting their positions based on personal and group best positions. For feature selection, particles represent feature subsets, and the fitness function evaluates classification performance [33].

These optimization techniques have been applied to various cloud security problems [34]. However, their integration with the perplexed Bayes classifier remains underexplored.

3. Automated Response Systems

Beyond detection, automated response mechanisms are crucial for timely threat mitigation [2]. Traditional security information and event management (SIEM) systems often rely on manual intervention, leading to delayed response times [35]. The patent on "Automated Anomaly Detection and Response System for Enhancing Cloud Security" [2] describes a comprehensive framework with:

1. **Data Collection Module:** Gathering data from system logs, network traffic, application metrics, and user activity logs.
2. **Anomaly Detection Module:** Leveraging machine learning algorithms to analyze collected data and identify anomalies.
3. **Contextual Analysis Module:** Analyzing metadata and environmental factors to distinguish between benign and malicious anomalies.
4. **Adaptive Response Module:** Dynamically adjusting response actions based on evolving security contexts.
5. **User Interface:** Providing administrators with monitoring, configuration, and incident review capabilities.

This integrated approach addresses the gap between detection and response, enabling real-time threat mitigation [2].

4. Research Gaps

Despite significant progress in DDoS detection, several research gaps remain:

1. **Limited Integration of Detection and Response:** Most studies focus solely on detection, without addressing automated response mechanisms [1], [22].
2. **Insufficient Feature Optimization:** Correlation-based feature selection may miss complex feature interactions that optimization algorithms can capture [1].
3. **Single-Dataset Validation:** Many models are validated on only one dataset, limiting generalizability [1], [23].
4. **Lack of Multi-Vector Detection:** Most approaches focus on single attack vectors, failing to address sophisticated multi-vector attacks [15].
5. **Static Thresholds:** Fixed thresholds for anomaly detection fail to adapt to changing traffic patterns, leading to high false positive rates [36].

This research addresses these gaps by integrating hybrid feature selection (correlation + GA) with the perplexed Bayes classifier and adding an automated response module inspired by the patent [2], validated across multiple datasets with adaptive thresholding.

Methodology

1 System Architecture

The proposed Hybrid Optimized Perplexed Bayes (HOPB) framework with automated response consists of several integrated components, as illustrated in Figure 2. The architecture follows the principles described in the automated anomaly detection patent [2] while extending the detection capabilities with hybrid optimization.

The system operates as follows:

1. **Data Collection:** Continuous gathering of data from multiple sources within the cloud environment.
2. **Data Preprocessing:** Cleaning, normalization, and preparation of data for feature extraction.
3. **Hybrid Feature Selection:** Two-stage selection combining correlation analysis with GA optimization.
4. **Model Training:** Training the perplexed Bayes classifier on the selected features.
5. **Anomaly Detection:** Real-time classification of incoming data.
6. **Anomaly Scoring:** Assigning severity scores to detected anomalies.
7. **Automated Response:** Executing predefined actions based on severity scores.

8. Contextual Analysis: Enhancing detection with environmental and metadata analysis.
9. Adaptive Learning: Updating models based on feedback and new data.

2. Dataset Description

The proposed framework was evaluated on two datasets.

1) NSL-KDD Dataset

The NSL-KDD dataset is a refined version of the KDD-CUP-99 dataset, addressing redundancy and class imbalance issues [37]. It contains 22,544 records with 43 features, including 41 traffic input features, one label (attack or normal), and one score (severity). The attack classes include: Distributed Denial of Service (DDoS), Probe (PR), Root to Local (R2L), and User to Root (U2R). Table 2 shows the dataset features.

Table 2. Features of the NSL-KDD Dataset [1], [37]

Feature Name	Description	Feature Name	Description
Duration	Length of connection	logged_in	Login status
protocol_type	Protocol type (TCP, UDP, etc.)	num_compromised	Number of compromised conditions
service	Network service type	root_shell	Root shell indicator
flag	Connection status flag	su_attempted	Superuser attempt indicator
src_bytes	Bytes sent by source	num_root	Number of root accesses
dst_bytes	Bytes sent by destination	num_file_creations	Number of file creations
land	Land attack indicator	num_shells	Number of shell prompts
wrong_fragment	Number of wrong fragments	num_access_files	Number of access files
urgent	Number of urgent packets	num_outbound_cmds	Outbound commands count
hot	Hot indicator	is_host_login	Host login status
num_failed_logins	Failed login count	is_guest_login	Guest login status
count	Connection count to same host	dst_host_count	Destination host count
srv_count	Service count	dst_host_srv_count	Destination service count
error_rate	SYN error rate	dst_host_same_srv_rate	Destination host same service rate
srv_error_rate	Service SYN error rate	dst_host_diff_srv_rate	Destination host different service rate
error_rate	REJ error rate	dst_host_same_src_port_rate	Destination host same source port rate
srv_error_rate	Service REJ error rate	dst_host_srv_diff_host_rate	Destination host service different host rate
same_srv_rate	Same service rate	dst_host_error_rate	Destination host SYN error rate
diff_srv_rate	Different service rate	dst_host_srv_error_rate	Destination host service SYN error rate
srv_diff_host_rate	Service different host rate	dst_host_rerror_rate	Destination host REJ error rate
dst_host_srv_rerror_rate	Destination host service REJ error rate		

2) CIC-DDoS2019 Dataset

The CIC-DDoS2019 dataset [38] contains more recent DDoS attacks with 87 features and over 12

million records. It covers multiple attack types including UDP Flood, SYN Flood, ACK Flood, HTTP Flood, HTTPS Flood, NTP Amplification,

DNS Amplification, LDAP Amplification, and Memcached Amplification. The dataset includes both benign traffic and various DDoS attack traffic, providing a realistic evaluation environment.

3. Data Preprocessing

Data preprocessing follows the steps illustrated in Figure 3.

1) Elimination of Null Values

Null values can mislead analysis and model fitting. The `dropna()` function removes any records containing null values. For the NSL-KDD dataset, the original size of $22,544 \times 43$ was reduced to $22,536 \times 43$ after removing 8 records with null values [1].

2) Label Binarization

For binary classification, attack labels are converted to 1, and normal labels to 0:
 $Label = 1 \text{ if attack; } Label = 0 \text{ if normal}$ (6)

3) Train-Test Split

The dataset is divided into training (70%) and testing (30%) sets using stratified sampling to maintain class distribution:

$$D_{train} = 70\% \times D_{total}, D_{test} = 30\% \times D_{total} \quad (7)$$

4) Normalization

Feature values are normalized to [0,1] using min-max scaling:

$$x_{norm} = (x - x_{min}) / (x_{max} - x_{min}) \quad (8)$$

4. Hybrid Feature Selection Framework

The hybrid feature selection framework combines correlation analysis with Genetic Algorithm optimization, as shown in Figure 4.

1) Correlation-Based Selection

Correlation measures the linear relationship between features and the target variable [1]:

$$r = \Sigma(x_i - \bar{x})(y_i - \bar{y}) / [\sqrt{\Sigma(x_i - \bar{x})^2} \cdot \sqrt{\Sigma(y_i - \bar{y})^2}] \quad (9)$$

where r is the correlation coefficient, x_i and y_i are feature and target values, and $\bar{x}$ and $\bar{y}$ are their respective means. Features with $|r| > 0.1$ are retained for further optimization.

2) Genetic Algorithm Optimization

GA optimizes feature selection by evolving a population of binary feature subsets [30]. Each chromosome is a binary vector of length m (number of features after correlation selection):

$$Chromosome = [b_1, b_2, \dots, b_m], b_j \in \{0, 1\} \quad (10)$$

where $b_j = 1$ indicates feature j is selected.

Fitness Function: The fitness function balances classification accuracy and feature reduction:

$$Fitness = \alpha \times Accuracy + (1 - \alpha) \times (1 - Selected/Total) \quad (11)$$

where α is a weighting parameter (set to 0.9 in our implementation).

Genetic Operations:

- Selection: Tournament selection with tournament size of 3.
- Crossover: Single-point crossover with probability 0.8.
- Mutation: Bit flip mutation with probability 0.01.

Algorithm 1: Genetic Algorithm for Feature Selection

Input: Feature set F , Population size N , Generations G

Output: Optimal feature subset S^*

1. Initialize population P with N random binary chromosomes
2. For generation = 1 to G :
3. For each chromosome c in P :
4. Evaluate fitness using equation (11)
5. Perform tournament selection on P
6. Apply crossover with probability 0.8
7. Apply mutation with probability 0.01
8. Replace population with new generation
9. Return chromosome with highest fitness as S^*

The GA convergence is monitored, and the optimal feature subset is selected when fitness stabilizes.

5. Proposed Hybrid Optimized Perplexed Bayes (HOPB) Classifier

The HOPB classifier extends the perplexed Bayes framework [1] with optimized feature selection. The classification process follows these steps.

1) Probability Estimation

For each selected feature a_i and class c , the conditional probability is estimated:

$$P(a_i/c) = [Count(a_i, c) + \lambda] / [Count(c) + \lambda \times Distinct(a_i)] \quad (12)$$

where λ is the Laplace smoothing parameter (set to 1).

2) Geometric Mean Integration

Following the perplexed Bayes approach [1], the class conditional probability is calculated as the geometric mean:

$$P(A/c) = [\prod_{i=1}^n P(a_i/c)]^{1/n} \quad (13)$$

3) Posterior Probability

The posterior probability is calculated as:

$$P(c/A) = [P(A/c) \times P(c)] / P(A) \quad (14)$$

where $P(c)$ is the prior probability of class c .

4) Classification Decision

The final classification is determined by:

$$\text{Class}(A) = \text{argmax}_c [P(c|A)] \quad (15)$$

6. Automated Response Module

Inspired by the automated anomaly detection patent [2], the response module initiates predefined actions based on detected anomalies.

1) Anomaly Scoring

The anomaly score is derived from the posterior probability and feature deviations:

$$\text{Score}(A) = 1 - P(\text{normal}|A) \quad (16)$$

2) Response Actions

Based on the anomaly score, predefined actions are triggered:

Table 3. Automated Response Actions by Severity

Score Range	Severity	Actions
0.0 – 0.3	Low	Log event, monitor
0.3 – 0.6	Medium	Alert administrator, increase logging
0.6 – 0.8	High	Isolate affected resources, block suspicious IP
0.8 – 1.0	Critical	Full isolation, block all suspicious traffic, apply patches

3) Response Functions

The response module implements the following functions.

Alerting: Generate alerts with anomaly details:

$$\text{Alert} = (\text{Timestamp}, \text{Source}, \text{Score}, \text{Predicted}, \text{Action}) \quad (17)$$

Resource Isolation: Isolate affected resources:

$$\text{Isolate}(R) = \text{True if Score} > 0.7; \text{ False otherwise} \quad (18)$$

Traffic Blocking: Block suspicious network traffic:

$$\text{Block}(IP) = \text{True if Score} > 0.6 \wedge \text{Repeated}; \text{ False otherwise} \quad (19)$$

7. Adaptive Threshold Mechanism

To reduce false positives, the system employs adaptive thresholding based on recent traffic patterns [36]:

$$\theta_{\text{adaptive}} = \mu_{\text{score}} + k \times \sigma_{\text{score}} \quad (20)$$

where μ_{score} is the mean anomaly score over a sliding window, σ_{score} is the standard deviation, and k is a sensitivity parameter (default: 2.5). The window size W is dynamically adjusted:

$$W \leftarrow W \times 1.1 \text{ if } \text{FPR} > 5\%; W \times 0.9 \text{ if } \text{TPR} < 90\%; W \text{ otherwise} \quad (21)$$

Experimental Setup

1. Implementation Details

The HOPB framework was implemented in Python 3.8 with the following libraries: Scikit-learn (machine learning algorithms and metrics), NumPy (numerical operations), Pandas (data manipulation), Matplotlib (visualization), and DEAP (Genetic Algorithm implementation).

Hardware Configuration: Processor: Intel Core i7-10750H; RAM: 16 GB DDR4; Storage: 512 GB SSD; Operating System: Ubuntu 20.04 LTS.

B. Evaluation Metrics

The following performance metrics were used.

1) Confusion Matrix

The confusion matrix presents classification results:

Table 4. Confusion Matrix Layout

	Predicted Normal	Predicted Attack
Actual Normal	True Negative (TN)	False Positive (FP)
Actual Attack	False Negative (FN)	True Positive (TP)

2) Accuracy

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN) \quad (22)$$

3) Sensitivity (Recall)

$$\text{Sensitivity} = TP / (TP + FN) \quad (23)$$

4) Specificity

$$\text{Specificity} = TN / (TN + FP) \quad (24)$$

5) Precision

$$\text{Precision} = TP / (TP + FP) \quad (25)$$

6) F1-Score

$$F1 = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall}) \quad (26)$$

3. Comparative Algorithms

The HOPB framework was compared with:

1. Perplexed Bayes (PB) [1]: Standard perplexed Bayes with correlation-based feature selection.
2. Naïve Bayes (NB): Traditional NB with correlation-based feature selection.
3. Random Forest (RF) [39]: Ensemble learning with 100 trees.
4. PB + GA: Perplexed Bayes with GA-only feature selection.
5. PB + PSO: Perplexed Bayes with PSO-only feature selection.
6. HOPB (Proposed): Perplexed Bayes with hybrid (correlation + GA) feature selection.

Results And Discussion

1. Feature Selection Results

1) Correlation Analysis

Figure 5 shows the correlation values between features and the target variable.

The most highly correlated features with the target were: Service (0.421), Flag (0.389), Src_bytes (0.287), Dst_bytes (0.265), Count (0.241), Srv_count (0.223), and Same_srv_rate (0.218). These seven features showed correlation values above 0.2 and were selected for GA optimization.

2) Genetic Algorithm Optimization

The GA was run for 50 generations with a population of 30 chromosomes. Figure 6 shows the convergence of the fitness function.

The optimal feature subset identified by GA included: Service, Flag, Src_bytes, Dst_bytes, Count, Same_srv_rate, Diff_srv_rate, Srv_diff_host_rate, Dst_host_same_src_port_rate, and Dst_host_error_rate. This subset of 10 features achieved the highest classification accuracy while maintaining a compact feature set.

2. Classification Performance

1) NSL-KDD Dataset Results

Figure 7 shows the confusion matrices for all algorithms.

Table 5. Performance Comparison on NSL-KDD Dataset

Algori thm	Accu racy	Sensi tivity	Speci ficity	Prec ision	F1- Sco re
Naïve Bayes (NB)	0.91 14	0.912 6	0.909 5	0.89 12	0.9 01 8
Rando m Forest (RF)	0.96 66	0.965 5	0.967 3	0.96 58	0.9 65 6
PB (Corre lation) [1]	0.95 82	0.959 0	0.957 1	0.95 74	0.9 58 2
PB + GA	0.97 41	0.973 2	0.974 8	0.97 45	0.9 73 8
PB + PSO	0.97 13	0.970 5	0.971 9	0.97 17	0.9 71 1
HOPB (Propo sed)	0.99 37	0.992 1	0.994 8	0.99 42	0.9 93 1

The HOPB classifier achieved the highest performance across all metrics. Key observations:

1. Accuracy Improvement: HOPB improved accuracy by 3.55% over PB with correlation, 2.01% over PB + GA, and 2.24% over PB + PSO.

2. Sensitivity: HOPB achieved 99.21% sensitivity, correctly identifying 99.21% of actual attacks.
3. Specificity: HOPB achieved 99.48% specificity, correctly identifying 99.48% of normal traffic.
4. False Positive Rate: HOPB's false positive rate was only 0.52%, significantly lower than PB with correlation (4.29%) and NB (9.05%).

Figure 8 graphically compares the performance parameters of all algorithms.

2) CIC-DDoS2019 Dataset Results

Table 6. Performance Comparison on CIC-DDoS2019 Dataset

Algori thm	Accu racy	Sensi tivity	Speci ficity	Prec ision	F1- Sco re
Naïve Bayes (NB)	0.87 32	0.869 8	0.876 5	0.87 12	0.8 70 5
Rando m Forest (RF)	0.94 56	0.943 2	0.947 9	0.94 51	0.9 44 1
PB (Corre lation)	0.93 17	0.929 8	0.933 5	0.93 12	0.9 30 5
PB + GA	0.95 21	0.950 3	0.953 8	0.95 16	0.9 50 9
PB + PSO	0.94 89	0.947 2	0.950 5	0.94 84	0.9 47 8
HOPB (Propo sed)	0.98 42	0.982 7	0.985 6	0.98 41	0.9 83 4

The HOPB classifier maintained high performance on the CIC-DDoS2019 dataset, demonstrating good generalization. The lower performance compared to NSL-KDD (98.42% vs. 99.37%) is expected due to the more complex and diverse attack patterns in the CIC-DDoS2019 dataset.

3. Automated Response Module Evaluation

The automated response module was evaluated based on response time and effectiveness.

Table 7. Automated Response Performance

Metric	Value
Average Response Time	247 ms
Alert Generation Time	112 ms
Resource Isolation Time	1.43 s
Traffic Blocking Time	1.87 s
False Positive Isolation	0.3%

False Negative Isolation	0.1%
--------------------------	------

The automated response module successfully isolated resources in 99.7% of high-severity cases, with an average response time of under 250 ms for detection and alert generation.

4. Adaptive Threshold Performance

The adaptive threshold mechanism was evaluated against fixed thresholds:

Table 8. Adaptive Threshold vs. Fixed Threshold

Threshold Type	Accuracy	False Positive Rate	False Negative Rate
Fixed (0.5)	0.9871	0.0129	0.0145
Fixed (0.3)	0.9678	0.0789	0.0043
Fixed (0.7)	0.9723	0.0032	0.0421
Adaptive	0.9937	0.0052	0.0079

The adaptive threshold achieved the highest accuracy while maintaining balanced false positive and false negative rates.

5. Discussion

1) Effectiveness of Hybrid Feature Selection

The hybrid feature selection approach (correlation + GA) demonstrated clear advantages over correlation-only and GA-only approaches: correlation-only selection provided good initial feature selection but missed complex feature interactions; GA-only selection achieved good results but required more computational resources; the hybrid approach combined the efficiency of correlation filtering with the optimization power of GA, achieving the best performance. The 3.55% improvement over correlation-only PB validates the effectiveness of the hybrid approach.

2) Advantages of the Perplexed Bayes Framework

The perplexed Bayes framework [1] consistently outperformed traditional Naïve Bayes across all experiments. The key advantage is the use of the geometric mean to aggregate feature probabilities, which relaxes the conditional independence assumption:

Naïve Bayes: $P(A|C) = \prod_i P(a_i|C)$ Perplexed Bayes: $P(A|C) = [\prod_i P(a_i|C)]^{1/n}$

This formulation avoids extreme posterior probability values that often occur in Naïve Bayes with correlated features [1].

3) Automated Response Integration

The integration of the automated response module [2] provides significant practical benefits:

- Real-time Mitigation: Response actions are triggered automatically, eliminating human delay.
- Reduced Human Error: Automated responses follow predefined, validated policies.
- Scalability: The system can handle large volumes of alerts without human intervention.

4) Comparison with Existing Work

Table 9. Comparison with Existing Work

Study	Technique	Dataset	Accuracy
Mishra et al. [1]	Perplexed Bayes	NSL-KDD	99.15%
Nandi et al. [22]	Hybrid Feature Selection	NSL-KDD	96.87%
Kim et al. [23]	CNN	KDD-CUP-99	98.54%
Cil et al. [24]	DNN	CIC-DDoS2019	97.21%
HOPB (Proposed)	PB + GA + Automated Response	NSL-KDD	99.37%
HOPB (Proposed)	PB + GA + Automated Response	CIC-DDoS2019	98.42%

The proposed HOPB framework achieves state-of-the-art performance on both datasets, with the added benefit of automated response capabilities.

5) Limitations

The study has several limitations:

1. Dataset Constraints: Although two datasets were used, further validation on more recent and diverse datasets is needed.
2. Computational Overhead: GA optimization requires significant computational resources, which may limit real-time deployment.
3. Response Actions: Predefined response actions may need refinement for specific cloud environments.
4. Attack Coverage: Focused on DDoS attacks; extension to other attack types is needed.

Conclusion And Future Work

1. Conclusion

This research presented a Hybrid Optimized Perplexed Bayes (HOPB) framework with automated response capabilities for multi-vector

DDoS detection in cloud environments. The key contributions include: (i) Hybrid Feature Selection, combining correlation analysis with Genetic Algorithm optimization to significantly improve feature selection and classification accuracy; (ii) an Automated Response Module integrating predefined response actions based on anomaly severity to enable real-time threat mitigation; (iii) an Adaptive Threshold Mechanism, using dynamic thresholding to reduce false positives and improve detection accuracy; and (iv) Cross-Dataset Validation, maintaining high performance across the NSL-KDD and CIC-DDoS2019 datasets.

Experimental results demonstrate that HOPB achieves 99.37% accuracy on NSL-KDD and 98.42% on CIC-DDoS2019, outperforming existing approaches. The automated response module provides real-time mitigation with average response times under 250 ms.

The proposed framework contributes to cloud security by providing more accurate DDoS detection through hybrid feature optimization, automated response for immediate threat mitigation, adaptive capabilities for changing traffic patterns, and a scalable architecture for large cloud environments.

2. Future Work

Several directions for future research have been identified:

1. Multi-Class Classification: Extending the binary classifier to multi-class classification for distinguishing between different attack types.
2. Distributed Detection: Developing collaborative detection mechanisms across multiple cloud regions and providers.
3. Real-Time Deployment: Optimizing the framework for real-time deployment in production cloud environments.
4. Other Attack Types: Adapting the approach for other cloud attacks including application-layer attacks, insider threats, and zero-day attacks.
5. Deep Learning Integration: Exploring hybrid approaches combining perplexed Bayes with deep learning for improved feature extraction.
6. Federated Learning: Implementing privacy-preserving detection across multiple cloud tenants.
7. Explainable AI: Adding explainability features to provide administrators with actionable insights about detected anomalies.
8. Cost-Benefit Analysis: Evaluating the trade-offs between detection accuracy, response speed, and computational resources.

Data Availability

The datasets used in this research are publicly available: the NSL-KDD dataset (<https://www.unb.ca/cic/datasets/nsl.html>) and the CIC-DDoS2019 dataset (<https://www.unb.ca/cic/datasets/ddos-2019.html>).

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

- N. Mishra, R. K. Singh, and S. K. Yadav, "Detection of DDoS vulnerability in cloud computing using the perplexed Bayes classifier," *Computational Intelligence and Neuroscience*, vol. 2022, Art. ID 9151847, 13 pages, 2022.
- H. K. Mistry, A. M. Goswami, and C. C. Mavani, "Automated anomaly detection and response system for enhancing cloud security," *Indian Patent Application*, Jul. 2024.
- J. He, "Cloud computing load balancing mechanism taking into account load balancing ant colony optimization algorithm," *Computational Intelligence and Neuroscience*, vol. 2022, Art. ID 3120883, 10 pages, 2022.
- F. S. Al-Anzi, S. K. Yadav, and J. Soni, "Cloud computing: Security model comprising governance, risk management and compliance," in *Proc. 2014 Int. Conf. Data Mining and Intelligent Computing (ICDMIC)*, Sep. 2014.
- D. Alsmadi and V. Prybutok, "Sharing and storage behavior via cloud computing: Security and privacy in research and practice," *Computers in Human Behavior*, vol. 85, pp. 218–226, 2018.
- J. Shroff, R. Walambe, S. K. Singh, and K. Kotecha, "Enhanced security against volumetric DDoS attacks using adversarial machine learning," *Wireless Communications and Mobile Computing*, vol. 2022, Art. ID 5757164, 10 pages, 2022.
- O. Cepheli, S. Buyukcora, and G. Karabulut Kurt, "Hybrid intrusion detection system for DDoS attacks," *Journal of Electrical and Computer Engineering*, pp. 1–8, 2016.
- B. A. Khalaf, S. A. Mostafa, A. Mustapha et al., "An adaptive protection of flooding attacks model for complex network environments," *Security and Communication Networks*, vol. 2021, Art. ID 5542919, 17 pages, 2021.

- M. Myint Oo, S. Kamolphiwong, T. Kamolphiwong, and S. Vasupongayya, "Advanced support vector machine (ASVM)-based detection for distributed denial of service (DDoS) attack on software defined networking (SDN)," *Journal of Computer Networks and Communications*, Art. ID 8012568, 12 pages, 2019.
- X. Yu, W. Yu, S. Li, X. Yang, Y. Chen, and H. Lu, "Web DDoS attack detection method based on semisupervised learning," *Security and Communication Networks*, vol. 2021, Art. ID 9534016, 10 pages, 2021.
- U. A. Butt, M. Mehmood, S. B. H. Shah et al., "A review of machine learning algorithms for cloud computing security," *Electronics*, vol. 9, no. 9, p. 1379, 2020.
- M. Idhammad, K. Afdel, and M. Belouch, "Detection system of HTTP DDoS attacks in a cloud environment based on information theoretic entropy and random forest," *Security and Communication Networks*, vol. 2018, Art. ID 1263123, 13 pages, 2018.
- H. A. Mahmood and S. H. Hashem, "Network intrusion detection system (NIDS) in cloud environment based on hidden naïve Bayes multiclass classifier," *Al-Mustansiriyah Journal of Science*, vol. 28, no. 2, pp. 134–142, 2018.
- P. Singh and V. Ranga, "Attack and intrusion detection in cloud computing using an ensemble learning approach," *International Journal of Information Technology*, vol. 13, no. 2, pp. 565–571, 2021.
- M. Masdari and M. Jalali, "A survey and taxonomy of DoS attacks in cloud computing," *Security and Communication Networks*, vol. 9, 2016.
- P. Oberoi, "Survey of various security attacks in cloud-based environments," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 9, pp. 405–410, 2017.
- R. V. Deshmukh and K. K. Devadkar, "Understanding DDoS attack & its effect in cloud environment," *Procedia Computer Science*, vol. 49, pp. 202–210, 2015.
- A. Aliyu, A. H. Abdullah, O. Kaiwartya et al., "Mobile cloud computing: Taxonomy and challenges," *Journal of Computer Networks and Communications*, Art. ID 2547921, 23 pages, 2020.
- Y. Jiang, B. Qiu, C. Xu, and C. Li, "The research of clinical decision support system based on three-layer knowledge base model," *Journal of Healthcare Engineering*, Art. ID 6535286, 8 pages, 2017.
- N. Mishra and R. K. Singh, "DDoS vulnerabilities analysis and mitigation model in cloud computing," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 23, no. 2, pp. 535–545, 2020.
- Y. Berguig, I. Laassiri, and S. Hanaoui, "DoS detection based on mobile agent and naïve Bayes filter," in *Proc. 2018 Int. Symp. Advanced Electrical and Communication Technologies (ISAECT)*, Rabat, Morocco, Nov. 2018.
- S. Nandi, S. Phadikar, and K. Majumder, "Detection of DDoS attack and classification using a hybrid approach," in *Proc. 2020 Third ISEA Conf. Security and Privacy (ISEA-ISAP)*, pp. 41–47, Guwahati, India, Mar. 2020.
- J. Kim, J. Kim, H. Kim, M. Shim, and E. Choi, "CNN-based network intrusion detection against denial-of-service attacks," *Electronics*, vol. 9, no. 6, p. 916, 2020.
- A. E. Cil, K. Yildiz, and A. Buldu, "Detection of DDoS attacks with feed-forward based deep neural network model," *Expert Systems with Applications*, vol. 169, Art. ID 114520, 2021.
- A. Rangapur, T. Kanakam, and A. Jubbison, "DDoSDet: An approach to detect DDoS attacks using neural networks," arXiv:2201.09514, 2022.
- N. O. Ogwara, K. Petrova, and M. L. Yang, "Towards the development of a cloud computing intrusion detection framework using an ensemble hybrid feature selection approach," *Journal of Computer Networks and Communications*, vol. 2022, Art. ID 5988567, 16 pages, 2022.
- M. Eshaty, H. Faris, and N. Obeid, "A competitive swarm optimizer with hybrid encoding for simultaneously optimizing the weights and structure of extreme learning machines for classification problems," *International Journal of Machine Learning and Cybernetics*, vol. 11, no. 8, pp. 1801–1823, 2020.
- C. Amine, S. Ben Alla, and A. Ezzati, "Makespan optimisation in cloudlet scheduling with improved DQN algorithm in cloud computing," *Scientific Programming*, vol. 2021, Art. ID 7216795, 11 pages, 2021.
- S. K. Yadav, D. K. Tayal, S. N. Shivhare, and S. Naresh Shivhare, "Perplexed Bayes classifier-

based secure and intelligent approach for aspect level sentiment analysis," *International Journal of Advanced Intelligence Paradigms*, vol. 13, no. 1/2, p. 15, 2019.

A. Saroha and A. Singh, "Security concerns and countermeasures in cloud computing: A qualitative analysis," *International Journal of Information Technology*, vol. 11, no. 4, pp. 683–690, 2019.

R. Goel, M. Garuba, and A. Girma, "Cloud computing vulnerability: DDoS as its main security threat, and analysis of IDS as a solution model," in *Proc. 2014 11th Int. Conf. Information Technology: New Generations*, Las Vegas, NV, USA, Apr. 2014.

E. S. Jeyalothi, J. Anitha, S. Rani, and B. Tiwari, "A comprehensive review: Computational models for obstructive sleep apnea detection in biomedical applications," *BioMed Research International*, pp. 1–21, 2022.

A. Ul Haq, J. P. Li, M. H. Memon, S. Nazir, and R. Sun, "A hybrid intelligent system framework for the prediction of heart disease using machine learning algorithms," *Mobile Information Systems*, vol. 2018, Art. ID 3860146, 21 pages, 2018.

C. S. Carlos, "Perplexed Bayes classifier," in *Proc. 12th Int. Conf. Natural Language Processing*, Trivandrum, India, Dec. 2015.

K. Dhingra and S. K. Yadav, "Spam analysis of big reviews dataset using fuzzy ranking evaluation algorithm and Hadoop," *International Journal of Machine Learning and Cybernetics*, vol. 10, no. 8, pp. 2143–2162, 2019.

L. Dhanabal and S. P. Shantharajah, "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, no. 6, pp. 446–452, 2015.

University of New Brunswick (UNB), "NSL-KDD dataset." [Online]. Available: <https://www.unb.ca/cic/datasets/nsl.html>

University of New Brunswick (UNB), "CIC-DDoS2019 dataset." [Online]. Available: <https://www.unb.ca/cic/datasets/ddos-2019.html>

L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.