Hindawi Publishing Corporation
Mobile Information Systems
Volume 2015, Article ID 260594, 13 pages
http://dx.doi.org/10.1155/2015/260594

Research Article

EVFDT: An Enhanced Very Fast Decision Tree Algorithm for Detecting Distributed Denial of Service Attack in Cloud-Assisted Wireless Body Area Network

Rabia Latif,[1] Haider Abbas,[1,2] Seemab Latif,[1] and Ashraf Masood[1]

[1]*National University of Sciences and Technology, Islamabad 44000, Pakistan*
[2]*King Saud University, Riyadh 11451, Saudi Arabia*

Correspondence should be addressed to Haider Abbas; hsiddiqui@ksu.edu.sa

Received 12 May 2015; Accepted 9 August 2015

Academic Editor: Basit Shahzad

Due to the scattered nature of DDoS attacks and advancement of new technologies such as cloud-assisted WBAN, it becomes challenging to detect malicious activities by relying on conventional security mechanisms. The detection of such attacks demands an adaptive and incremental learning classifier capable of accurate decision making with less computation. Hence, the DDoS attack detection using existing machine learning techniques requires full data set to be stored in the memory and are not appropriate for real-time network traffic. To overcome these shortcomings, Very Fast Decision Tree (VFDT) algorithm has been proposed in the past that can handle high speed streaming data efficiently. Whilst considering the data generated by WBAN sensors, noise is an obvious aspect that severely affects the accuracy and increases false alarms. In this paper, an enhanced VFDT (EVFDT) is proposed to efficiently detect the occurrence of DDoS attack in cloud-assisted WBAN. EVFDT uses an adaptive tie-breaking threshold for node splitting. To resolve the tree size expansion under extreme noise, a lightweight iterative pruning technique is proposed. To analyze the performance of EVFDT, four metrics are evaluated: classification accuracy, tree size, time, and memory. Simulation results show that EVFDT attains significantly high detection accuracy with fewer false alarms.

1. Introduction

Nowadays, cloud-assisted WBAN for patient health monitoring have attracted researchers' attention. Besides other open issues in WBAN environment such as energy efficiency, quality of service, and standardization, security and privacy are the key issues that need special attention. Among these security issues, data availability is the most nagging security issue. The Distributed Denial of Service (DDoS) attack is one of the most powerful attacks on the availability of patients health data and services of health care professional. DDoS attack severely affects the capacity and performance of a WBAN network if not handled in a timely and appropriate manner [1].

For detecting a DDoS attack in cloud-assisted WBAN, there is a need for a defensive approach that understands the network semantics and flow of traffic in the networks. When a victim node is flooded with huge amount of packets that exceeds its processing ability, the excess must be dropped. The packet based dropping strategy helps in distinguishing the legitimate traffic from the flood traffic and is used to avoid the impact of attack traffic on legitimate users. Observing the network traffic flow shows that there is no regular structure of patterns existing in the network and therefore statistical pattern identification techniques are needed. Integrating existing attack detection and defense mechanism in a resource constrained WBAN network increases the computation and communication cost [2].

The network resources are not enough to mitigate the huge amount of traffic generated by DDoS attack. Therefore, there is a need for an approach that is lightweight and capable of handling real-time streaming data. For this research, data mining techniques have been studied and explored. Among the data mining techniques, VFDT is proved to be the most prevalent due to the simplicity and interpretability of their rules and thus considered as more appropriate for low-power sensor networks. The underlying reasons for the selection of VFDT are as follows: (1) it lightweight; that is, it does not require a dataset to be stored in memory, thus making it suitable for resource constraint WBAN; (2) it can progressively build decision tree from scratch which helps in detecting DDoS attack at any stage; (3) each time a new segment of sensor data arrives, a testing and training process is performed over it keeping the stored data up to date; (4) it does not require reading full dataset and yet adjusts decision tree according to the newly incoming and gathered statistical attributes, thus consuming less memory space; (5) it is appropriate for huge amount of nonstationary and streaming data obtained from WBAN sensors; (6) it provides a transparent learning process. These features make VFDT a suitable candidate for implementing an autonomous decision maker for DDoS attack detection in cloud-assisted WBAN.

In this paper an improvement of VFDT [3], namely, enhanced VFDT (EVFDT), is proposed which differs from the existing algorithms in terms of classification accuracy, tree size, memory, and time. Our proposal is to build a decision tree based classification algorithm capable of handling noisy data and detecting a DDoS attack efficiently with high accuracy and low false alarm rate while allowing legitimate requesters to access the resources. The proposed algorithm is deployed at the victim node.

The contributions of this paper include the following:

(1) It proposed a novel EVFDT classification algorithm that is capable of classifying network traffic and detecting DDoS attack with high accuracy and less false alarms. EVFDT has achieved classification accuracy of about 96.5% with 0% noise and 81.5% with 20% noise in dataset.

(2) It is analyzing the effect of noise on stream data and its impact on the accuracy and false alarm rate while detecting a malicious behavior in the network.

(3) It highlights the shortcomings of the existing DDoS detection techniques.

(4) For simulation experiment, a DDoS attack algorithm is written and simulated in order to generate attack traffic for evaluation.

(5) It presents evaluation of the proposed algorithm and results.

The remainder of this paper is organized as follows: Section 2 discusses the existing data mining and stream mining techniques for detecting DDoS attack along with their limitation when applied in resource scarce WBAN environment. Section 3 elucidates the proposed system model and proposed algorithms. Section 4 presents the simulation experiment with proposed algorithm for generating DDoS attack strategy. At the end of Section 4, a detailed performance analysis and results comparison are given. Finally, Section 5 concludes the paper with recommendation for future work.

2. Related Work

2.1. Distributed Denial of Service (DDoS) Attack. A Distributed Denial of Service (DDoS) attack is defined as an explicit attempt by an attacker to exhaust the resources of a victim node. Multiple nodes are deployed to launch an attack by sending a stream of packets towards the victim, thus consuming the key resources of victim node and making them unavailable to legitimate nodes. These resources mainly include the network bandwidth, computing power, and memory resources [4].

DDoS attack is mainly divided into two classes [5], bandwidth depletion attack and resource depletion attack. In bandwidth depletion attack, the goal of an attacker is to flood the victim node with huge amount of traffic in order to prevent the legitimate traffic from reaching the victim node. It is further divided into flood attack and amplification attack. In resource depletion attack, the goal of an attacker is to degenerate the critical resources (processor and memory) of a victim node in order to prevent the legitimate user from using these resources.

A detailed analysis of DDoS attack in cloud-assisted WBAN environment and its implication shows that DDoS attack has following characteristics:

(1) During an attack, the packet length, sequence number, and window size remain fixed.

(2) Source IP and destination IP address along with port numbers are spoofed and generated randomly.

(3) Packet throughput decreases for legitimate users, which is defined as the number of bytes transferred from source to destination per unit time.

(4) Packet loss increases for legitimate users, which occurs due to the interaction of legitimate traffic with attack traffic.

(5) Packet delay increases as network congestion builds up.

(6) Traffic jitter may also increase significantly to impact especially real-time traffic.

Taking into account these characteristics, the key challenge lies in identifying the attack traffic from legitimate traffic of incoming data stream.

2.2. Data Mining Techniques. In the recent past, data mining techniques have been considered as one of the most promising solutions for identifying the malicious behavior of nodes in the network. For this research, data mining techniques have been studied and evaluated for the detection of DDoS attack in cloud-assisted WBAN environment. From the perspective of DDoS attack detection, existing data mining techniques (Subbulakshmi et al. [6], Wu et al. [7], Lee et al. [8], Arun and Selvakumar [9], and Thwe and Thandar [10]) can be broadly classified into source-based and destination-based detection techniques. Source-based detection techniques are deployed near the source of an attack whereas destination-based detection techniques are deployed near the victim of an attack.

Subbulakshmi et al. [6] proposed an Intrusion Detection System (IDS) based defense mechanism to counter DDoS attack. The IDS is trained using the datasets obtained from the extraction of attack traffic features. To strengthen the detection process, weights are added with these datasets at regular intervals.

Wu et al. [7] proposed a destination-based DDoS attack detection technique. In this technique, a decision tree is deployed for attack detection and a traffic pattern matching technique for attack identification and its traceback. For the classification of network traffic, fifteen different network and packet features are selected. C4.5 classification algorithm is deployed to classify the network traffic on the basis of identified packet features.

In [8], a source-based attack detection technique is proposed based on an enhanced traffic matrix approach. Traffic matrix parameters are optimized using a genetic algorithm (GA). To construct a traffic matrix, two features of the IP header are used, namely, the packet arrival time and source IP address. From the resultant traffic matrix, variance is calculated and used to classify the traffic as normal (high variance) or a DDoS attack (low variance). Finally alerts are generated upon the detection of an attack.

In [9], the author explores the ensemble based neurofuzzy classifier. The author contributes to existing classifier by including a weight update distribution policy, error cost reduction, and ensemble output combination approach. For performance evaluation, training and testing datasets are separately maintained. The result shows that this scheme efficiently detects DDoS attack.

Thwe and Thandar [10] proposed a statistical anomaly detection technique based on K-Nearest Neighbor (KNN) deployed at the victim node. A user specified threshold is defined. When the current state of the system differs from the defined model by a specified threshold, an anomaly is raised. At this stage KNN is used to detect an attack.

A detailed comparison of these techniques along with their limitations can be found in [11]. The major drawback of data mining techniques is that they are not suitable for real-time data mining of network traffic and require a huge amount of memory to store the datasets. In recent past, stream mining techniques have been proposed to overcome the limitations of data mining techniques.

2.3. Stream Mining Techniques: Very Fast Decision Tree (VFDT). Taking into account the resource constrained nature of WBAN sensors, VFDT proves to be a lightweight data mining technique that is able to process a large amount of high speed streaming data consuming less memory space. It turns out to be efficient in the detection of DDoS attack at any stage due to its ability of building decision tree from scratch. In [11] VFDT is applied for detection of DDoS attack and objective based comparison is done. The results show that the VFDT proves to be an accurate tool for DDoS attack detection. Therefore, VFDT is selected and improved for detecting DDoS attack efficiently.

2.3.1. Variants of VFDT. VFDT is a stream-based data classification method that learns using a complete set of N training samples expressed as (X, y), where X is a vector of n attributes given as $\{X_1, X_2, \ldots, X_n\}$. The aim is to construct a model of a mapping function $y = f(X)$ that will predict the classes of subsequent samples x with maximum accuracy. To design a VFDT for DDoS attack detection, the mathematical preliminaries used for the classification are first discussed (see [3, 12]).

Hoeffding Bound. This gives a certain level of confidence about the best attribute to split the node. Suppose we have N independent observations of a real-valued random variable r whose bounded range is R. The Hoeffding bound states that, with confidence level $1-\delta$, the true mean of variable r is at least $r - \epsilon$, where ϵ can be calculated using

$$\epsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2N}}. \tag{1}$$

Information Gain. VFDT uses the information gain as heuristic evaluation function to find the upper and lower bounds with high confidence. The upper bound $G(\cdot)^+$ and lower bound $G(\cdot)^-$ are calculated using

$$\begin{aligned} G(A,T)^+ &= \sum_{v\in A} P(T,A,v) + \sqrt{\frac{\ln(1/\delta)}{2N}} H(\mathrm{Sel}(T,A,v))^+ \\ G(A,T)^- &= \sum_{v\in A} P(T,A,v) + \sqrt{\frac{\ln(1/\delta)}{2N}} H(\mathrm{Sel}(T,A,v))^-, \end{aligned} \tag{2}$$

where A is an attribute in the T set of training samples. $P(T, A, v)$ is a fragment of the training samples in set T that holds the value v for attribute A. $\mathrm{Sel}(T, A, v)$ selects all the training samples having value v for attribute A from set T.

Although VFDT classifies stream data efficiently, it has limitations like the fact that it cannot handle noisy data and classification accuracy decreases with the increase in noise. In recent past, few variations of VFDT have been proposed. In this section, variants of VFDT are discussed and briefly analyzed for their feasibility long with their limitations when employed for DDoS attack detection in cloud-assisted WBAN. The variations of VFDT are analyzed on the following parameters: attack detection accuracy, time, memory, and tree size.

Domingos and Hulten [3] proposed VFDT based on Hoeffding bound (HB) using (1) to control over error in the attribute splitting distribution selection. Information gain $G(\cdot)$ given in (2) is used as heuristic evaluation function (HEF) in order to decide the split attribute to convert the decision nodes to leaves. $\Delta G = G(X_a) - G(X_b)$ defines the difference between the two best attributes X_a (highest $G(\cdot)$) and X_b (second highest $G(\cdot)$). If $\Delta G > \epsilon$, then X_a is considered as highest value attribute in $G(\cdot)$. At this stage, the splitting occurs on attribute X_a and the decision node is converted into leaf node. The major drawback of this technique lies in the fact that there exist certain cases, when the two information gains have very small differences and are equally good to become a leaf node. At this stage, a tie condition occurs and the process gets stuck. Resolving the tie-breaking is a computation intensive task that increases the processing time and decreases the overall accuracy of decision tree. At the same time, it is considered to be inappropriate for resource constrained WBAN.

To overcome the limitation of VFDT, Hulten et al. [12] proposed a fixed tie-breaking threshold τ. Whenever the difference between two information gains is very small, τ acts as a quick decisive parameter to solve the tie condition. The node splitting occurs on the current best attribute regardless of how good the second best attribute might be. The value of τ is chosen randomly and remains fixed throughout the process. An excessive tie-breaking condition reduces the performance of VFDT-τ significantly on noisy and complex streaming data, even with the use of parameter τ. VFDT-τ does not support pruning as the tree size itself is very small. While improving the accuracy, the tree size explodes. Therefore, a suitable pruning mechanism is required at this stage.

To overcome the limitation of fixed tie-breaking threshold, Yang and Fong [13] proposed an algorithm based on adaptive tie-breaking threshold computed directly from the Hoeffding bound mean. The value of HB mean fluctuates intensively with the increase in the noise percentage thus reducing the accuracy of attack classification.

Concept Adaptive VFDT (CVFDT) [12] maintains two trees simultaneously in memory. The tree with the shortest depth is retained and the other one is discarded. The main drawback of this technique is that it consumes more memory and time to maintain two trees. Also CVFDT does not handle noisy data efficiently.

2.4. Effect of Noise in Streaming Data. Noisy data is considered as meaningless or extraneous data that makes the identification of data patterns more difficult. As the noise increases in data stream, the number of outliers also increases. In sensor networks, noise arises due to the changes in system behavior and malicious activity in the network. There are two major sources of noise [14].

Error. An error is defined as a noisy value coming from an erroneous sensor. Outliers caused by such errors have a very high probability of occurrence.

Event. An event refers to a particular phenomenon which, in this case, is an attack occurrence event that changes the state of a system. Outliers caused by events occur with small probability but they are lasting and modify the historical patterns of sensor data.

In both cases, the presence of outliers due to noisy data decreases the attack detection accuracy and increases the false alarm rate. At the same time, the tree size increases which results in added memory consumption. The goal is to remove these outliers from the sensor data in order to ensure the high detection accuracy while keeping the resource consumption of the network minimal.

Figure 1 shows the detrimental effect of noisy data on classification accuracy (Figure 1(a)) and tree size (Figure 1(b)). The experimental data is synthetic and supplied as input to VFDT-τ algorithm with $\tau = 0.05$ [12]. The error rate significantly affects both the accuracy and tree size of decision tree when the number of instances increases manifold. Even a small error rate leads to increase in tree size by several times.

3. Proposed Model

3.1. System Architecture. The proposed Distributed Denial of Service (DDoS) attack detection system studies the network traffic behavior and classifies it as a normal or malicious traffic based on the observed traffic patterns. The proposed system architecture is shown in Figure 2. It consists of following phases starting from data collection phase up to response generation phase.

3.1.1. Data Collection Phase. In this phase, the incoming data stream is captured online and stored in database for training purposes. The captured data is supplied as an input to the preprocessing phase for feature extraction. Each instance of an incoming traffic is defined by a collection of features and is represented in feature vector space.

3.1.2. Preprocessing Phase. Preprocessing phase is further divided into the packet feature extraction phase followed by the labeling phase. In feature extraction phase, the real-time packets are captured from the network traffic in order to construct the new statistical features. These statistical features are listed in Table 1 and are used for DDoS attack detection and analysis. The identified features are important in defining the quality of service (QoS) of the network and to classify the network traffic pattern under DDoS attack. In labeling phase, classes are assigned to these statistical features. The entire dataset is divided into two classes labeled as "1" for attack and "0" for nonattack packet. After labeling the resulting dataset consists of both attack and nonattack data and is used for training the classification tree.

Mapping the preprocessing phase to feature vector space is given as follows:

(i) Let "x" be the N-dimensional vector of extracted features; that is, $x = \{x_1, x_2, x_3, \ldots, x_n\}$, where $\{1, 2, 3, \ldots, n\}$ are the individual packet features.

(ii) Let P_x be the packet of x features.

(iii) Let C_x be the vector space of labeled packets of dimension C_n.

3.1.3. Attack Classification. In this phase, the incoming traffic is classified as attack or nonattack by building a classification tree using the preprocessed data defined in feature vector space. For building the classification tree, we have proposed an algorithm, which is discussed in the next section.

3.1.4. Attack Response. The goal of attack response module is to minimize the impact of DDoS attack on the victim node while allowing the legitimate traffic to move forward. When a DDoS attack is detected, an appropriate traceback mechanism is applied to trace an attacker and block his traffic. The traceback technique will be explored in future work.

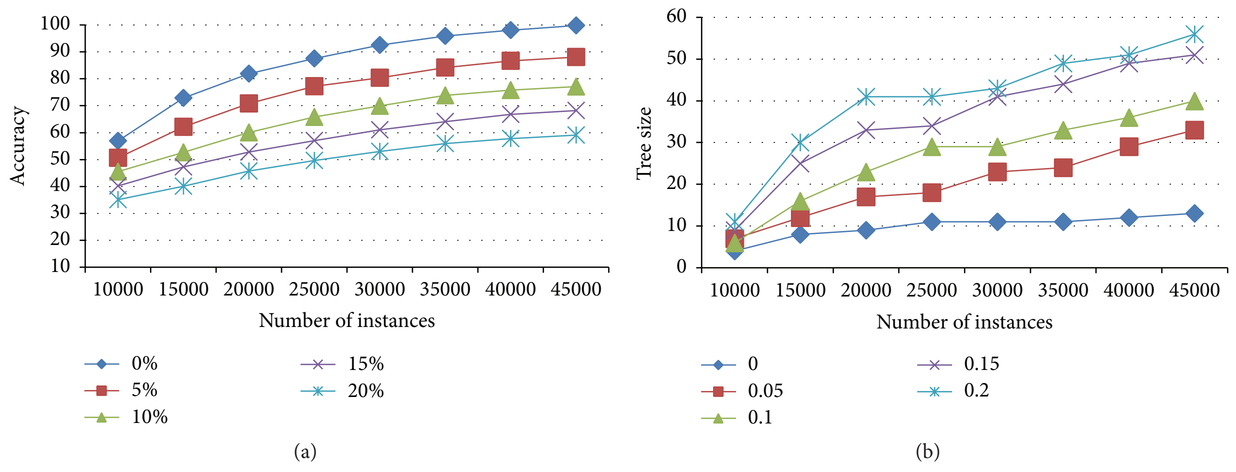


Figure 1: (a) Effect of noise on accuracy. (b) Effect of noise on tree size.

Table 1: List of features.

Features	Description	Formula for calculation
Packet loss (PL) percentage	Number of packets lost due to the interaction of the legitimate traffic with the attack. It is the presence of congestion in the network due to DDoS flooding attacks.	$\text{Loss} = \left[\frac{\sum \text{PL}}{\sum \text{PS}}\right] * 100.$ (i) PL is the packet lost. (ii) PS is the total number of packets sent towards the destination.
Delay or latency	The time taken by the packets to reach from source to destination.	$\text{Delay} = \sum (\text{PATime}_i - \text{PSTime}_i).$ (i) PATime_i is the packet arrival time. (ii) PSTime_i the packet start time.
Jitter (delay variation)	The variation in the time between packets arriving within a particular window. Jitter is used as an indicator of consistency and stability of network.	$\text{Jitter} = \sum_{i=0}^{n} \left[\frac{(\text{Delay}_i - \text{Delay})}{N}\right].$ (i) Delay_i is packet duration. (ii) Delay is the last packet delay. (iii) N is the difference of packet sequence number
Throughput (TH)	Bytes transferred per unit time from source to destination. The DDoS defense mechanism ideally increases the throughput for the legitimate users.	$\text{TH} = \frac{\sum_{i=0}^{n} \text{PacketReceived}}{\sum_{i=0}^{n} (\text{PacketStartTime} - \text{StopTime})}.$

3.2. Enhanced Very Fast Decision Tree (EVFDT): A Proposed Classification Algorithm. The proposed classification algorithm is an enhancement of original VFDT-τ [12] to make it efficient for the detection of DDoS attack in resource constrained cloud-assisted WBAN. The EVFDT classification algorithm simultaneously trains and tests the decision tree based on learning traffic patterns and classifies malicious behavior based on these learned patterns as shown in Figure 2.

Considering the severity of DDoS attack, the scarce resources, and missing protection mechanism of WBAN, the EVFDT improves the existing VFDT algorithms in terms of following parameters: (1) accuracy; (2) tree size; (3) time; (4) memory.

3.2.1. EVFDT Tree Building Process. EVFDT is based on original VFDT-τ [12] and is improved in two aspects: accuracy and the tree size. Algorithm 1 presents the pseudocode of EVFDT. It is divided into four subprocedures as described below.

(A) Procedure for Tree Initialization. In this procedure, the tree is initialized using a single leaf node, which is a root node. The tree grows as a new data stream arrives at the root node. Algorithm 2 shows the pseudocode of Tree Initialization Procedure. The procedure is executed when the first data stream arrives and it is the same as VFDT-τ [12].

(B) Procedure for New Stream Sample. In this procedure, data from the stream is traversed starting from the root node.

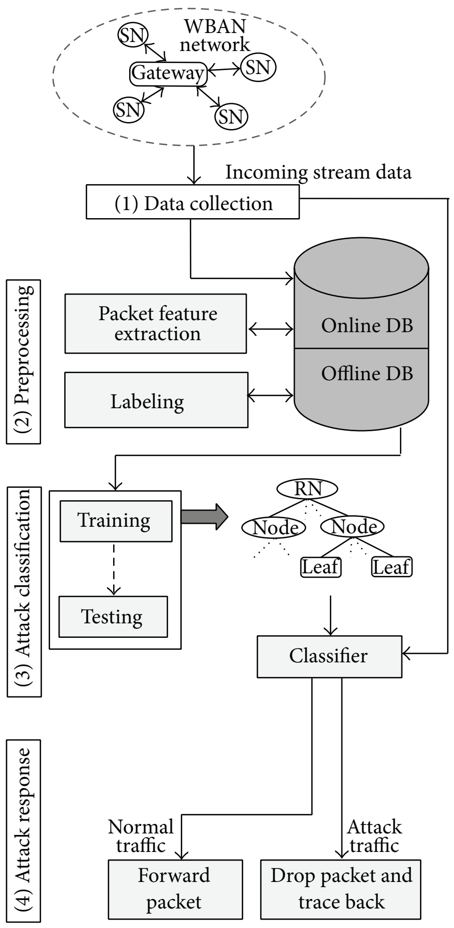


Figure 2: Proposed system architecture.

Each time a new data stream arrives, this procedure traverses the stream from root node to the leaf node and at the same time the tree statistics are updated. Algorithm 3 shows the pseudocode for traversing a new sample. This procedure is same as tree traversing of VFDT-τ [12].

(C) Procedure for Accuracy Improvement. Algorithm 4 shows the proposed pseudocode for EVFDT, that is, enhancing the accuracy of VFDT-τ [12]. VFDT-τ takes a fixed value of τ to break the tie as discussed in Section 2.3, which can be deviating because of presence of noise. To overcome this, optimized VFDT (OVFDT) takes the mean of Hoeffding bound (HB) values for tie-breaking. As noise ratio is high in sensor data, a simple mean does not depict the true mean of HB because of the presence of outliers. To cater for such outliers, EVFDT incorporates a procedure for accuracy enhancement as given in Algorithm 4 and explained as follows.

Let XS be the sorted list of HB values, m the total number of HB values in XS, and n the new HB value seen at the node. Starting with $m \geq 3$, the mean of difference between HB values in XS is calculated. This mean is stored as threshold T_r. Let XS_n be the position of n in XS and the PrunedMean is calculated with new value n. If this PrunedMean is less than the threshold T_r, then n is added to XS; otherwise, n is discarded as an outlier.

(D) Procedure for Tree Pruning. Algorithm 5 shows the pseudocode for the proposed tree pruning and explained as follows.

Let HT be the Hoeffding tree to be pruned and S the stream of examples belonging to S_0 and DataSeenAtLeafn_0 is the number of samples seen at leaf node. For every example S in S_0 is passed through the tree starting from the root node. If S_0 is filtered to the leaf node n_0, then increment DataSeenAtLeafn_0 otherwise continues growing the HT.

If DataSeenAtLeafn_0 is less than τ, where τ is the threshold for checking the eligibility of a node to be part of HT, then prune the tree by deleting the leaf node n_0 and updating the HT. The eligibility of the node to be part of HT is checked by comparing the number of samples seen at the leaf node with τ. The comparison will tell us that this leaf node has less contribution towards classification as a smaller number of instances are filtered to this leaf node on the current HT.

4. Simulation Experiments and Evaluation

In this section, we evaluate the performance of proposed EVFDT in detecting DDoS attack from incoming data stream in cloud-assisted WBAN environment. EVFDT is compared with VFDT and its variants in terms of the following parameters: accuracy, time, tree size, and memory. These evaluation parameters are important for comparison. Analysis shows that a higher classification accuracy produces a bigger tree size which in turn consumes more memory space and takes more learning time. Similarly, a faster learning speed brings a smaller tree size, which results in less memory space but lower classification accuracy.

The proposed EVFDT algorithm has been implemented using Very Fast Machine Learning (VFML) libraries [15]. The experiments are run on windows7-64-bit workstation with 2.8 GHz CPU and 8 GB RAM with all background processes switched off.

4.1. Synthetic Datasets. The Low Energy Adaptive Clustering Hierarchy (LEACH) protocol [16] was implemented in NS-2 for generating the synthetic data stream containing 1 million data values, which are divided into five datasets. Each dataset contains different number of instances and noise percentage. LEACH protocol is selected because it is lightweight and closely reflects the WBAN scenario. Cluster heads act as body control unit (BCU) and all other nodes as sensor nodes as depicted in Figure 3. LEACH is responsible for transferring the data from WBAN sensor node to BCU. The simulation runs for 900 s, 1200 s, 1500 s, 1800 s, and 2000 s to generate the data streams. Other simulation parameters and network configurations are shown in Table 2.

The resulting dataset includes both attack and nonattack data. The DDoS attack code is attached to sensor nodes to make them malicious. The number of malicious sensor nodes varies with different attack dataset.

```
Input:
$S$: a stream of examples
$X$: a set of symbolic attributes
$G(\cdot)$: heuristic evaluation function for node splitting
$\delta$: one minus desired probability of choosing the correct attribute at any given node.
$n_{\min}$: number of samples between estimation of growth
$XS$: sorted list of Hoeffding bound values
$m$: total number of values in $XS$
$n$: new Hoeffding Bound value seen at the node
$T_r$: adaptive threshold
$S_0$: subset of $S \rightarrow S_0 \in S$
$\tau$: 5% of examples in $S_0$. Threshold for checking the eligibility of a node to be part of HT
Size of $S_0 = n_{\min}$
Output:
  A decision tree HT
Procedure EnhancedVFDT($S, X, G, \delta, n_{\min}$)
BEGIN:
  A stream of examples $S$ arrives
    IF (HT = $\phi$), THEN TreeInitialization($S, X$)
        Get an Initialized HT with a single root node
    IF (HT $\neq \phi$), THEN NewStreamSample($S, X$)
    Label $l$ with the majority class among the samples seen so far at $l$
    Let $n_1$ be the number of samples seen at $l$
    IF the samples seen so far at $l$ are not all of the same class and ($n_l$ mode $n_{\min}$) = 0
    THEN
        Compute $G_l(X_i)$ for each attribute $X_i \in X_l - \{X_\phi\}$ using $n_{ijk}(l)$
        PrunedMean = AccuracyEVFDT($XS, m, n, T_r$)
        Let $X_a$ be the attribute with highest $G_l$ and $X_b$ be the attribute with second-highest $G_l$
        Compute $\varepsilon$ using (1)
        Let $\Delta G_l = G_l(X_a) - G_l(X_b)$
        IF (($\Delta G_l$) > $\varepsilon$ or ($\Delta G_l$) $\leq$ PrunedMean) and $X_a \neq X_\phi$, THEN split $X_a$ as a branch
            FOR each branch of split
              Add a new leaf $l_m$ and let $X_m = X - \{X_a\}$
              Let $G_m(X_\phi)$ be the $G$ obtained by predicting the most frequent class at $l_m$
              FOR each class $y_k$ and each value $x_{ij}$ of each attribute $X_i \in X_l - \{X_\phi\}$
                Let $n_{ijk}(l_m) = 0$.
              END-FOR
            END-FOR
        END-IF
    ELSE Pruning($S, S_0, n_{\min}, \tau$, HT)
Return HT
END:
```

Algorithm 1: EVFDT procedure: enhanced EVFDT.

```
Procedure TreeInitialization($S, X$)
BEGIN:
    Let HT be a tree with a single leaf $l_1$ (the root)
    Let $X_1 = X \cup (X_\phi)$
    Let $G_1(X_\phi)$ be the $G$ obtained by predicting the most frequent class in $S$
    FOR each class $y_k$
        FOR each value $x_{ij}$ of each attribute $X_i \in X$
            Let $n_{ijk}(l) = 0$
        END-FOR
    END- FOR
    Return HT
END:
```

Algorithm 2: EVFDT procedure: tree initialization.

```
Procedure NewStreamSample($S$, $X$)
BEGIN:
  FOR each new instance $(x, y)$ in $S$
    Sort $(x, y)$ into a leaf $l$ using HT
  FOR each $x_{ij}$ in $X$ such that $X_i \in X$
      Increment $n_{ijk}(l)$
  END-FOR
END:
```

Algorithm 3: EVFDT procedure: new stream sample.

Table 2: Simulation parameters.

Parameters	Values
Sensing field	50 × 50
Topology	Star
Simulation time	900 s, 1200 s, 1500 s, 1800 s, 2000 s
Packet size	1000 bytes
Radio communication range	2 m standard, 5 m special use
Number of nodes	50
BAN coordinator	Directional mode
Sensor nodes	Omnidirectional mode
Routing protocol	LEACH
Transport protocol	TCP

4.1.1. DDoS Attack Strategy: Generation and Analysis. In this section, an attack dataset is generated and analyzed at TCP layer. In ongoing simulation experiment, first the normal TCP traffic was considered for analysis. Secondly, the attack was generated and its intensity under flooding attack with TCP traffic was analyzed. An attack algorithm was written and generated online. The resulting dataset (attack dataset) is stored in database at base station for preprocessing. The proposed EVFDT algorithm was applied on the preprocessed dataset. The DDoS attack algorithm is presented in Algorithm 6.

Java code is written to randomly add noise in the dataset. For this purpose, N-dimensional feature vector is multiplied with vector of random variables taken from the Normal Distribution $N(0, \sigma 2)$, where $\sigma 2$ is noise variance adjusted according to the percentage of noise added.

To evaluate the performance of EVFDT, noise is attached to datasets in order to compare the result of EVFDT with VFDT and its variants under different noise percentage. Each dataset contains different number of instances and divided into 20% testing data and 80% training data to learn the classifier. The performance is evaluated using the following parameters.

(1) Attack Detection Accuracy. In general, the accuracy of EVFDT is directly proportional to the number of data stream samples. With the increase in data stream samples, the EVFDT becomes more and more accurate as long as there is noise-free data. But as soon as the noise is injected, the classification accuracy starts decreasing as shown in Figure 4.

For the conducted simulation experiments, attack detection accuracy was calculated using

$$\text{Detection Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total number of Tested Examples}}, \quad (3)$$

where True Positive (TP) is number of samples correctly classified as attack class, True Negative (TN) is number of samples correctly classified as nonattack class, False Positive (FP) is number of samples incorrectly classified as attack class, and False Negative (FN) is number of samples incorrectly classified as nonattack class.

Figure 5 shows the accuracy comparison of EVFDT with existing VFDT and its variants on datasets with noise percentage of 10% and 20%. On all experimental datasets, EVFDT maintains higher accuracy with less false alarm rate and more sensitivity.

Table 3 shows the sensitivity, specificity, and false alarm rate (FAR) of algorithms with respect to different noise percentages. These can be calculated using (4) (sensitivity), (5) (specificity), and (6) (FAR)

$$\text{Sensitivity} = \frac{(\text{True Positives})}{(\text{True Positives} + \text{False Negatives})}, \quad (4)$$

$$\text{Specificity} = \frac{(\text{True Negatives})}{(\text{True Negatives} + \text{False Positives})}, \quad (5)$$

$$\text{FAR} = \frac{(\text{False Positives})}{(\text{False Positives} + \text{True Negatives})}. \quad (6)$$

Results show that the average accuracy of EVFDT is greater than VFDT and its variants. VFDT-τ has lowest accuracy among all algorithms.

(2) Tree Size. A significant characteristic of EVFDT lies in its ability to build a decision tree with reduced tree size and increased classification accuracy simultaneously. The tree size gets bigger with increase in noise percentage as shown in Figure 6. Although VFDT-τ obtains smallest tree size in our simulation, it results in increased classification error. As shown in Table 4, notably, EVFDT maintains a smaller tree size. In few cases, EVFDT and VFDT-τ produce same tree size, but the average tree size of EVFDT is smaller than VFDT-τ. The tree size is the depth of the decision tree. The simulation experiment shows the tree size (TS): $\text{TS}_{\text{EVFDT}} < \text{TS}_{\text{VFDT-}\tau} < \text{TS}_{\text{CVFDT}} < \text{TS}_{\text{OVFDT}}$.

(3) Computation Time. Early detection of an attack is a desirable property of any algorithm. A detection mechanism should be fast enough in detecting an attack. Computational time is the total time taken in seconds for processing a full set of data stream. Computational complexity of proposed algorithm is proportional to $O(lpdvc)$, where lp is the length of a pruned tree, d is the total number of attributes, v is the number of values per attribute, and c is the number of classes. VFDT-τ has a small running time due to small and fixed value

```
Procedure AccuracyEVFDT($XS, m, n, T_r$)
BEGIN
  IF ($m == 1$)
    $T_r = \sum \frac{XS}{m}$
  ELSE-IF ($m == 2$)
    $T_r = \sum \frac{XS}{m}$
    FOR ($j = 1$)
        $XD_j = XS_{j+1} - XS_j$
        Increment $j = j + 1$
        $j < m$
    END-FOR
  ELSE // Calculate mean difference of $XS$ without $n$
    FOR ($j = 1$)
        $XD_j = XS_{j+1} - XS_j$
        Increment $j = j + 1$
        $j < m$
    END-FOR
    $T_r = (\sum_{i=1}^{m} XD_i)/m$ // Updated Threshold
    Let $XS_n$ be the position of "$n$" in sorted list
        PrunedMean $= \frac{(XS_n - XS_{n-1}) + (XS_{n+1} - XS_n)}{2}$
    IF (PrunedMean $\leq T_r$)
    THEN
        Add $n$ in sorted list $XS$
        $XS \longleftarrow XS + n$
    ELSE
        Discard $n$ as it is outlier
        $XS \longleftarrow XS$
Return PrunedMean
END
```

Algorithm 4: EVFDT procedure: accuracy EVFDT.

```
Procedure Pruning($S, S_0, n_{\min}, \tau$, HT)
BEGIN:
    Let DataSeenAtLeaf be the number of samples seen at leaf node $n_0$
    FOR each example $S \in S_0$
        IF the sample $S$ traverses to the node $n_0$ which is a leaf node
        THEN
            Start counter on node $n_0$
            Increment: DataSeenAtLeaf $n_0$
        ELSE
            Continue growing EVFDT
    END-FOR
    IF (DataSeenAtLeaf $< \tau$)
    THEN
        Prune the tree: Delete $n_0$
        UPDATE HT
END:
```

Algorithm 5: EVFDT procedure: pruned EVFDT.

τ. EVFDT takes slightly more time than VFDT-τ because of pruning and threshold computation as shown in Figure 7. The computation time (CT) is compared as $\text{CT}_{\text{VFDT-}\tau} < \text{CT}_{\text{EVFDT}} < \text{CT}_{\text{OVFDT}} < \text{CT}_{\text{CVFDT}}$.

(4) Memory. Taking into account the resource scarcity of WBAN environment, the proposed mechanism should consume less memory resources as shown in Figure 8. The advantage of VFDT over existing machine learning techniques is

```
Input:
    SN: Set of sensor nodes for cluster head selection
    R: Number of rounds
    Simulation Parameters of LEACH protocol
Output:
    DDoS Attack Dataset
Procedure DDoSAttackAlgorithm(SN, R)
BEGIN:
        IF ($r = 0$)
            Initial round CH selection
        FOR maximum number of rounds "r"
            Choose r rounds CH randomly
            CH announces schedule time T to all SN
        END-FOR
        Attach attack code with random nodes $N_r$
        Randomly initiates malicious nodes towards victim node V
            Malicious nodes start and stop randomly according to time T during formation
        Malicious nodes start compromise victim node V
            Malicious nodes forward flooding packets to V with high rate to overflow and consume resources available to victim
            Victim node V receives packet with high rate
        More malicious nodes start compromising victim at their schedule time T causing DDoS flooding attack
END:
```

Algorithm 6: DDoS attack algorithm.

Table 3: Sensitivity, specificity, and FAR of the algorithms.

Noise	VFDT-τ (%)				CVFDT (%)				OVFDT (%)				EVFDT (%)			
	SN	SP	AC	FA	SN	SP	AC	FA	SN	SP	AC	FA	SN	SP	AC	FA
0%	90.8	98.4	92.2	5.4	89.8	99.4	94.6	5.1	97.6	94.9	96.2	2.4	97.9	94.7	96.5	1.1
5%	79.2	91.7	87.4	5.9	78.9	91.7	87.5	5.7	79.2	91.7	87.6	2.7	82.3	90.0	89.8	1.3
10%	78.2	88.8	82.3	6.8	69.2	90.8	79.6	6.2	77.0	90.7	83.6	3.2	80.2	88.4	84.2	1.7
15%	76.3	88.3	81.2	7.3	67.7	88.9	78.0	7.0	76.9	86.8	80.4	3.5	79.2	86.8	83.0	2.2
20%	77.7	81.3	78.2	7.9	66.9	79.1	76.9	7.8	79.9	81.0	79.3	4.1	78.1	83.4	81.5	2.7
Avg.	**80.4**	**88.9**	**84.2**	**6.7**	**74.5**	**89.1**	**83.3**	**6.3**	**82.5**	**89.0**	**85.4**	**3.2**	**83.6**	**88.6**	**87.0**	**1.8**

Table 4: Tree size comparison with different noise percentage.

Dataset	Noise	VFDT-τ	CVFDT	OVFDT	EVFDT
Dataset-10	0%	4	5	6	3
	5%	6	7	8	6
	10%	9	9	11	8
	15%	9	10	13	9
	20%	9	9	15	8
	Average	**7**	**8**	**9**	**6**
Dataset-10	0%	5	6	5	5
	5%	7	7	8	7
	10%	8	9	10	6
	15%	10	11	11	8
	20%	10	11	13	8
	Average	**8**	**9**	**10**	**6**

that it does not require a full dataset to be stored in memory. Memory required for running EVFDT is proportional to $O(ndvc)$, where n is the number of decision nodes in a tree, d is the total number of attributes, v is the number of values per attribute, and c is the number of classes. The total amount of memory required to run EVFDT is the sum of memory allocated for learning and memory allocated for training. EVFDT consumes less memory than existing algorithms and is compared as (M): $M_{\text{EVFDT}} < M_{\text{OVFDT}} < M_{\text{CVFDT}} < M_{\text{VFDT-}\tau}$.

4.2. Comparison of EVFDT with VFDT and Its Variants. Comparison of EVFDT classification algorithm with existing VFDT and its variants is shown in Table 5. Only OVFDT and EVFDT can handle noisy data. At the same time, OVFDT handles noisy data to some extent and becomes inaccurate with the increase in noise percentage due to the presence of outliers. VFDT-τ, CVFDT, and IVFDT do not provide tree pruning. They maintain small tree size from the beginning but with increased error percentage in classification. Only proposed EVFDT algorithm efficiently handles noisy data and at the same time maintains small tree size with less resource usage.

Table 5: Comparison of existing machine learning techniques with VFDT.

Features	VFDT-τ	CVFDT	OVFDT	EVFDT
Detection accuracy	Very low	Very low	Good; does not handle outliers	Excellent
Resource usage (time/memory)	Less time; more memory	More time in building two trees; requires additional memory	Less time; less memory	Having same time as VFDT but consuming very less memory space
Noisy data handling	Does not handle noisy data	Not appropriate under noisy data	HB fluctuation intensifies under noisy data; accuracy decreases	Handles noisy data efficiently
Tree size/pruning	Small tree size; no pruning	Same tree size as VFDT; no pruning	Small tree size; incremental pruning	Small tree size; iterative pruning
Computational resources	Consuming less resources	Consuming more resources by maintaining two trees	Consuming less resources	Consuming very less resources by cutting of HB outliers

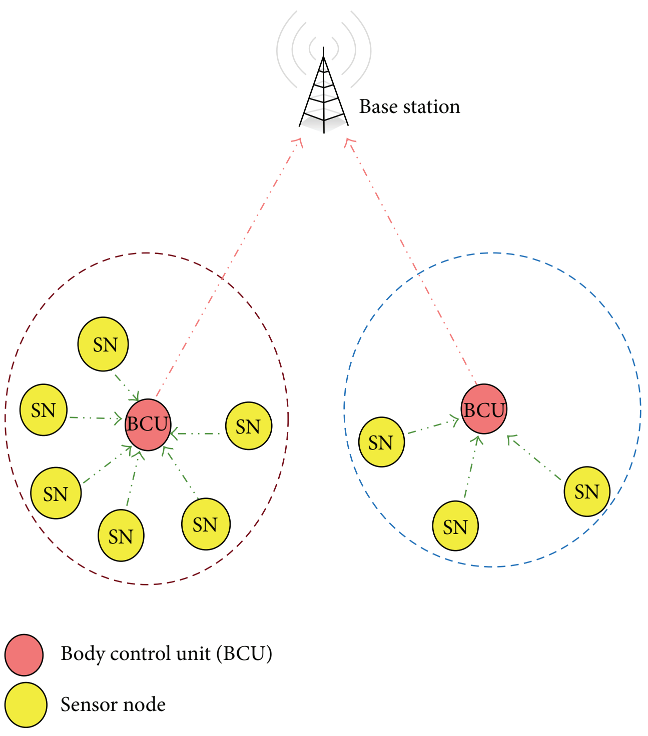


Figure 3: Illustration of LEACH protocol.

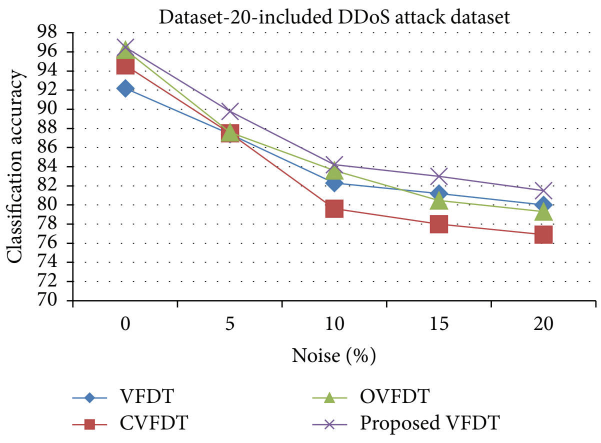


Figure 4: Accuracy in different noise percentage.

5. Conclusion and Future Work

Nowadays, zombies-based DDoS attack occurs with legitimate flow of traffic. Therefore, it is very difficult to detect such attacks even with the presence of stored attack traffic signatures. The challenge is to distinguish the legitimate traffic and DDoS attack traffic. Data mining techniques for data classification fail for real-time streaming data and also they require a sufficient amount of memory for data storage. On the other hand, stream mining techniques handle high speed streaming data originating from WBAN sensors and are efficient for resource scarce WBAN network. An algorithm based on VFDT is proposed in this paper. Our main contributions include a novel enhanced Very Fast Decision Tree (EVFDT) classification algorithm and it differs from existing algorithms in terms of attack classification accuracy and tree size. The performance of EVFDT algorithms is evaluated on the synthetic dataset generated by implementing a LEACH protocol. An attack code is written to generate DDoS attack. Experimental result shows that the proposed algorithm is able to detect an attack with high classification accuracy (96.5%) and low false alarm rate (1.1%) with less memory overhead. The proposed model is deployed at victim node and is deployed in real-time network environment. In future, the proposed architecture is deployed on real WBAN test bed to evaluate the performance of proposed algorithm.

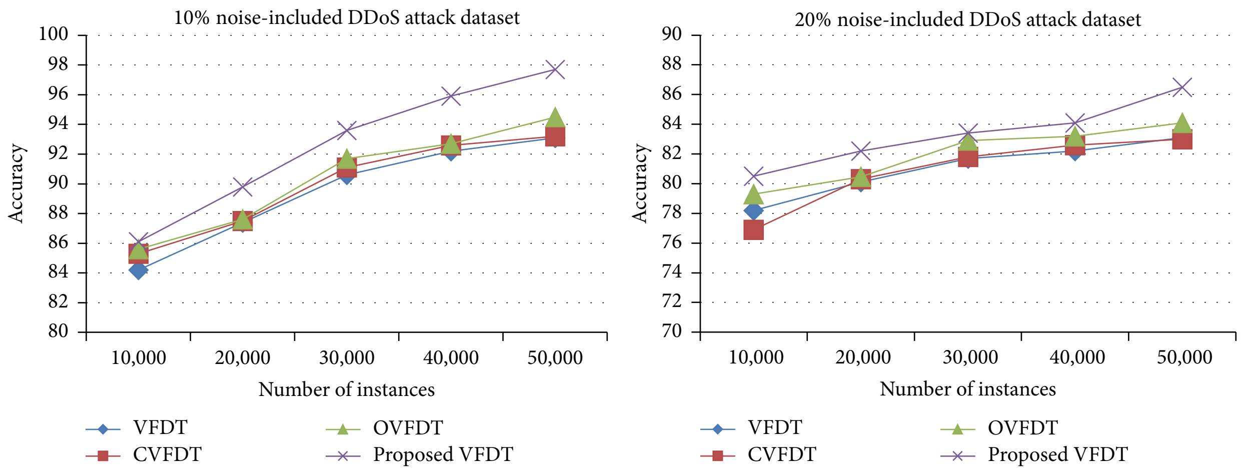


Figure 5: Accuracy versus number of instances in different noise ercentages.

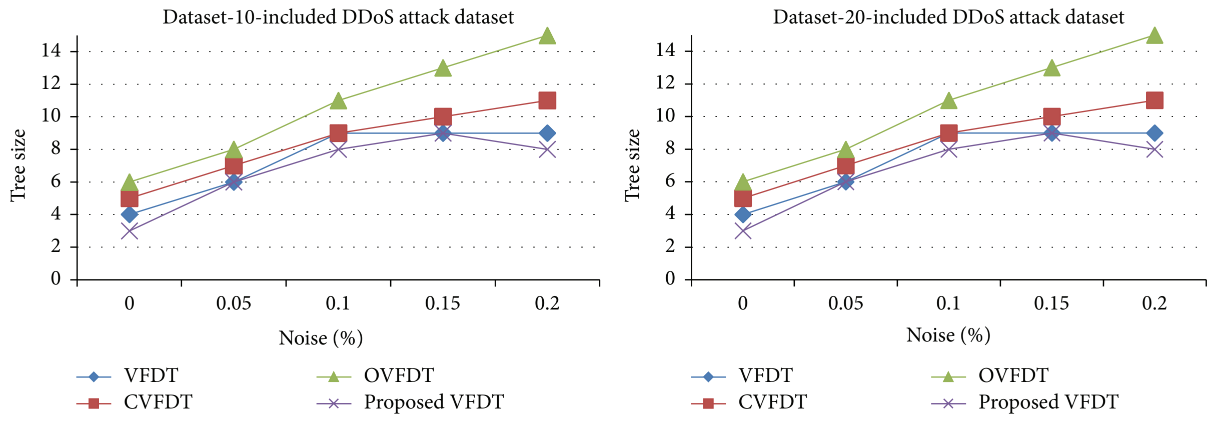


Figure 6: Tree size comparison with different noise percentage.

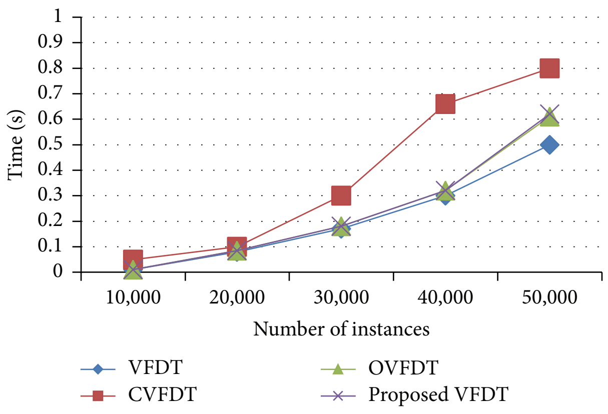


Figure 7: Computation time comparison.

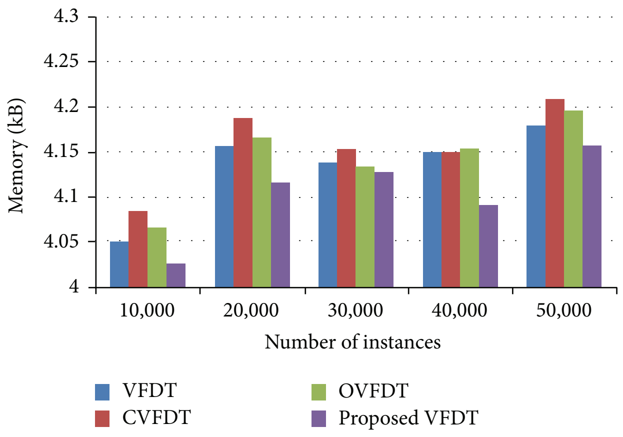


Figure 8: Memory resource comparison.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors would like to extend their sincere appreciation to the Deanship of Scientific Research at King Saud University for its funding of this research through the Research Group Project no. RG-1435-048. The authors would also like to thank the National University of Sciences and Technology, Islamabad, Pakistan, for its support during this research.

References

[1] R. Latif, H. Abbas, S. Assar, and S. Latif, "Analyzing feasibility for deploying very fast decision tree For DDoS attack detection in cloud-assisted WBAN," in *Intelligent Computing Theory: Proceedings of the 10th International Conference, ICIC 2014, Taiyuan, China, August 3-6, 2014*, Lecture Notes in Computer Science, pp. 507–519, Springer, Berlin, Germany, 2014.

[2] R. Latif, H. Abbas, and S. Assar, "Distributed denial of service (DDoS) attack in cloud- assisted wireless body area networks: a systematic literature review," *Journal of Medical Systems*, vol. 38, article 128, 2014.

[3] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 71–80, Boston, Mass, USA, August 2000.

[4] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDOS) flooding attacks," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013.

[5] D. Arora, P. Singh, and V. Singh, "Impact analysis of denial of service (DoS) due to packet flooding," *International Journal of Engineering Research and Applications*, vol. 4, no. 6, pp. 144–149, 2014.

[6] T. Subbulakshmi, S. M. Shalinie, V. Ganapathisubramanian, K. Balakrishnan, D. Anandkumar, and K. Kannathal, "Detection of DDoS attacks using enhanced support vector machines with real time generated dataset," in *Proceedings of the 3rd International Conference on Advanced Computing (ICoAC '11)*, pp. 17–22, IEEE, Chennai, India, December 2011.

[7] Y.-C. Wu, H.-R. Tseng, W. Yang, and R.-H. Jan, "Ddos detection and traceback with decision tree and grey relational analysis," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 7, no. 2, pp. 121–136, 2011.

[8] S. M. Lee, D. S. Kim, J. H. Lee, and J. S. Park, "Detection of DDoS attacks using optimized traffic matrix," *Computers and Mathematics with Applications*, vol. 63, no. 2, pp. 501–510, 2012.

[9] R. K. Arun and S. Selvakumar, "Detection of distributed denial of service attacks using an ensemble of adaptive and hybrid neuro-fuzzy systems," *Computer Communications*, vol. 36, no. 3, pp. 303–319, 2013.

[10] T. Thwe and P. Thandar, "Statistical anomaly detection of DDoS attacks using K-nearest neighbour," *International Journal of Computer & Communication Engineering Research*, vol. 2, no. 1, pp. 315–319, 2014.

[11] R. Latif, H. Abbas, and S. Assar, "Distributed Denial of Service (DDoS) attack in cloud—assisted wireless body area networks: a systematic literature review," *Journal of Medical Systems*, vol. 38, article 12, 2014.

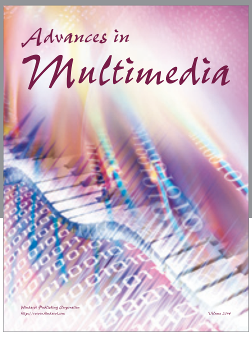
[12] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '01)*, pp. 97–106, San Francisco, Calif, USA, August 2001.

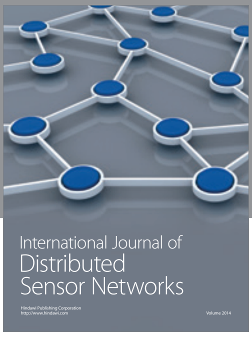
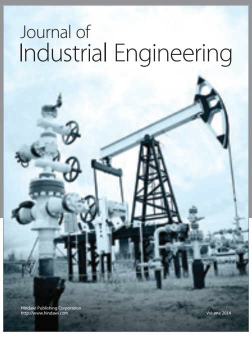
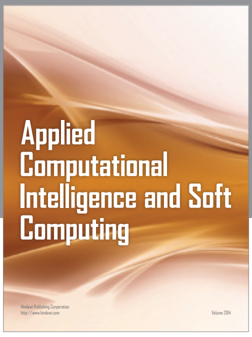
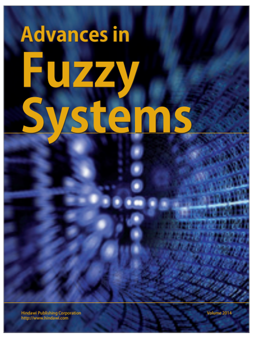
[13] H. Yang and S. Fong, "Moderated VFDT in stream mining using adaptive tie threshold and incremental pruning," in *Proceedings of the 13th International Conference on Data Warehousing and Knowledge Discovery (DaWaK '11)*, pp. 471–483, Toulouse, France, August 2011.

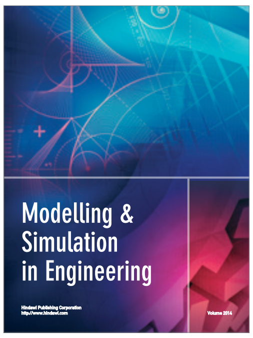
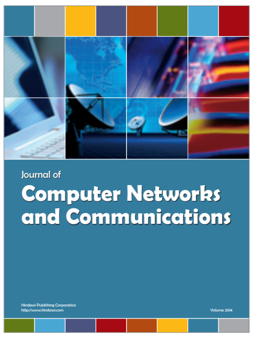
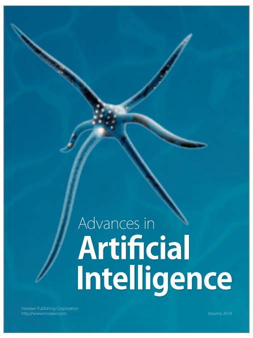
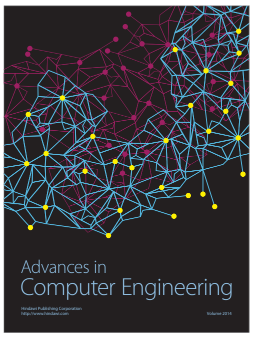
[14] A. Fawzy, H. M. O. Mokhtar, and O. Hegazy, "Outliers detection and classification in wireless sensor networks," *Egyptian Informatics Journal*, vol. 14, no. 2, pp. 157–164, 2013.

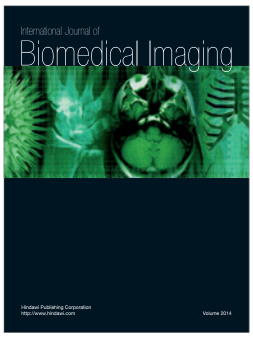
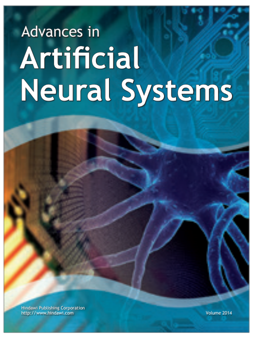
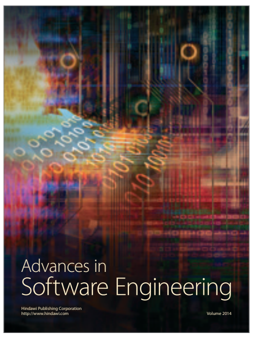
[15] VFML (Very Fast Machine Learning) toolkit, 2014, http://www.cs.washington.edu/dm/vfml/.

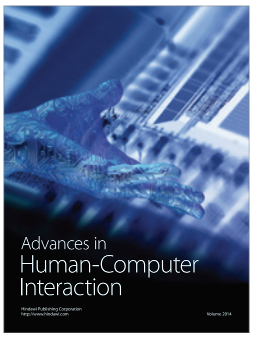
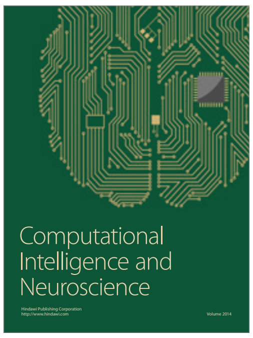
[16] L. Hughes, X. Wang, and T. Chen, "A review of protocol implementations and energy efficient cross-layer design for wireless body area networks," *Sensors*, vol. 12, no. 11, pp. 14730–14773, 2012.


Advances in
Multimedia

The Scientific World Journal

International Journal of
Distributed Sensor Networks

Journal of
Industrial Engineering

Applied Computational Intelligence and Soft Computing

Advances in
Fuzzy Systems


Modelling & Simulation in Engineering

Journal of
Computer Networks and Communications

Advances in
Artificial Intelligence

Advances in
Computer Engineering

International Journal of
Computer Games Technology

International Journal of
Biomedical Imaging

Advances in
Artificial Neural Systems

Advances in
Software Engineering

Journal of
Robotics

Advances in
Human-Computer Interaction

Computational Intelligence and Neuroscience

International Journal of
Reconfigurable Computing

Journal of
Electrical and Computer Engineering