# Intrusion Detection System: Time Probability Method and Hyperbolic Hopfield Neural Network

Jabez. J
Research Scholar, Department of Computer Science
Sathyabama University
Chennai, India
jabezme@gmail.com

Anadha Mala. G. S
Professor & Head, Department of Computer Science
St.Joseph's College of Engineering
Chennai - 600119. India

***ABSTRACT:*** *Nowadays, high level security maintenance between the two companies is the significant issue when they are communicating between them. However, the goal of achieving complete secure communication between those companies communicating over the internet with various networks is still lacked by the following factors such as misuses and intrusions. Thus, the Intrusion Detection Systems are very important components to detect these types of attacks. There are various approaches being utilized to these intrusion detections, but any of the systems so far is not completely flawless. The proposed framework utilized novel methods such as time probability based pattern detection method and Hyperbolic Hopfield Neural Network for detect the intrusion from the real time network datasets. The experimental results shows our proposed framework is obtained better results rather than other frameworks.*

**Keywords:** Data Mining, Intrusion Detection Systems (IDS), Network Security, Anomaly Detection, Time probability Method, Hyperbolic Hopfield Neural Network (HHNN)

**Received:** 1 May 2017, Revised 24 June 2017, Accepted 29 June 2017

## 1. Introduction

In internet there are number of online system were developed such as online shopping, foreign currency exchange, internet banking, online bookstore, online sales, online stock, and also online gaming etc., Moreover, as internet open culture, many security system and data also at danger for eternity. A spacious development of internet has to optimistic the data interruption detection which has to develop into an important part of the infrastructure security system for its mechanism. As a result, the accessibility of data which has to be detection using IDS for recognizing a group of nasty actions which treats as threats, privacy data. An anomaly data through IDS using adapt for new attacks by generated by "normal" pattern data discovered on its network traffic. These network systems have to find the anomalies by analyzing incoming datasets with as "normal" data model. For everything that is measured a statistically abnormal is classified as anomalous. It allow the systems to find automatically detect new attacks through normal behaviour. For using two methods for detect anomalous dataset detection on data stream paradigm. To evaluate network traffic, by Den Stream algorithm adapting the clustering on network traffic for its streaming. This algorithm, which can access individual dataset are treated as points and flag as normal or abnormal condition. This algorithm utilize a histogram model which has to evolving on incoming network traffic by using Pearson correlation.

Initially the set of network datasets which can be used to identify the patterns as anomaly and non-anomaly dataset with help of time probability method as well as generation rules and these patterns are trained by Hyperbolic Hopfield neural network.

Hyperbolic Hopfield neural network passes on the Hopfield neural networks. These neural networks have been extensive to complex-valued into Hopfield neural networks. Hyperbolic Hopfield neural network are the easy and most popular model of Hopfield neural network. In exacting, quantized Hopfield neural networks, also passes to as multivolume neural networks, and should applied into multilevel data such as gray-scale images. A numerous extensions to Clifford Hyperbolic Hopfield neural network have been attempted. Isokawa et al. studied Hyperbolic Hopfield neural network using quaternion and also Kuroe constructed hyperbolic neural network type.

## 2. Related Work

Some applications for network intrusion detection of soft computing techniques are explained in this section. Different Genetic programming (GP) and Genetic Algorithm (GA) are used to detect intrusion detection for various kinds in several scenarios. A few uses GA for derive categorization rules [1] [2] [3] [4]. GA is used to select require features and also find the minimal and optimal parameters of a few core function such a several AI methods be use to obtain acquirement of rules [5] [6] [7]. They are various papers related to IDS [8] [9] [10] [11] the network security for level of impact.

Different intrusion-detection prototypes are created for seminal work in 1981. The intrusion detection system contain emerge of computer security area so the complexity of ensure the information systems will free for security flaw [12]. The classification of security flaws are show the computer system attack from security vulnerable not consider of their purpose, origin, manufacturer and also it is difficult for technical and economically costly to make certain computer system and vulnerable to attack for network.

However the system working well usually, selects the fuzzy membership function parameter through the experience it may be lead some false alarm. In this paper use genetic algorithm to mechanically optimize fuzzy-membership function parameters. The main approach of this paper, he define a chromosome to contain the sequence of fuzzy function parameters. To start the process of chromosomes with random initial population where each and every chromosome is feasible set of parameters. This process evolve the population of chromosomes to be come with optimize set of parameters. Continuation of these work proposed prototype (IIDS) Intelligent Intrusion Detection System utilize genetic algorithm and fuzzy mining algorithm.

The highly referred article regarding intrusion detection using neural networks [13]. In this article, he studies about this application of advantages, disadvantages of Neural Network. Conclusion of this article has very suitable of neural network for IDS. Though article is shows that the training of neural network is not small and may be in fact need the substantial effort.

The real time intrusion detection system is able to detect penetration break-ins and further form of computer mistreatment is described [8]. This model is depending on hypothesis then that security violate are used to detect the abnormal patterns from monitor the system audio record of system usage. This model consist profile for represent the performance of subjects through respect to objects of statistical and metrics and also the rules of acquiring knowledge regarding this performance from detecting anomalous performance and audit records.

In 1987, the proposed intrusion detection is an approach used to count networking attacks, computer and misuses [9]. The intrusion detection is implementing by an intrusion detection system. But today here available lot of intrusion detection systems. Generally the lot of this commercial implement is relatively insufficient and ineffective, which gives increase to require for research on additional dynamic intrusion detection systems. Basically an intruder is also defining as system, person or program who tries to legally allow performing an action or successfully breaking into the information system.

In the previous decade different approaches to be developed and future in order to intrusions are detect [7]. In the previous stage, statistical approaches and rule-based expert system are used to intrusions detected. The rule based expert IDS detect intrusion for well known with include high detection rate, other then it is not easy to detecting new intrusion and its name need to updated frequently and manually in database. Statistical based IDS employ different statistical methods such as cluster analysis, multivariate analysis, principal component analysis, frequency, Bayesian analysis, simple significance tests. In this type of IDS wants to collect sufficient data to construct a difficult mathematical model that is unfeasible in case of difficulty network traffic.

Data security and recent technology for computer is generally depend on Access Control List (ACL) methodology (e.g. UNIX-style password authentication), data encryption or monitored environments [14]. In Addition, make use of encryption grow significantly after introduction of public key technology [16] and Data Encryption Standard [15] together the late 70s. In these paper, demonstrate the new security technique depend on monitoring encrypted interchange in order to intrusions detect. But these technique is less accurate of detect intrusion, in our proposed framework, to detect the intrusion datasets by using Hyperbolic Hopfield neural network.

### 2.1 Problems With Existing System

Initially, the information used through the intrusion detection system is obtained from datasets lying on a network. Information has to pass through a longer path from its source to the IDS and in the method can potentially be destroyed by an attacker. Moreover, the intrusion detection system has to suppose the behavior of the system from its data collected, which can result in missed events. This is referred to as reliability problem.

The intrusion detection system always uses extra resources in the system monitoring yet when there is no intrusions occurring as components of the intrusion detection system have to be successively all the time. It is resource usage problem.

These components of the intrusion detection system are implemented as separate programs as susceptible to tampering. An intruder can potentially disable the programs running on a system and to expose the intrusion detection system useless or unreliable problem.

## 3. Proposed Work

The proposed framework utilized a new method called time probability method for detection of patterns and Hyperbolic Hopfield Neural Network for detect of the intrusion dataset. The proposed time probability method discovers the patterns from the datasets based on their occurrence in frequently. The detected patterns are trained with our proposed intrusion detection system which is using Hyperbolic Hopfield neural network machine learning mechanism where those patterns are utilized to detect the testing dataset become a normal or anomaly dataset. The subsection are discussed our proposed frame work in briefly.

### 3.1 Network Datasets

The following environments would be generating the network datasets.

**a) File Access:** Its operations are read, write, delete, and access list and to create directories and files access in an online system.

**b) System Access:** It access login, logout, terminate of user and access password into this category.

**c) Resource Consumption:** It includes CPU, I/O, and MU as usage. It has been presently able to gather this information on a preprocess basis, though it is obtainable merely after a process has terminated. It is probable to find this information at arbitrary interval from an executing process; however, this would absorb more program effort and can boost the load on the object system, as it requires polling the process and scanning various kernel data structures with all polls. For these reasons, they haven’t tried to obtain resource consumption information for executing process.

**d) Process Creation/Command Invocation:** It indicates the design process and this information is frequently accessible after invoking a command.

Anomalies can easily hacking the data from network communication. So identify these types of anomalies from the malicious network data using our proposed framework.

### 3.2 Pattern Discover: Time Probability

The anomaly patterns are generated with following method.

#### 3.2.1 Time Probability Method

Pattern mining, a well-studied and important issue in data mining field, is for discovering frequent subsequences as patterns in a network dataset database. Patterns discover is an important problem for many applications as also in our intrusion detection system for more reliable datasets has needed to detect the intrusion dataset from the incoming network datasets . A lot of efforts have been devoted to developing efficient algorithms for searching frequent patterns.

As a result, the time probability method utilized to discover more reliable patterns based on time duration of received dataset. Also, this method utilized the following generation rules to generate the more reliable patterns.

### 3.2.2 Some rules to generate the pattern

Discard the size of file as below 1MB. (Applies to all size)

Discard if any of the protocol UDP. (Applies to all protocols)

Keep the protocol is (HTTP, TCP/IP). (Applies to all protocols)

Discard if any of the IP addresses same. (Applies to all sender IP address)

Keep transaction rate is above 20 kbps. (Applies to all transaction rate)

Discard if any of the filename same. (Applies to all filename)

Keep total time of sending 10-11 minutes. (Applies to all transaction)

### 3.3 Algorithm

**Input:** Network Dataset, Generation Rules

**Output:** Anomaly patterns

**Method**

Obtain all frequent datasets

Compute arrival rate of frequent dataset

Arrival rate of frequent dataset ($\lambda$) = (Number of occurrence) / (Frequent time occurrence with remain dataset- First occurrence time of frequent dataset)

Also check with generation rules

Compute expected same data dataset

Expected possibility = 1-e$\lambda$ td

*td* - is the expected time

## 4. Anomaly Detection: Hyperbolic Hopfield Neural Network

An anomaly detection will assume that an intrusion always return a number of deviations form normal patterns. An anomaly detection can be classified into static and dynamic. A static anomaly detector depends on the statement ie. a portion of system being monitored that should stay regular. frequently, static detectors only address of the software.

Segment of a system is based on the assumption of hardware configure didn't checked. The static segment of a system code and the constant portion of data has to be depending on the exact performance of the system. The static segment of the system can be represented as binary bit string such as files. If the static segment of the system always deviates from its original shape, an error has occurred has changed the static segment of the system. Static anomaly detectors are meant for its checking the data integrity.

Dynamic anomaly detectors involve meaning of performance to categorize as normal or anomalous. Repetitively, system designers utilize the perception of event. Performance is defined as a sequence of discrete actions that reason events which can be recorded in audit report. Since audit report of operating system only proof events of interest, then only performance that can be experimental this result in an event in an audit record. Events may occur in a series. In various cases like distributed systems, incomplete ordering of actions is adequate. In further cases, the order isn't directly represented; only cumulative information such as cumulative processor resource used during a time gap, is maintained. In this case, thresholds are defined into a split normal from anomalous resource consumption.

### 4.1 Hyperbolic Hopfield Neural Network

It is an algorithmic technique used to first study the relationship between the two sets of information, and then "generalize" to get new input-output pairs in a reasonable way. Neural networks can be hypothetically used in knowledge-based intrusion-detection systems to identify the attacks and look for them in the audit stream. However, as there is presently no reliable method to realize what triggered the association; the neural network cannot clarify the reasoning that led to the classification of the attack. In intrusion detection system, neural networks contain mainly used in the performance of the system. Correspondence between the neural network models and statistics model has been demonstrated in the network. The benefit of using neural networks relatively than statistics lies in having an easy method to say nonlinear relations between variables, and retraining the neural network automatically. However, these neural networks are still a computational intensive technique, and aren't broadly used by the intrusion detection community. However, our proposed neural network is testing the intrusion network datasets very efficiently.

The following Hyperbolic Hopfield Neural Network Structure is explained in below figure 1. An input of hyperbolic neurons is *R*. Outputs of hyperbolic neurons are hyperbolic numbers in *S*. For an input *z*, a hyperbolic neuron by activation function $g(z)$ produces the output $g(z)$. For $z = p1 + q1 \in R$ and it term the activation function $g(z)$ as follows:

$$g(z) = \Sigma\,(p1 + p2 +, \ldots, + pn, q1 + q2 +, \ldots + qn)$$

Next its call this activation function $g(z)$ the hyperbolic activation function. As it represent $z = r\,exp$ in hyperbolic polar form, and write $g(z)$ as output. From a geometrical point of view, $g(z)$ is the intersection of curve *S* and the line passing through the source and point *z* on the hyperbolic plane.

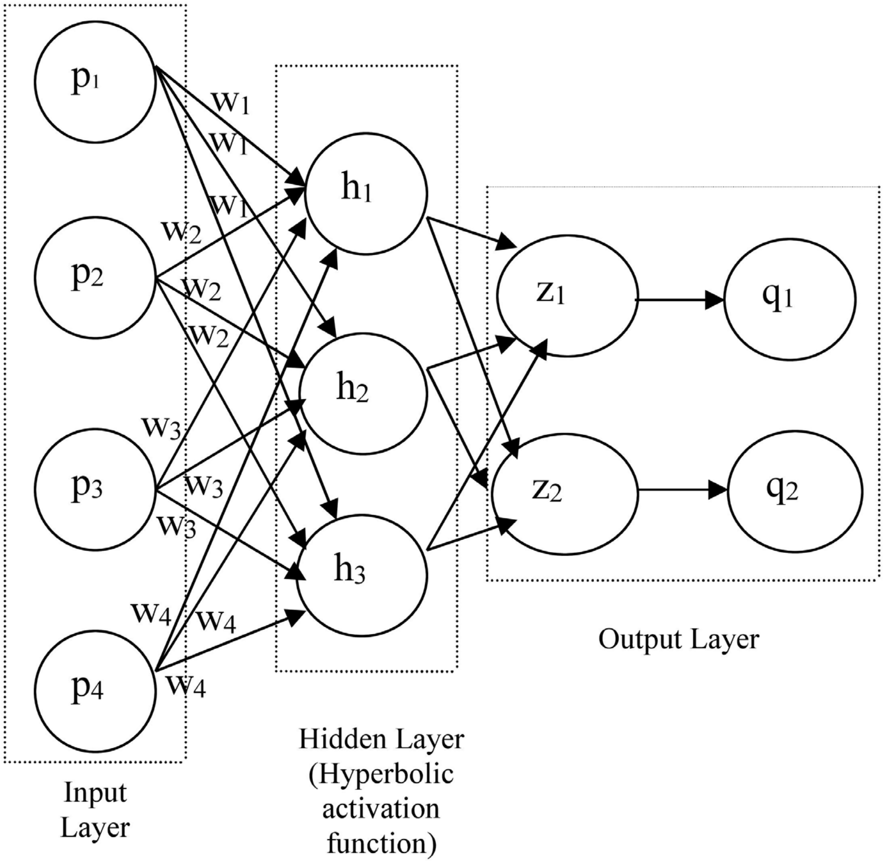


Figure 1. Hyperbolic Hopfield Neural Network Structure

All inputs are joined into a distinct number, *z*, using the following weighted amount:

$$Z = \sum_{i=1}^{m} w_i p_i + \mu$$

Where *m* is the number of inputs and *wi* is the weight connected with the $i^{th}$ input (attribute) *pi*. The term m calculation is referred to numbers of input layer as bias terms. In geometric terminology it may be referred to as the intercept. The weights and bias terms calculation are expected during network training. The neural doesn't respond to its inputs unless *z* is greater than zero. If *z* is greater than zero then the output from this neural is set to 1. If *z* is less than or equal to zero then the output is zero as follows:

$$y = \begin{cases} 1 \text{ if } z > 0 \\ 0 \text{ if } z \leq 0 \end{cases}$$

Where *y* is neural output. The function used for this calculation is referred to as the activation function as shown in Figure 2 below.

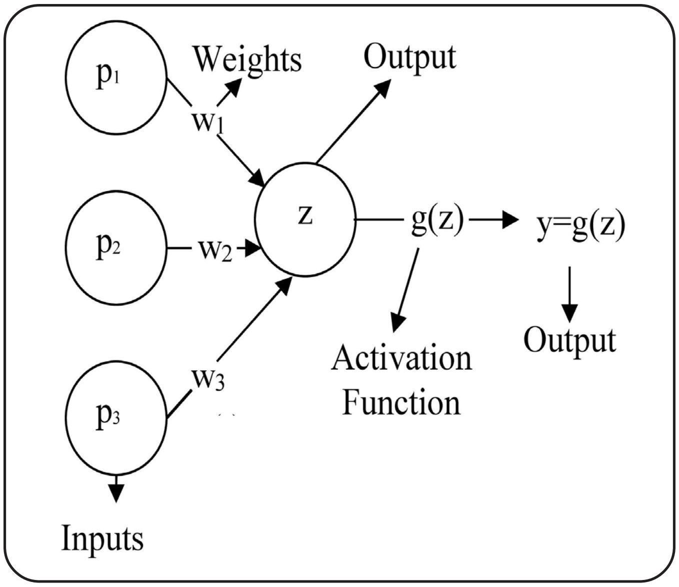


Figure 2. Neural Networks as Activation Function Perceptron

As shown above Figure 2 consist of the following five components:

1. Inputs is $p_1$, $p_2$, and $p_3$,

2. Input Weights – $w_1$, $w_2$, and $w_3$,

3. Potential $Z = \sum_{i=1}^{m} w_i p_i + \mu$, where $\mu$ is a bias correction,

4. Activation Function – $g$ ($z$), and

5. Output is $y = g$ ($z$).

An input can be either the original raw data inputs or the output from another perceptron. The primary purpose of neural network is training to estimate the weights associated with each perceptron's potential. The activation function maps this potential to the perceptron's output.

Initially, it defines connection weights. All neuron networks are connected through all other neurons. It denotes the connection weight from neuron j$^{th}$ to neuron k$^{th}$ by w$_{kj}$. The connection weights should be satisfying the following conditions:

1) $w_{kj} \epsilon R$;

2) $w_{kj} = w_{jk}$.

It denotes the output of neuron $k^{th}$ by *zk* and the weighted sum input k$^{th}$ to neuron k as follows:

$$g(z) = \sum_{j=k} w_{kj} z_j$$

Thus, *Ik* belongs to *R*. Neuron *k* receives the weighted sum input $w_{k,} p_{i,} q_i$ and produces *g* (*z*) as output.

**5. Results and Discussion**

The effectiveness and efficiencies of the proposed method are investigated using KDD network datasets. The features of the proposed method are summarized as follows time probability method with rule-based analysis. The network datasets are used to identify the patterns as anomaly and non-anomaly with help of these time probability method as well as generation rule. Then, these patterns are trained by our proposed neural network method. The intrusion detection can be flexibly applied to anomaly detection with specific designed classifiers with of these patterns.

**5.1 Experimental Result**

| S.No | Attributes |
|---|---|
| 1. | KDD Train |
| 2. | Protocol type |
| 3. | service |
| 4. | flag |
| 5. | src_bytes |

Table 1. Fragmentation of Attributes from the datasets

The table 1 shown the partial attributes names which are obtained from network datasets to detects normal or anomaly data based these attribute values. The Hyperbolic networks using hyperbolic tangent activation function with these attribute values for training the intrusion detection system. However, the entire datasets could not be trained by the intrusion detection system so, the system training based on the discovered patterns that are obtained by the frequent occurrence in the network communication.

**5.2 Pattern Discovered**

The following table 5.2 shown fragmentations of KDD datasets that are used in reliable patterns and trained into intrusion detection system.

The table 3 shows the partial discovered patterns. The arrival rate of every pattern is shown the pattern when it was occurred and the number of times in the network communication. Also, the Expected Time Probability indicates the future occurrence possibility for every pattern where those values are obtained by 7 minutes i.e. those patterns may occur every 7 minutes.

| ID | Duration | Flag | Src_byte | Dst_byte | Classification |
|---|---|---|---|---|---|
| 1. | 81 | 18 | 522 | 0 | Normal |
| 2. | 12 | 61 | 0 | 0 | Normal |
| 3. | 22 | 334 | 0 | 0 | Anomaly |
| 4. | 65 | 184 | 520 | 0 | Normal |
| 5. | 45 | 47 | 0 | 0 | Normal |
| 6. | 66 | 28 | 0 | 0 | Anomaly |
| 7. | 78 | 132 | 18 | 0 | Normal |
| 8. | 45 | 454 | 0 | 0 | Anomaly |
| 9. | 74 | 58 | 89 | 0 | Normal |
| 10 | 35 | 1 | 0 | 0 | Anomaly |

Table 2. shows the fragmentation of datasets that are used to identify the reliable patterns and to train the intrusion detection system

| ID | Arrival Rate | Expected Time Possibility |
|---|---|---|
| 1.0 | 0.02073 | 0.1562 |
| 2.0 | 0.00154 | 0.01893 |
| 3.0 | 0.05941 | 0.51574 |
| 4.0 | 0.08855 | 0.85861 |
| 5.0 | 0.04138 | 0.33595 |

Table 3. Fragmentation of pattern discovered from the datasets with Arrival rate and Time probability

### 5.3 Testing

In our proposed method, the network datasets (patterns) are using to train the hyperbolic Hopfield neural network. For example of trained network dataset is mention above Table 3. These type of datasets finally detect the receiving dataset normal or anomaly that are as shown in the below Table 5.

The proposed neural network is testing the network datasets using hyperbolic function. For example of test network datasets is given below Table 4. In testing process, the incoming dataset compared with the trained datasets using activation function then it will decided normal or anomaly dataset.

### 5.4 Discussion

In our proposed intrusion detection system was used the Hyperbolic Hopfield Neural Network with trained anomaly patterns. The patterns are generated by time probability based method with generation rules. These patterns are used to find the anomaly occurrence in network datasets. The Network dataset vs Intrusion occurring rate the dataset pattern using time probability, the proposed system is handle pattern discover frequently that is occurred on the datasets this system uses Hyperbolic Hopfield neural network model to pattern the different dataset of flow in the dataset patterns. The proposed system is more robust and flexible. A prototype of the proposed method and demonstrated that the search time is reduced maxima. The proposed system extends the time based probability method and security over the heterogeneous network in the industrial detection automation system.

| ID | Duration | Flag | Src_byte | Dst_byte |
|---|---|---|---|---|
| 1. | 10 | SF | 491 | 0 |
| 2. | 22 | 334 | 0 | 0 |
| 3. | 56 | 146 | 0 | 0 |
| 4. | 78 | 199 | 420 | 0 |
| 5. | 66 | 28 | 0 | 0 |
| 6. | 98 | 233 | 616 | 0 |
| 7. | 569 | 147 | 105 | 0 |
| 8. | 45 | RSTR | 0 | 0 |
| 9. | 87 | 255 | 861 | 0 |
| 10 | 35 | 1 | 0 | 0 |

Table 4. Fragmentation of Testing from datasets

| ID | Classification |
|---|---|
| 1. | Normal |
| 2. | Anomaly |
| 3. | Normal |
| 4. | Normal |
| 5. | Anomaly |
| 6. | Normal |
| 7. | Normal |
| 8. | Anomaly |
| 9. | Normal |
| 10 | Anomaly |

Table 5. Fragmentation of Detected Normal and Anomaly Dataset

The below Figure 3 shown the network datasets vs. intrusion occurring rate. The intrusion occurring rate is depends upon the network datasets size. The intrusion detection rate is gradually increased while increasing the testing data set because of trained data sets more reliable and suitable to detect all anomaly and normal datasets.

The Figure 4 shown the dataset size vs. discovered pattern rate where pattern size is increased when various types of training data sets frequently occurs. Moreover, these patterns are very helpful to detect the normal and anomaly datasets that are in testing datasets. Also, the time probability is increased while increasing the arrival rate.

The Figure 5 had shown the anomaly detection accuracy rate from the testing dataset where anomaly detection rate is increased when increasing the testing data is increased because of every testing time the system trained with new patterns.

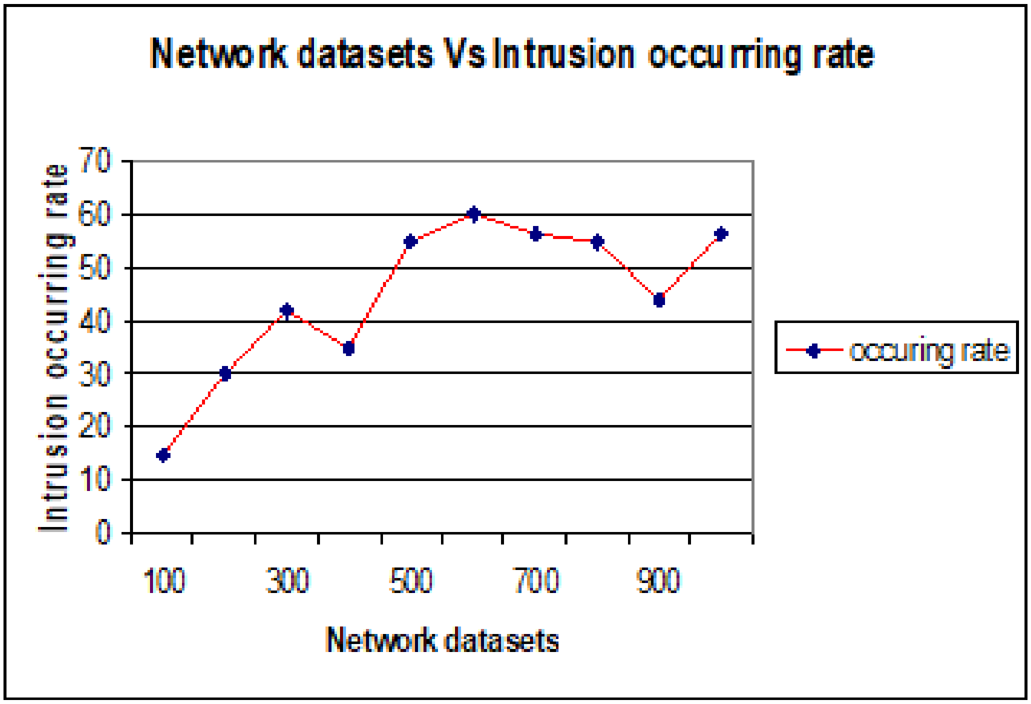


Figure 3. Network datasets Vs Intrusion Occurring Rate

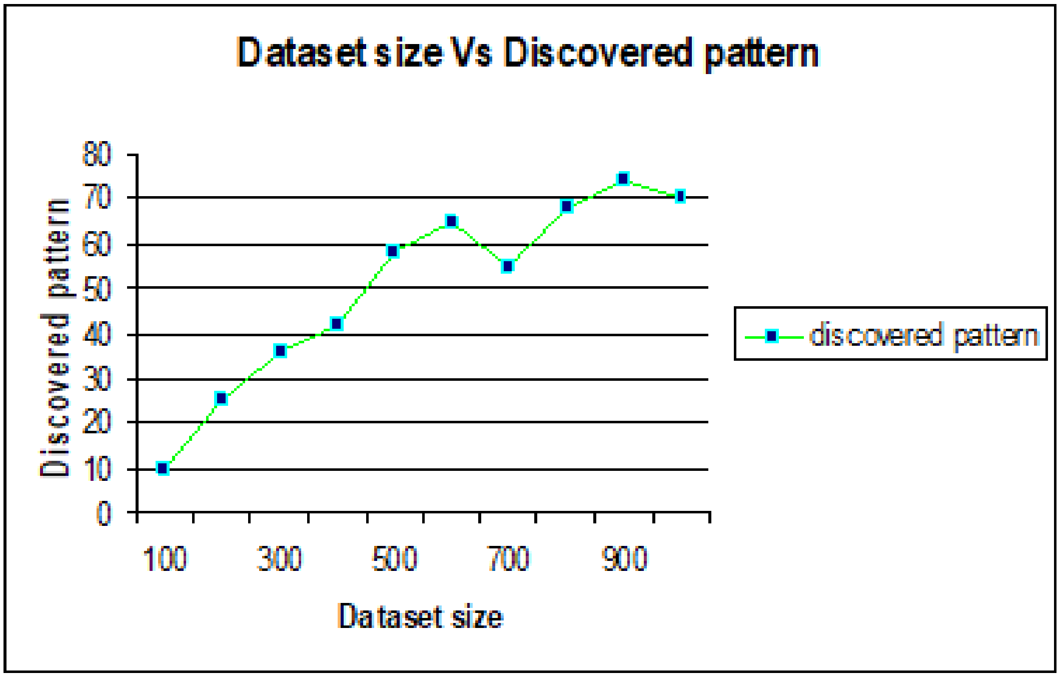


Figure 4. Dataset Size Vs Discovered Pattern

In previous techniques, to detect the anomaly datasets using genetic algorithms are extensively high but our proposed work more efficient by using Hyperbolic Hopfield Neural Network.

The performance of the proposed frame work will increase the detection of anomaly from the testing datasets when compared with other existing techniques. The Figure 6 shows the proposed work with existing work comparison where our proposed work detects the maximum anomaly from the given testing dataset.

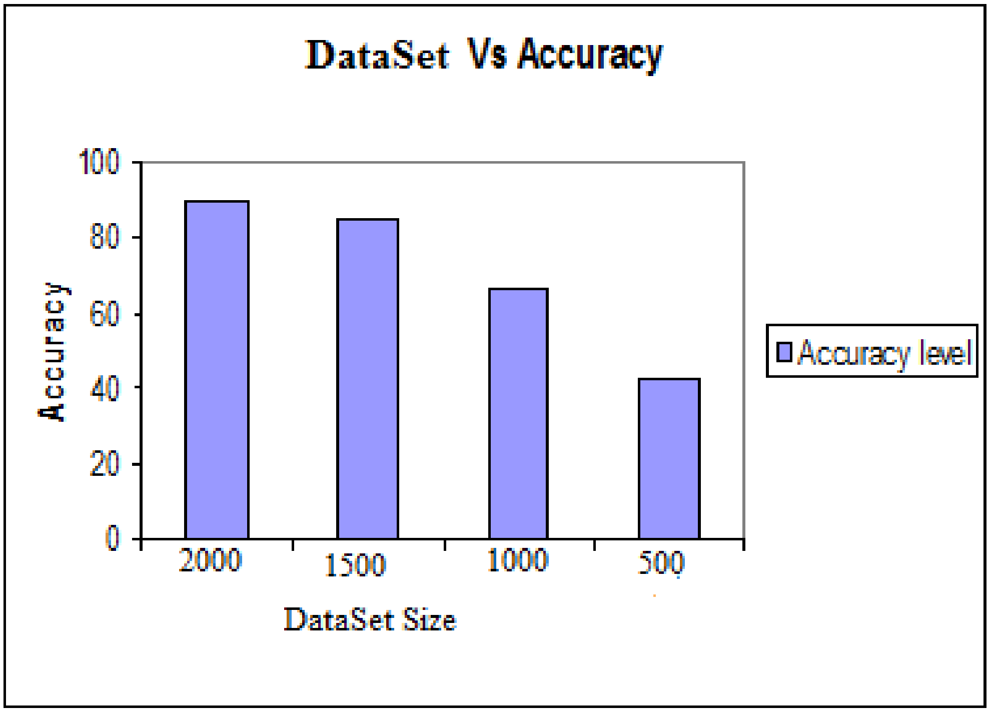


Figure 5. Datasets vs. Accuracy

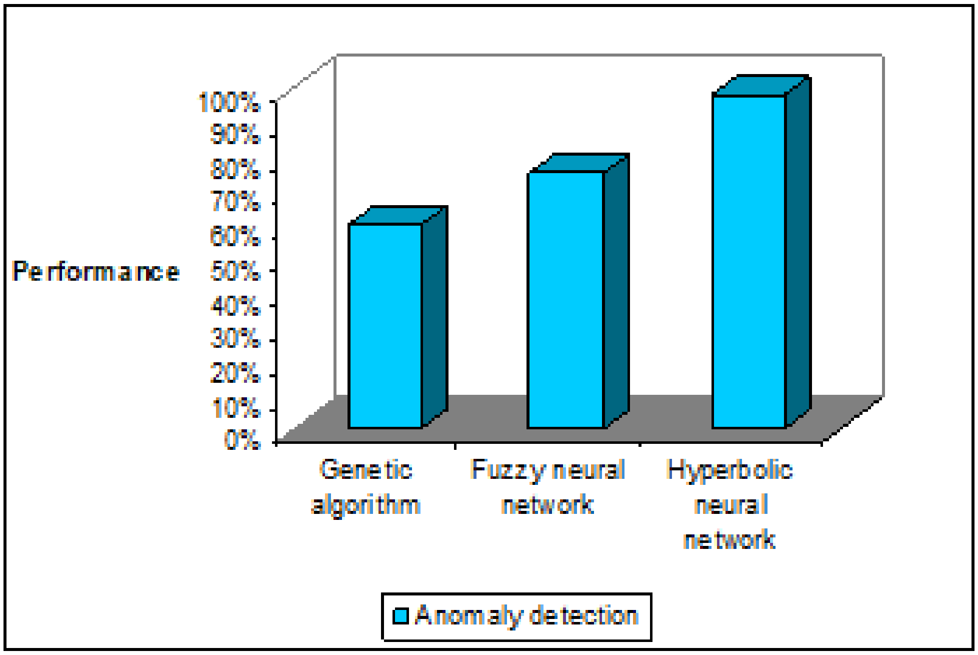
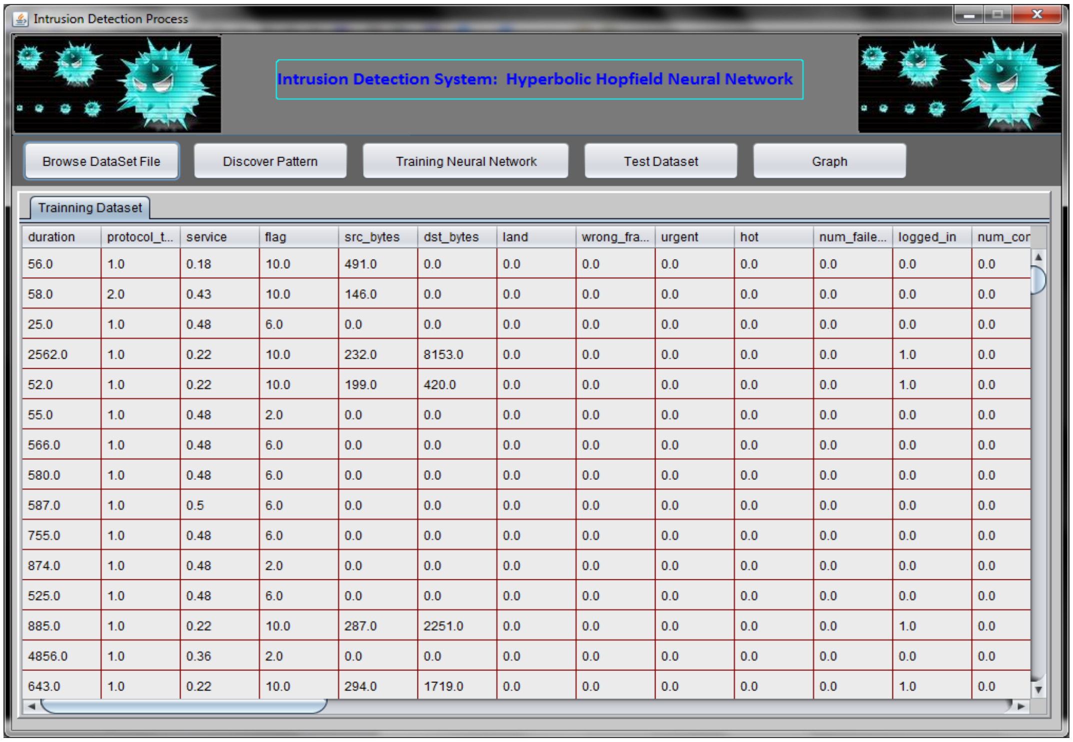


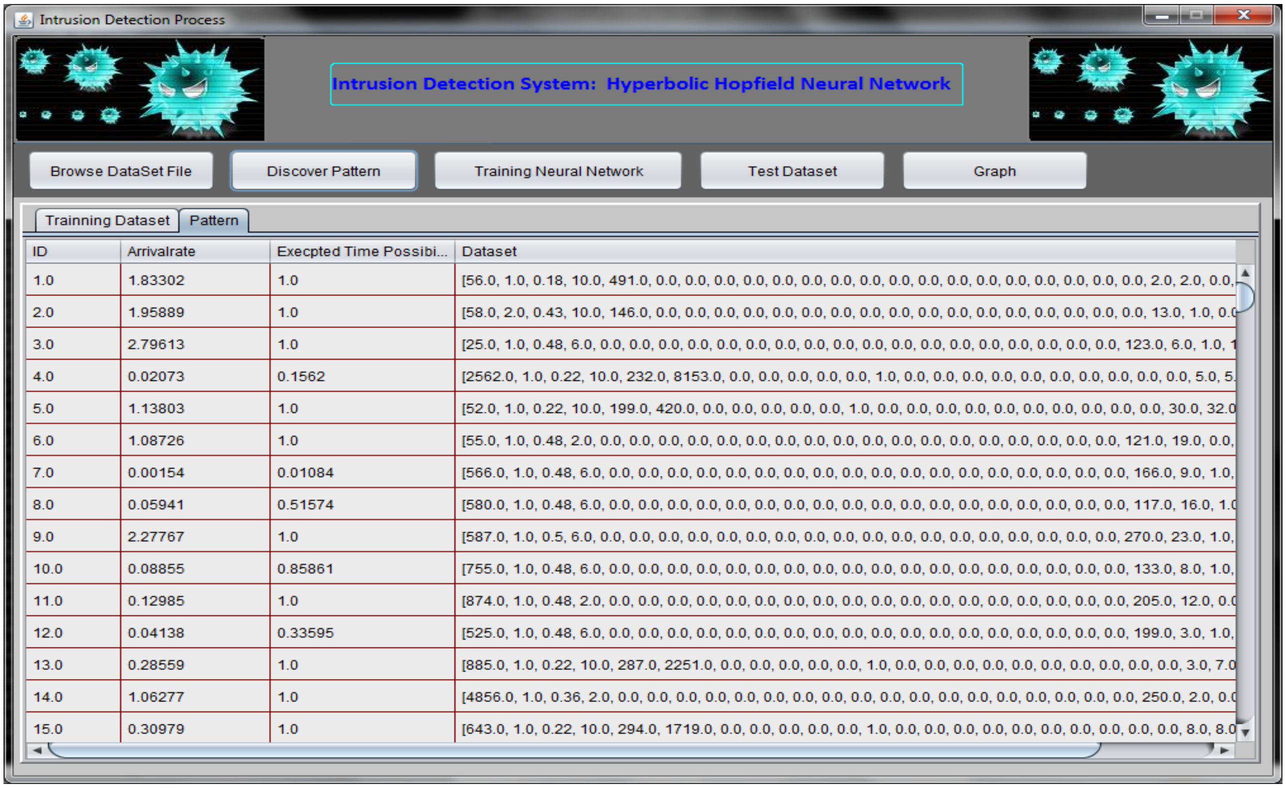
Figure 6. Comparison of Anomaly detection: Hyperbolic Hopfield Neural Network vs. Existing Techniques

### 5.4.1 Implementation Results



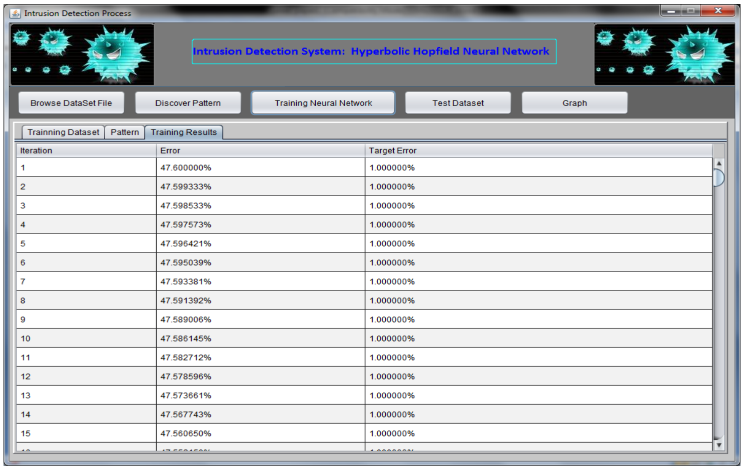
| duration | protocol_t... | service | flag | src_bytes | dst_bytes | land | wrong_fra... | urgent | hot | num_faile... | logged_in | num_cor |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 56.0 | 1.0 | 0.18 | 10.0 | 491.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 58.0 | 2.0 | 0.43 | 10.0 | 146.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 25.0 | 1.0 | 0.48 | 6.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 2562.0 | 1.0 | 0.22 | 10.0 | 232.0 | 8153.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| 52.0 | 1.0 | 0.22 | 10.0 | 199.0 | 420.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| 55.0 | 1.0 | 0.48 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 566.0 | 1.0 | 0.48 | 6.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 580.0 | 1.0 | 0.48 | 6.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 587.0 | 1.0 | 0.5 | 6.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 755.0 | 1.0 | 0.48 | 6.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 874.0 | 1.0 | 0.48 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 525.0 | 1.0 | 0.48 | 6.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 885.0 | 1.0 | 0.22 | 10.0 | 287.0 | 2251.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| 4856.0 | 1.0 | 0.36 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 643.0 | 1.0 | 0.22 | 10.0 | 294.0 | 1719.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 |

Figure 7. Training Datasets



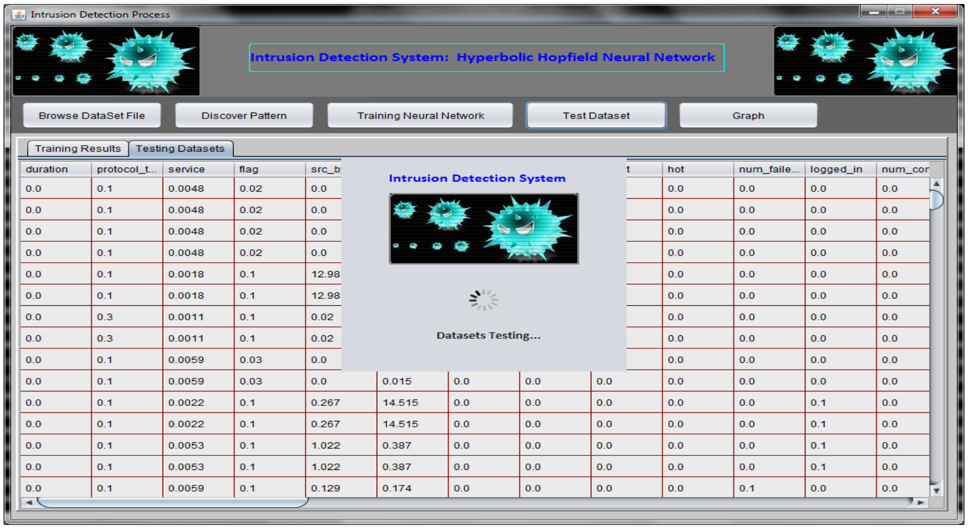
| ID | Arrivalrate | Execpted Time Possibi... | Dataset |
|---|---|---|---|
| 1.0 | 1.83302 | 1.0 | [56.0, 1.0, 0.18, 10.0, 491.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2.0, 2.0, 0.0, |
| 2.0 | 1.95889 | 1.0 | [58.0, 2.0, 0.43, 10.0, 146.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 13.0, 1.0, 0.0 |
| 3.0 | 2.79613 | 1.0 | [25.0, 1.0, 0.48, 6.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 123.0, 6.0, 1.0, 1 |
| 4.0 | 0.02073 | 0.1562 | [2562.0, 1.0, 0.22, 10.0, 232.0, 8153.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 5.0, 5. |
| 5.0 | 1.13803 | 1.0 | [52.0, 1.0, 0.22, 10.0, 199.0, 420.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 30.0, 32.0 |
| 6.0 | 1.08726 | 1.0 | [55.0, 1.0, 0.48, 2.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 121.0, 19.0, 0.0, |
| 7.0 | 0.00154 | 0.01084 | [566.0, 1.0, 0.48, 6.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 166.0, 9.0, 1.0, |
| 8.0 | 0.05941 | 0.51574 | [580.0, 1.0, 0.48, 6.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 117.0, 16.0, 1.0 |
| 9.0 | 2.27767 | 1.0 | [587.0, 1.0, 0.5, 6.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 270.0, 23.0, 1.0, |
| 10.0 | 0.08855 | 0.85861 | [755.0, 1.0, 0.48, 6.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 133.0, 8.0, 1.0, |
| 11.0 | 0.12985 | 1.0 | [874.0, 1.0, 0.48, 2.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 205.0, 12.0, 0.0 |
| 12.0 | 0.04138 | 0.33595 | [525.0, 1.0, 0.48, 6.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 199.0, 3.0, 1.0, |
| 13.0 | 0.28559 | 1.0 | [885.0, 1.0, 0.22, 10.0, 287.0, 2251.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3.0, 7.0 |
| 14.0 | 1.06277 | 1.0 | [4856.0, 1.0, 0.36, 2.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 250.0, 2.0, 0.0 |
| 15.0 | 0.30979 | 1.0 | [643.0, 1.0, 0.22, 10.0, 294.0, 1719.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 8.0, 8.0 |

Figure 8. Discovered Patterns with Expected Probability



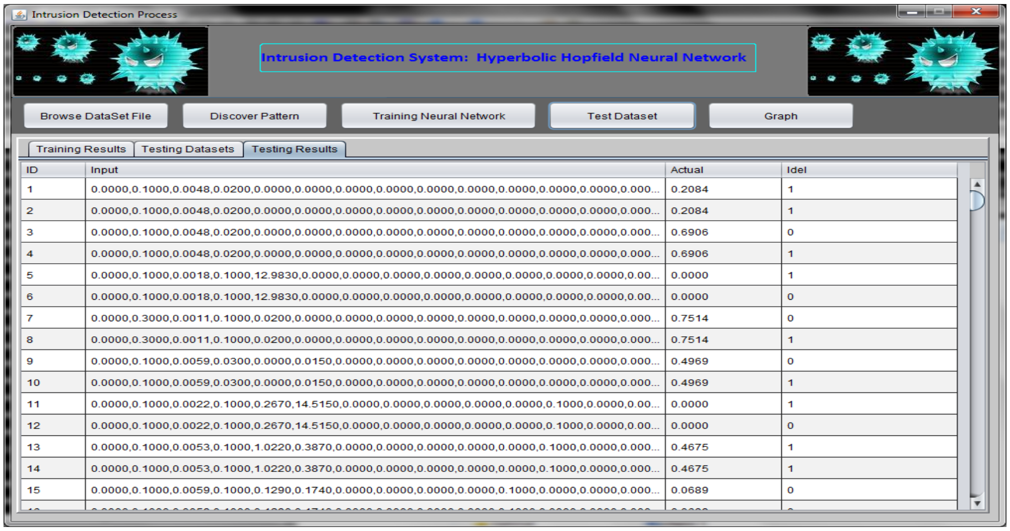
| Iteration | Error | Target Error |
|---|---|---|
| 1 | 47.600000% | 1.000000% |
| 2 | 47.599333% | 1.000000% |
| 3 | 47.598533% | 1.000000% |
| 4 | 47.597573% | 1.000000% |
| 5 | 47.596421% | 1.000000% |
| 6 | 47.595039% | 1.000000% |
| 7 | 47.593381% | 1.000000% |
| 8 | 47.591392% | 1.000000% |
| 9 | 47.589006% | 1.000000% |
| 10 | 47.586145% | 1.000000% |
| 11 | 47.582712% | 1.000000% |
| 12 | 47.578596% | 1.000000% |
| 13 | 47.573661% | 1.000000% |
| 14 | 47.567743% | 1.000000% |
| 15 | 47.560650% | 1.000000% |

Figure 9. HHNN Training Results



| duration | protocol_t... | service | flag | src_b |  |  |  | t | hot | num_faile... | logged_in | num_cor |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.0 | 0.1 | 0.0048 | 0.02 | 0.0 |  |  |  |  | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.1 | 0.0048 | 0.02 | 0.0 |  |  |  |  | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.1 | 0.0048 | 0.02 | 0.0 |  |  |  |  | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.1 | 0.0048 | 0.02 | 0.0 |  |  |  |  | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.1 | 0.0018 | 0.1 | 12.98 |  |  |  |  | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.1 | 0.0018 | 0.1 | 12.98 |  |  |  |  | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.3 | 0.0011 | 0.1 | 0.02 |  |  |  |  | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.3 | 0.0011 | 0.1 | 0.02 |  |  |  |  | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.1 | 0.0059 | 0.03 | 0.0 |  |  |  |  | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.1 | 0.0059 | 0.03 | 0.0 | 0.015 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.1 | 0.0022 | 0.1 | 0.267 | 14.515 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 0.0 |
| 0.0 | 0.1 | 0.0022 | 0.1 | 0.267 | 14.515 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 0.0 |
| 0.0 | 0.1 | 0.0053 | 0.1 | 1.022 | 0.387 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 0.0 |
| 0.0 | 0.1 | 0.0053 | 0.1 | 1.022 | 0.387 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 0.0 |
| 0.0 | 0.1 | 0.0059 | 0.1 | 0.129 | 0.174 | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 0.0 | 0.0 |

Figure 10. HHNN Testing Datasets



| ID | Input | Actual | Idel |
|---|---|---|---|
| 1 | 0.0000,0.1000,0.0048,0.0200,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.000... | 0.2084 | 1 |
| 2 | 0.0000,0.1000,0.0048,0.0200,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.000... | 0.2084 | 1 |
| 3 | 0.0000,0.1000,0.0048,0.0200,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.000... | 0.6906 | 0 |
| 4 | 0.0000,0.1000,0.0048,0.0200,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.000... | 0.6906 | 1 |
| 5 | 0.0000,0.1000,0.0018,0.1000,12.9830,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.00... | 0.0000 | 1 |
| 6 | 0.0000,0.1000,0.0018,0.1000,12.9830,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.00... | 0.0000 | 0 |
| 7 | 0.0000,0.3000,0.0011,0.1000,0.0200,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.000... | 0.7514 | 0 |
| 8 | 0.0000,0.3000,0.0011,0.1000,0.0200,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.000... | 0.7514 | 1 |
| 9 | 0.0000,0.1000,0.0059,0.0300,0.0000,0.0150,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.000... | 0.4969 | 0 |
| 10 | 0.0000,0.1000,0.0059,0.0300,0.0000,0.0150,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.000... | 0.4969 | 1 |
| 11 | 0.0000,0.1000,0.0022,0.1000,0.2670,14.5150,0.0000,0.0000,0.0000,0.0000,0.0000,0.1000,0.0000,0.00... | 0.0000 | 1 |
| 12 | 0.0000,0.1000,0.0022,0.1000,0.2670,14.5150,0.0000,0.0000,0.0000,0.0000,0.0000,0.1000,0.0000,0.00... | 0.0000 | 0 |
| 13 | 0.0000,0.1000,0.0053,0.1000,1.0220,0.3870,0.0000,0.0000,0.0000,0.0000,0.0000,0.1000,0.0000,0.000... | 0.4675 | 1 |
| 14 | 0.0000,0.1000,0.0053,0.1000,1.0220,0.3870,0.0000,0.0000,0.0000,0.0000,0.0000,0.1000,0.0000,0.000... | 0.4675 | 1 |
| 15 | 0.0000,0.1000,0.0059,0.1000,0.1290,0.1740,0.0000,0.0000,0.0000,0.0000,0.1000,0.0000,0.0000,0.000... | 0.0689 | 0 |

Figure 11. HHNN Testing Results (Ideal :1-Anomaly, 0-Normal)

## 6. Conclusion

This paper is designed and implemented a Knowledge-Based Intrusion Detection framework to detect the attacks on secure communication. The proposed intrusion detection system was used a Hyperbolic Hopfield Neural Network with well-formed anomaly patterns that are trained the proposed neural network. Also, these reliable patterns were generated by our proposed time probability method with generation rules. These reliable patterns are also given the time duration information of future anomaly occurrence. The implemented results are shown the proposed work identify the maximum (100%) anomaly datasets from the tested network datasets. It also takes less execution time and less storage for detects the intrusion attacked datasets. In future, the different types of pattern detection method can be used to detect the maximum reliable patterns.

## References

[1] Chittur, A. (2005). Model Generation for an Intrusion Detection System Using Genetic Algorithms. January.

[2] Li, W. (2004). Using Genetic Algorithm for Network Intrusion Detection. A Genetic Algorithm Approach to Network Intrusion Detection. SANS Institute, USA.

[3] Lu, W., Traore, I. (2004). Detecting New Forms of Network Intrusion Using Genetic Programming. *Computational Intelligence*, 20, p. 3, Blackwell Publishing, *Malden*, p. 475-494.

[4] Pillai, M. M., Eloff, J. H. P., Venter, H. S. (2004). An Approach to Implement a Network Intrusion Detection System using Genetic Algorithms, *In:* Proceedings of SAICSIT, p. 221-228.

[5] Bridges, S. M., Vaughn, R. B. (2000). Fuzzy Data Mining And Genetic Algorithms Applied To Intrusion Detection, *In:* Proceedings of 12th Annual Canadian Information Technology Security Symposium, p. 109-122.

[6] Gomez, J., Dasgupta, D. (2002). Evolving Fuzzy Classifiers for Intrusion Detection, *In:* Proceedings of the IEEE.

[7] Middlemiss, M., Dick, G., Feature selection of intrusion detection data using a hybrid genetic algorithm/KNN approach, Design and application of hybrid intelligent systems, IOS Press Amsterdam, p. 519-527, 2003.

[8] Srinivas Mukkamala., Andrew, H., Sung., Ajith Abraham. (2005). Intrusion detection using an ensemble of intelligent paradigms, *Journal of Network and Computer Applications*, 28 (2) April 2005, p.167-182.

[9] Peddabachigari, S., Ajith Abraham, Grosan, C., Thomas, J. (2007). Modeling intrusion detection system using hybrid intelligent systems, *Journal of Network and Computer Applications*, 30 (1) January, p. 114–132.

[10] Saniee Abadeh,M., Habibi, J., Lucas, C. (2007). Intrusion detection using a fuzzy genetics-based learning algorithm, *Journal of Network and Computer Applications*, 30 (1) January 2007, p. 414-428

[11] Tao Peng, C. Leckie, Kotagiri Ramamohanarao, Information sharing for distributed intrusion detection systems, *Journal of Network and Computer Applications*, 30 (3) August 2007, p. 877–899.

[12] Carl E. Landwehr, Alan R. Bull, John P. McDermott, and William S. Choi. (1994). A taxonomy of computer program security flaws. *ACM Computing Surveys*, 26 (3) 211–254, September.

[13] Cannady, J. (1998). Artificial neural networks for misuse detection. *In:* Proceedings of the 1998 National Information Systems Security Conference (NISSC'98) October 5-8 1998. Arlington, VA., p. 443–456.

[14] Dorothy E. Denning. An Intrusion-Detection Model, From 1986 IEEE computer Society Symposium on Research in Security and Privacy.

[15] National Bureau of Standards (NBS). Data Encryption Standard. Dederal Information Processing Standard, Publication 46, NBS, Washington, D.C., January.

[16] Rivest, R. L., Shamir, A., Adleman, L. M. (1978). A Method for Obtaining Digital Signatures and Public Key Cryptosystems, CACM, 21 (2), Feburary 1978, p. 120-126.