Kernelized Extreme Learning Machine with Levenberg-Marquardt Learning Approach towards Intrusion Detection

V. Jaiganesh
Doctoral Research Scholar
Department of Computer Science
Manonmaniam Sundaranar University
Tirunelveli, Tamil Nadu, India

P. Sumathi, PhD.
Doctoral Research Supervisor,
Assistant Professor, Post Graduate & Research
Department of Computer Science,
Government Arts College, Coimbatore,
Tamil Nadu, India.

ABSTRACT

Network and system security is of vital importance in the present data communication environment. Hackers and intruders can create many successful attempts to cause the crash of the networks and web services by unauthorized intrusion. New threats and associated solutions to prevent these threats are emerging together with the secured system evolution. Intrusion Detection Systems (IDS) are one of these solutions. The main function of Intrusion Detection System is to protect the resources from threats. It analyzes and predicts the behaviours of users, and then these behaviours will be considered an attack or a normal behaviour. There are several techniques which exist at present to provide more security to the network, but most of these techniques are static. On the other hand, intrusion detection is a dynamic one, which can give dynamic protection to the network security by observing the attack. In recent times, Extreme Learning Machine (ELM) has been extensively applied to provide potential solutions for the IDS problem. But, the practicability of ELM is affected because of the complexity in choosing the suitable ELM parameters. Hence, in this paper sigma (σ) of the radial basis kernel function is tuned using Levenberg-Marquardt (LM) learning and proposed kernelized Extreme Learning Machine with LM. In order to obtain a converged solution, LM learning is utilized. The experiment is carried out with the help of WEKA by using KDD Cup 1999 dataset and the results indicate that the proposed technique can achieve higher detection rate, very low false alarm rate and to achieve high accuracy than the regular ELM algorithms. This method is used to decrease the space densisty of the data.

Keywords

Intrusion Detection System (IDS), Extreme Learning Machine (ELM), Radial Basis Function (RBF), Kernel Function, Leave-One-Out (LOO), Kernel Partial Least Squares (K-PLS)

1. INTRODUCTION

NFORMATION security is a matter of serious worldwide concern as the incredible development in connectivity and accessibility to the internet has generated a tremendous security threat to information systems worldwide [1]. Security is turning out to be a serious issue as the internet applications are developing rapidly. The majority of current security system mostly concentrates on encryption, firewall and access control. However all these approaches cannot promise perfect security, hence the security of a system can be improved by the introduction of intrusion detection system. The capability of an IDS in categorizing a wide range of intrusions in real time with correct results is more significant. The patterns of user behavior and inspection records are observed and the intrusions are traced [2].

Intrusion detection attacks are segmented into two groups,

- Host-based attacks [3-5] and
- Network-based attacks [6, 7].

In case of host-based attacks, the intruders aim at a particular machine and attempt to get access to privileged services or resources on that particular machine. Detection of these kind of attacks typically uses routines that acquire system call data from an audit-process which monitors all system calls made with the support of each user. In case of network-based attacks, it is extremely complicated for legitimate users to use various network services by purposely occupying or disrupting network resources and services. Intruders attack these systems by transmitting huge amounts of network traffic, utilizing familiar faults in networking services, overloading network hosts, etc. Detection of these kind of attacks uses network traffic data (i.e., tcpdump) to look at traffic addressed to the machines being monitored.

Several intrusion detection systems are available and they do not meet the challenges of a susceptible internet atmosphere [8, 9]. In the present scenario, an IDS is much essential for a modern computer system. Intrusion detection technologies can be categorized into two major groups:

- Misuse detection and
- Anomaly detection.

A misuse detection system traces intrusion activities that follow recognized patterns. These patterns explain a suspect collection of sequences of activities or operations that can possibly be dangerous. The major drawback of this detection is that it doesn’t have the capability to trace or detect new kind of intrusions, i.e., certain events that have never occured in the past. An anomaly detection system examines event data and identifies pattern of activities that appear to be ordinary. If an event lies outside of the patterns, it is considered as a possible intrusion [18].

But these techniques are not adequate to detect the new kind of attacks. Hence, at present Support Vector Machines (ELMs) are used in IDS. ELM is a machine learning technique that plots the training vectors in high dimensional feature space, labeling each vector by its class. ELMs categorizes data by finding out a collection of support vectors, which are members of the set of training inputs that outline a hyper plane in the feature space. ELM is the kind of classifiers which were initially developed for binary classification [10] and they can be exploited to classify the attacks.

The major reason for using ELM in IDS is because of its speed and its scalability. Moreover the classification complexity of ELM is not based on the dimensionality of the feature space and hence it can potentially learn a huge collection of patterns. Also ELM provides a standard mechanism to fit the surface of the hyper plane to the data by utilizing the kernel function. In this paper, radial basis kernel function of ELM is tuned using Levenberg-Marquardt (LM) learning and tested using KDD Cup 1999 dataset based on the detection rate and very low false alarm rate.

2. LITERATURE SURVEY

A model is offered by Zhang Hongmei [11] to provide a solution for the crisis of low accurateness and high false alarm rate in network model of ELM with the help of rough set feature able to reduce low accurateness and high fasle alarm. Rough set approach can be utilized to maintain the discermibility of the reduction of original datasets, the original dataset reducts is computed and is used to train individual ELM classifier for collection and to increase the variety between individual classifiers, and to increase the chance of discovering accuracy.

ELM has been confirmed to be a efficient classifier in several applications, however its practice is still imperfect as the data allocation information is underutilized in determining the result hyperplane. The majority of existing kernels are working in nonlinear ELMs establishing the likeness between a pair of pattern images depending on the Euclidean inner product or the Euclidean detachment of equivalent input patterns, that omits data sharing tendency and makes the ELM fundamentally a ldquolocalrdquo classifier. Toward a paradigm of kernels, Defeng Wang et al., [12] provided a step by adding exact data knowledge into existing kernels. Find the data structure first for each class adaptively in the input space via Agglomerative Hierarchical Clustering (AHC), then construct the Weighted Mahalanobis Distance (WMD) kernels using the data distribution information which is deducted. Similarity between two pattern images is determined not only by the Mahalanobis Distance (MD) between their related input patterns but also by the sizes of the clusters they reside in WMD. However WMD kernels are not assured to be positive definite (pd) or conditionally positive definite (cpd), acceptable categorization outcome can still be achieved of regularizers in ELMs both WMD kernels and empirically optimistic in pseudo-Euclidean (pE) spaces.

The individuality of security attacks in ad hoc networks has given rise to the requirement for intending novel intrusion detection approaches, different from those exist in conventional networks. Joseph et al., [13] developed an autonomous host-dependent IDS for identifying malicious sinking behavior. This system increases the detection accuracy by using cross-layer features to describe a routing behavior. For learning and adjustment to new kind of attack circumstances and network surroundings, two machine learning approaches are exploited. ELMs and Fisher Discriminant Analysis (FDA) are utilized collectively to develop better accuracy of ELM and quicker speed of FDA. Rather than using all cross-layer features, features from MAC layer are connected/interrelated with features from additional layers, in that way reducing the feature set without reducing the information content.

3. METHODOLOGY

3.1 Extreme Learning Machine (ELM)

Extreme learning machine (ELM) was proposed in Huang, et al. [15]. Suppose we are training SLFNs with K hidden neurons and activation function g(x) to learn N distinct samples (x_i, t_i), where $x_i = [x_{i1}, x_{i2}, \dots, x_{in}]^T \in R^n$ and $t_i = [t_{i1}, t_{i2}, \dots, t_{im}]^T \in R^m$. In ELM, the input weights and hidden biases are randomly generated instead of tuning. By doing so, the nonlinear system has been converted to a linear system:

$\mathrm{H}\beta = \mathrm{T}$

$$\hat{\beta} = \mathrm{H}^{\dagger}\,\mathrm{T} \qquad (2)$$

where $H^{\dagger}$ is the MP generalized inverse of matrix H. The minimum norm LS solution is unique and has the smallest norm among all the LS solutions. As analyzed by Huang,et al. [15], ELM using such MP inverse method tends to obtain good generalization performance with dramatically increased learning speed.

ELM Algorithm

In the case of learning an arbitrary function with zero training error, Baum had presented several constructions of SLFNs with sufficient hidden neurons [14]. However, in practice, the number of hidden neurons required to achieve a proper generalization performance on novel patterns is much less. And the resulting training error might not approach to zero but can be minimized by solving the following problem:

$$\min_{w_i,\, b_i,\, \beta} \|H(W_1, \dots, W_{\tilde{N}}, b_1, \dots, b_{\tilde{N}})\beta - T\|^2, \qquad (3)$$

Where

$$T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m} \qquad (4)$$

ELM randomly assigns and fixes the input weights w_i and biases b_i based on some continuous probability distribution function in the case of learning a structured function, only leaving output weights β_i to be adjusted according to:

$$\min_{\beta} \|H\beta - T\|^2 \qquad (5)$$

The above problem is well established and known as a linear system optimization problem. Its unique least-squares solution with minimum norm is given by[15]:

$$\hat{\beta} = H^{\dagger}T, \qquad (6)$$

where $H^{\dagger}$ the Moore-Penrose is generalized inverse of the matrix H. As analyzed by Bartlett [21] and Huang,[22],[21] the generalization performance of a SLFN tends to be better with smaller magnitude of output weights. From this sense, the solution produced by ELM in Eq. (6) not only achieves the minimum square training error but also the best generalization performance on novel patterns. We now summarize ELM as the follows: ELM Algorithm: Given a training set $\aleph = \{(X_k, t_k) \mid X_k \in R^n, t_k \in R^m, k = 1, \dots, N\}$, an activation function $g(x)$, and the number of hidden neurons $\tilde{N}$,

(i) Randomly assign input weights wi and biases bi according to some continuous probability density function.

(ii) Calculate the hidden layer output matrix H.

(iii) Calculate the output weights β_i: $\hat{\beta} = \mathrm{H}^{\dagger}\mathrm{T}$.

In our experiments with ELM in this paper, the activation function is a sigmoid function: $g(x) = 1/(1 + e^{-x})$. And the probability density function is a uniform distribution function in the range from −1 to 1.

Principles of ELM – A survey

ELM [21] designed as a SLFN with L hidden neurons can learn L distinct samples with zero error. Even if the number of hidden neurons (L)$<$ the number of distinct samples (N), ELM can still assign random parameters to the hidden nodes and calculate the output weights using pseudoinverse of H giving only a small error $\in > 0$. The hidden node parameters of ELMa_i and b_i(input weights and biases or centers and impact factors) need not be tuned during training and may simply be assigned with random values. The following theorems state the same.

Theorem 1: (Liang et.al.[19]) Let an SLFN with

L additive or RBF hidden nodes and an activation function $g(x)$ which is infinitely differentiable in any interval of R be given. Then, for arbitrary L distinct input vectors $\{x_i | x_i \in R^n, i = 1, \dots, L\}$, and $\{(a_i, b_i)\}_{i=1}^{L}$ randomly generated with any continuous probability distribution, respectively, the hidden layer output matrix is invertible with probability one, the hidden layer output matrix H of the SLFN is invertible and $\|H\beta - T\| = 0$

Theorem 2: (Liang et.al.[19])Given any small positive value $\in > 0$ and activation function $g(x) : R \to R$which is infinitely differentiable in any interval, there exists $L \leq N$ such that for N arbitrary distinct input vectors $\{x_i | x_i \epsilon R^n, i = 1, \dots, L\}$, for any $\{(a_i, b_i)\}_{i=1}^{L}$ randomly generated according to any continuous probability distribution $\|H_{N\times L}\beta_{L\times m} - T_{N\times m}\| < \in$ with probability one.

Since the hidden node parameters of ELM need not be tuned during training and since they are simply assigned with random values, eqn (5) becomes a linear system and the output weights can be estimated as follows.

$$\beta = H^{\dagger}T$$

where $H^{\dagger}$the Moore-Penrose is generalized inverse [21] of the hidden layer output matrix H and can be calculated using several methods including orthogonal projection method, orthogonalization method, iterative method, singular value decomposition (SVD) etc. The orthogonal projection method can be used only when $H^T H$ is nonsingular and $H^{\dagger} = (H^T H)^{-1} H^T$. Due to the use of searching and iterations, orthogonalization method and iterative method have limitations. Implementations of ELM uses SVD to calculate the Moore-Penrose generalized inverse of H, since it can be used in all situations. ELM is thus a batch learning method.
Universal approximation capability of ELM has been analyzed in [15] in an incremental method and it has been shown that single SLFNs with randomly generated additive or RBF nodes with a widespread of piecewise continuous activation functions can universally approximate any continuous target function on any compact subspace of the Euclidean space R^n.

Theorem 3: (Huang et.al. [15])Given any bounded non constant piecewise continuous function $g : R \to R$for additive nodes or any integrable piecewise continuous function $g : R \to R$ and $\int_R g(x)dx \neq 0$ for RBF nodes, for any continuous target function f and any randomly generated function sequence $\{g_n\}, \lim_{n\to\infty} \|f - f_n\| = 0$ holds with probability one if

$$\beta_n = \frac{\langle e_{n-1}, g_n \rangle}{\|g_n\|^2} \qquad (7)$$

Incremental algorithm (also called I-ELM) has been proposed by Huang et.al.[31] for SLFN and TLFN (Two hidden layer feedforward neural network) which increases the hidden neurons one-by-one until the error becomes less than a predefined constant ϵ.

Convex incremental ELM (CI-ELM) [20] is another extension of ELM. In CIELM, the output weights of existing nodes are recalculated based on the Barron’s convex optimization concept [20], when a new hidden node is randomly added using$f_n = (1 - \beta_n) f_{n-1} + \beta_n g_n$, where $0 \leq \beta_n \leq$1.

Theorem 4: (Huang et.al [15]) Given any nonconstant piecewise continuous function $g : R \to R$, if span$\{G(x, a, b) : (a, b) \epsilon R^d \times R\}$ is dense in L^2, then for any continous target function f and any function sequence $\{g_n(x) = G(x, a_n, b_n)\}$ randomly generated based on any continuous sampling distribution, $lim_{n\to inf} \|f - ((1 - \beta_n) f_{n-1} + \beta_n g_n\| = 0)$ holds with probability 1 if

$$\beta_n = \frac{\langle e_{n-1}, g_n - f_{n-1} \rangle}{\|g_n - f_{n-1}\|} \qquad (8)$$

where G(x,a,b) is the output of hidden nodes.

Based on the above theorem, the output weight for the newly added hidden node is

$$\beta_L = (E.[E - (F - H_L)]^T)/([E - (F - H_L)].[E - (F - H_L)]^T) = \left(\sum_{p=1}^{N} e(p)[e(p) - (f(p) - h(p))]\right) / \left(\sum_{p=1}^{N} [e(p) - (f(p) - h(p))]\right)$$

the output weights of existing hidden nodes are recalculated by $\beta_i = (1 - \beta_L)\beta_i$ and the residual error after addeding the new hidden node L is $E = (1 - \beta_L)E + \beta_L(F - H_L)$, where the estimates based on the training set are $H = [h(1), \dots, h(N)]^T$ is the activation vector of the new node for all the N training samples, $E = [e(1), \dots, e(N)]^T$is the residual vector before the new hidden node is added and $F = [t_1, \dots, t_N]^T$ is the target vector
The following theorem states universal approximator for any type of piecewise continuous computational hidden nodes.

Theorem 5: (Huang et.al [15]) Given any nonconstant piecewise continuous function $g : R \to R$, if $span\{G(x, a, b) : (a, b) \epsilon R^d \times R\}$ is dense in L^2, for any continuous target function f and any function sequence$\{g_n(x) = G(x, a_n, b - n)\}$randomly generated based on any continuous sampling distribution, $lim_{n\to\infty} \|f - f_n\| = 0$ holds with probability 1 if the output parameters are determined by ordinary least square to minimize $\|f(x) - \sum_{i=1}^{n} \beta_i g_i(x)\|$.

Later Huang et.al. came up with EI-ELM (Enhanced random search based incremental learning machine) [15] and they found that some of the hidden nodes in networks play a very minor role in the network output thereby increasing the complexity of the system. So in EI-ELM, at each learning step several hidden nodes are randomly generated and among them the hidden node leading to the largest residual error decreasing will be added to the existing network. The output wieght is calculated as in I-ELM. The following theorem states the same.

Theorem 6: Given an SLFN with any nonconstant piecewise continuous hidden nodes$G(x,a,b)$, if $span\{G(x,a,b) : (a,b) \in R^d \times R\}$ is dense in L^2, for any continuous target function f and any randomly generated function sequence $\{g_n\}$
and any positive integer k, $lim_{n\to\infty}||f - f_n^*|| = 0$ holds with probability 1 if if

$$\beta_n = \frac{\langle e_{n-1}^*, g_n^* \rangle}{||g_n^*||^2} \qquad (10)$$

where

$$f_n^* = \sum_{i=1}^{n} \beta_i^* g_i^*, e_n^* = f - f_n^* \text{ and } g_n^* = \{g_i | min_{(n-1)k+1 \beta \le i \le nk} ||(f - f_n^*) - \beta_n g_i||\}.$$

Gradient-based algorithms cannot directly train neural network with threshold functions as they are nondifferentiable. Hence most of the literature uses sigmoid function as an approximation to threshold functions. The following lemma 1,
theorems 7,8 by Huang et.al. [15] states the use of threshold functions for extreme learning machines.
Lemma 1: (Huang et.al. [15]) A SLFN with N hidden neurons with the activation function $g(x) = 1/(1 + e^{-\lambda x})$and with randomly chosen input weights and hidden biases can learn N distinct observations with any arbitrarily small error.

Theorem 7: (Huang et.al. [15]) For a SLFN with the activation function $g(x) = 1/(1 + e^{-\lambda x})$ in the hidden layer, given any constant $\in > 0$, there always exists an integer $L \le N$such that a SLFN with Lsuch hidden neurons and with randomly chosen input weights and hidden biases can learn N distinct observations with a training error less than $\in$.
Theorem 8: (Huang et.al. [15]) suppose that threshold activation function $h(x) = 1_{x \ge 0} + 0_{x<0}$ is used in the hidden layer. Given any nonzero constant $\in > 0$ there always exists an integer $L \le N$ such that a SLFN with L such hidden neurons and with randomly chosen input weights and hidden biases can learn N distinct observations with its training error less than$\in$. Online Sequential learning algorithm [19] has been proposed by Liang et.al. this can learn data one-by-one or chunk by chunck. For this, first the type of node (additive or RBF) must be selected, the corresponding activation function g, and the number of hidden neurons L. Then initialize the learning using a small chunck of data $\aleph_0 = \{(x_i, t_i)\}_{i=1}^{N_0}$ from the given training set $\aleph = \{(x_i, t_i) | x_i \in R^n, t_i \in R^m, i = 1, \dots\}, N_0 \ge N.$ then find hidden layer output matrix
H_0 and the initial output weight $\beta^{(0)} = P_0 H_0^T T_0$, where $P_0 = (H_0^T H_0)^{-1}$ and $T_0 = [t_1, \dots, t_{N_0}]^T$.
Then for each $(K + 1)$th chunk of data,

$$\aleph_{k+1} = \{(x_i, t_i)\} \sum_{j=0}^{k+1} N_j i = \left(\sum_{j=0}^{k} N_j\right) + 1 \qquad (11)$$

where N_{k+1} denotes the number of observations in the $(k + 1)$th chunk, the sequential learning phase is given as
1) Calculate H_{k+1}.
2) Set $T_{k+1} = \left[t\left(\sum_{j=0}^{k} N_j\right) + 1, \dots, t \sum_{j=0}^{k+1} N_j\right]^T$.
3) Calculate β^{k+1} using

$$P_{k+1} = P_k - P_k H_{k+1}^T (I + H_{k+1} P_k H_{k+1}^T)^{-1} H_{k+1} P_k \qquad (12)$$

$$\beta^{k+1} = \beta^k + P_{k+1} H_{k+1}^T (T_{k+1} - H_{k+1} \beta^k \qquad (13)$$

4) Do these same procedure for other chunks of new data Error Minimized ELM (EM-ELM) with automatic Growth of Hidden Nodes and fast Incremental output weight Learning has been proposed by Feng et.al.[18] with Lemma 2 and theorem 9.
Lemma 2: (Feng et.al. [18]) Given an SLFN, let $H_1 = H(a_1, \dots, a_{L_0}, b_1, \dots, b_{L_0}, x_1, \dots, x_N)$ denote the hidden layer output matrix of the SLFN with L_0hidden nodes$\{(a_i, b_i)\}_{i=1}^{L_0}$. If $L_1 - L_0$ new hidden nodes are added to the SLFN,
the new hidden layer output matrix of the SLFN becomes $H_2 = H(a_1, \dots, a_{L_1}, b_1, \dots, b_{L_1}, x_1, \dots, x_N)$then
$E(H_2) = \min||H_2\beta_2 - T|| \le E(H_1) = \min ||H_1\beta_1 - T||$where $E(H)$denotes the output error functions of SLFNs.

Theorem 9: (Feng et.al. [18]) (Convergence Theorem): For a given set of distinct training samples$\{(x_i, t_i)\}_{i=1}^{N}$, given an arbitrary positive value ϵ, there exists a positive integer k such that $E(H_k) = \min||H_k\beta_k - T|| < \epsilon$.
Given a set of training data$\{(x_i, t_i)\}_{i=1}^{N}$, the maximum number of hidden nodes L_{max}, a small positive integer $L_0$and the expected learning accuracyϵ, the recursive EM-ELM algorithm will randomly add δL_{k-1} hidden nodes (total hidden nodes is $L_k = L_{k+1} + \delta L_{k-1}$) until the learning error $E(H_k) > \epsilon$ and the output weights β is updated recursively by$\beta_{k+1} = H_{k+1}^{\dagger} T = [U_k \; D_k]^T T$, where
$D_K = ((I - H_k H_K^{\dagger}) \delta H_K)^{\dagger}, U_K = H_K^{\dagger}(I - \delta H_K^T D_K)$and $H_{k+1} = [H_k, \delta H_k]$
is the hidden layer output matrix with

$$\delta H_k = \begin{bmatrix} G(a_{L_{k-1}} + 1, b_{L_{k-1}} + 1, X_1) \dots G(a_{L_k} + 1, b_{L_k} + 1, X_1) \\ \dots \\ G(a_{L_{k-1}} + 1, b_{L_{k-1}} + 1, X_N) \dots G(a_{L_k}, b_{L_k}, X_N) \end{bmatrix} \qquad (14)$$

Levenberg-Marqurадt Learning
Levenberg-Marquardt (LM) learning developed from expansion of Error Back Propagation (EBP) algorithm dependent techniques. It provides a significant exchange between the speed of the Newton algorithm and the strength of the steepest descent technique. There are the two fundamental theorems of LM algorithm. In the back-propagation algorithm, the performance index $F(w)$ to be reduced is defined as the sum of squared errors between the target outputs and the network's simulated outputs,

$$F(w) = e^T e \qquad (15)$$

Where $w = [w_1, w_2 \dots, w_N]$ includes all weights of the network, e indicates the error vector comprising the error for all the training samples.
The increment of weights Δw, when training with the LM method is calculated as follows:

$$\Delta w = [J^T J + \mu I]^{-1} J^T e \tag{16}$$

Where J denotes the Jacobian matrix, μ represents the learning rate which is to be updated using the β depending on the resultant. Especially, μ is multiplied by decay rate β $(0 < \beta < 1)$ whenever $F(w)$ diminishes, while μ is divided by β whenever $F(w)$ increases in a new step.
This is achieved by adding a small scalar λ to the diagonal elements in the Hessian H, as expressed by:

$$(H + \lambda I)\Delta\sigma = -\nabla E(\sigma_0) \tag{17}$$

In this method, the algorithm begins with a first-order approach and regularly proceeds towards the second-order approach outlined below. The equation (17) is solved for $\Delta\sigma$.
It is to be noted that each element $\frac{\partial E}{\partial \sigma_i}\big|_{\sigma=\sigma_0}$in the right side of equation (10) will be computed by numerical perturbation as given below:

$$\frac{\partial E}{\partial \sigma_i}\Big|_{\sigma=\sigma_o} \approx \frac{\Delta E}{\varepsilon}\Big|_{\sigma=\sigma_o} = \frac{E(\sigma_i + \varepsilon) - E(\sigma_i)}{\varepsilon}\Big|_{\sigma=\sigma_c} \tag{18}$$

Where ε represents a small perturbation value acting on the i^{th} component inσ. $E(\sigma_i)$ is the performance metric Q^2 obtained from the change in the i^{th} component of σ only.
A second approximation will be introduced before solving the above equations. Because the elements of the Hessian are expensive to evaluate, a fast and efficient approximation for the Hessian matrix is introduced. Each element in the Hessian matrix is originally defined by:

$$H(i,j) = \frac{\partial^2 E}{\partial \sigma_i \partial \sigma_j} \tag{19}$$

In general, the second partial derivatives can be numerically computed. On the other hand, in order to speed up the calculation process, the second-order partial derivatives are approximated by:

$$\frac{\partial^2 E}{\partial \sigma_i \partial \sigma_j} \approx \frac{\partial E}{\partial \sigma_i}\frac{\partial E}{\partial \sigma_j} \tag{20}$$

$\Delta\sigma$ is then solved numerically from equation (17) with a fast conjugate gradient based equation solver in order to avoid calculating the inverse of the Hessian matrix, H. As a result of the fairly accurate evaluation of the Hessian, a heuristic coefficient α will be introduced in the iterative updating procedure for the elements of σ leading to:

$$\sigma = \alpha\Delta\sigma + \sigma_0 \tag{21}$$

The value of α is fixed with a constant value of 0.5 which turns out to be a robust choice based on hundreds of experiments with this algorithm on different datasets. A more detailed description for the implementation of the algorithm is the sigma tuning algorithm is illustrated in the following,

1. Begin with an initial guess σ_0 and compute the initial Q^2 error metric from a leave-one-out K-PLS model and estimate$E(\sigma_0)$. Start with λ =1.
2. ΔE Calculation: For each scalar σ_i calculate the corresponding element in ΔE by perturbation.
3. $\Delta\sigma$ Calculation: Solve equation (17) for $\Delta\sigma$ by using a fast conjugate gradient-based equation solver.
4. λ Adjustment: If the Q^2 error gets smaller, update σ and decrease $\lambda = 0.93\lambda$; otherwise, make no change for σ and increase $\lambda = 3.5\lambda$. If $\lambda > 1$, cap λ to unity.
5. Iterate the process: Use the new solution as a new starting point and go to step 2. If the error can not be improved or the process reaches the iteration number limit, halt the procedure.

Note that both coefficients 0.93 and 3.5 are empirical values based on several experiments.

4. EXPERIMENTAL RESULTS

The proposed IDS is experimented using the Waikato Environment for Knowledge Analysis (WEKA) and the dataset used is KDD Cup99 dataset. WEKA is a complete set of Java class libraries that execute several state-of-the-art machine learning and data mining approaches [16]. KDD Cup99 dataset comes from DARPA 98 Intrusion Detection Evaluation handled by Lincoln laboratory at MIT [17].
Initially pre-processing is carried out, at this phase the data is segmented into training and testing. Then applied ELM and the proposed KELM is tuned with LM on training dataset with the purpose of constructing and training the models. At last, the trained models are evaluated on testing dataset to observe the effectiveness of both the approaches.
KDD Cup99 is an audited set of standard dataset which includes training and testing set. Data has the following four major groups of attacks:

- Denial-of-Service (DoS) like smurf, apache2, pod, etc.
- Remote-to-Local (R2L) like imap, worm, phf, etc.
- User to Root (U2R) like perl, rootkit and so on.
- Probing like nmap, portsweep, etc.

Detection of attack can be calculated by using the following metrics:

- False positive (FP): Or false alarm, Corresponds to the number of detected attacks but it is in fact normal.
- False negative (FN): Corresponds to the number of detected normal instances but it is actually attack, in other words these attacks are the target of intrusion detection systems.
- True positive (TP): Corresponds to the number of detected attacks and it is in fact attack.
- True negative (TN): Corresponds to the number of detected normal instances and it is actually normal.

Performance Measures

The performance measure used to evaluate the proposed KELM with LM against ELM is

- Detection rate and
- False-alarm rate

The accuracy of an intrusion detection system is computed based on the detection rate and false alarm rate.

Detection Rate Comparison

Detection rate indicates the percentage of detected attack among all attack data, and is given as,

$$Detection\ Rate = \frac{TP}{TP + TN} \times 100 \tag{22}$$

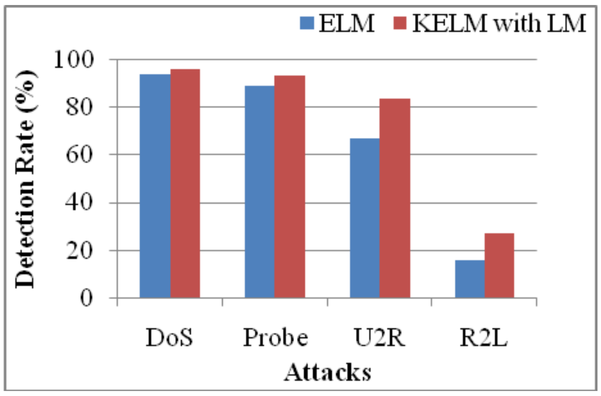

Figure 1.1: Comparison of Detection Rate on Four Attacks

The results of detection rate for different types of attacks is shown in figure 1.1. From the results it is observed that in case of DoS attacks, detection rate for ELM and proposed KELM with LM is 93.84 and 97.89, respectively. Similarly the detection rate for KELM with LM is better than ELM in all other attacks.

False Alarm Rate Comparison
False alarm rate indicates the percentage of normal data which is wrongly recognized as attack, and is defined as follows:

$$False\ Alarm\ Rate = \frac{FP}{FP + TN} \times 100 \qquad (23)$$

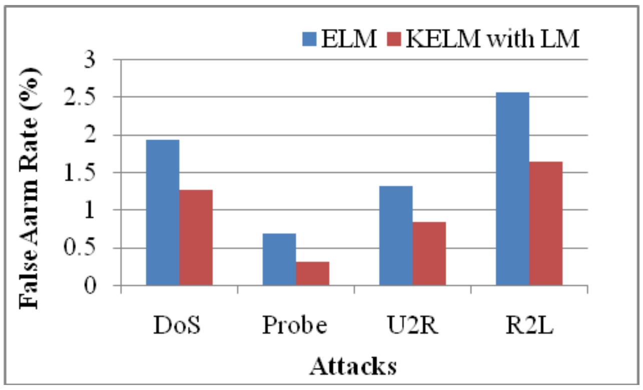

Figure 1.2: Comparison of False Alarm Rate on Four Attacks

The results of false alarm rate for different types of attacks is shown in figure 1.2. From the figure it is observed that for DoS attacks, false alarm rate for ELM and proposed KELM with LM is 1.92 and 1.06, respectively. In the same way the false alarm rate for KELM with LM is better than the ELM in all other attacks.

5. CONCLUSION

At present, security inside the network communication is of a major concern. Being the fact that data are considered as the valuable asset of an organization, providing security against the intruders is very essential. Intrusion detection system tries to identify security attacks of intruders by investigating several data records observed in processes on the network. This paper proposed Kernelized Extreme Learning Machine with Levenberg-Marquardt (LM) learning, in which the sigma of radial basis kernel function are tuned using LM. LM learning is utilized to obtain a converged solution. The experiment is carried out in WEKA by using KDD Cup 1999 dataset and the results indicate that the proposed system can provide better detection rate and low false alarm rate than the ELM.

6. REFERENCES

[1] S. Mukkamala, G. Janoski and A. Sung, “Intrusion detection using neural networks and support vector machines”, Proceedings of International Joint Conference on Neural Networks (IJCNN '02), Vol. 2, Pp. 1702–1707, 2002.

[2] Snehal A. Mulay, P.R. Devale and G.V. Garje, “Intrusion Detection System using Support Vector Machine and Decision Tree”, International Journal of Computer Applications, Vol. 3, No. 3, Pp. 40-43, 2010.

[3] D. Anderson, T. Frivold and A. Valdes, “Next-generation intrusion detection expert system (NIDES): a summary”, Technical Report SRI-CSL-95-07. Computer Science Laboratory, SRI International, Menlo Park, CA, 1995.

[4] S. Axelsson, “Research in intrusion detection systems: a survey”, Technical Report TR 98-17 (revised in 1999). Chalmers University of Technology, Goteborg, Sweden, 1999.

[5] S. Freeman, A. Bivens, J. Branch and B. Szymanski, “Host-based intrusion detection using user signatures”, Proceedings of the Research Conference. RPI, Troy, NY, 2002.

[6] K. Ilgun, R.A. Kemmerer and P.A. Porras, “State transition analysis: A rule-based intrusion detection approach”, IEEE Trans. Software Eng, Vol. 21, No. 3, Pp. 181–199, 1995.

[7] D. Marchette, “A statistical method for profiling network traffic”, Proceedings of the First USENIX Workshop on Intrusion Detection and Network Monitoring, Santa Clara, CA, Pp. 119–128, 1999.

[8] R.G. Bace, “Intrusion Detection”, Macmillan Technical Publishing, 2000.

[9] B.V. Dasarathy, “Intrusion detection, Information Fusion”, Vol. 4, No. 4, Pp. 243-245, 2003.

[10] M. Cramer, James Cannady and Jay Harrell, “New Methods of Intrusion Detection using Control-Loop Measurement”, Proceedings of the Technology in Information Security Conference, Pp 1-10, 1995.

[11] Zhang Hongmei, “ELM ensemble intrusion detection model based on Rough Set feature reduct”, Chinese Control and Decision Conference (CCDC '09), Pp. 5604–5608, 2009.

[12] Defeng Wang, D.S. Yeung and E.C. Tsang, “Weighted Mahalanobis Distance Kernels for Support Vector Machines”, IEEE Transactions on Neural Networks, Vol. 18, No. 5, Pp. 1453-1462, 2007.

[13] J.F.C. Joseph, Bu-Sung Lee, A. Das and Boon-Chong Seet, “Cross-Layer Detection of Sinking Behavior in Wireless Ad Hoc Networks Using ELM and FDA”, IEEE

Transactions on Dependable and Secure Computing, Vol. 8, No. 2, Pp. 233-245, 2011.

[14] Wun-Hwa Chen, Sheng-Hsun Hsu and Hwang-Pin Shen, "Application of ELM and ANN for intrusion detection", Computers & Operations Research, Vol. 32, Pp. 2617-2634, 2003.

[15] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: a new learning scheme of feedforward neural networks, in: Proceedings of the International Joint Conference on Neural Networks (IJCNN2004), Budapest, Hungary, 25–29 July 2004.

[16] Witten, I. H., and Frank E. (1999) Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann, San Francisco.

[17] KDD Cup network intrusion dataset, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[18] Kyaw Thet Khaing, "Enhanced Features Ranking and Selection using Recursive Feature Elimination (RFE) and k-Nearest Neighbor Algorithms in Support Vector Machine for Intrusion Detection System", International Journal of Network and Mobile Technologies, Vol. 1, No. 1, Pp. 8-14, 2010.

[19] Liang N-Y, Huang G.B, Saratchandran P, Sundararjan N (2006) A fast and accurate on-line sequential learning algorithm for feed forward networks.IEEE Trans Neural Netw 17(6):1411-1423.

[20] A.R. Barron, Universal approximation bounds for superposition of sigmoidal function, IEEE Trans.Inf Theory 39(3) (1993) 930-945.

[21] P.L Bartlett,"The Sample complexity of pattern classification with Neural Networks:The size of the Weight is More Important than the Size of the Network," IEEE Trans. Information Theory,Vol.44,no.2,pp.525 536,Mar.1998.

[22] G.B Huang, Q. Y. Zhu, and C.K.Siew,"Extreme Learning Machine: Theory and Application," Neurocomputing,Vol.70,pp, 489-501, 2006.