



Model of Improved a Kernel Fast Learning Network Based on Intrusion Detection System

Mohammed Hasan Ali^(✉) and Mohamed Fadli Zolkipli

Faculty of Computer Systems and Software Engineering,
University Malaysia Pahang, Gambang 26300, Malaysia
Mh180250@gmail.com

Abstract. The detection of network attacks on computer systems remains an attractive but challenging research scope. As network attackers keep changing their methods of attack execution to evade the deployed intrusion-detection systems (IDS), machine learning (ML) algorithms have been introduced to boost the performance of the IDS. The incorporation of a single parallel hidden layer feed-forward neural network to the Fast Learning Network (FLN) architecture gave rise to the improved Extreme Learning Machine (ELM). The input weights and hidden layer biases are randomly generated. In this paper, the particle swarm optimization algorithm (PSO) was used to obtain an optimal set of initial parameters for Reduce Kernel FLN (RK-FLN), thus, creating an optimal RKFLN classifier named PSO-RKELM. The derived model was rigorously compared to four models, including basic ELM, basic FLN, Reduce Kernel ELM (RK-ELM), and RK-FLN. The approach was tested on the KDD Cup99 intrusion detection dataset and the results proved the proposed PSO-RKFLN as an accurate, reliable, and effective classification algorithm.

Keywords: Fast learning network · Kernel extreme learning machine
KDD Cup99 · Particle swarm optimization algorithm
Intrusion detection system

1 Introduction

Both network security and computer security systems collectively make up cybersecurity systems. Each of the security systems basically has an antivirus software, a firewall, and an IDS. The IDSs is involved in the discovery, determination, and identification of unauthorized access, usage, alteration, destruction, or duplication of an information system [1]. The security of these systems can be violated through external (from an outsider) and internal attacks (from an insider). Until now, much efforts are devoted to studies on the improvement of network and information security systems, and several studies exist on IDS and its taxonomy [1–4]. Machine learning has recently gained much interest from different fields such as control, communication, robotics, and several engineering fields. In this study, a machine learning approach was deployed to address the issues of intrusion detection in computer systems. It is a challenging task to automate ID processes, as has earlier been ascertained by Sommer and Paxson who applied ML techniques to ID systems and outlined the challenges of automating

network attack detection processes [5]. The specific approaches of using ML techniques for network intrusion detection and their challenges have been previously outlined [6]. Some of the major problems of the current network ID systems such as high rates of false-positive alarms, false negative or missed detections, as well as data overload (a situation where the network operator is overloaded with information, making it difficult to monitor data) have been discussed [6].

Several ML algorithms have been used to detect anomalies in the behavior of ID systems. This is achieved by training the ML algorithms with the normal network traffic patterns, making them capable of determining traffic patterns that differ from the normal pattern [5]. Although some ML techniques can effectively detect certain forms of attack, no single method has been developed that can be universally applied to detect multiple types of attack. Intrusion detection systems can be generally divided into two system (anomaly and misuse) based on their mode of detection [6]. The anomaly-based detection system flags any abnormal network behavior as an intrusion, but the misuse-based detection system relies on the signature of established previous attacks to detect new intrusions. Several anomaly-based detection systems have been developed based on different ML techniques [6, 9, 11]. For instance, several studies have used single learning techniques like neural networks, support vector machines, and genetic algorithms to design ID systems [5]. Other systems such as the hybrid or ensemble systems are designed by combining different ML techniques [10, 11]. These techniques are particularly developed as classifiers for the classification or recognition of the status of an incoming Internet access (normal access or an intrusion). One of the significant algorithms of machine learning is the ELM first proposed by Huang. The ELM has been widely investigated and applied severally [12]. Several ID systems have been proposed based on the use of ELM as the core algorithm [6, 13, 14]. Furthermore, there is a heavy influx of network traffic data through the ID system which needs to be processed [7]. This study, therefore, focuses on the development of a scalable method that can improve the effectiveness of network ID systems in the detection of different classes of network attack.

2 Overview of Fast Learning Network

In the past few decades, the demand for even the high performing single hidden layer Feedforward Neural Network (FNN) has waned due to some application challenges [4]. To solve these issues, Guang Bin Huang proposed the Extreme Learning Machine (ELM) [3] whose major function is the transformation of a single hidden layer FNN into a linear least square solving a problem; it then, calculates the output weights through the Moore–Penrose (MP) generalized inverse. There are several advantages of ELM, first, it avoids repeated calculation of iteration, has a fast learning speed; cannot be trapped at the local minimum, ensure output weights uniqueness, has a simplified network framework, presents a better generalization ability and regression accuracy. Several scholars have successfully implemented the ELM learning algorithm and theory [5, 6] in pattern classification, function approximation [7–9], system identification and so on [10, 12]. The other issue is the handling of information incorporation in the ELM when multiple varying data sources are available [15]. Therefore, the

kernel-based ELM (KELM) has been proposed by comparing the modeling process between SVM and ELM [16]. The results show that KELM performs better and is more robust compared to the basic ELM [15] in solving non-separable linearly samples. The KELM also performed better than ELM, KELM in solving regression prediction tasks. It achieved a comparative or better performance with a faster learning speed and easier implementation in several applications, including 2-D profiles reconstruction, hyperspectral remote sensing image classification [17, 18], activity recognition, and diagnosis of diseases [19, 20]. KELM has also been used for online prediction of hydraulic pumps features, location of damage spot in aerospace, and behavior identification [21, 22]. However, [15] the training of KELM is an unstable process; the learning parameters must be manually adjusted; and it utilizes randomly generated hidden node parameters. The adjustment of the learning parameters requires human input, and could influence the classification performance. Its kernel function parameters also need a careful selection process to achieve an optimal solution. There are many works that provide optimization methods based on KELM parameters. The metaheuristics have been suggested for tackling the problems of parameter setting in KELM. Some of the suggested metaheuristics are genetic algorithm (GA) [18], and AdaBoost framework [23]. [24] used adaptive artificial bee colony (AABC) algorithm for parameter optimization and selection of KELM features. The features were evaluated based on Parkinson's disease dataset. In [25], the authors proposed the chaotic moth-flame optimization (CMFO) strategy to optimize KELM parameters. Also, an active operators particle swarm optimization algorithm (APSO) was proposed in [15] for obtaining an optimal initial set of KELM parameters. The evaluated model (APSO-KELM) based on standard genetic datasets show the APSO-KELM to have a higher classification performance compared to the current ELM and KELM. However, the results of this work show KELM to have a better accuracy compared to ELM, showing the need to introduce the kernel function. In other words, the optimize kernel parameter results showed no fluctuation and an increasing coverage with iteration. Meanwhile, there are some issues with ELM such as the need for additional hidden neurons in some regression applications compared to the conventional neural network (NN) learning algorithms. This may cause the trained ELM to require more reaction time when presented with new test samples. Furthermore, any increase in the number of hidden layer neurons also results to an exponential increase in the number of thresholds and weights of random initialization. These values may not be the optimized parameters [26, 27]. In 2013, Li et al. suggested a novel ELM-based artificial neural network for fast learning network (FLN) [13]. FLN is a double parallel FNN made up of a single layer FNN and a single hidden layer FNN. The received information at the input layer is transmitted to the hidden and output layers (first, the message gets to the neurons of the hidden layer before being transmitted to the output layer). Therefore, the FLN can perform nonlinear approximation like other general NN. Contrarily, the information is directly transferred from the input layer to the output layer, giving the FLN the ability to establish the linear relationship between the input and the output. Hence, the FLN can handle linear problems with a high accuracy, and can also infinitely approximate nonlinear systems. FLN can also solve the issue associated with the conventional NN which does not demand iterative calculation. This work start with Sect. 2 provides the

introduction. Section 3 explain the data set KDD details. Section 4 provides overview of the methodology. Section 5 provides the experiments and analysis of the results.

3 Overview of the Methodology

3.1 Fast Learning Network

The FLN was presented by [37] as a novel variant of the ELM [38]. It is structured as a combination of two NNs, the first one as an SLFNN and the second an MLFNN. The FLN depends on three layers, namely, input, hidden, and output layers. The FLN structure is shown in Fig. 1,

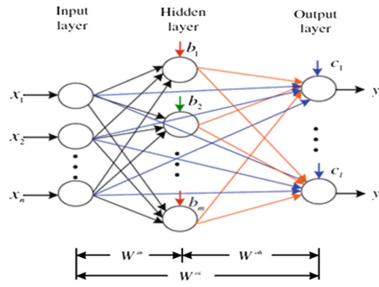


Fig. 1. Structure of FLN

The equations of deriving the output of the FLN based on the provided matrices and vectors and they are presented in the following equations.

$$y_j = f(w^{oi}x_j + c + \sum_{k=1}^m w_k^{oh} g(w_k^{in}x_j + b_k)) \tag{1}$$

Where

- $w^{oi} = [w_1^{oi}, w_2^{oi}, \dots, w_i^{oi}]$ is the weight vector connecting the output nodes and input nodes.
- $w_k^{in} = [w_{k1}^{in}, w_{k2}^{in}, \dots, w_{km}^{in}]$ is weight vector connecting the input nodes and hidden node
- $w_k^{oh} = [w_{1k}^{oh}, w_{2k}^{oh}, \dots, w_{ik}^{oh}]$ is weight vector connecting the output nodes and hidden node, a more compact representation is given as follows

The matrix $W = [W^{oi} \ W^{oh} \ c]$ can be called output weights, and G is the hidden layer output matrix of FLN, the *i*th row of G is the *i*th hidden neuron’s output vector with respect to inputs $x_1.x_2. \dots.x_N$. To solve the model, the minimum norm least-squares solution of the linear system can be written as follows: A more compact representation is given as follows:

$$Y = f(w^{oi}x + w^{oh}G + c) = f\left([w^{oi}w^{oh}c] \begin{bmatrix} X \\ G \\ I \end{bmatrix} \right) \tag{2}$$

$$= f\left(W \begin{bmatrix} X \\ G \\ I \end{bmatrix} \right) \tag{3}$$

Where

$$H = [XGI]^T \tag{4}$$

$$G(W_1^{in}, \dots, W_m^{in}, b_1, \dots, b_m, \dots, X_N) \tag{5}$$

$$= \begin{bmatrix} g(W_1^{in}X_1 + b_1) & \dots & g(W_1^{in}X_N + b_1) \\ \vdots & \ddots & \vdots \\ g(W_m^{in}X_1 + b_m) & \dots & g(W_m^{in}X_N + b_m) \end{bmatrix}_{m \times N}$$

$$W = [W^{oi}W^{oh}C]_{1 \times (n+m=1)} \tag{6}$$

$$I = [11 \dots 1]_{1 \times N} \tag{7}$$

In order to resolve the model, a Moore Penrose based equation is given as follows.

$$\hat{w} = \mathbf{f}^{-1}(Y) \left([X^T G^T I^T] \begin{bmatrix} X \\ G \\ I \end{bmatrix} \right)^{-1} \mathbf{H}^T \tag{8}$$

$$\hat{w} = \mathbf{f}^{-1}(Y) (X^T X + G^T G + I^T I)^{-1} \mathbf{H}^T \tag{9}$$

An algorithm that explains the learning of the FLN is presented in the flowchart depicted in Fig. 2. The algorithm starts by random initialization of the weights between the input and hidden layers and the biases of the hidden layer. Then, the G matrix is determined depending on the input-hidden matrix. This matrix represents the output matrix of the hidden layer. Next, the input-output matrix w^{oi} and w^{oh} are determined based on the Moore–Penrose equations. As a result, a complete FLN model is formulated

3.2 Kernel Fast Learning Network

A kernel function in machine learning is a measurement of the closeness between input sample data defined over a feature denoted as $K(x, x')$ [39]. In a recent study, [16] suggested that the hidden layer of an SLFN does not need to be formulated as a nodes

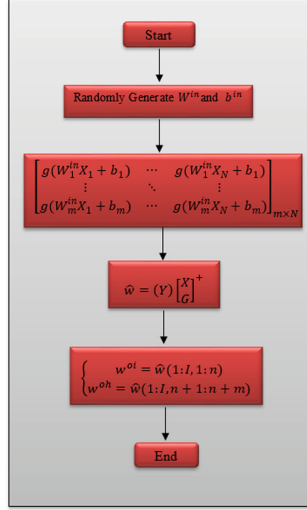


Fig. 2. Flowchart of the FLN learning model

of single layer; instead, a mechanisms of feature mapping can be a wide range used to replace the hidden layer.

This approach has been exemplified in this work as previously reported by [16] to convert an FLN model to a kernel-based model. By recalling the output of Eq. (3), replacing the output weight W with the output weight based on the Moore–Penrose generalized inverse \hat{w} as Eq. (8), and replacing H with Eq. (4), we can obtain Eq. (13) as follows:

$$\begin{aligned}
 W^\wedge &= Y^\wedge H^T (H H^T)^{-L} \\
 Y &= f \left(Y^\wedge H^T (H H^T)^{-1} H^T \right) \\
 &= f \left(Y^\wedge (H H^T)^{-1} H^T H \right) \\
 Y &= f \left(f(Y)^{-1} \left(\begin{bmatrix} X \\ G \\ I \end{bmatrix} \right)^{-1} \begin{bmatrix} X \\ G \\ I \end{bmatrix} \right) \\
 Y &= f \left(f(Y)^{-1} (X^T X + G^T G + I^T I)^{-1} (X^T X + G^T G + I^T I) \right) \\
 Y &= f \left(f(Y)^{-1} \left(\begin{bmatrix} X \\ G \\ I \end{bmatrix} \right)^{-1} \begin{bmatrix} X \\ G \\ I \end{bmatrix} \right) \quad (10)
 \end{aligned}$$

$$Y = f\left(f(Y)^{-1}(X^T X + G^T G + I^T I)^{-1}(X^T X + G^T G + I^T I)\right) \quad (11)$$

Moreover, through the addition of a small positive quantity (i.e., the stability factor) $1/\lambda$ to the diagonal of $\mathbf{H}^T \mathbf{H}$, thus yielding a more “stable” solution, we can represent Y as [16] follows:

$$Y = f\left(f(Y)^{-1}(k_1(x.x') + k_2(x.x') + k_3(x.x') + 1/\lambda)^{-1}(k_1(x.x') + k_2(x.x') + k_3(x.x'))\right) \quad (12)$$

Generally, the selection of the output neurons’ active function $f(\cdot)$ is often linear, such that $f(x) = x$. Then, Eq. (12) can be written as follows:

$$Y = Y (k_1(x.x') + k_2(x.x') + k_3(x.x') + 1/\lambda)^{-1}(k_1(x.x') + k_2(x.x') + k_3(x.x')) \quad (13)$$

Substituting in the location of k_1, k_2, k_3 three kernels, and in the location of λ a regularization factor, we obtain a MKFLN model. The power of this model is with using 3 kernels for performing the separation which is expected to outperform the classical one kernel ELM variant. However, there are two problems to be addressed. The first one is the computational concern when using kernels calculation, especially if the size of the dataset is huge. The second one is the criticality of selection suitable kernels for classification due to the sensitivity of the performance to the kernel type. Therefore, a framework for to make the developed MKFLN feasible for practical applications is designed.

However, the kernel-based learning methods usually utilize a large memory system for learning dilemmas with large data sets and are therefore modified to reduced kernel extreme learning machine (RKELM) (Deng et al. 2013). A modification to RKFLN called reduced kernel FLN has also been proposed. The kernel-based and basic ELMs have shown superior generalization and better scalability for multiclass problems with superior generalization, and considerable scalability for multiclass classification problems with much lower training times than those of SVMs [16]. These issues create the ELM an appealing learning paradigm for applied in large-scale problems, such as the IDSs.

Nevertheless, kernels are utilized in learning approaches, particularly those with potentially large amounts of memory for ML problems, with huge datasets such as IDS, which requires the collection of wide data in traffic network. To treat this issue, RKELM takes Huang’s kernel-based ELM and, instead of computing $k(x,x)$ over the entire input data, computes $k(\tilde{x}, x)$, where \tilde{x} is a randomly chosen subset of the input data. [40] adapted the method of reduced kernel by selecting a small random subset $\tilde{X} = \{x_i\}_{i=1}^{\tilde{n}}$ from the original data points $X = \{x_i\}_{i=1}^n$ with $\tilde{n} \ll n$ and using $k(x, \tilde{x})$ in place of $k(X, X)$ to cut the problem size and computing time. As mentioned previously, FLN is better than ELM; to address this issue, RKELM is swapped out for RKFLN. Hence, RKFLN is expected to be better than RKELM. An assumption supposes that multiplying each kernel in RKFLN by a weight provides better results.

3.3 Particle Swarm Optimization

The PSO was first introduced by Li et al. [23] as a parallel evolutionary computation technique which was inspired by the social behavior of swarm. The performance of PSO can be significantly affected by the selected tuning parameters (commonly called exploration– exploitation tradeoff). Exploration is the ability of an algorithm to explore all the segments of the search space in an effort to establish a good optimum, better known as the global best. Exploitation on the other hand is the ability of an algorithm to concentrate on its immediate environment and within the surrounding of a better performing solution to effectively establish the optimum. Irrespective of the research efforts in recent times, the selection of algorithmic parameters is still a great problem [41]. In the PSO algorithm, the objective function is used for the evaluation of its solutions, and to operate on the corresponding fitness values. The position of each particle is kept (including its solution), and its velocity and fitness are also evaluated [42]. The PSO algorithm has many practical applications [43–46]. The position and velocity of each particle is modified to establish an best solution for each iteration using the following relationship:

$$v_i(k + 1) = wv_ik + c1r1(xbest, local - x_i) + c2r2(xbest, global) - x_i \tag{14}$$

$$x_i(k + 1) = xk + v(k + 1) \tag{15}$$

Each particle’s velocity and position are denoted as the vectors $v_k = (v_{k1}, \dots, v_{kd})$ and $x_i = (x_{i1}, \dots, x_{id})$, respectively. In (14), x vectors is the best local and best global positions; $c1$ and $c2$ represents the acceleration factors referred to as cognitive and social parameters; $r1$ and $r2$ represents randomly selected number in the range of 0 and 1; k stands for the iteration index; and w is the inertia weight parameter [47]. x_i of a particle is updated using (15).

4 Experiments and Analysis

4.1 Implementing of Multi Kernel Based on Fast Learning Network

As it has been mentioned in the previous section, RKFLN has two problems to be solved before we can apply it, the first one is the computational complexity of applying three kernels at the same time, and the second one is the selection of the kernels and its parameters. The initial one is solved through using reduced kernel approach, and the other one is solved through preparing a set of kernels and selecting out of them. In order to make the model more optimized, three weighting factors of using the kernels are imported in the model $\alpha_1, \alpha_2, \alpha_3$ and λ . The parameters α_1, α_2 , and α_3 are weighting factors for the kernels k_1, k_2 , and k_3 , the parameter λ is the regularization parameter. The new model after optimizing is written as

$$Y = f(Y^\wedge \left(\alpha_1^* K_1 + \alpha_2^* K_2 + \alpha_3^* K_3 + \frac{1}{\lambda^*} \right)^{-1} (\alpha_1^* K_1 + \alpha_2^* K_2 + \alpha_3^* K_3))$$

where the vector $(\alpha_1^*, \alpha_2^*, \alpha_3^*, \lambda^*)$ denotes the optimal parameters of the model, they have the constraint

$$\alpha_1^* + \alpha_2^* + \alpha_3^* = 1, \alpha_1, \alpha_2, \alpha_3 \in [0, 1], \lambda^* \in [0, 1]$$

In order to determine convenient values for these weights that multiplying each kernel in RKFLN. PSO will be used. Herein, $k_1, k_2,$ and $k_3,$ are multiplied by the weights $\alpha_1, \alpha_2,$ and α_3 respectively, the optimization goal is to find the best weights values that give the highest testing accuracy over validation samples. Beside the searching for the weights values, the optimization process will also look for the best value for the regularization coefficient.

4.2 Experiments of Optimization Multi Kernel Fast Learning Network

This section describe our experimental also provide classification performanc results. In order to evaluate the PSO-RKFLN developed model, this work cover teasting results with the KDD Cup99 dataset. Optimize the RKFLN parameters to enhance the accuracy of IDS, we proposed several models such as RKFLN, RKELM, and PSO-RKELM as benchmarks. Figure 3 shown classification evaluation measures, and Table 1 shown the compresion results between the models.

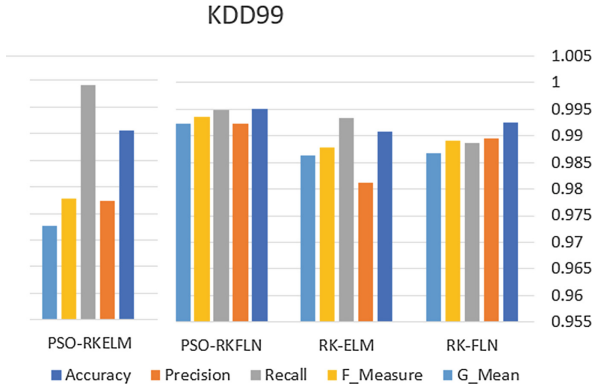


Fig. 3. Evaluation measures of classification

For PSO parameters formulae such as $c1 = c2 = 1.42,$ $w = 0.75$ and number of particles = 5. The KDD 99 training set with duplicates removed for the first set of sets with a dataset consisting of 145,585 split into a training set of 72,792 training examples and 72,792 testing samples. For this study we used all 41 attributes of the data.

Table 1. Results of evaluation measures of different models

Models	G_Mean	F_Measure	Recall	Precision	Accuracy
RK-ELM	0.9863	0.98773	0.99344	0.981103	0.99073
RK-FLN	0.9868	0.98905	0.98856	0.98955	0.99243
PSO-RKELM	0.98841	0.98973	0.99513	0.97971	0.99105
PSO-RKFLN	0.99223	0.99358	0.99492	0.99224	0.99509

5 Summary

In this work, using Machine learning based on the intrusion-detection system, it's attractive for many researchers. This work provides model based on optimize of a multi kernel of fast learning network. The derived model was rigorously compared to four models, including basic ELM, basic FLN, Reduce Kernel ELM (RK-ELM), and RK-FLN. The approach was tested on the KDD Cup99 intrusion detection dataset. The accuracy of our model (PSO-RKFLN) is slightly higher than other models. For a future work, we recommend checking this model with a different number of neurons to measures and evaluate the complexity of the model.

References

1. Buczak, A., Guven, E.: A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Commun. Surv. Tutorials*, vol. PP, no. 99, p. 1, 2015
2. Patel, A., Taghavi, M., Bakhtiyari, K., Celestino Jr J.: An intrusion detection and prevention system in cloud computing: a systematic review. *J. Netw. Comput. Appl.* **36**(1), 25–41 (2013)
3. Liao, H.-J., Lin, C.-H.R., Lin, Y.-C.: Intrusion detection system: a comprehensive review. *J. Netw. Comput. Appl.* **36**(1), 16–24 (2012)
4. Liao, H.J., Richard Lin, C.H., Lin, Y.C., Tung, K.Y.: Intrusion detection system: a comprehensive review. *J. Netw. Comput. Appl.* **36**(1), 16–24 (2013)
5. Tsai, C., Hsu, Y., Lin, C., Lin, W.: Expert systems with applications intrusion detection by machine learning: a review. *Expert Syst. Appl.* **36**(10), 11994–12000 (2009)
6. Fossaceca, J.M., Mazzuchi, T.A., Sarkani, S.: MARK-ELM: Application of a novel multiple kernel learning framework for improving the robustness of network intrusion detection. *Expert Syst. Appl.* **42**(8), 4062–4080 (2015)
7. Mishra, P., Pilli, E.S., Varadharajan, V., Tupakula, U.: Intrusion detection techniques in cloud environment: a survey. *J. Netw. Comput. Appl.* **77**, pp. 18–47, October 2016
8. Jaiganesh, V., Sumathi, P.: Kernelized extreme learning machine with levenberg-marquardt learning approach towards intrusion detection. *Int. J. Comput. Appl.* **54**(14), 38–44 (2012)
9. Udaya Sampath, X.W., Perera Miriya Thantrige, K., Samarabandu, J.: Machine learning techniques for intrusion detection. *IEEE Can. Conf. Electr. Comput. Eng.* 1–10 (2016)
10. Aslahi-Shahri, B.M., et al.: A hybrid method consisting of GA and SVM for intrusion detection system. *Neural Comput. Appl.* **27**(6), 1669–1676 (2016)
11. Atefi, K., Yahya, S., Dak, A.Y., Atefi, A.: A Hybrid Intrusion detection system based on differen machine learning algorithms. In: *Proceedings of the 4th International Conference Computing Informatics*, no. 22, pp. 312–320 (2013)

12. Ding, S., Xu, X., Nie, R.: Extreme learning machine and its applications. *Neural Comput. Appl.* **25**(3–4), 549–556 (2014)
13. Singh, R., Kumar, H., Singla, R.K.: An intrusion detection system using network traffic profiling and online sequential extreme learning machine. *Expert Syst. Appl.* **42**(22), 8609–8624 (2015)
14. Ali, M.H., Zolkipli, M.F., Mohammed, M.A., Jaber, M.M.: Enhance of extreme learning machine-genetic algorithm hybrid based on intrusion detection system. *J. Eng. Appl. Sci.* **12** (16), 4180–4185 (2017)
15. Lu, H., Du, B., Liu, J., Xia, H., Yeap, W.K.: A kernel extreme learning machine algorithm based on improved particle swam optimization. *Memetic Comput.* **9**(2), 121–128 (2017)
16. Huang, G.-B., Zhou, H., Ding, X., Zhang, R.: Extreme learning machine for regression and multiclass classification. *IEEE Trans. Syst. man, Cybern. Part B, Cybern.* **42**, (2), 513–529 (2012)
17. Pal, M., Maxwell, A.E., Warner, T.A.: Kernel-based extreme learning machine for remote-sensing image classification. *Remote Sens. Lett.* **4**(9), 853–862 (2013)
18. Liu, B., Tang, L., Wang, J., Li, A., Hao, Y.: 2-D defect profile reconstruction from ultrasonic guided wave signals based on QGA-kernelized ELM. *Neurocomputing* **128**, 217–223 (2014)
19. Deng, W.Y., Zheng, Q.H., Wang, Z.M.: Cross-person activity recognition using reduced kernel extreme learning machine. *Neural Netw.* **53**, 1–7 (2014)
20. Chen, H.L., Wang, G., Ma, C., Cai, Z.N., Bin Liu, W., Wang, S. J.: An efficient hybrid kernel extreme learning machine approach for early diagnosis of Parkinson’s disease. *Neurocomputing* **184**, 131–144 (2016)
21. Chen, C., Li, W., Su, H., Liu, K.: Spectral-spatial classification of hyperspectral image based on kernel extreme learning machine. *Remote Sens.* **6**(6), 5795–5814 (2014)
22. Fu, H., Vong, C.-M., Wong, P.-K., Yang, Z.: Fast detection of impact location using kernel extreme learning machine. *Neural Comput. Appl.* 1–10 (2014)
23. Li, L., Wang, C., Li, W., Chen, J.: Hyperspectral image classification by AdaBoost weighted composite kernel extreme learning machines. *Neurocomputing* **275**, 1725–1733 (2018)
24. Wang, Y., Wang, A.N., Ai, Q., Sun, H.J.: An adaptive kernel-based weighted extreme learning machine approach for effective detection of Parkinson’s disease. *Biomed. Signal Process. Control* **38**, 400–410 (2017)
25. Wang, M., et al.: Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses. *Neurocomputing* **267**, 69–84 (2017)
26. Li, X., Niu, P., Li, G.: An adaptive extreme learning machine for modeling NOx emission of a 300 MW circulating fluidized bed boiler (2017)
27. Li, G., Niu, P.: Combustion optimization of a coal-fired boiler with double linear fast learning network (2014)
28. Abadeh, M.S., Mohamadi, H., Habibi, J.: Design and analysis of genetic fuzzy systems for intrusion detection in computer networks. *Expert Syst. Appl.* **38**(6), 7067–7075 (2011)
29. Chen, T., Zhang, X., Jin, S., Kim, O.: Efficient classification using parallel and scalable compressed model and its application on intrusion detection. *Expert Syst. Appl.* **41**(13), 5972–5983 (2014)
30. Bolón-Canedo, V., Sánchez-Marroño, N., Alonso-Betanzos, A.: Feature selection and classification in multiple class datasets: an application to KDD Cup 99 dataset. *Expert Syst. Appl.* **38**(5), 5947–5957 (2011)
31. Mitchell, R., Chen, I.-R.: A survey of intrusion detection techniques. *Comput. Secur.* **12**(4), 405–418 (2014)

32. Tavallae, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the KDD CUP 99 data set. In: IEEE Symposium on Computational Intelligence for Security and Defense Applications CISDA 2009 (June 2009)
33. Engen, V., Vincent, J., Phalp, K.: Exploring discrepancies in findings obtained with the KDD Cup'99 data set. *Intell. Data Anal.* **15**(2), 251–276 (2011)
34. Hu, W., Gao, J., Wang, Y., Wu, O., Maybank, S.: Online adaboost-based parameterized methods for dynamic distributed network intrusion detection. *IEEE Trans. Cybern.* **44**(1), 66–82 (2014)
35. Weller-Fahy, D.J.: Network intrusion dataset assessment, p. 114 (2013)
36. Chou, T.-S., Fan, J., Fan, S., Makki, K.: Ensemble of machine learning algorithms for intrusion detection. In: 2009 IEEE International Conference System Man and Cybernetics, pp. 3976–3980 (2009)
37. Li, G., Niu, P., Duan, X., Zhang, X.: Fast learning network: a novel artificial neural network with a fast learning speed. *Neural Comput. Appl.* **24**(7–8), 1683–1695 (2014)
38. Guang-Bin, H., Qin-Yu, Z., Chee-Kheong, S.: Extreme learning machine: a new learning scheme of feedforward neural networks. In: Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference, vol. 2, pp. 985–990. August 2004
39. Smola, A.J., Schölkopf, B.: Learning with Kernels. February 2002
40. Flynn, H., Cameron, S.: Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013, vol. 226 (2013)
41. Trelea, I.C.: The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Inf. Process. Lett.* **85**(6), 317–325 (2003)
42. J. Blondin, “Particle swarm optimization: A tutorial,” ... *Site Http//Cs. Armstrong. Edu/Saad/Csci8100/Pso Tutor.* ..., pp. 1–5, 2009
43. Sengupta, A., Bhadauria, S., Mohanty, S.P.: TL-HLS: Methodology for Low Cost Hardware Trojan Security Aware Scheduling with Optimal Loop Unrolling Factor during High Level Synthesis. *IEEE Trans. Comput. Des. Integr. Circuits Syst.* **36**(4), 660–673 (2017)
44. Mishra, V.K., Sengupta, A.: Swarm-inspired exploration of architecture and unrolling factors for nested-loop-based application in architectural synthesis. *Electron. Lett.* **51**(2), 157–159 (2015)
45. Sengupta, A., Bhadauria, S.: User power-delay budget driven PSO based design space exploration of optimal k-cycle transient fault secured datapath during high level synthesis. In: Proceedings of the International Symposium Quality Electronic Design ISQED, vol. 2015, no. 6, pp. 289–292 (2015)
46. Mishra, V.K., Sengupta, A.: MO-PSE: Adaptive multi-objective particle swarm optimization based design space exploration in architectural synthesis for application specific processor design. *Adv. Eng. Softw.* **67**, 111–124 (2014)
47. Shi, Y., Eberhart, R.: A modified particle swarm optimizer. In: Proceedings of the IEEE International Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence (Cat. No.98TH8360), pp. 69–73 (1998)