Intrusion Detection System Using PCA and Kernel PCA Methods

Z. Elkhadir, K. Chougdali, and M. Benattou

***Abstract*—The network traffic data used to build an intrusion detection system is frequently enormous and redundant with important useless information which decreases IDS efficiency. In order to overcome this problem, we have to reduce as much as possible this meaningless information from the original high dimensional data. To do this, we have compared the performance of two features reduction techniques namely, Principal Component Analysis (PCA) and Kernel Principal Component Analysis (KPCA). After the step of dimension reduction, data samples are classified using k nearest neighbor (K-NN) or decision tree algorithm in order to check whether these samples are normal or anomalous network connection. In this paper, the two well-known KDDcup99 and NSL-KDD databases have been used for testing the proposed approaches. Experimental results show that KPCA with the power kernel performs better than many other types of kernels, especially once we have used the KNN classifier. Additionally, we have noted that KPCA method also overcomes PCA in detecting denial of service (DOS) and probing attacks. Lastly, when we have employed a decision tree classifier, KPCA with the spherical kernel takes the advantage over the same kernels used with KNN.**

***Index Terms*—Network security, Intrusion detection system (IDS), PCA, KPCA.**

I. Introduction

THE security of a computer network is compromised when an intrusion takes place. An Intrusion Detection System (IDS) is an important mechanism that attempts to identify any set of actions or malicious activities which can compromise network security policy. Practically, there are two main intrusion detection techniques: misuse detection and anomaly detection. The misuse detection recognizes a suspicious behavior by comparing it to a specific attack signature that has been already stored in a database of attacks signatures; unfortunately it cant detect new attacks. STAT [1] and Snort [2] are examples of IDS using misuse detection techniques. On the other side, anomaly detection defines normal behavior as a model, and tries to check any deviation from the model and thus decides to generate or not the corresponding alert. Anomaly detection was originally introduced by Anderson [3] and Denning [4] and then implemented in some IDS like IDES [5] or EMERALD [6].

Many concepts have been developed for the anomaly-based IDS, such as machine learning, data mining, neural networks, statistical methods. All of them have been applied directly on the rough high dimensional data without any dimension reduction technique. It can be considered as one of the principal factors contributing in IDS inefficiency.

The main idea behind our proposed work is to reduce original features of database connection records by extracting its relevant information. A simple technique to extract the relevant information contained in a collection of TCP/IP connections is to capture the variance in these connection records. Thus, the extracted information will be used to classify these network connections as normal or attack.

Mathematically speaking, we want to find the principal components of the connection records distribution. To do this, the approach extracts the relevant information using the eigenvectors of the covariance matrix of all connection records [7]. These eigenvectors can be defined as a set of features used to deduce the variation between record connections. Indeed, each connection is expressed using only the eigenvectors with the largest eigenvalues given by the most variance within the set of connection records. The new subspace spanned by these eigenvectors is constructed using the Principal Component Analysis (PCA) which has proven to be efficient in intrusion detection [9][10][11][12].

However, PCA allows only a linear dimensionality reduction [8]. So, if the data has more complicated nonlinear structures, which cannot be well represented in a linear subspace, standard PCA will not be very helpful. As a solution to this weakness, KPCA (Kernel Principal Component Analysis) was introduced to extract principal components by adopting a non-linear kernel method [13] and has also shown a satisfactory results in the field of intrusion detection [14][15][16].

On the other hand, we have noted that the most researchers use KPCA with conventional kernels such as polynomial or Gaussian kernel. In this paper, we propose new kernels namely spherical and power kernel that have not been used before with KPCA.

This paper is organized as follows: Section II is dedicated to present briefly the two dimensionality reduction methods PCA and KPCA with special attention to the proposed kernels. Section III presents the proposed model for IDS. In Section IV, we will describe and discuss the experimental results. Finally, Section V gives the concluding remarks and outlines our future works.

II. PCA and Kernel PCA

In this section, we present a modeling concepts and theoretical analysis of PCA and KPCA methods

A. PCA

Principal component analysis (PCA) is a mathematical technique that transforms a number of correlated variables

Manuscript received July 27, 2015; revised Nov 03, 2015.

K. Chougdali is with GREST Research Group, National School of Applied Sciences (ENSA), Kenitra, Morocco (corresponding author, e-mail: chougdali@yahoo.fr).

Z. Elkhadir and M. Benattou are with LASTID laboratory, Faculty of Science, Ibn tofail University, Kenitra, Morocco.

into a number of uncorrelated variables called principal components (PCs). Generally, the number of these principal components is less than or equal to the number of original variables. The main goal of PCA is to reduce dimensionality of the initial variables, while retaining as much as possible the variance present in these samples. This is achieved by taking only the first few PCs [8].

Suppose we have a training set of M vectors $w_1, w_2, ..., w_M$ each vector contain n features. To get n' $(n' << n)$ principal components of the training set the procedure is based on the following steps:

1) Compute the average σ of this set :

$$\sigma = (\frac{1}{M}) \sum_{i=1}^{M} w_i \tag{1}$$

2) Subtract the mean σ from w_i and get ρ_i :

$$\rho_i = w_i - \sigma \tag{2}$$

3) Compute the covariance matrix C where :

$$C_{n \times n} = (\frac{1}{M}) \sum_{i=1}^{M} \rho_i \rho_i^T = AA^T \tag{3}$$

and

$$A_{n \times M} = (\frac{1}{\sqrt{M}}) \rho_i \tag{4}$$

4) Let U_k be the k^{th} eigenvector of C corresponding to the λ_k associated eigenvalue and $U_{n \times n'} = [U_1, ..., U_{n'}]$ the matrix of these eigenvectors, so we have

$$CU_k = \lambda_k U_k \tag{5}$$

5) Sort the eigenvalues in decreasing order and choose the first corresponding eigenvectors, those eigenvectors are called principal components (PC_i). Practically, the number of the principal components chosen depends on the precision explicitly expressed by

$$\tau = \frac{\sum_{i=1}^{n'} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \tag{6}$$

This ratio defines the information rate kept from the whole rough input data, by the corresponding n eigenvalues.

Finally, the projection of a new column vector sample x_{new} on the space constructed by principal components can be obtained as

$$t_i = PC_i^T x_{new} \tag{7}$$

B. Kernel PCA

Kernel PCA allows us to generalize PCA to nonlinear dimensionality reduction. This can be done by a nonlinear mapping function Φ, that transform all samples input into a higher-dimensional feature space F as follows:

$$\Phi : w \in R^n \rightarrow \Phi(w_i) \in F$$

Where $\Phi(w_i)$ is a sample of F and $\sum_{i=1}^{M} \Phi(w_i) = 0$. The mapping of w_i is simply noted as $\Phi(w_i) = \Phi_i$ and the covariance matrix of this sample in the feature space F can be constructed by

$$C = (\frac{1}{M}) \sum_{i=1}^{M} (\Phi_i - mean)(\Phi_i - mean)^T \tag{8}$$

Where $mean = \sum_{i=1}^{M} \frac{\Phi_i}{M}$. The covariance matrix C can be diagonalized with nonnegative eigenvalues λ satisfying

$$Cv = \lambda_i v \tag{9}$$

Its easy to see that every eigenvector v of C can be linearly expanded by

$$v = \sum_{i=1}^{M} (\alpha_i \Phi_i) \tag{10}$$

To obtain the coefficients α_i, a kernel matrix K with size $M \times M$ is defined and its elements are determined as follows

$$K_{ij} = \Phi_i^T \Phi_j = \Phi_i . \Phi_j = k(w_i, w_j) \tag{11}$$

Where $k(w_i, w_j) =< \Phi_i, \Phi_j >$ is the inner product of two vectors in F. If the projected dataset $\Phi(w_i)$ does not have zero mean, we can use the Gram matrix K' to substitute the kernel matrix K using

$$K' = K - 1_M K - K 1_M + 1_M K 1_M \tag{12}$$

such that $1_M = (1/M)_{M \times M}$. In order to solve the eigenvalue problem in (9), we can reformulate this equation as [13]

$$K' \alpha = M \lambda \alpha \tag{13}$$

Let column vectors α_i be the orthonormal eigenvectors of K' corresponding to the p largest positive eigenvalues $\lambda_1 \geq \lambda_2 \geq .. \geq \lambda_p$. Hence the orthonormal eigenvectors v_i of C can be expressed as

$$v_i = (\frac{1}{\sqrt{\lambda_i}}) \Phi_i \alpha_i \tag{14}$$

For a new column vector sample x_{new}, the mapping to the feature space F is $\Phi(x_{new})$ and then the projection of x_{new} onto eigenvectors v_i is:

$$t = (v_1, v_2, ..., v_p)^T \Phi(x_{new}) \tag{15}$$

The i^{th} KPCA transformed feature t_i can be obtained by

$$t_i = v_i^T \Phi(x_{new}) = (\frac{1}{\sqrt{\lambda_i}}) \alpha_i^T k(w_i, x_{new}) \tag{16}$$

It should be noted that the kernel matrix could be directly constructed from the training dataset. The common kernel functions mostly used are :

⋆ Gaussian kernel :

$$k(x, y) = e^{(\frac{-\|x-y\|^2}{2 \times sigma^2})} \tag{17}$$

⋆ Polynomial kernel :

$$k(x, y) = (x^T y + 1)^d \ where \ d \in N \tag{18}$$

In this paper, we propose to use other kernels which have not received much attention from the scientific community. These kernel functions are:

⋆ Power kernel :

$$k(x, y) = \|x - y\|^d \ where \ d \geq 1 \tag{19}$$

⋆ Rational Power kernel :

$$k(x,y) = \|x-y\|^d \ \ where\ 0 < d < 1 \quad (20)$$

⋆ Log kernel :

$$k(x,y) = -log(\|x-y\|^d + 1) \ where\ d \geq 1 \quad (21)$$

⋆ Spherical kernel :

$$k(x,y) = 1 - \frac{3}{2}(\frac{\|x-y\|}{d}) + \frac{1}{2}(\frac{\|x-y\|}{d})^3 \ , d \geq 1 \quad (22)$$

III. Proposed model for IDS

The architecture and the main idea of our IDS have been shown in the Fig 1. There are different phases in the proposed architecture for an efficient network IDS.

Firstly, one database (KDDcup or NSL-KDD) is selected during this phase. Secondly, we randomly split the original dataset into two parts, training subset and testing subset. These two parts of raw dataset are preprocessed in order to have a standard feature format.

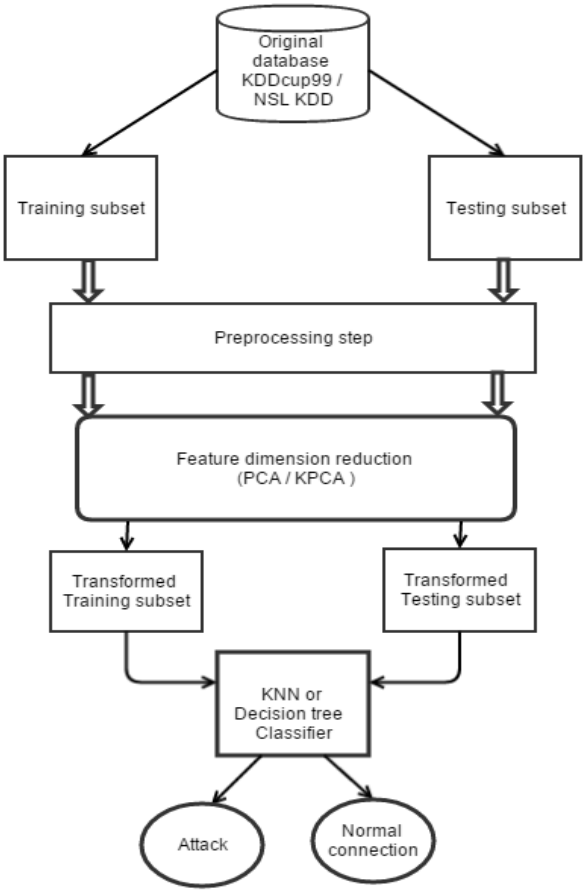


Fig. 1: The diagram flow of the proposed IDS.

In the training phase, a feature reduction method is used to extract the optimal features from the original high-dimensional data and get a new training subset. During the testing phase, features which were selected in the training phase will be required to obtain a new testing subset.

Finally, in the classification phase, thanks to the training subset and the employment of a classifier (KNN [17] or Decision trees [18]) the IDS will decide whether the testing samples are normal or not.

A. Selection of suitable Dataset

1) KDDcup99: The KDDcup99 dataset is the most popular database that has ever been used in the intrusion detection field. It has been widely used in many contest [19] in order to present a predictive model able to recognize legitimate (normal) and illegitimate (called intrusion or attacks) connections in a computer network. The entire training dataset contained about 5,000,000 connection records. In this paper we work only with the 10% training dataset consisted of 494,021 records which contain 97,278 normal connections (i.e. 19.69%). Each TCP connection record is composed of 41 different attributes that describe the corresponding connection, and the value of the connection is labeled either as an attack with one specific attack type, or as normal. Each attack type falls exactly into the following four categories:

1) Probing: surveillance and other probing, e.g., port scanning;
2) DOS: denial-of-service, e.g. syn flooding;
3) U2R: unauthorized access to local superuser (root) privileges, e.g., various buffer overflow attacks;
4) R2L: unauthorized access from a remote machine, e.g. password guessing.

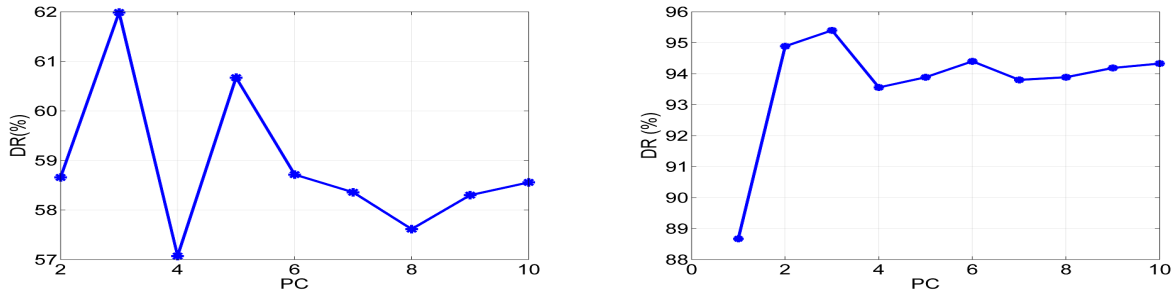
The test database is composed of 311,029 connections. It is important to note that the test data includes some specific attack types which doesn't exist in the training data. In details, there are 4 new U2R attack, 7 new R2L attack types, 4 new DOS attack, and 2 new Probing attacks types in the test dataset that are not present in the training dataset.

2) NSL-KDD: This dataset has been suggested to solve some of the inherent problems of the KDDcup99 dataset [20]. The advantages of NSL-KDD over the original KDDcup99 can be resumed as follows :

1) It does not include redundant records in the training set, so the classifiers will not be biased towards more frequent records.
2) There is no duplicate records in the proposed test dataset; therefore, the performance of the learners are not biased by the methods which have better detection rates on the frequent records.
3) The number of selected records from each difficulty level group is inversely proportional to the percentage of records in the original KDD data set. As a result, the classification rates of distinct machine learning methods vary in a wider range, which makes it more efficient to have an accurate evaluation of different learning techniques.
4) The number of records in the training and testing datasets are reasonable, which makes it affordable to run the experiments on the complete set without the need to select a small portion. Consequently, evaluation results of different research works will be consistent and comparable

B. Preprocessing step

The datasets are defined by continuous and discrete attributes values. We have transformed the discrete attributes values to continuous values by applying the transformation concept used in [9]. Let's describe this process briefly: If a discrete attribute i has k values. we correspond i to k coordinates composed of one's and zero's. After that, we will



(a) (b)

Fig. 2: Detection rate (%) vs. number of principal components (PC) for two datasets: (a) KDDcup99 and (b) NSL-KDD

have one coordinate for every possible value of the attribute. For instance if we consider the protocol type attribute which can take the following discrete attributes tcp, udp or icmp. According to the idea, there will be three coordinates for this attribute. As a consequence, suppose a connection record contains a tcp (resp. udp or icmp) then the corresponding coordinates will be (1,0,0) (resp. (0,1,0) or (0,0,1)). With this technique, each connection record in the datasets will be represented by 125 coordinates.

IV. Experiments and discussion

This section is dedicated to evaluate the results obtained when applying the two dimensionality reduction methods PCA and KPCA in combination with K-NN and decision tree classifiers. In order to show the effectiveness of the proposed approaches, we have conducted extensive experiments using both KDDcup99 and NSL-KDD datasets previously presented. For KDDCup99, as a training set, we have selected randomly 1000 normal, 100 DOS, 50 U2R, 100 R2L and 100 PROBE from the 10% training dataset. The test samples is composed of 100 normal data, 100 DOS data, 50 U2R data, 100 R2L data and 100 PROBE randomly selected from the test dataset. For NSL-KDD, the simulation settings are the same as those used in KDDcup99, nevertheless the training and testing sets are selected from the same original dataset.

The performance of an IDS is evaluated by its ability to make correct predictions. To examine the performance of the proposed system we have used two measures: detection rate (DR) and false positive rate (FPR) defined as follows:

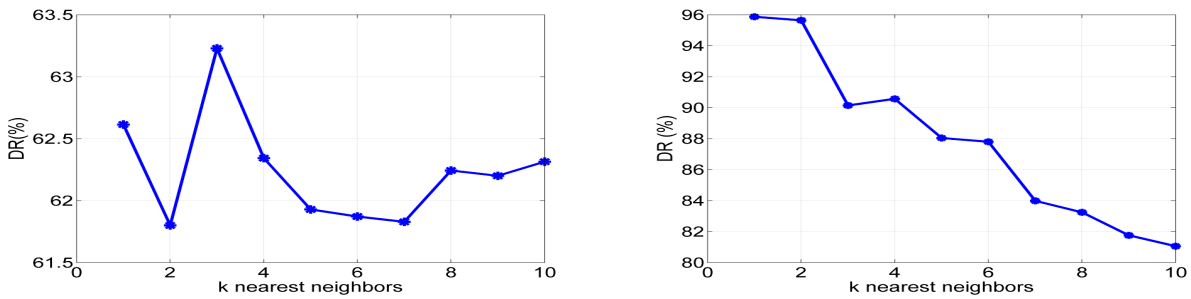
$$DR = \frac{TP}{TP + FN} \times 100 \quad (23)$$

$$FPR = \frac{FP}{FP + TN} \times 100 \quad (24)$$

Where true positives (TP) correspond to intrusions correctly predicted. False negatives (FN) refer to intrusions wrongly classified; false positive (FP) are normal instances wrongly classified, and true negatives (TN) are normal instances successfully predicted. Hence, based on these performance indicators, an efficient IDS should have a high DR and a low FPR.

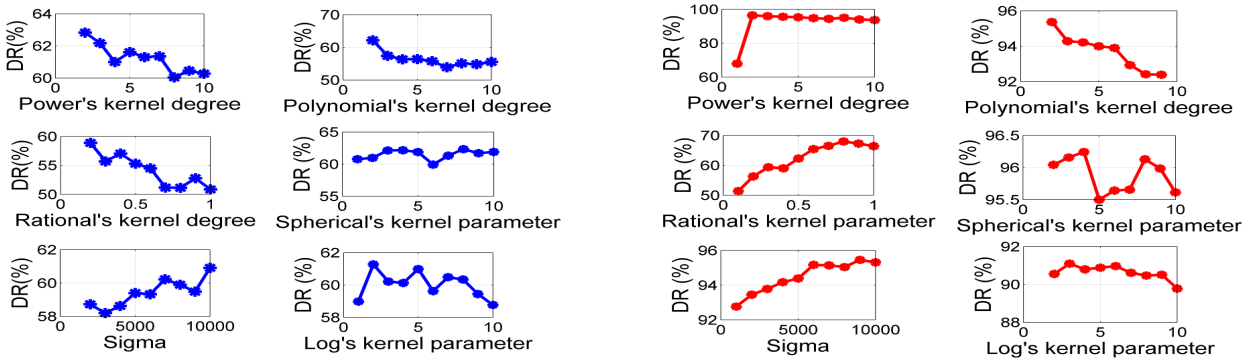
In the first experiment, we have performed PCA on training samples and hence we have obtained the principal components (PC). The number of PC determines the dimension of the new reduced samples. Then, we project the test samples on the subspace spanned by these principal components, varying their numbers. The objective of this experiment is to seek the optimal number of PCs which contribute significantly in increasing detection rate (DR). Fig 2 shows that, only the first three principal components give a highest detection rate with inertia ratio $\tau > 0.99$ (equation (6)) for both datasets.

A second experience tries to determine the number of neighbors (i.e., k) that yields the best detection rate. To do that, we have fixed the number of Principal Components at three and we have varied the number of nearest neighbors from a wide range of values. For KDDcup99, as shown in Fig 3a, we can choose $k = 3$ nearest neighbors which gives the optimal detection rate. Whereas, for NSL-KDD we take only one nearest neighbor (Fig 3b) to achieves a maximum detection rate for PCA. According to the first and the second experience, we have fixed the number of PCs and the number of nearest neighbors at their adequate values and try to find the optimal rate in detecting every type of attacks (DOS, U2R, R2L, and PROBE). From the Table I, we can see that for KDDCup99, the two categories of attacks DOS and PROBE are detected with a rate of 95,13 % for DOS and



(a) KDDcup99 (b) NSL-KDD

Fig. 3: Detection rate (%) vs. number of nearest neighbors on two datasets



(a) KDDCup99 dataset

(b) NSL-KDD dataset

Fig. 4: Detection rate (%) of KPCA versus different kernels parameters

TABLE I: Attacks detection rate (%) of PCA

Database	DOS	U2R	R2L	PROBE
KDDCup99	95,1333	8,1333	3,5667	69,8667
NSL-KDD	87,95	98,80	93,85	80,15

69,8 % for PROBE. However, the U2R and R2L attacks are not well detected with only 8,13 % and 3,5 %, respectively.

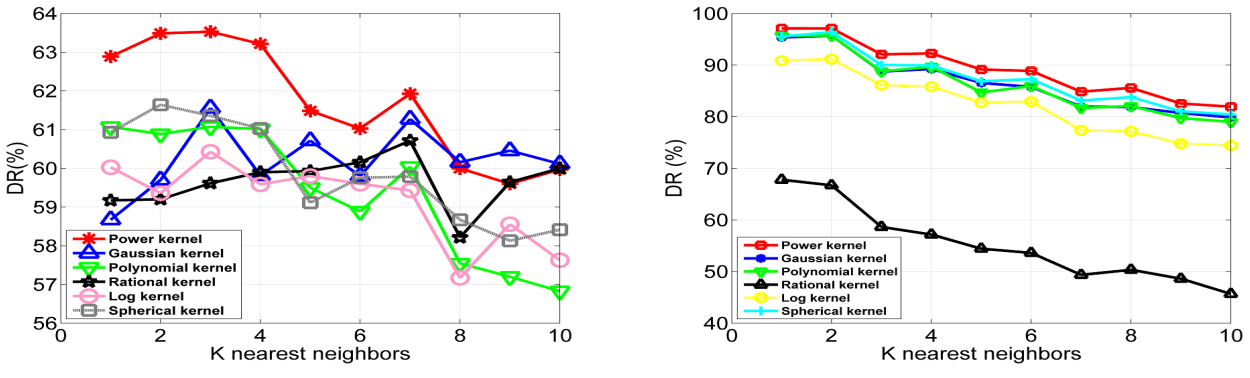
In the other hand, the results changed when it comes to NSL-KDD as shown in Table I. We can note that DOS and PROBE attacks are highly detected with a rate of 87,95 % and 80,15 %. Furthermore, in contrast to the first dataset, U2R and R2L are also well identified as attacks with 98,80 % and 93,85 % respectively.

In the second part of our experiments, we will show how we can tune Kernel PCA to be more efficient. For that, we have evaluated the effectiveness of KPCA in intrusion detection system by executing many steps. Firstly, we implement six kernels, described by equations (17), (18), (19), (20), (21), (22). Secondly, we try to pick up the maximum detection rate for KPCA by varying the different kernels parameters. As illustrated in Fig 4a, the results which concern KDDcup99 reveal that we can take the following optimal values for the different kernels parameters: degree $d = 2$ for polynomial, power and log kernels, $d = 0.2$ for rational power, $sigma = 10000$ for Gaussian kernel and $d = 8$ for spherical kernel. On the other side, from Fig 4.b, the best values for kernels parameters which give the maximum DR for KPCA on NSL-KDD are: $d = 2$ for polynomial and power kernel, $d = 3$ for log kernel, $d = 0.8$ for rational power, $sigma = 9000$ for Gaussian kernel and $d = 4$ for spherical kernel.

A next step seeks to identify the best kernel for KPCA. To achieve this goal, we exploit the previous results and fix the kernels parameters to their optimal values. From Fig 5, we can observe that the power kernel has a higher detection rate for the two databases in comparison with other kernels. Once the appropriate kernel for KPCA method is found, a next experiment will compare this method to the classical PCA on the both database. For KDDCup99, it can be easily seen that KPCA with the power kernel outperforms PCA when number of nearest neighbors is between 1 and 4 (Fig 6a). However, PCA gives fewer false positive alarms (Fig 6c). For the other NSL-KDD database, it is observable that the same kernel maintains it superiority over PCA in term of attacks detection (Fig 6b) and also in producing less false positives (Fig 6d). To further demonstrate the efficiency of KPCA, Table II illustrates the detection rate for every type of attacks

TABLE II: Attack's detection rate (%) of KPCA

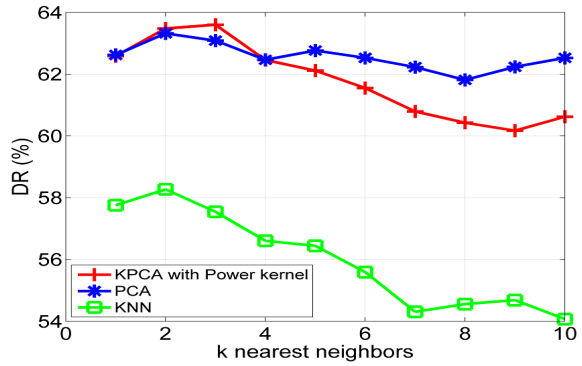
Database	DOS	U2R	R2L	PROBE
KDDcup99	96,133	9,9333	3,4667	73,2667
NSL-KDD	93	99	95,30	89.05



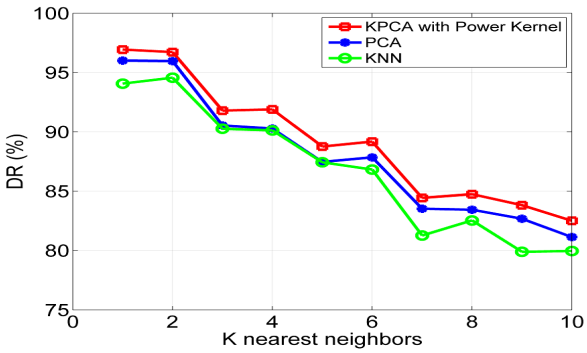
(a)

(b)

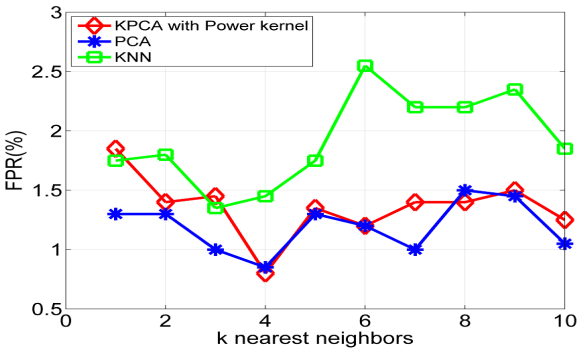
Fig. 5: Detection performance of KPCA using different kernels for the (a) KDDCup and (b) NSL-KDD datasets.



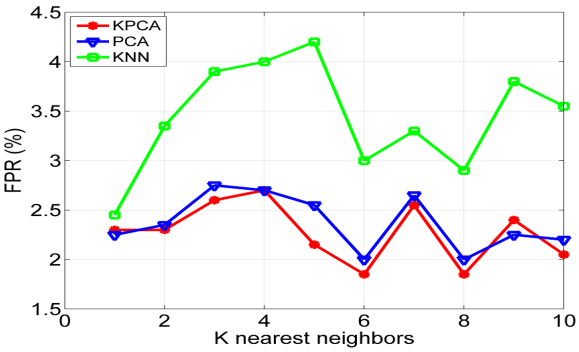
(a) KDDcup



(b) NSL-KDD



(c) KDDCup



(d) NSL-KDD

Fig. 6: Detection rate (%) and FPR (%) of KPCA, PCA and KNN methods with different neighbor number.

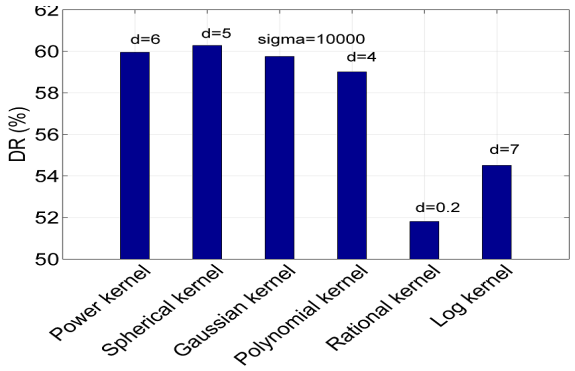
(DOS, U2R, R2L and PROBE), and tells us that the two categories of attacks DOS and PROBE are well detected with a rate of 96,13% for DOS and 73,8% for PROBE attacks. Furthermore, we can conclude that these detection rates are better than those found with PCA (95,13% and 69,8%). In the other hand, U2R and R2L attacks are not well detected with only 9,93% for U2R (slightly better than PCA which gives 8,13%) and just 3,46%. Moreover, we note that for NSL-KDD the identification of attacks is also more accurate than those found with PCA.

In this stage of our experiments, instead of using KNN classifier we work with decision trees classifier and we look for the kernel which gives highest DR. So, like what we have done with KNN, we compare the performance of the kernels at their best and the result is illustrated in Fig 7. We can see clearly that the spherical kernel outperforms all the other

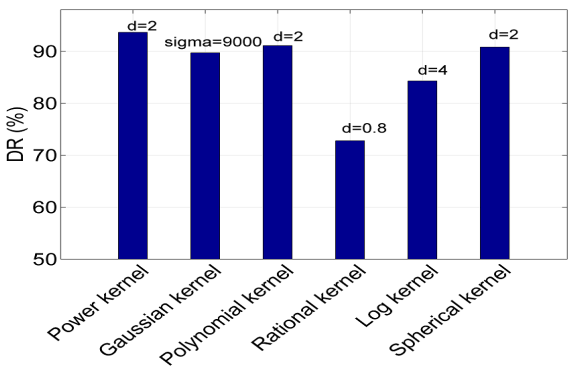
TABLE III: Attack's detection rate (%) of PCA and KPCA

Database	The method	DOS	U2R	R2L	PROBE
KDDcup99	PCA	81,7	21,6	2,3	70,5
	KPCA	93,7	13,7	3,15	78
NSL-KDD	PCA	90,35	93,6	87,2	85,15
	KPCA	90,2	92,6	87,25	85,45

kernels for KDDcup99 but for NSL-KDD the power kernel still the best one. To go deeper in detection attacks we expose in TABLE III the detection of every type of attacks for PCA and KPCA. It is shown that the detection rates of KPCA for DOS and PROBE attacks are globally the best compared

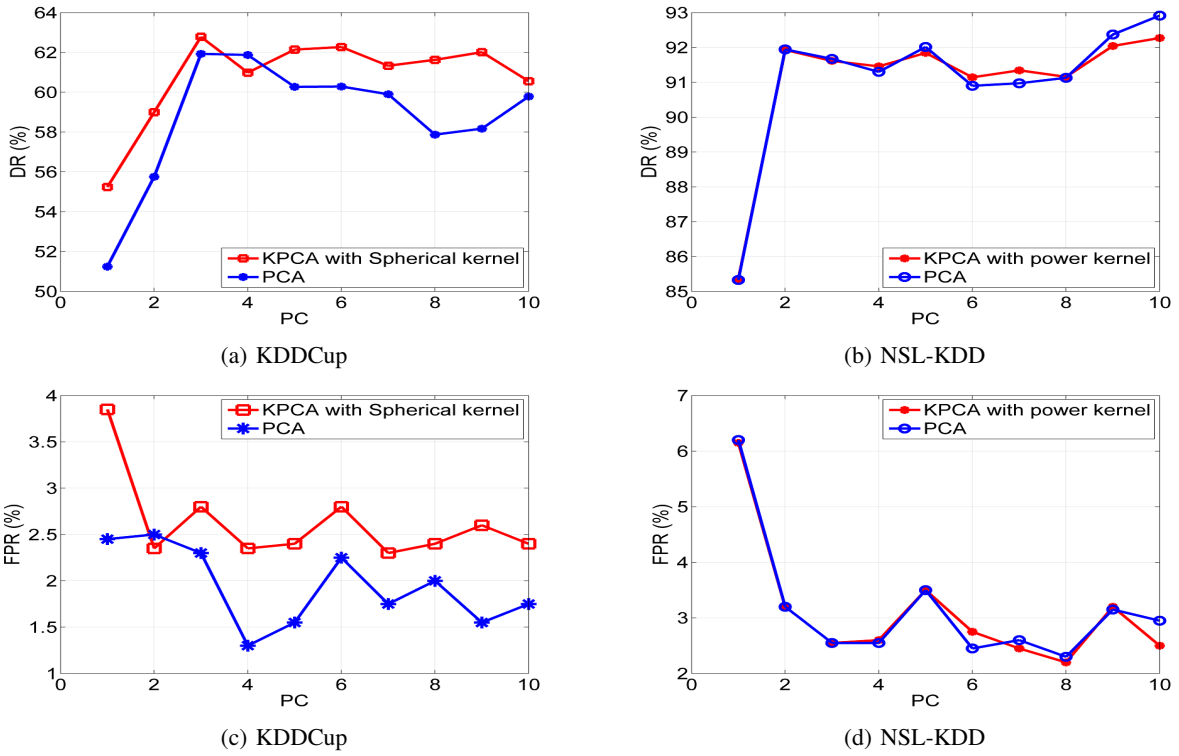


(a) KDDCup99



(b) NSL-KDD

Fig. 7: Performance comparison of various kernels for KPCA



(a) KDDCup (b) NSL-KDD

(c) KDDCup (d) NSL-KDD

Fig. 8: Detection rate (%) and FPR(%) of KPCA and PCA methods under different dimensionality

to those of PCA, even if the detection rate of U2R is not acceptable.

In the last experience, using decision tree classifier, we have compared the DR and FPR for the two algorithms. From Fig 8a and Fig 8b, which summarizes the comparison of KPCA and PCA performance, we retain that KPCA on KDDCup99 with spherical kernel outperforms PCA when we vary the number of principal components from one to ten. Otherwise, KPCA with power kernel give nearly the same results as PCA on NSL-KDD. Unfortunately, in comparison with PCA, KPCA method has a worse false alarm rate even if it has a better detection rate especially for KDDCup (Fig 8c).

V. Conclusion

The main idea behind the work presented in this paper is to reduce the original features that represent all connection records stored in a dataset for the purpose of intrusion detection. The proposed work shows, how we can extract relevant information using PCA and KPCA in order to build a robust IDS with the maximum detection rate and minimum false alarms. Experimental results show that KPCA with the power kernel performs better than many other types of kernels, especially once we have used the KNN classifier. Additionally, we have noted that KPCA method also overcomes PCA in detecting denial of service (DOS) and probing attacks. Lastly, when we have employed a decision tree classifier, KPCA with the spherical kernel takes the advantage over the same kernels used with KNN. Our future works will be oriented towards advanced dimension reduction techniques in order to improve the performance of an IDS, particularly for the CPU time consuming.

References

[1] S. Kumar and E.H. Spafford, “A Software architecture to support misuse intrusion detection”, *Proceedings of the 18th National Information Security Conference*, pp. 194-204, 1995.

[2] J. Beale, “Snort 2.1 Intrusion Detection”, *Caswell (Editor)*, Syngress, 2004.

[3] J.P. Anderson, “Computer Security Threat Monitoring and Surveillance”, *Technical report*, James. P. Anderson Co., Fort Washington, Pennsylvania, 1980.

[4] D. Denning, “An Intrusion Detection Model”, *IEEE Transactions on Software Engineering*, pp. 222-232, 1987.

[5] T.Lunt, A.Tamaru, F.Gilham, et al, “A Real-time Intrusion Detection Expert System (IDES) final technical report”, *Technical report*, Computer Science Laboratory, SRI International, Menlo Park, California, Feb. 1992.

[6] P.A. Porras, P.G. Neumann, “EMERALD:Event monitoring enabling responses to anomalous live disturbances”, *Proceedings of National Information Systems Security Conference*, Baltimore MD, Oct. 1997.

[7] I.T. Jolliffe, Principal Component Analysis. Springer-Verlag, New York, 2002.

[8] M. Kirby and L. Sirovich, “Application of the Karhunen Loeve Procedure for the Characterization of Human Faces”, *IEEE Transactions On Pattern Analysis and Machine Intelligence*, vol. 12, no. 1, pp.103-107, January 1990.

[9] Y. Bouzida, F.Cuppens, N.Cuppens-Boulahia and S.Gombault, “Efficient intrusion detection using principal component analysis”, In *3eme Conference sur la Securite et Architectures Reseaux (SAR)*, La Londe, France, June, 2004.

[10] I.Ahmad, A.Abdullah, A.Alghamdi, M.Hussain and K.Nafjan, ”Intrusion detection using feature subset selection based on MLP”, Sci Res Essays, vol. 6, no. 34, pp.6804-6810, 2011.

[11] M.Panda, A.Abraham and M. R.Patra, ”A hybrid intelligent approach for network intrusion detection”, Procedia Engineering, vol. 30, pp.1-9, 2012.

[12] I.Ahmad, M.Hussain, A.Alghamdi and A.Alelaiwi, ”Enhancing SVM performance in intrusion detection using optimal feature subset selection based on genetic principal components”, Neural Computing and Applications, vol. 24, no. 7, pp.1671-1682, 2014.

[13] B. Scholkopf, A. Muller and K. R.: ”Nonlinear component analysis as a kernel eigenvalue problem”. Neural Computation, vol. 10, no. 5, pp.1299-1319, 1998.

[14] F.Kuang, W.Xu, and S.Zhang, ”A novel hybrid KPCA and SVM with GA model for intrusion detection”, Applied Soft Computing, vol. 18, pp.178-184, 2014.

[15] L.S.Chen and J.S.Syu, "Feature Extraction based Approaches for Improving the Performance of Intrusion Detection Systems", *In Proceedings of the International MultiConference of Engineers and Computer Scientists*, pp.286-291, 2015.
[16] R.Beghdad, Y.Ziraoui and N.Kouache, "Kernel methods to detect intruders", In IEEE 3rd International Conference Control, Engineering and Information Technology (CEIT), pp.1-6, 2015.
[17] T. M. Cover and P. E. Hart, Nearest neighbor pattern classification, IEEE Trans. Inform. Theory, vol. 13, no. 1, pp.2127, 1967
[18] L.Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Boca Raton, CRC Press, 1984.
[19] KDDcup99 task. Available at: http://kdd.ics.uci.edu/databases/kddcup 99/task.html.
[20] NSL-KDD dataset. Available at http://nsl.cs.unb.ca/NSL-KDD/