Lopez-Martin, M.; Carro, B.; Sánchez-Esguevillas, A.; Lloret, J. (2019). Shallow neural network with kernel approximation for prediction problems in highly demanding data networks. Expert Systems with Applications. 124:196-208. https://doi.org/10.1016/j.eswa.2019.01.063

Elsevier Editorial System(tm) for Expert Systems With Applications

Manuscript Draft

Manuscript Number:

Title: Shallow neural network with kernel approximation for prediction problems in highly demanding data networks

Article Type: Full length article

Keywords: shallow neural network; kernel approximation; intrusion detection; network traffic classification

Corresponding Author: Mr. Manuel Lopez-Martin, M.D.

Corresponding Author's Institution: Universidad de Valladolid

First Author: Manuel Lopez-Martin, M.D.

Order of Authors: Manuel Lopez-Martin, M.D.; Belen Carro, PhD; Antonio Sanchez-Esguevillas, PhD; Jaime Lloret, PhD

Research Data Related to this Submission

There are no linked research data sets for this submission. The following reason is given:
The data sets used are public.

Shallow neural network with kernel approximation for prediction problems in highly demanding data networks

Manuel Lopez-Martin, Belen Carro, Antonio Sanchez-Esguevillas and Jaime Lloret

Abstract— Intrusion detection and network traffic classification are two of the main research applications of machine learning to highly demanding data networks e.g. IoT/sensors networks. These applications present new prediction challenges and strict requirements to the models applied for prediction. The models must be fast, accurate, flexible and capable of managing large datasets. They must be fast at the training, but mainly at the prediction phase, since inevitable environment changes require constant periodic training, and real-time prediction is mandatory. The models need to be accurate due to the consequences of prediction errors. They need also to be flexible and able to detect complex behaviors, usually encountered in non-linear models and, finally, training and prediction datasets are usually large due to traffic volumes. These requirements present conflicting solutions, between fast and simple shallow linear models and the slower and richer non-linear and deep learning models. Therefore, the perfect solution would be a mixture of both worlds. In this paper, we present such a solution made of a shallow neural network with linear activations plus a feature transformation based on kernel approximation algorithms which provide the necessary richness and non-linear behavior to the whole model. We have studied several kernel approximation algorithms: Nystrom, Random Fourier Features and Fastfood transformation and have applied them to three datasets related to intrusion detection and network traffic classification.

This work presents the first application of a shallow linear model plus a kernel approximation to prediction problems with highly demanding network requirements. We show that the prediction performance obtained by these algorithms is positioned in the same range as the best non-linear classifiers, with a significant reduction in computational times, making them appropriate for new highly demanding networks.

Index Terms—shallow neural network; kernel approximation; intrusion detection; network traffic classification.

* Correspondence: manuel.lopezm@uva.es; Tel.: +34-983-423-980, Fax: +34-983-423-667
The authors declare that there is no conflict of interest regarding the publication of this paper

I. INTRODUCTION

Considering the importance of current security attacks on modern networks with the highest demands (e.g. IoT networks), the economic importance of services running on these networks and the increased demands on data networks imposed by these services, it is more important than ever to rely on new automatic systems capable of detecting intrusions in a fast and reliable manner. Such a system is an Intrusion Detection Systems (IDS) [1], being its final goal to fast and accurately analyze network traffic and to predict potential threats. Similarly, Network Traffic Classification (NTC) [2][3] is also an important prediction task in data networks. NTC uses network traffic information to predict the type of services of a network flow. This prediction is important to address network management and quality of service tasks which are dependant of the type of service of the data connection.

IDS and NTC are identified as two of the main research applications of machine learning for data networking [4][5] and are representative of many other applications. Both IDS and NTC present important prediction problems for data networks. Both deal with large, noisy, unbalanced and complex data. The labels to predict have a very (sometimes extremely) unbalanced distribution. The data to process, in both problems, is usually large, and, the features extracted from the network traffic are complex with usually a noisy assignment of labels to its corresponding ground-truth state, due to the difficulty to ascertain the true value of the intrusion state (with manual detection or intrusion detection tools) or type of traffic (with deep packets inspection tools).

This work has manifold objectives: (1) To demonstrate that shallow linear models can be a competitive alternative to more complex models, when former models are used together with a feature transformation based in kernel approximation (KA) theory. Simple linear models with a feature transformation based on KA have the prediction and training speeds of a simple

M. Lopez, B. Carro and A. Sanchez are with Dpto. TSyCeIT, ETSIT, Universidad de Valladolid, Paseo de Belén 15, Valladolid 47011, Spain; (manuel.lopezm@uva.es; belcar@tel.uva.es; antoniojavier.sanchez@uva.es).

J. Lloret is with Instituto de Investigación para la Gestión Integrada de Zonas Costeras, Universitat Politècnica de València, Camino Vera s/n, Valencia 46022, Spain; (jlloret@dcom.upv.es).

linear model and the detection performance of complex non-linear models (e.g. kernel-Support Vector Machines) or deep learning models (e.g. Convolutional Neural Networks), making them a perfect alternative for prediction problems in data networks. (2) To analyze the different architectural alternatives for the linear model and KA transformations and justify the reasons for the selected architecture. (3) To propose a model that can be trained with a differentiable loss function in modern high-performance platforms (e.g. Tensorflow). (4) To provide a thorough comparison of our proposed model with several Machine Learning (ML) models for some important and representative prediction problems in data networks (e.g. IDS and NTC), with a focus in prediction performance, prediction and training times, model flexibility, model richness and capacity to deal with large datasets.

In order to generalize as much as possible the results obtained, to as many prediction problems as possible, we have compared all the models using three different datasets: NSL-KDD [6], ADFA UNSW-NB15 [7][8] and Moore [9] datasets, each with different characteristics and objectives (Section III.A): two datasets are related with intrusion detection systems (IDS) and one with network traffic classification (NTC).

The proposed model consists of a linear classifier based on a Neural Network (NN) with linear activations, with a previous KA transformation (Section 4.3) of the input features. The study covers several variants of KA transformations based on different algorithms: Nystrom, RFF and Fastfood transform [10][11][12].

The KA transformations provide the nonlinearity among features, which is needed to give flexibility and richness for pattern detection. We have also studied different configurations for the linear model, all of them based on a linear NN with different activation and loss functions, extracting interesting conclusions about the best architectures for this task.

To evaluate the model's performance, we have compared it with the following supervised machine learning models: Linear and Radial Basis Function (RBF) Support Vector Machine (SVM), Multilayer Perceptron (MLP), Gradient Boosting Machine (GBM), Random Forest, AdaBoost, Multinomial Logistic Regression and Convolutional Neural Networks (CNN). The comparison was implemented for the three datasets.

Considering the dependence of the results when performing a classification with a different number of labels, we repeated the experiment for the three datasets considering two groups of results: one with a binary classification and another with a multilabel classification. In addition, a Wilcoxon signed-rank test guarantees the significance of the results.

Along with the importance given to classification performance metrics, we provide an exhaustive study of the training and prediction times of the different algorithms considered for the study. Computation times are of crucial importance when the classification must be carried out in time-critical scenarios or when changes in the environment demand flexibility and rapid reaction of the proposed classifier.

The paper is organized as follows: Section II identifies related works. Section III presents the work performed and the models analyzed in the paper. Section IV shows the results obtained and finally, Section V provides discussion and conclusions.

II. Related works

There are no works presenting results of the application of linear models plus KA transformations for IDS and NTC problems, but there are many considering other prediction models. In this section we present the most representative of these works applied to the NSL-KDD, UNSW-NB15 and Moore datasets.

Being a mature dataset there are many works providing results for classification models applied to the NSL-KDD [6] dataset: In [13], applying an MLP with three layers, they report an accuracy of 79.9% for test data for the 5-labels prediction scenario. For the scenario of 2-label prediction, it is obtained an accuracy of 81.2% for test data. Authors in [14], for the 2-labels scenario and using Self Organizing Maps (SOM), report a recall of 75.49% on test data. The work in [15] for the 2-labels scenario, and using Naive Bayes with several feature engineering methods, reports an accuracy of 96.5% but the test set used is unclear. Similarly, in [16] they report an accuracy of 99.1% using several methods (SVM, NB...) with a previously performed dimensionality reduction on the features, but again, it is not clear the test set used, and the metrics are given on subsets of the anomaly types. Again, in [17] they report an accuracy of 99.9% with AdaBoost and a selection of features using a wrapper model; they use a subset of the NSL-KDD dataset for training and an unclear test set. In [18] for the 2-labels scenario and using AdaBoost with weak learners being simple decision stumps, they report a detection rate of 90% on test data. In [19] using AdaBoost with Naive Bayes as weak learner and a previous feature selection, they report an F1 of 98% for the 5-labels scenario; test results are based on 10-fold cross validation over the training data, not on the test set. The work in [1] explains why and how the NSL-KDD data set was created. They provide results of applying several methods to the NSL-KDD data. The best accuracy reported is 82.02% with Naive Bayes Tree using Weka. They use the full NSL_KDD dataset for training and testing for the 2-labels prediction scenario. Finally, in [20] is proposed a variational autoencoder to perform detection on NSL-KDD with a 5-labels configuration obtaining an overall accuracy of 80%.

Compared with NSL-KDD, the UNSW-NB15 [7][8] is a much modern dataset for intrusion detection with a larger training and test set. There are several works that present prediction results for this dataset: In [21] three classification algorithms;

Expectation-Maximization (EM) clustering, Logistic Regression (LR) and Naive Bayes (NB) are applied to UNSW-NB15 achieving a best accuracy of 83% for LR, with a previous feature selection based on the central points of attribute values and an Association Rule Mining algorithm. These results are obtained with the 2-labels configuration of the dataset. With the same 2-labels configuration, five models are used in [8]: NB, Decision Tree (DT), Artificial Neural Network (ANN), LR and EM Clustering, with DT obtaining the best classification accuracy (85.56%). In [22] the authors employ a DT to achieve an accuracy of 81.42% for the 2-labels configuration; previously they apply a feature selection wrapper approach, based on a combination of genetic and logistic regression algorithms to select the best subset of features. In [23] a previous separation of TCP and UDP traffic is performed for training with a Reduced Error Pruning Tree (REPTree) algorithm, obtaining an accuracy of 88.9% for the 2-labels and 81.2% for the 10-labels configuration of the UNSW-NB15 dataset.

The Moore [9] dataset is a well-known dataset for classification of network traffic from Cambridge University. There are several works reporting prediction results for this dataset: In [24] an MLP is applied in to the Moore dataset obtaining a classification accuracy of 96% for a 10-labels configuration of the dataset. A very similar accuracy is achieved in [25], for the same dataset, using a Directed Acyclic Graph-Support Vector Machine. And, in [26] is presented a modification of the C4.5 decision tree algorithm to classify a 12-labels configuration of the Moore dataset, reporting a one vs. rest accuracy moving between 60-90%, depending on the label, with only two labels with an accuracy higher than 90%. Authors of the Moore dataset propose in [27] a Naïve Bayes classifier with an average accuracy of 66%, which is further improved by refinements in [28] to achieve an accuracy of 95%. These refinements consist in a Naïve Bayes model based on kernel-estimates and feature selection based on Fast Correlation-Based Filter (FCBF). Finally, in [9] same authors propose a better model using C4.5 obtaining an accuracy of 94-99% depending on the data configuration used for training and test.

There are few works that apply KA techniques to data networks prediction problems: In [29][30] are presented several KA methods applied to intrusion detection. They use KA with the LS-SVM model to reduce the dimensionality of the input dataset. They apply the model to the KDD-99 dataset with results not comparable to results from NSL-KDD.

There is no reported work applying KA methods for NTC. There is a modern work [31] presenting the application of KA methods for speech recognition. It has been applied to visual recognition in [32] with results comparable to best known methods with a significant reduction in training and prediction times.

III. WORK DESCRIPTION

In this Section we describe the datasets chosen to carry on the experiments, the different variants of KA algorithms used for feature transformation and the best model that we believe can tackle the diverse requirements imposed to data networks problems, which are: (1) fast prediction and training times, (2) to deal with unbalanced, noisy and large datasets, and (3) to detect complex and non-linear patterns in the data.

The datasets employed are described in Section III.A. The proposed model is presented in detail in Section III.B.

A. Selected datasets

To verify the capabilities of the proposed model in an environment as close as possible to the different requirements imposed by a prediction problem in a data network, we have selected three well-known datasets designed with different objectives: the NSL-KDD datasets [6], ADFA UNSW-NB15 [7] [8] and Moore [9]. Each one offers the opportunity to evaluate the performance of the models under different restrictions and goals. Two of the datasets are related to IDS and one to NTC. They vary in size from medium to large and with different distributions of continuous and categorical variables. All datasets are intended for classification.

1) NSL-KDD dataset (intrusion detection)

The NSL-KDD [6] dataset is a classic well-known IDS dataset. It solves the problem of redundant records in the KDD-99 [6] dataset, which causes the learning algorithms to be biased towards the more frequent records. For this reason, the detection scores on the KDD99 are usually much higher than for the NSL-KDD dataset.

The NSL-KDD dataset provides 125973 training samples and 22544 test samples, with 41 features, being 38 continuous and 3 categorical (discrete valued). We have performed an additional data transformation: scaling all continuous features to the range [0–1] and one-hot encoding all categorical features. This provides a final dataset with 122 features: 38 continuous and 84 with binary values ({0, 1}) associated to the three one-hot encoded categorical features. This is a very unbalanced dataset with a frequency of 43.1% and 1.7% for the most and least frequent labels.

Each training sample has a label output from 23 possible labels (normal plus 22 labels associated to different types of anomaly). The test data has the same number of features (41) and output labels from 38 possible values. That means that the test data has anomalies not presented at training time. The 23 training and 38 testing labels have 21 labels in common; 2 labels only

appear in training and 17 labels are unique to the test dataset. Up to 16,6% of the samples in the test dataset correspond to labels unique to the test dataset, and which were not present at training time. The existence of new labels at testing introduces an additional challenge to the learning methods.

To facilitate interpretation of results the labels have been aggregated into meaningful categories. As presented in [6], the training/testing labels can be associated to one of five possible categories: NORMAL, PROBE, R2L, U2R and DoS. All the above categories correspond to an intrusion except the category: NORMAL, which implies that no intrusion is present. We have considered these five categories as the final labels driving our results (Section IV). These new labels are useful for characterizing intrusions, maintaining a fairly unbalanced distribution (an important feature of intrusion data) and with a number of samples, in each category, large enough to provide significant results.

The results presented in Section IV for this dataset also include a different prediction setup for two labels (NORMAL and INTRUSION values). In this case, the INTRUSION value corresponds to any original label different to NORMAL. This two-labels setup has been included to assess the performance of the model under different number of prediction labels.

2) UNSW-NB15 dataset (intrusion detection)

The UNSW-NB15 [7][8] is a much more modern IDS dataset than NSL-KDD. It was released in 2015. It includes a mixture of normal network activity with modern attack type behaviors and modern normal traffic scenarios.

The dataset consists of 2540044 samples with a test set of 82332 samples. It is also provided a training subset of 175341 samples, which has been used in this work. This is a larger dataset than NSL-KDD and includes nine attack types, and 42 useful features that are created by summarizing the information of the data packets exchanged in real network dataflows. The 42 features are divided into continuous (39) and categorical (3), which after one-hot encoding produce a final dataset with 196 features. The continuous features are also scaled to the range [0-1].

Similar to the NSL-KDD dataset, labels (attack types) are very unbalanced, with a frequency of 44.9% for the most frequent label (NORMAL) and 0.05% for the least frequent. Unlike NSL-KDD, the distribution of labels for the training and test sets is similar for this dataset.

The results in Section IV for this dataset also include a configuration of two labels (NORMAL and INTRUSION values), with the value of INTRUSION associated with any of the nine attack types. For the same IDS classification problem, this dataset provides different constraints and challenges to the ones imposed by the NSL-KDD dataset, which allows a better generalization of the results obtained (Section IV)

3) Moore dataset (network traffic classification)

The Moore [9] dataset is intended for NTC classification. It is a large dataset with 12 continuous features and over 1 million samples. All the features are continuous. The dataset does not provide a test set; therefore, we have constructed one from a random subset of 20% of the original samples, holding the distribution of labels in both the resulting training and test sets.

The features for this dataset are obtained by summarizing (e.g. median, variance of bytes in packets…) information contained in the first five packets of each network data flow. A window size of five packets is considered by the authors of this dataset as optimal to achieve good classification results.

The dataset includes 15 labels associated to different types of network traffic (e.g. WWW, MAIL, P2P, VoIP…). It is extremely unbalanced with a frequency of 83.9% and 0.0001% for the most and least frequent labels. In order to have a still unbalanced dataset but not so extremely unbalanced, we have aggregated the labels with a frequency less than 0.1% into a single label, resulting in a final dataset with 9 labels with the frequency for the most and least frequent labels being now 83.99% and 0.11% respectively.

Similarly to the two other datasets we have included in Section IV the results for this dataset when the 9 labels are aggregated into only two (WWW and REST). The only intention of this result is to present the prediction performances under a variety of comparable scenarios between datasets.

This dataset provides a different scenario for classification, having only a small set of continuous features.

B. Model description

When doing classification with linear models we try to find a linear decision surface that separate the classes. This surface is obtained as an inner product of a vector of weights (w), with the vector of features (x), given by: $\langle w, x \rangle$. But, in general, the classes are not linearly separable. In this case, a solution is to perform a projection of the vector of features into a higher dimensional vector space ($\phi(x)$), where the associated classes can be linearly separable with a new linear decision surface, given by: $\langle w, \phi(x) \rangle$. The parameters w can be obtained by some optimization mechanism (e.g. Stochastic Gradient Descent - SGD). To ensure linear separation, the best approach would be to perform the projection into a transformed space with a large number of dimensions, ideally an infinite number of dimensions, but that would make it impossible to calculate the required inner product. The Representer Theorem [33] and Reproducing Kernel Hilbert Space theory [34] provide a solution to this problem with the

kernel trick that applies a kernel function: $k(x, x')$, such that:

$$k(x, x') = \langle \phi(x), \phi(x') \rangle \quad (1)$$

That is, the inner product in the possibly infinite transformed space can be easily calculated with the kernel function in the original feature space. Applying this solution, the new decision surface is obtained by:

$$\langle w, \phi(x) \rangle = \langle \sum_{i=1}^{N} \alpha_i \phi(x_i), \phi(x) \rangle = \sum_{i=1}^{N} \alpha_i k(x_i, x) \quad (2)$$

The parameters α_i in (2) can be obtained by quadratic optimization (dual solution) [35] as an alternative solution to obtaining the parameters w (primal solution) [35].

These principles are applied in the kernel-SVM model, usually with an RBF kernel. SVM is one of the best models for classification, being able to capture nonlinearity in the features, what is important to detect complex patterns. As will be shown later on this paper, SVM produces very good prediction results but has real problems in terms of time and resources needed to accomplish both training and prediction.

The applicability of the dual solution depends on the number of samples and features, not being applicable when the number of samples is large, since that implies solving a problem of quadratic programming with huge matrices. When it is necessary to deal with datasets with a large number of samples (as is usually the case in IoT predictions), it is necessary to use the primal solution with its associated problem of non-applicability of the kernel trick. The kernel approximation (KA) algorithms comes as a way out in these cases, since they provide a projection into a higher dimensional space (but not infinite dimensional) with the property than the inner products in this transformed space are approximately equal to those obtained by applying the associated kernel.

Classification problems in data networks generally have a large number of samples which require using the primal solution, and it makes it necessary to use a KA transform before applying a linear model. To implement the KA transformation there are several techniques. Section III.C gives details on the techniques used in this paper.

Our proposed model is presented in Fig. 1, where the input data is the matrix X, composed of N samples of dimension d (d features). The input data is transformed to a higher dimensional space by a KA transform producing a new matrix X_t of dimensions $N \times D$, where $D \gg d$, and D being the dimension of the KA transformed data. The matrix X_t is multiplied by a matrix of weights W of dimensions $D \times M$, generating an output matrix of results Y of dimensions $N \times M$. The dimension M of the output vector corresponds with the M values of the predicted label (one-hot encoded). The matrix Z contains the ground truth values for the labels of the training data, with dimensions $N \times M$.

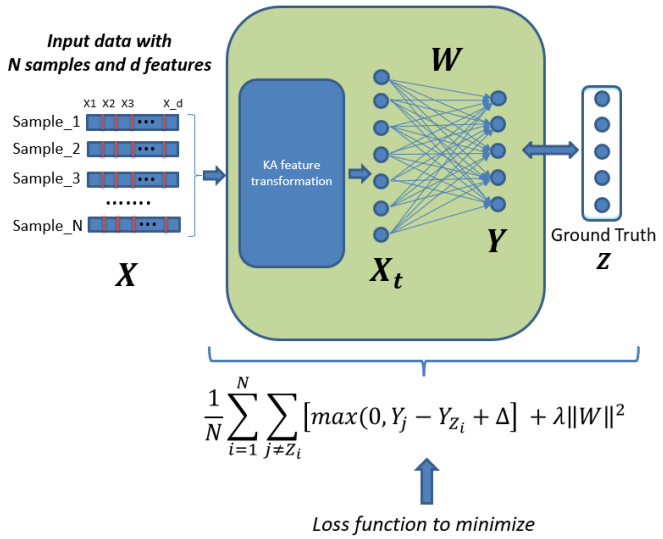


Fig 1. Linear model with KA transformation of features.

To learn the parameters of the weights matrix: W, we use SGD with a loss function shown in Fig. 1. The loss function corresponds to the average value through the samples of the sum of a hinge function applied to the difference between the values of Y_j (the j position of vector Y) with the value of Y_{Z_i} associated with the index Z_i of the correct output in Z. The parameter Δ corresponds to the margin value, being $\Delta = 1$ for the hinge loss (linear SVM) or $\Delta = 0$ for the perceptron loss. The final term is a regularization term to penalize weights with big values; in this case we have used a L2 norm for regularization, the parameter

λ being a hyperparameter to control the importance of regularization.

To implement the model we have used Tensorflow as an efficient platform to perform automatic differentiation, to obtain the necessary gradients to minimize the loss function during training, and capable of handling large datasets.

C. *Kernel approximation variants*

In order to implement kernel approximation, there are several techniques. We have employed Random Fourier Features (RFF) [10] and Nystroem Method [11] to approximate a Radial Basis Function kernel. Similarly, we have also applied a KA based on the Fastfood transformation [12] which is a fast implementation of Random Kitchen Sinks [36]

In the case of RFF [10], we create the new features using random Fourier basis (3):

$$\cos(\omega' x + b) \quad (3)$$

where $\omega, x \in \mathcal{R}^d$ and $b \in \mathcal{R}$, being x the original data to be transformed, ω a random variable from a distribution obtained from the Fourier transform of the kernel, and b a random variable from a uniform distribution on $[0, 2\pi]$; both ω and x have the same dimensionality (d), which is the same as the number of initial features. The random mapping works using Bochner's theorem [37] which states that the Fourier transform of a shift-invariant positive definitive kernel is a proper probability distribution [10]

The feature transformation will provide a feature map:

$$\psi: \mathcal{R}^d \rightarrow \mathcal{R}^D \quad (4)$$

Which transforms d features to D features, where D is bigger than d, but not infinity, as would be necessary if we want to do the real mapping associated to the RBF feature map (5):

$$\phi: \mathcal{R}^d \rightarrow \mathcal{R}^\infty \quad (5)$$

The important property of the transformation given by (4) is that the inner products of the approximated function (in this case the one associated to the RBF kernel) can be made similar to the inner products of the transformed data (as given in (6)):

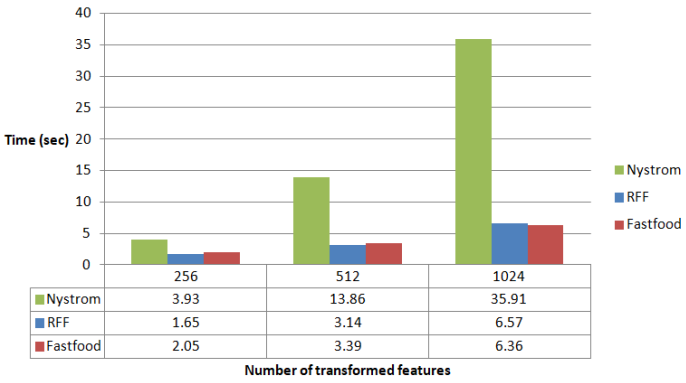
$$k(x, y) = \langle \phi(x), \phi(y) \rangle \approx \psi(x)'\psi(y) \quad (6)$$

Where $k(x, y)$ is the RBF kernel applied to samples x and y

The objective of Nystroem Method is similar to RFF in obtaining a transformation similar to (4) but the method used is different. Nystroem Method implements a low-rank approximation to the Gram matrix (kernel matrix) using sampled columns of the original d x d Gram matrix [11][38][39]. Nystroem method generally provides a better approximation to the true kernel transform, at the cost of increasing the required time to perform the transformation when the number of features is big.

The Fastfood transformation is a fast implementation of Random Kitchen Sinks which approximate the function $\phi(x)$ by multiplying the input vector (original features) with a dense Gaussian random matrix, followed by the application of a non-linearity. Fastfood operates in a similar way, but replacing the dense Gaussian matrix by a combination of Hadamard and diagonal Gaussian matrices, which are more efficient to multiply and store [12][36].

In Fig. 2, are presented the times needed to perform the feature transformation for the different KA algorithms, depending on the number of transformed features. The dataset used has been NSL-KDD with 122 features (Section III.A). We can see that Nystroem requires more time and that this time increases with the number of features. Fastfood starts to be competitive when the number of features is greater than 1000 (as predicted in [12]). Similar results are obtained for the UNSW-NB15 and Moore datasets.



	256	512	1024
Nystrom	3.93	13.86	35.91
RFF	1.65	3.14	6.57
Fastfood	2.05	3.39	6.36

Fig 2. Time required to perform the KA transformation of features.

IV. RESULTS

In this section, we compare the results of applying different machine learning models to the NSL-KDD, UNSW-NB15 and Moore datasets, specifically the models are: Multinomial Logistic Regression, SVM with linear kernel and RBF kernel, Random Forest, GBM, AdaBoost with one weak learner (simple trees), MLP, CNN and our proposed model: Linear model + KA transformation.

All results presented in this section are based in a neural network with no hidden layers (it consists just of the input and output layers), linear activation function for the last layer, and linear loss function (hinge loss) with a margin value for the hinge loss equal to zero ($\Delta = 0$) and an L2 regularization parameter for the weights equal to 0.001 ($\lambda = 0.001$). This architecture, for the shallow neural network, is applied for all datasets and all configurations of labels (2-labels or multi-label). We have observed that this configuration provides best results, with small differences, for all datasets and models.

The KA transformation that achieves best results does depend on the dataset. For the NSL-KDD the best KA transform has been the Nystroem method, with a number of transformed features equal to 400. The final model for the UNSW-NB15 consists of a KA transformation based in the Fastfood method with 512 features, and, finally, the Moore dataset obtains best prediction results with the RFF method with 512 generated features.

It is interesting that the element that creates a difference is the selection of the KA method and the number of features created by the transformation.

All results presented in this paper are based in the test sets defined in Section III.A. To analyze the prediction performance for the different models, and considering the highly unbalanced distribution of labels, we provide the following performance metrics: accuracy, F1 score, precision and recall. We base our definition of these performance metrics on the usually accepted ones [1].

Regarding highly unbalanced datasets, F1 is considered a better metric for prediction performance than accuracy, precision and recall. This metric (F1) will be the metric used to rank the different algorithms applied to the three datasets.

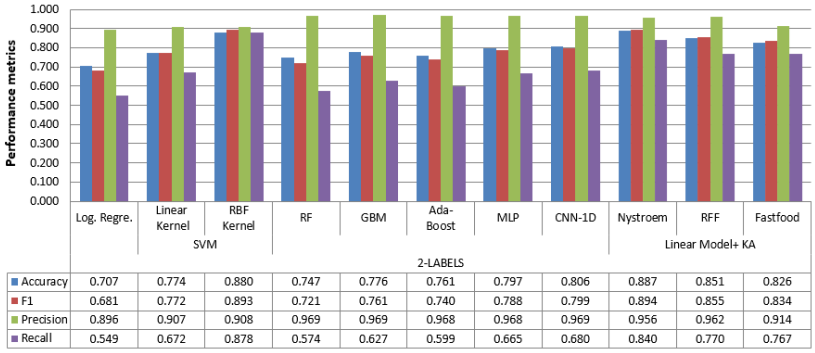
For a binary classification there is no ambiguity to provide results, but, when facing a multi-class classification problem, there are two possible ways to give results: aggregated and One vs. Rest. For One vs. Rest, we focus in a particular class (label) and consider the other classes as a single alternative class, simplifying the problem to a binary classification task for each particular class (one by one). In the case of aggregated results, we try to give a summary result for all classes. There are different alternatives to perform the aggregation (micro, macro, samples, weighted), varying in the way the averaging process is done [40]. Considering the results presented in this paper, we have used the weighted average provided by scikit-learn [40], to calculate the aggregated F1, precision and recall scores.

Aggregated performance results for the NSL-KDD, UNSW-NB15 and Moore datasets are summarized in Fig. 3, 4 and 5 respectively. These figures are divided into two parts, with the upper part giving results for the 2-labels configuration of the datasets and the lower part for the multi-label configuration. (Section III.A)

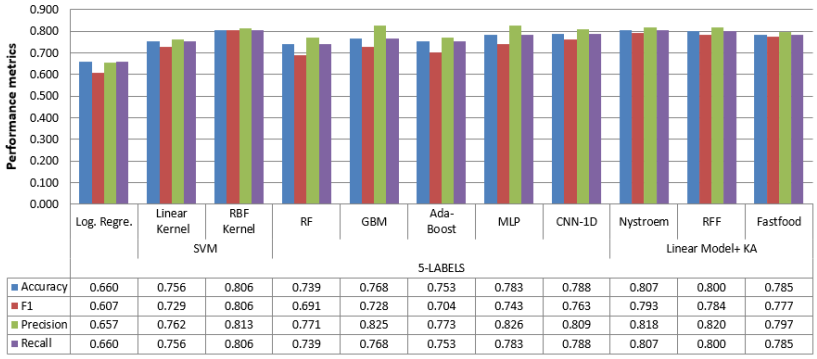
Considering F1 as our main performance metric, the results in Fig. 3-5 can be interpreted as follows: The NSL-KDD dataset (Fig. 3), highly noisy and with a difficult test set, has the SVM-RBF model as its best model for the configuration of 5-labels (followed immediately by the Linear+Nystroem model) and the Linear+Nystroem model as the best for the 2-labels configuration. The UNSW-NB15 dataset (Fig. 4) provides best results for the CNN-1D model, closely followed by Linear+Fastfood. This behavior occurs for both the configuration of 2 and 10-labels. For the Moore dataset (Fig. 5) with a large number of samples and a small number of features the best results are for bagging and/or boosting models based on decision trees (random forest, GBM, Adaboost) and for the CNN-1D model, all of them highly non-linear and complex models. In this case, Linear+RFF is with the best models for the 2-label configuration and immediately behind them for the 5-labels configuration. The performances of MLP and SVM-RBF are not good for this dataset.

From the results in Fig. 3-5 we can conclude that Linear+KA models are among the best models for all three datasets and in all label configurations.

It is important to mention several details about the different models applied in the study. For SVM with linear kernel, we have used the primal solution which provides a much faster implementation in our specific case (high number of samples and small number of features). For the SVM with RBF, we had to use the dual implementation. Results related to linear models plus KA transformation are provided using three KA methods: Nystroem (400 features), RFF (512 features) and Fastfood (512 features). We have implemented the MLP with several hidden layers (3 layers with 1024, 512 and 128 nodes). Considering the CNN model [41], we have applied the one-dimensional CNN due to the nature of the features in all datasets (no spatial allocation of features).

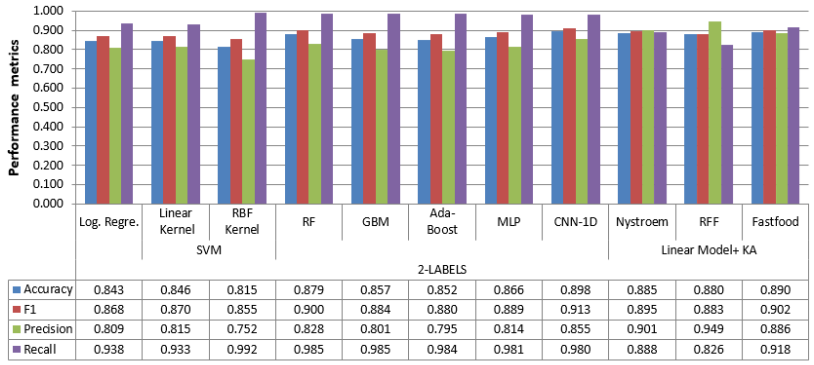


	Log. Regre.	Linear Kernel	RBF Kernel	RF	GBM	Ada-Boost	MLP	CNN-1D	Nystroem	RFF	Fastfood
Accuracy	0.707	0.774	0.880	0.747	0.776	0.761	0.797	0.806	0.887	0.851	0.826
F1	0.681	0.772	0.893	0.721	0.761	0.740	0.788	0.799	0.894	0.855	0.834
Precision	0.896	0.907	0.908	0.969	0.969	0.968	0.968	0.969	0.956	0.962	0.914
Recall	0.549	0.672	0.878	0.574	0.627	0.599	0.665	0.680	0.840	0.770	0.767

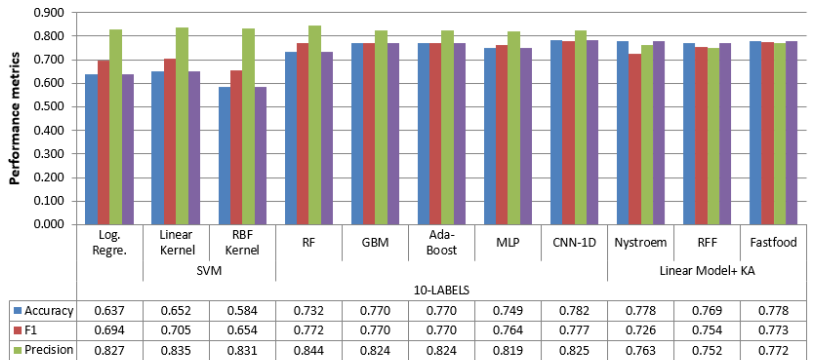


	Log. Regre.	Linear Kernel	RBF Kernel	RF	GBM	Ada-Boost	MLP	CNN-1D	Nystroem	RFF	Fastfood
Accuracy	0.660	0.756	0.806	0.739	0.768	0.753	0.783	0.788	0.807	0.800	0.785
F1	0.607	0.729	0.806	0.691	0.728	0.704	0.743	0.763	0.793	0.784	0.777
Precision	0.657	0.762	0.813	0.771	0.825	0.773	0.826	0.809	0.818	0.820	0.797
Recall	0.660	0.756	0.806	0.739	0.768	0.753	0.783	0.788	0.807	0.800	0.785

Fig 3. Aggregated performance scores for the NSL-KDD dataset.



	Log. Regre.	Linear Kernel	RBF Kernel	RF	GBM	Ada-Boost	MLP	CNN-1D	Nystroem	RFF	Fastfood
Accuracy	0.843	0.846	0.815	0.879	0.857	0.852	0.866	0.898	0.885	0.880	0.890
F1	0.868	0.870	0.855	0.900	0.884	0.880	0.889	0.913	0.895	0.883	0.902
Precision	0.809	0.815	0.752	0.828	0.801	0.795	0.814	0.855	0.901	0.949	0.886
Recall	0.938	0.933	0.992	0.985	0.985	0.984	0.981	0.980	0.888	0.826	0.918



	Log. Regre.	Linear Kernel	RBF Kernel	RF	GBM	Ada-Boost	MLP	CNN-1D	Nystroem	RFF	Fastfood
Accuracy	0.637	0.652	0.584	0.732	0.770	0.770	0.749	0.782	0.778	0.769	0.778
F1	0.694	0.705	0.654	0.772	0.770	0.770	0.764	0.777	0.726	0.754	0.773
Precision	0.827	0.835	0.831	0.844	0.824	0.824	0.819	0.825	0.763	0.752	0.772
Recall	0.637	0.652	0.584	0.732	0.770	0.770	0.749	0.782	0.778	0.769	0.778

Fig 4. Aggregated performance scores for the UNSW-NB15 dataset.

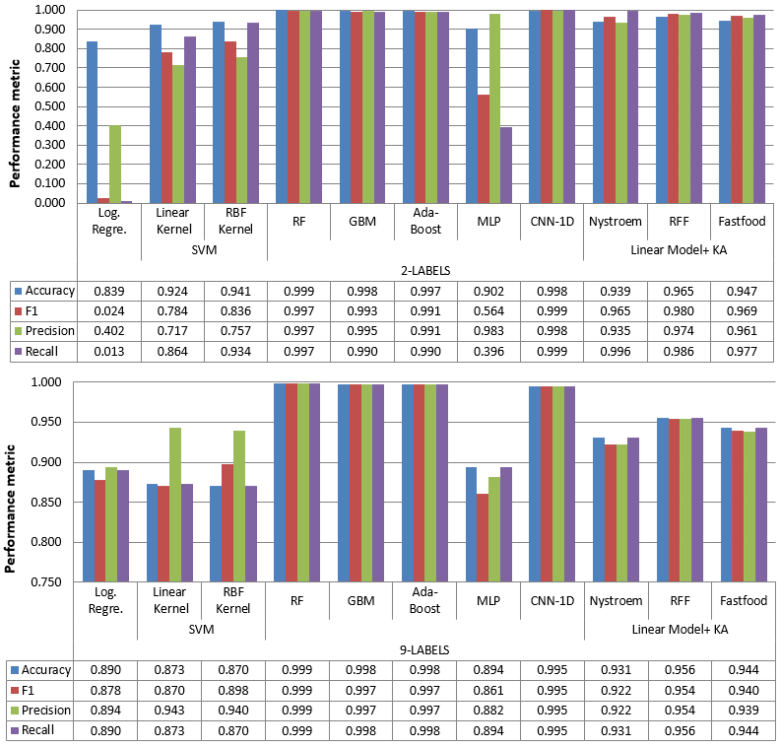


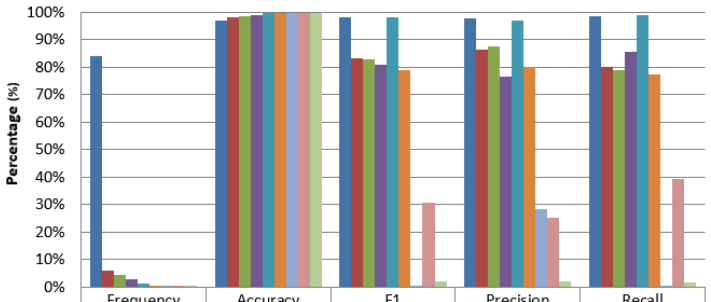
Fig 5. Aggregated performance scores for the Moore dataset.

To ensure that the good results obtained by the Linear + KA model are significant, Table I presents the application of the Wilcoxon signed-rank test for the comparison of the prediction results, for each metric, between the best Linear+KA model by dataset and the other models for that dataset. The test is performed including all datasets. The last row in Table I presents the results when all the metrics are included in the comparison. The conclusion is that the Linear+KA model has results with a higher average value and significantly different from the rest of the models. Only the precision metric, even having a higher average value, presents a non-significant result at a level of significance of 1%

			Mean values	
Metric	p-value	Significance Level (1%)	Best Linear+KA model	Rest of models
Accuracy	7.937E-06	Yes	0.881	0.832
F1	1.082E-05	Yes	0.883	0.804
Precision	1.178E-01	No	0.893	0.862
Recall	2.985E-03	Yes	0.881	0.801
All metrics	1.313E-11	Yes	0.884	0.825

Table I. Wilcoxon signed-rank test: significance of results

Fig. 6 shows One vs. Rest detailed performance metrics for the linear model + RFF transformation applied to the Moore dataset with the multi-label configuration. We can observe how the frequency distribution for the labels is highly unbalanced (column "Frequency" in Fig. 6). We get an F1 score greater than 0.8 for the most frequent labels. The accuracy obtained is always greater than 0.98 regardless of the label and usually much higher.

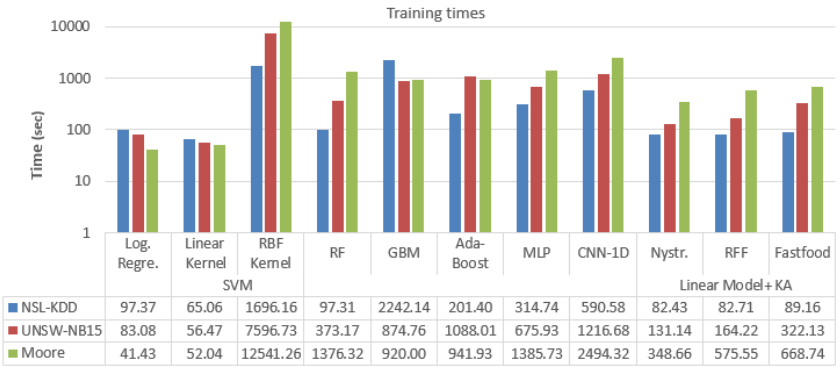


	Frequency	Accuracy	F1	Precision	Recall
WWW	83.99%	0.969	0.982	0.977	0.986
MAIL	5.90%	0.981	0.832	0.863	0.803
P2P	4.46%	0.986	0.831	0.876	0.790
BULK	2.86%	0.988	0.809	0.767	0.856
DATABASE	1.45%	0.999	0.980	0.970	0.991
ATTACK	0.67%	0.997	0.788	0.803	0.774
SERVICES	0.34%	0.997	0.006	0.286	0.003
OTHERS	0.21%	0.996	0.309	0.254	0.394
VOIP	0.11%	0.998	0.019	0.021	0.018

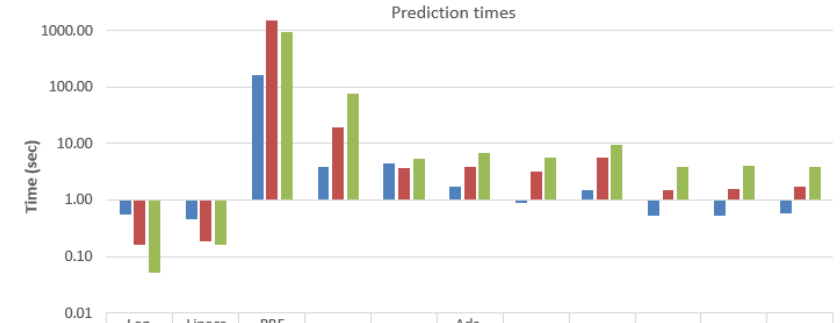
Fig. 6. One vs. Rest performance scores for Linear+RFF model (Moore dataset).

Prediction and training times are essential for IDS and NTC predictions, since traffic is permanently changing. Good prediction metrics by themselves are not enough for deciding the best model. In Fig. 7 are shown the computing times for training and prediction for all models applied to the three datasets.

As expected Linear-SVM and Logistic regression present the best prediction times, followed by Linear Models + KA transformation. Linear-SVM model is implemented as a Linear Model without KA transformation, hence its better performance regarding prediction time. For training times, the best values are obtained with Linear-SVM and Linear Models + KA transformation closely followed by Logistic regression.



	Log. Regre.	Linear Kernel	RBF Kernel	RF	GBM	Ada-Boost	MLP	CNN-1D	Nystr.	RFF	Fastfood
	SVM								Linear Model+KA		
NSL-KDD	97.37	65.06	1696.16	97.31	2242.14	201.40	314.74	590.58	82.43	82.71	89.16
UNSW-NB15	83.08	56.47	7596.73	373.17	874.76	1088.01	675.93	1216.68	131.14	164.22	322.13
Moore	41.43	52.04	12541.26	1376.32	920.00	941.93	1385.73	2494.32	348.66	575.55	668.74



	Log. Regre.	Linear Kernel	RBF Kernel	RF	GBM	Ada-Boost	MLP	CNN-1D	Nystr.	RFF	Fastfood
	SVM								Linear Model+KA		
NSL-KDD	0.55	0.46	158.65	3.87	4.39	1.69	0.89	1.52	0.52	0.52	0.57
UNSW-NB15	0.16	0.18	1536.25	19.34	3.67	3.85	3.23	5.50	1.50	1.56	1.68
Moore	0.05	0.16	925.46	77.41	5.41	6.65	5.59	9.58	3.84	4.00	3.93

Fig. 7. Training (upper chart) and prediction times (lower chart) for all learning models and datasets.

The main gains in computational time obtained by the Linear+KA models are less evident for the Moore dataset, which after

applying the KA transformation increases the number of features from 12 to 512. This large increase justifies that some non-linear models that work with 12 features can bring their computing times closer to those of the Linear+KA models.

As a final check of the good balance presented by the Linear+KA models between the prediction times and the predictive capacity, Fig. 8 presents a graph that shows all the models against their F1 metric using the size of the point to indicate the time needed to make a complete prediction of the test data set. Fig. 8 provides the values for the NSL-KDD dataset; similar graphs can be obtained for the other datasets. In this graph we can see how the Linear+KA models have a predictive performance close to the best non-linear model (SVM-RBF) but with a much lower prediction time required.

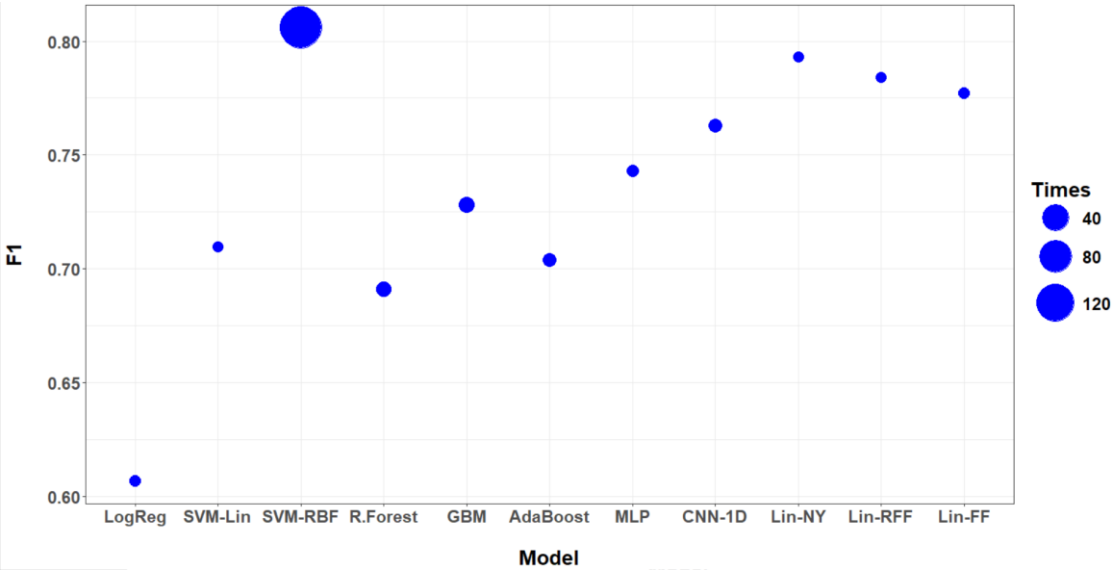


Fig. 8. Prediction times vs. F1 score for all learning models.

It is interesting to compare the low prediction performance of all linear models (logistic regression and SVM with a linear kernel) with the excellent performance results of the different variants of the proposed model: Linear Model+KA. This largely justifies the application of the initial KA transformation to the features, and demonstrates that the prediction problems generally dealt with in data networks require the application of non-linear models.

In addition, a curious finding of this work is that the KA transformation provides a performance improvement only when used with models strictly linear, which means that the loss function is linear and the activation functions are also linear, otherwise the results obtained are worse than without the KA transformation. This behavior has been verified with the Multinomial Logistic Regression, CNN and MLP which all include non-linearities (e.g. ReLU, Softmax activations or cross-entropy loss function), and, that provide poorer results when combined with a KA transformation.

We have implemented all the models in python using the scikit-learn package [40], except all linear models (including linear-SVM and Multinomial Logistic Regression), MLP and CNN models for which we have used Tensorflow.

V. Conclusion

Intrusion detection and network traffic classification are two of the main research applications of machine learning to highly demanding data networks e.g. IoT/sensors networks. We propose and analyze a prediction model that is appropriate for these applications, requiring high detection performance with reduced computation times.

The proposed model consists of a shallow linear architecture with a multiclass hinge loss and a feature transformation based in kernel approximation theory. This combination provides a fast and flexible model with non-linear behavior.

This paper presents the first application of a linear model plus a transformation KA of the features to prediction problems of data networks.

We provide a thorough comparison study of the model with alternative models, considering three performance aspects: prediction scores, prediction and training times. We also consider different variants for the proposed model, including the analysis of three KA transformations: Nystroem, RFF and Fastfood. The results of the study show that the proposed model is positioned uniquely in the upper part for the three performance aspects, being similar in detection performance to the best non-linear models (e.g. Kernel-SVM, CNN...) and with computation times similar to linear models (e.g. Logistic regression and Linear-SVM).

We have chosen IDS and NTC as two representative prediction problems in modern data networks (e.g. IoT), and selected three well-known datasets (NSL-KDD, UNSW-NB15 and Moore) in these areas. These datasets were created under a variety of objectives and constraints and represent a good example of the different requirements imposed on prediction problems in data networks.

The observed low detection performance of all linear models seems to imply that linear relationships are not enough to capture the underlying structure of the datasets used in prediction problems for data networks, then the importance of providing the models with non-linear behavior while still achieving the best prediction and training times.

ACKNOWLEDGMENTS

This work has been partially funded by the Ministerio de Economía y Competitividad del Gobierno de España and the Fondo de Desarrollo Regional (FEDER) within the project "Inteligencia distribuida para el control y adaptación de redes dinámicas definidas por software, Ref: TIN2014-57991-C3-2-P", and the Project "Distribucion inteligente de servicios multimedia utilizando redes cognitivas adaptativas definidas por software", Ref: TIN2014-57991-C3-1-P, in the Programa Estatal de Fomento de la Investigación Científica y Técnica de Excelencia, Subprograma Estatal de Generación de Conocimiento.

REFERENCES

[1] M.H. Bhuyan, D.K. Bhattacharyya, and J.K. Kalita, "Network Anomaly Detection: Methods, Systems and Tools", IEEE Communications Survey & Tutorials, Vol. 16, No. 1, First Quarter 2014, 2014

[2] T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," IEEE Commun. Surv. Tutorials, vol. 10, no. 4, pp. 56–76, 2008.

[3] M. Lopez-Martin et al., "Network Traffic Classifier with Convolutional and Recurrent Neural Networks for Internet of Things," IEEE Access, vol. 5, 2017, pp. 18042-18050.

[4] M. Wang et al., "Machine learning for networking: workflow, advances and opportunities", IEEE Network, vol.32, no.2, pp. 92-99, March-April 2018.

[5] S. Kim et al., "Building Resilient and Autonomous Systems for IoT Network Management - Advantages and Difficulties in adopting Machine Learning Techniques", Internet Engineering Task Force (IETF), 6Lo Working Group, Internet-Draft: draft-kim-ml-iot-00, Seoul National University, 2018

[6] M. Tavallaee et al, "A Detailed Analysis of the KDD CUP 99 Data Set", Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2009), pages 53-58

[7] M. Nour and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)."Military Communications and Information Systems Conference (MilCIS), 2015. IEEE, 2015.

[8] M. Nour and J. Slay, "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set." Information Security Journal: A Global Perspective (2016): 1-14.

[9] L. Wei et al., "Efficient Application Identification and the Temporal and Spatial Stability of Classification Schema", Computer Network Vol. 53, no.6, pp. 780-809, 2008

[10] A. Rahimi, and B. Recht, "Random features for large-scale kernel machines", Advances in neural information processing 2007.

[11] T. Yang et al, "Nystrom Method vs Random Fourier Features: A Theoretical and Empirical Comparison", Advances in Neural Information Processing Systems 25 (NIPS 2012).

[12] Q. Le, T. Sarlos, and A. Smola. "Fastfood – approximating kernel expansions in loglinear time". In ICML, 2013.

[13] B. Ingre and A. Yadav., "Performance Analysis of NSL-KDD dataset using ANN", SPACES-2015, Dept of ECE, K L University, 2015.

[14] L. M. Ibrahim et al, "A comparison study for intrusion database (KDD99, NSL-KDD) based on self-organization map (SOM) artificial neural network", Journal of Engineering Science and Technology, School of Engineering, Taylor's University, Vol. 8, No. 1 (2013) 107 – 119.

[15] M. Panda, A. Abraham, and M. R. Patra, "Discriminative Multinomial Naïve Bayes for Network Intrusion Detection", Proceedings of 6th Intl. conf. on information assurance and security (IAS-2010), Aug. 2010, USA, 5-10, IEEE Press, 2010

[16] L.Dhanabal , S.P. Shantharajah, "A Study on NSL- KDD Dataset for Intrusion Detection System Based on Classification Algorithms", International Journal of Advanced Research in Computer and Communication Engineering Vol. 4, Issue 6 , June 2015, 2015

[17] O.S.M. Kame et al, "AdaBoost Ensemble Learning Technique for Optimal Feature Subset Selection", International Journal of Computer Networks and Communications Security VOL. 4, NO. 1, January 2016, 1–11, 2016

[18] D.R. Patil, T.M. Pattewar, "A Comparative Performance Evaluation of Machine Learning-Based NIDS on Benchmark Datasets", International Journal of Research in Advent Technology, Vol.2, No.2, April 2014 E-ISSN: 2321-9637, 2014

[19] Y. Wahb et al, "Improving the Performance of Multi-class Intrusion Detection Systems using Feature Reduction", IJCSI International Journal of Computer Science Issues, Volume 12, Issue 3, May 201, 2015

[20] M. Lopez-Martin et al., "Conditional Variational Autoencoder for Prediction and Feature Recovery Applied to Intrusion Detection in IoT", Sensors 17 (9), 1967, 2017.

[21] N. Moustafa and J. Slay, "A hybrid feature selection for network intrusion detection systems: central points and association rules", arXiv:1707.05505 [cs.CR], 2017.

[22] C. Khammassi and S. Krichen, "A GA-LR wrapper approach for feature selection in network intrusion detection," Computers & Security, vol. 70, pp. 255–277, 2017.

[23] M. Belouch, S. El Hadaj and M. Idhammad, "A Two-Stage Classifier Approach using RepTree Algorithm for Network Intrusion Detection", International Journal of Advanced Computer Science and Applications, Vol. 8, No. 6, 2017

[24] W. Zhou, L. Dong, L. Bic, M. Zhou and L. Chen, "Internet traffic classification using feed-forward neural network," *2011 International Conference on Computational Problem-Solving (ICCP)*, Chengdu, 2011, pp. 641-646.

[25] S. Hao, J. Hu, S. Liu, T. Song, J. Guo and S. Liu, "Network traffic classification based on improved DAG-SVM," 2015 International Conference on Communications, Management and Telecommunications (ComManTel), DaNang, 2015, pp.256-26

[26] Z. Yuan and C. Wang, "An improved network traffic classification algorithm based on Hadoop decision tree," 2016 IEEE International Conference of Online Analysis and Computing Science (ICOACS), Chongqing, 2016, pp. 53-56.

[27] D. Zuev and A. W. Moore, "Traffic Classification using a Statistical Approach", Passive and Active Network Measurement Workshop. PAM 2005. Lecture Notes in Computer Science, vol 3431. Springer, Berlin, Heidelberg. 2005.

[28] A. W. Moore and D. Zuev. "Internet traffic classification using bayesian analysis techniques". Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems (SIGMETRICS '05). ACM, New York, NY, USA, 50-60. 2005.

[29] H. Gao, X. Wang and H. Yang, "LS-SVM Based Intrusion Detection using Kernel Space Approximation and Kernel-Target Alignment," 2006 6th World Congress on Intelligent Control and Automation, Dalian, 2006, pp. 4214-4218.
[30] P. Movahedi et al. “Fast regularized least squares and k-means clustering method for intrusion detection systems”. 2015. Proceedings of the International Conference on Pattern Recognition Applications and Methods. DOI: http://dx.doi.org/10.5220/0005246802640269
[31] A. May, “Kernel Approximation Methods for Speech Recognition”. PhD Thesis, Columbia University, 2018.
[32] L. Bo and C. Sminchisescu. “Efficient match kernel between sets of features for visual recognition”, Proceedings of the 22nd International Conference on Neural Information Processing Systems (NIPS'09), 2009
[33] G. S. Kimeldorf and G. Wahba. “A correspondence between Bayesian estimation on stochastic processes and smoothing by splines”. Annals of Mathematical Statistics, 41:495–502, 1970.
[34] F. Girosi. “An equivalence between sparse approximation and support vector machines”. Neural Computation, 10(6):1455–1480, 1998.
[35] B. Scholkopf et al., “Input Space Versus Feature Space in Kernel-Based Methods”, IEEE Transactions on Neural Networks, Vol. 10, No. 5, September 1999
[36] A. Rahimi and B. Recht. “Random features for large-scale kernel machines”. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, Advances in Neural Information Processing Systems 20. MIT Press, Cambridge, MA, 2008.
[37] W. Rudin. “Fourier Analysis on Groups”. Wiley Classics Library. Wiley-Interscience, New York, 1994.
[38] P. Drineas et al, “On the Nystrom Method for Approximating a Gram Matrix for Improved Kernel-Based Learning”, Journal of Machine Learning Research 6 (2005) 2153–2175.
[39] C.K.I. Williams and M. Seeger, “Using the Nystrom Method to Speed Up Kernel Machines”, NIPS'00 Proceedings of the 13th International Conference on Neural Information Processing Systems (2000), pp 661-667
[40] Pedregosa et al., “Scikit-learn: Machine Learning in Python”, JMLR 12, pp. 2825-2830, 2011.
[41] I. Goodfellow. Y. Bengio and A. Courville, “Deep Learning”. Book. MIT Press, Ch 9. 2016

***Highlights (for review)**

- Most prediction problems of data networks are non-linear
- Shallow neural networks are fast
- Kernel approximation is a non-linear data transformation
- Shallow neural networks with kernel approximation are non-linear and fast models
- Proposed model is faster with prediction results comparable to deep models