Detection of Malicious SCADA Communications via Multi-Subspace Feature Selection

Ehsan Hallaji, Roozbeh Razavi-Far, and Mehrdad Saif

Department of Electrical and Computer Engineering
University of Windsor
Windsor ON, Canada
E-mails: {hallaji, roozbeh, msaif}@uwindsor.ca

Abstract—Security maintenance of Supervisory Control and Data Acquisition (SCADA) systems has been a point of interest during recent years. Numerous research works have been dedicated to the design of intrusion detection systems for securing SCADA communications. Nevertheless, these data-driven techniques are usually dependant on the quality of the monitored data. In this work, we propose a novel feature selection approach, called MSFS, to tackle undesirable quality of data caused by feature redundancy. In contrast to most feature selection techniques, the proposed method models each class in a different subspace, where it is optimally discriminated. This has been accomplished by resorting to ensemble learning, which enables the usage of multiple feature sets in the same feature space. The proposed method is then utilized to perform intrusion detection in smaller subspaces, which brings about efficiency and accuracy. Moreover, a comparative study is performed on a number of advanced feature selection algorithms. Furthermore, a dataset obtained from the SCADA system of a gas pipeline is employed to enable a realistic simulation. The results indicate the proposed approach extensively improves the detection performance in terms of classification accuracy and standard deviation.

Index Terms—Feature selection, ensemble learning, intrusion detection, supervised learning, mutual information, cyber-physical systems, SCADA

I. INTRODUCTION

Safe and reliable operation of SCADA systems can be disrupted through the interference of intruders who launch malicious attacks on the application layer of these cyber-physical systems [1]. Catastrophic consequences of such intrusions, on the other hand, necessitate prompt detection and isolation of cyber-attacks [2]. For this mean, various Intrusion Detection Systems (IDS) have been proposed and studied in the literature [1], [3].

An IDS generally makes use of a data-driven approach, in which a detection model is constructed based on the available prior knowledge on the types of cyber-attacks [4], [5]. Therefore, data patterns that resemble a type of cyber-attack can be identified and classified in the traffic data w.r.t. the constructed IDS model.

The performance of intrusion detection using the constructed model is heavily dependant on the quality of the collected data [6]. On one hand, the recorded data may contain non-informative features. On the other hand, raw data measurements often require proper feature extraction [7], [8], which usually produces redundant features along with informative features [9]–[11]. This redundancy in the data results in the shortage of efficiency by exposing excessive computational burden to the system. Moreover, including non-informative dimensions of the feature space usually deteriorates the accuracy of the IDS model.

Redundancy in data can be eliminated using Feature Selection (FS) and dimensionality reduction [11]–[14]. The former is mainly used to find the best set of informative features while disregarding the rest of the features. The latter, on the other hand, aims to compute a transformation matrix that transforms data onto a lower-dimensional feature space. While both approaches aim to find a feature space, in which all classes are well-discriminated, reaching this goal is usually more challenging via FS, as it keeps the nature of features intact rather than transforming them.

In this paper, we propose a novel FS algorithm, called Multi-Subspace Feature Selection (MSFS), to improve the data quality in terms of redundancy and relevancy. The main idea behind MSFS is to find a set of feature subsets, each of which obtained by focusing on separating a specific class from others. By this mean, in contrast to the traditional approaches that model all classes in a unified feature space, MSFS tries to maximize the discrimination among classes by modeling each class in a separate subspace, where only features that optimally present this class are used. Note that MSFS is different from Embedding FS methods [15], [16] that use subspace clustering to learn clustering labels and a similarity matrix in order to find an optimal feature set. To enable the usage of multiple subspaces within the same feature space, we propose an ensemble scheme. Another contribution of this work is to design an IDS by employing the proposed MSFS and a number of advanced FS algorithms to improve the detection accuracy. This enables a comparative study that shows the effect of FS on performance enhancement of the IDS. For the sake of evaluation, we consider the case of intrusion detection in the SCADA systems of a gas pipeline [3]. Finally, the results are analyzed in terms of accuracy and standard deviation.

The remainder of this paper is organized as follows. Section II conducts a brief literature review on the related FS approaches. Section III explicitly proposes the novel MSFS algorithm. The designed IDS is introduced in Section IV. Section V reports and analyzes the obtained experimental
results. Finally, the paper is concluded in Section VI.

II. RELATED WORKS

In this section, we briefly overview the FS algorithms that are employed in this study. Unless stated otherwise, the following algorithms are categorized under unsupervised learning.

A. Infinite Feature Selection (InfFS)

InfFS [17] constructs a graph by considering an infinite number of paths connecting all the features and uses the convergence properties of power series of matrices. It evaluates the importance and redundancy of a feature w.r.t. all the remaining features.

B. Infinite Latent Feature Selection (ILFS)

ILFS [18] is a graph-based FS method that makes use of an affinity graph. Considering features as nodes of this graph, the importance of each node is evaluated w.r.t. Eigenvector centrality, while considering this factor for nodes in the neighbourhood as well. These nodes are then ranked similar to InfFS. The main difference between ILFS and InfFS is the former models a relevancy latent variable.

C. Eigenvector Centrality Feature Selection (ECFS)

ECFS [19] follows a graph-based approach in the same fashion as InfFS and ILFS. ECFS ranks features according to a graph centrality measure. By this mean, the importance of each feature is calculated by taking the importance of its neighbours into account.

D. Relief Feature Selection (ReliefF)

ReliefF [20] is a supervised and randomized FS technique that measures feature qualities in an iterative manner. To do so, ReliefF determines to what extent features values differentiate samples in a small neighbourhood. Nevertheless, feature redundancy may not be perceived by this algorithm, and thus, the best feature set may not be attained.

E. Mutual Information Feature Selection (MutInfFS)

MutInfFS [21] finds the best set of features in a greedy approach. In this process, a feature with the highest influence on the class relevance is determined at each step. The selection, on the other hand, is conducted based on a proportional term, which indicates the intersection of the nominated feature and the pool of features at hand.

F. Minimum Redundancy Maximum Relevance (mRMR)

mRMR [22] is a supervised search algorithm that uses an efficient incremental approach. Given a subset of selected features and a candidate feature, relevance scores are estimated through maximizing the joint information that is mutual between them. mRMR uses Parzen Gaussian windows to enable efficient estimations in this process.

G. Feature Selection via Concave minimization (FSV)

FSV [23] is an embedded FS technique that makes use of linear programming approach to inject the FS procedure into the training phase of a support vector machine.

H. Laplacian Score for Feature Selection

Laplacian Score (LS) [24] mainly relies on Laplacian Eigenmaps and Locality Preserving Projection. LS uses the locality preserving power of features in order to evaluate their importance. This has been done by means of a nearest neighbour graph, which is constructed to model the geometric structure of data.

I. Multi Cluster Feature Selection Technique (MCFS)

MCFS [25] aims to find the most informative set of features using cluster analysis. MCFS assumes that the selected features should preserve the cluster structure of the data, for which the manifold structure has been used. Additionally, MCFS ensures that all possible clusters are covered using by the selected features.

J. Recursive Feature Elimination (RFE)

RFE [26] is a wrapper FS algorithm that devises a sequential and backward elimination scheme for selecting features. RFE assigns a high rank to a feature if it results in significant separation of the data points by means of a support vector machine (SVM) with a linear kernel.

K. L0-Norm Feature Selection (L0-norm)

L0-Norm [27] penalizes those features that lead to more regularization and parallel parameter estimation. This FS method solves L0 penalty problem through the selection of non-zero coefficients and regularization parameters at the same time, and finds an approximation solution for the L0 penalty problem.

L. Fisher Score for Feature Selection

Fisher filter [28] is a fast FS technique that calculates the score of a feature w.r.t. the ratio of between-class separation and within-class variance. The features are evaluated independently within this process.

M. Unsupervised Discriminative Feature Selection (UDFS)

UDFS [29] is an L2,1-norm regularized discriminative FS algorithm, which chooses the best subset of features from the pool of features in the batch mode.

N. Correlation Based Feature Selection (CFS)

CFS [26] is a FS technique that ranks features with regards to a correlation-based heuristic evaluation function. The bias of this function is toward features that are highly correlated with a class and also uncorrelated with each other.
III. MULTI-SUBSPACE FEATURE SELECTION

The main idea behind Multi-Subspace Feature Selection (MSFS) is that different subspaces in the feature space can be used for modeling each class of data, rather than using the same set of features for all classes. To this aim, we devise ensemble learning to use multiple subspaces for modeling a unified dataset, as illustrated in Fig. 1. In this process, the feature selection is inspired by mRMR due to its supervised nature and compatibility with the case study at hand.

Given a dataset \( X \in \mathbb{R}^n \) with \( m \) samples and \( n \) features, the goal is to find a set of optimal features \( \hat{F} = \{X_1, X_2, \ldots, X_{\lambda}\} \) from the set of all features \( F = \{X_1, X_2, \ldots, X_n\} \), where \( \lambda \) is the number of selected features. To ensure that each class \( c \) is characterized in the best possible subspace, we aim to find the optimal feature set \( \hat{F} \) for each class separately. By this mean, given a set of unique classes \( C = \{c_1, c_2, \ldots, c_\kappa\} \), data samples \( x_i \in X \) are initially divided into different subsets \( S_i \) (see Fig. 1). This has been done w.r.t the set of all labels \( Y = \{y_1, y_2, \ldots, y_m\} \) corresponding to \( X \), as follows:

\[
S_i = \{x_i \mid 1 \leq i \leq m_i\}, \quad m_i = \text{Card}(\{x_i \mid y_i = c_i\}),
\]

where \( \text{Card}(\cdot) \) returns the cardinality, \( m_i \) is the number of samples in \( S_i \), i.e., number of samples in class \( c_i \), and \( 1 \leq i \leq \kappa \).

Once the subsets are formed, the search for \( \hat{F} \) can be carried out w.r.t. two criteria, namely maximum relevancy and minimum redundancy, which are defined based on mutual information \( f \) as:

\[
f(z,h) = \int_{\Omega_z} \int_{\Omega_h} p(z,h) \log \frac{p(z,h)}{p(z)p(h)} dzdh,
\]

where \( z \) and \( h \) are two random variables, \( \Omega_z \) and \( \Omega_h \) are the random variable sample spaces, and \( p(\cdot, \cdot) \) and \( p(\cdot) \) are the joint probability and marginal density function, respectively. Equation (2) can cope with discrete variables by changing to:

\[
f(z,h) = \sum_{z \in \Omega_z} \sum_{h \in \Omega_h} p(z,h) \log \frac{p(z,h)}{p(z)p(h)}.
\]

The relevancy \( J_D \) is formulated as the average of all mutual information between \( X_j \in F \) and \( c_i \in C \), as in the following:

\[
J_D(\hat{F}_i, c_i, S_i) = \frac{1}{\lambda} \sum_{X_j \in \hat{F}_i} f(\Phi(S_i, X_j), c_i),
\]

where \( \hat{F}_i \) is the optimal feature set to be determined for the class \( c_i \). Also, \( \Phi(X, F) \) returns the representation of \( X \) in subspace \( F \), i.e., only features in \( \hat{F} \) are used to present \( X \):

\[
\Phi(X, \hat{F}) : X \rightarrow \hat{X}, \quad \hat{X} \in \hat{F}.
\]

The redundancy, on the other hand, measures the information redundancy as:

\[
J_R(\hat{F}_i, S_i) = \frac{1}{\lambda^2} \sum_{X_i, X_j \in \hat{F}_i} f(\Phi(S_i, X_i), \Phi(S_i, X_j)).
\]

Fig. 1. Block diagram of the Multi-Subspace Feature Selection (MSFS) algorithm. Train and test phases are specified within the dashed boxes, and the ensemble model is indicated with \( E \). \( E \) is constructed during training and used during the test phase.

The optimal \( \hat{F}_i \) is then estimated through solving the following optimization problem:

\[
\max J_D(\hat{F}_i, c_i, S_i) + \min J_R(\hat{F}_i, S_i),
\]

which can be simplified in terms of optimization into the following form:

\[
\max J(J_D, J_R), \quad J(J_D, J_R) = J_D - J_R.
\]

Once the training phase, i.e., specified within a dashed box in Fig. 1, is over, test samples should be mapped onto their optimal feature space. In order to determine the right subspace for test samples, a classification model \( \psi_i \) is constructed for each class \( c_i \) in their corresponding subspace \( \hat{F}_i \). By this mean, each classification model \( \psi_i \) returns the posterior probability of a test sample \( \hat{x}_i \) belonging to the class \( c_i \) within the subspace \( \hat{F}_i \) as follows:

\[
\psi_i(\hat{x}_i) = p(c_i \mid \Phi(\hat{x}_i, \hat{F}_i)),
\]

where \( x_i \in \hat{F}_i \) denotes the representation of \( x_i \) in the subspace \( \hat{F}_i \subset F \).

The ensemble model \( E \), showed with a dash-dotted box in Fig. 1, is then completed by adding pairs of feature sets and classification models for each class to the ensemble as:

\[
E = \bigcup_{i=1}^{\lambda} [\hat{F}_i, \psi_i],
\]

where \([\cdot, \cdot]\) resembles a tuple. Thus, the output of ensemble for each test sample \( \hat{x}_i \in X \) would be the representation of \( \hat{x}_i \) in an optimal subspace as shown in Fig. 1 and in the following:

\[
E(\hat{x}_i) = \Phi(\hat{x}_i, \hat{F}_\alpha), \quad \alpha = \arg \max_{1 \leq r \leq \lambda} \psi_r(\hat{x}_i).
\]
Notice that $\psi(\cdot)$ is defined as a general classification model. The type of classification model, on the other hand, depends on the user preference. Generally, the most efficient approach is to consider $\psi(\cdot)$ as a one-class model that simply determines the posterior probability of the test sample belonging to the selected class $c_k$. Alternatively, a binary model can be used through dividing classes into two categories of matching and opponent classes. This is while, multi-class models are the least efficient models that can be used in this process.

IV. INTRUSION DETECTION SYSTEM

The designed IDS aims to detect and isolate cyber-attacks in a gas pipeline SCADA system [3]. In this scenario, the IDS should distinguish between the safe traffic (or normal class) and the data that is exposed to cyber-attacks. Additionally, the type of cyber-attacks, if any, should be determined by categorizing them in eight different groups, as described in Table 1.

<table>
<thead>
<tr>
<th>Class labels</th>
<th>Types of Cyber-Attacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Sample does not resemble any attack pattern.</td>
</tr>
<tr>
<td>1</td>
<td>Naive malicious response injection.</td>
</tr>
<tr>
<td>2</td>
<td>Complex malicious response injection.</td>
</tr>
<tr>
<td>3</td>
<td>Malicious state command injection.</td>
</tr>
<tr>
<td>4</td>
<td>Malicious parameter command injection.</td>
</tr>
<tr>
<td>5</td>
<td>Malicious function command injection.</td>
</tr>
<tr>
<td>6</td>
<td>Denial-of-service (DoS).</td>
</tr>
<tr>
<td>7</td>
<td>Reconnaissance.</td>
</tr>
</tbody>
</table>

The aforementioned classification problem is solved by making use of Decision Tree (DT) and $k$ Nearest Neighbours (kNN) algorithms, as shown in Fig. 2. Although numerous state-of-the-art classifiers exist in the literature that are more advanced compared to DT and kNN, we selected these classifiers for two reasons. Firstly, the effect of feature selection on the performance of classification is more noticeable using simpler classifiers such as DT and kNN, as they are usually less robust against redundant and non-informative features. In other words, the more a classifier is sensitive to bad quality of features, the more it shows the accuracy improvement obtained via FS. Secondly, the selected techniques are computationally less expensive than advanced methods such as Deep Neural Networks.

As illustrated in Fig. 2, the designed IDS framework employs 14 advanced FS techniques in addition to the proposed MSFS algorithm. Each of these methods results in a set of selected features that are obtained as the output of FS algorithms, which is denoted by $F(\cdot)$ in Fig. 2.

To simulate the experiments, initially a training dataset is attained from the given SCADA network and used to train all FS techniques (see Fig. 2). Then, the classification models are constructed w.r.t. each FS model. Once the training phase is completed, the testing phase is initiated by passing the network traffic through the constructed FS models. These models will reduce the size of data using the estimated optimal feature sets. The improved samples are then fed to the corresponding classification models to enable the attack identification. Notice that MSFS makes use of ensemble learning, and, thus, it uses
an ensemble of FS models and classifiers in the described framework.

Since the focus of this work is on FS, we do not consider classification challenges such as the presence of non-stationary environments. Nevertheless, the designed framework can be adapted to the case of non-stationary environments by resorting to available adaptive frameworks \cite{30}, \cite{31} for dealing with concept drift.

V. EXPERIMENTAL RESULTS

Here, the experimental setting is initially explained in Subsection V-A. The obtained results are then analyzed and discussed in terms of accuracy and standard deviation in Subsection V-B.

A. Experimental Setting

The employed intrusion detection dataset has originally 26 features and 97020 samples. The optimal number of features to be selected is estimated via the naive search, where the search ranges are obtained empirically.

A nested 10-fold cross-validation procedure is used to the statistical reliability of the experiments. This nested structure, enables the parameter tuning of classifiers, such as the value of \( k \) for kNN and depth of tree for DT, and hyper-parameters of the FS algorithms. For this mean, the grid search algorithm is utilized to ensure the optimal classification accuracy achieved by using the outputs of the FS algorithms.

B. Results Analysis

Fig. 3 shows the obtained accuracies through the cross-validation iterations for each FS method. Considering the results of DT, Fig. 3(a) indicates that MSFS has outperformed the other methods. This is while mRMR and ECFS are ranked second and third, albeit with a slight difference. Furthermore, Relieff, ILFS, UDFS, MutInfFS, InfFs, CFS, Fisher, L0-norm, RFE, Laplacian, MCFS, and FSV are ranked from fourth to 15-th, respectively. Although FSV improves the variance of the classification results, i.e., see Fig. 3(a), it seems that it is not compatible with the existing distribution in this case study, as it results in accuracy deterioration. Moreover, based on Fig. 3(a), the combination of DT with any of the selected FS algorithm will always improve the classification variance in this case study. Nonetheless, the achieved variances through the combination of FSV, Laplacian and MCFS and DT are considerably higher than that of other FS techniques.

Fig. 3(b) illustrates the obtained accuracies using the combination of kNN with FS algorithms. Similar to the results of DT, which is shown in Fig. 3(a), MSFS, mRMR, ECFS, Relieff, ILFS and UDFS are ranked from first to sixth, respectively. Nevertheless, the rest of FS methods exhibit different performances, when combined with kNN. Here, CFS, RFE, L0-norm and Fisher are ranked from seventh to tenth. On the other hand, the accuracy resulted through the combination of kNN with Laplacian, MutInfFS, InfFS, MCFS and FSV fall under the baseline accuracy, which imply the incompatibility of these combinations with the case study at hand. Moreover, employing InfFS and MutInfFS significantly increases the classification variance of kNN. FSV, Laplacian and MCFS also
The reported rankings for the achieved accuracies can be also seen in Fig. 4(a–c), in terms of classification error. In order to perform a precise study on the stability of the FS methods, we devise the averaged standard deviations of classification that is resulted using each algorithm. To begin with, Fig. 4(a) implies that MSFS, ECFS and mRMR are ranked as the first three in terms of standard deviation, when DT is used for classification. This is while RFE, ILFS, ReliefF, InfFS, MutInfFS, L0-norm, CFS, Fisher, UDFS, FSV, MCFS and Laplacian are ranked from fourth to 13-th, respectively, as shown in Fig. 4(a).

The averaged standard deviations resulted by means of kNN are illustrated in Fig. 4(b). Based on this figure, ECFS, mRMR, MSFS, L0-Norm, ILFS, ReliefF, CFS, RFE, UDFS, Fisher, FSV, Laplacian and MCFS are ranked from first to 13-th. On the other hand, in contrast to the rest of FS techniques, InfFS and MutInfFS increase the standard deviation compared to the baseline, and gain the last two ranks.

The overall standard deviation w.r.t. both classifiers can be seen in Fig. 4(c). In this figure, MSFS outperforms other FS methods in terms of the overall standard deviation. However, ILFS, ReliefF, ECFS and mRMR, which are ranked from second to fifth, have a negligible difference with MSFS in terms of standard deviation. RFE, CFS, L0-Norm, UDFS and Fisher are ranked from sixth to tenth with a higher difference with the first five ranks. The rest of the FS methods, result in a lower overall standard deviations compared to the baseline. These methods, namely Laplacian, MCFS, FSV, InfFS and deterio...
A novel feature selection algorithm, called MSFS, is proposed in this paper. The proposed MSFS finds a different subspace for a selected class, where it is optimally discriminated. Estimated multiple subspaces based on mutual information estimation, an ensemble model is then formed to enable classification via multiple subspaces. In order to evaluate the proposed method, the case of cyber-attack identification in a SCADA network of a gas pipeline is considered. For this mean, an IDS is designed to distinguish between seven types of cyber-attacks and the normal state in the SCADA system. Moreover, fifteen advanced FS techniques, including the proposed MSFS, are employed within the designed IDS to enable a comparative study on the selected case study. The experimental results indicate the superiority of the proposed method in terms of accuracy and standard deviation for identifying injected cyber-attacks in the given SCADA system.

VI. Conclusion

MutInfFS are ranked from 11-th to 15-th, respectively.

The overall improvement achieved using each FS method is shown in Fig. 4(d). MSFS results in the highest overall improvement in terms of accuracy and standard deviation, as shown in Fig. 4(d). mRMR, ECFS, ReliefF, ILFS and UDFS are ranked from second to sixth in terms of accuracy improvement. It is worthwhile to mention that the obtained improvement by these methods are considerably higher than the rest of techniques. On the contrary, FSV is the only algorithm that results in the accuracy deterioration. However, we find this due to the incompatibility of this method to the structure of the utilized data. On the other hand, the achieved improvements in terms of standard deviation are almost similar for most of the algorithms, except for FSV, Laplacian and MCFS that brought about less stability improvement compared to the others.

Another issue of concern is the dimensionality size of the data when performing FS. While FS algorithms aim to increase the performance using the selected features, they also endeavour to minimize the dimensionality size as much as possible in order to enhance the computational efficiency. In this regard, Fig. 5 shows the number of features that are selected and disregarded by FS methods. It can be seen that MCFS, Laplacian and FSV that were generally outperformed by other techniques have selected the least number of features. Thus, their lower performance may be due to the failure in detecting some of the important features, which brings about information loss. On the other hand, MSFS robustly recognizes the informative features and select them for the sake of classification. In other words, although algorithms such as MCFS, Laplacian and FSV may seem more desirable than MSFS in terms of efficiency, this efficiency is followed by accuracy deterioration in this case study, which is not desirable.

![Fig. 5: Number of selected and disregarded features by each FS method, where the original data has 26 features.](image)

### References


