Neural Networks Based Feature Selection from KDD Intrusion Detection Dataset

Adel Ammar, Khaled Al-Shalfan
College of Computer Sciences and Information
Computer Sciences Department
Al-Imam Mohammad Ibn Saud Islamic University
Riyadh, KSA
adel.ammar@ccis.imamu.edu.sa

Abstract—We present the application of a distinctive feature selection method based on neural networks to the problem of intrusion detection, in order to determine the most relevant network features. We use the same procedure for feature selection and for attack detection, which gives more consistency to the method. We apply this method to a case study and show its advantages compared to some existing feature selection approaches. We then measure its dependence to the network architecture and the learning database.

Keywords—Intrusion detection, network security, feature selection, KDD dataset, neural networks.

I. INTRODUCTION

For Intrusion Detection Systems (IDS), ranking the importance of input features is a problem of significant interest, since the elimination of irrelevant or useless inputs leads to a simplification of the problem and may allow faster and more accurate detection. This is especially critical for the construction of an efficient real-time IDS able to comply with the constraints of high speed networks.

We present, in this article, a feature selection method based on Neural Networks (NN), classifying traffic features according to their relative contribution to attack detection.

Section II introduces the method and describes its theoretical basis. Section III details the results of a case study for a single output classification NN, and reviews the advantages and limitations of the method. Finally, section IV draws a conclusion for the present work and mentions some open issues for future works.

II. THEORETICAL BASIS

The method we propose here for selecting connection features is based on feed-forward neural networks. It has been applied in another application by [1] and theoretically formulated by [2] who called it HVS (Heuristic for Variable Selection). Nevertheless, it has not yet been applied to intrusion detection, to the best of our knowledge.

We introduce the features that need to be ranked as inputs of a feed-forward neural network (with a single hidden layer) used as a classifier that distinguishes attacks from normal traffic. After the training process on a representative learning database, we assess the relative contribution of each feature as follows. We expect the contribution $C_{js}$ of a neuron $j$ of the hidden layer to the output $s$ according to the formula:

$$C_{js} = \frac{|W_{js}|}{\sum_{k=1}^{N_h}|W_{ks}|}$$

Where $W_{js}$ is the weight of the connection between a hidden neuron $k$ and the output $s$ and $N_h$ is the number of hidden neurons. Then, we obtain the contribution of an input neuron $i$ to the output according to the formula:

$$C_{is} = \sum_{j=1}^{N_h} C_{js} \cdot \frac{|W_{ij}|}{\sum_{k=1}^{N_h}|W_{kj}|}$$
Where $W_{ij}$ is the weight of the connection between the input neuron $i$ and a hidden neuron $j$ and $N_i$, is the number of inputs. The sum of input contributions is, therefore, equal to 1.

### III. Case Study on KDD Database

#### A. Calculation of features’ contribution

We have applied the HVS method described above, in a case study, to the KDD 99 intrusion detection benchmark [3]. This database originated from the 1998 DARPA Intrusion Detection Evaluation Program that was prepared and managed by MIT Lincoln Labs. The objective was to assess and evaluate research in intrusion detection [4]. The dataset was summarized into network connections with 41 features per connection. In order to measure the relevance of these features, we constructed a NN with a single output that distinguishes between normal traffic and attacks. The learning database used to train the NN consists of a 1% random extraction (4,940 samples) from the original KDD learning set (containing 494,021 connection records). A learning database with such a size is sufficient to achieve an accuracy rate of 92% on the KDD test set (composed of 311029 independent connection records).

Figure 1 depicts the obtained results, after applying the HVS method following (1) and (2). Features # 20 and 21 take a null contribution because they are constant in the whole KDD learning set. The same can be noticed for features # 9 and 15, which are almost constant. In fact, more than 99.999% of the KDD learning set connection records contain a null value for these two features. Features 7, 11 and 18 could also be excluded from the learning database since their contribution is remarkably little; while the most significant features are # 10, 22, 23, 34, 36, 39.

#### B. Checking the consistence of the method

In order to verify the consistence of the results, we selected a set of most significant features (calculated as in the section above) to be set as inputs of the classification NN, and compared the results with those obtained with the full set of inputs. Figure 2 shows these results after applying the networks to the testing databases. We note that we can keep only the most influential 12 features (out of 41), without significantly deteriorating neither the overall accuracy rate (Figure 2) nor the false positive and false negative rates (Figure 3).

<table>
<thead>
<tr>
<th>KDD feature</th>
<th>Relative contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>7%</td>
</tr>
<tr>
<td>#2</td>
<td>6%</td>
</tr>
<tr>
<td>#3</td>
<td>5%</td>
</tr>
<tr>
<td>#4</td>
<td>4%</td>
</tr>
<tr>
<td>#5</td>
<td>3%</td>
</tr>
<tr>
<td>#6</td>
<td>2%</td>
</tr>
<tr>
<td>#7</td>
<td>1%</td>
</tr>
</tbody>
</table>

![Figure 1](image1.png)

**Figure 1.** Relative contribution of each of the KDD 41 features to the detection of attacks (distinction between normal traffic and attacks of various types)

![Figure 2](image2.png)

**Figure 2.** Evolution of the overall accuracy rate according to the number of selected inputs

![Figure 3](image3.png)

**Figure 3.** Evolution of the false positive rate according to the number of selected inputs
c. Advantages of the method

The results shown above are consistent with those obtained by [5] and [6]. The latter used a totally different method which consists in deleting one of the features and measuring its impact on the result, using either a Neural Network or an SVM classifier. Compared to this approach, the method we have presented above shows several advantages:

- The deletion-based method needs to run as many trainings as the number of features, each time deleting one of the features while the HVS method ranks all the features after a unique training, and does not imply any complicated computation.
- The HVS method tends to be more accurate in selecting relevant features than the method used by [6] as explained in section III.A.2.
- The HVS method distinguishes well between features than the SVM based feature ranking used by [6] which yields remarkably close accuracy results for most of the features, with so slight variations that they could be of random origin.
- The HVS method reveals to be more precise in detecting irrelevant features than the method presented in [6]. For example, while features 20 and 21 are constant in the whole KDD learning dataset (as previously noticed by [5]), and features 9 and 15 almost constant and they were not detected as the least important features in [6].

On the other hand, in term of consistence of HVS method, we note that we can keep only the most important 12 features (out of 41), without significantly deteriorating neither the overall accuracy rate (Figure 2) nor the false positive and false negative rates (Figure 3). This number of features is close to the one retained by [9] (11 features) using rough sets and genetic algorithms. [6] conducted a similar test but showed a significant deterioration when selecting the most important 34 features (the overall accuracy rate decreased from 87% to 81% and the false positive rate increased from 6.7% to 18%). This tends to prove that our selection feature method is considerably more accurate than other cited methods. It should be also noticed that these latter results shown by [6] are not consistent with the Figures they obtained during the feature ranking since the deletion of only one feature (#10 or #35) decreased the accuracy of their network to less than 55%. They did not precise on which database they tested their result. intuitively the results they gave for the SVM classification suggests that they tested on only a part of the KDD training dataset (so with a very close distribution to that of the learning database) while we tested on the independent KDD testing dataset (which an entirely different distribution of attacks, and containing new attack types), which is more realistic. Obviously, testing on the training data set yields an artificially high performance.

![Relative contribution](image1)

**Figure 9.** Relative contribution of each of the KDD 41 features to the detection of normal traffic, calculated for five different networks (with a number of hidden neuron varying from 16 to 20).

Furthermore, the contributions of the inputs, calculated using the HVS method, are largely independent of the network architecture, as shown in Figure 9. This Figure depicts the result of use of the HVS method to five networks with different internal architectures. The five tests show very close results. Nevertheless, this stands only if the number of hidden neurons is sufficient to resolve the classification problem.

![Relative contribution](image2)

**Figure 10.** Relative contribution of each of the KDD 41 features to the detection of normal traffic, calculated for five different randomly extracted learning databases of the same size.
IV. RELATED WORK

There exists other feature selection methods also based on neural networks, theoretically described in [7], which we should consider and compare in future works, in the context of intrusion detection. The one we used is the simplest to calculate. We need, thoroughly, to compare the HVS method to other feature selection methods mentioned here, such as SVDF-based method or the one used by [5] based on information gain.

Besides, several recent papers presented various feature selection techniques applied to the KDD features. Reference [8] proposed a hybrid approach combining the information gain ratio (IGR) and the k-means classifier. Reference [9] proposed a feature selection method based on Rough Sets, improved Genetic Algorithms and clustering. Then they used the SVM classifier for performance evaluation on the KDD database. Reference [10] proposed a clustering-based classifier selection method. The method selects the best classifier on similar clusters, compares it with the best classifier on the nearest cluster and chooses the better one to make the system decision. It showed better results than the Clustering and Selection (CS) method.

We should compare our method to these various techniques in a future work. Nevertheless, most of the cited works tested their methods on an extraction from the KDD learning database. They did not test them on the KDD database originally dedicated to testing and containing new attacks as we did in this paper. This demonstrates the potential of the method to detect new attacks and gives more realistic results than the results produced by testing on only a part of the KDD learning database.

V. CONCLUSION AND FUTURE WORK

We have shown that the HVS method we presented in this work can be directly and efficiently applied to the problem of intrusion detection, in order to assess the most important features that contribute to attack detection. We could then select a set of most relevant features to accelerate the detection process. An important advantage of the approach, compared to existing methods (like [9]), is that the same technique (feed-forward neural networks) can be used for both feature selection and attack detection, which gives more consistency to the method. Furthermore, the method is almost independent of the used networks’ architecture. Further rigorous tests should be conducted to measure accurately the dependence of the HVS method to the learning database, with databases of different sizes. This dependence should not be an obstacle, however, since, in most applications, the learning database is set once for all.

ACKNOWLEDGMENT

This work is a partial result of a research project supported by grant number INF 36-8-08 from King Abdul Aziz City for Sciences and Technology (KACST), Riyadh, Kingdom of Saudi Arabia.

REFERENCES


