A Hybrid Method for Intrusion Detection Using SVM and k-NN

Chapter · February 2021

DOI: 10.1007/978-3-030-67187-7_13

3 authors, including:

Maheep Singh
National Institute of Technology Uttarakhand
32 PUBLICATIONS **143** CITATIONS
SEE PROFILE

Krishan Berwal
99 PUBLICATIONS **1,937** CITATIONS
SEE PROFILE

A hybrid method for Intrusion Detection based on SVM and k-NN

Abstract. The growing amount of data has necessitated the use of Intrusion Detection Systems in the modern days. The performance of an IDS is determined by the feature selection and classifiers. The traditional IDS fail to give satisfactory performance in today's world of growing data. In this paper, we propose a hybrid, two step approach for intrusion detection. In the first step, the data is classified into different classes with the help of Support Vector Machines. After the first steps, the records whose classification is not certain are passed on to the second step in which we use k-Nearest Neighbor method to further classify the incoming request into its respective class.Then we show how our proposed model compares with some recent approaches.

1 Introduction

In this world of ever growing information, Network Security holds utmost importance. Network Intrusion detection is the field of network security which attempts to detect any unwanted intrusion in the network, which may be in the form of an attack or any attempt to steal or misuse the information.

There are two types of Intrusion Detection Systems

- Signature-Based Intrusion Detection System
- Anomaly Based Intrusion Detection System

The former take reference from a database of intrusions identified attacks and checks if the new request matches any of them. While the latter tries to detect anomalies in the behaviour of the incoming request and flags them as an intrusion if they seem inappropriate.

As the amount of data grows and takes varied form, the first type fails at providing satisfactory results in classification. This necessitates the development of new Anomaly Based Intrusion Detection approaches. Over the past few years it has been found out that SVM can come in handy at such kind of classification and various past works have proposed models based on the same. With a culmination of SVM and k-NN we propose a reliable two step approach for intrusion detection in this work.

1.1 Support Vector Machines

A Support Vector Machine is a definitive classifier that, on being given a labeled data set, produces an optimum hyperplane that maximizes the margin between

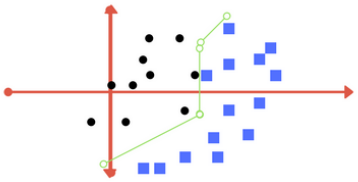

Fig. 1: Classification of example dataset using linear SVM [Chapter 2 : SVM (Support Vector Machine)-Theory]. Retrieved May 3, 2019 from https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72

the given classes. The vectors which define this hyperplane are termed as Support Vectors.

In the given figure 1 the black dots and the blue squares represent the two classes and the the green hyperplane acts as a divisor among the classes.

1.2 k-Nearest Neighbor

The k-Nearest Neighbor is yet another method for classification and regression. In this method a current data point is classified into a class depending upon the class which is most common among it's k nearest neighbors.

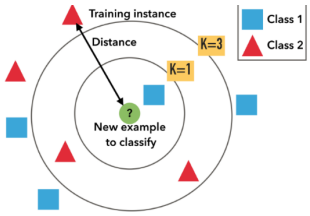

Fig. 2: Classification of a new entity based on k-NN for k=1 and k=3 [A Quick Introduction to K-Nearest Neighbors Algorithm]. Retrieved May 3, 2019 from https://blog.usejournal.com/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7

The rest of this paper is organized in the following way. Section 2 provides a description of the data-sets utilized for experimentation purposes. Section 3 provides for an evaluation criteria on which we measure the performance of the model. Section 4 provides the method itself and the inner functioning of the model in detail. Section 5 shows the various experiments and analysis performed on the dataset in order to provide a comparison between the different models. Section 6 concludes the work and provides the future scope and improvements in the proposed work.

2 Dataset used

In this work, we have taken the NSL-KDD dataset for evaluation purpose. It is a benchmark dataset for checking the performance of Intrusion Detection Systems. The NSL-KDD Dataset is an improvement over the KDD'99 data set. In each of the record in the dataset, there are 42 attributes. The first 41 attributes describe the flow of the data and the 42nd attribute is the label calssifying the given record as normal or one of the four attack types.

	Normal	DoS	Probe	U2R	R2L	Total
KDDTrain^{+}	67,343	45,927	11,656	52	995	125,973
KDDTest^{+}	9,711	7,458	2,421	200	2,754	22,544
KDDTest^{-21}	2,152	4,342	2,402	200	2,754	11,850

Table 1: Instances in the NSL-KDD dataset

Table 1 gives a description of number of instances of each attack type in the NSL-KDD Dataset. The 41 attributes in the dataset can be classified as 9 basic features, 13 content related features, 9 time related traffic features and 10 host based traffic features.

3 Evaluation Criteria

The performance of the model is analyzed with the help of a confusion matrix as shown in table 1. TP stands for True Positives, it means the number of positive records correctly classified. FP stands for False Positives, the number of negative records wrongly classified. FN stands for False Negatives, the number of positive records wrongly classified. TN stands for True Negatives, the number of negative records correctly classified.

		Predicted	
		Attack	Normal
Actual	Attack	TP	FN
	Normal	FP	TN

Table 2: Confusion Matrix

Based on the above matrix following measures calculated.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

$$Recall = \frac{TP}{TP + FN}$$

It is interesting to note here that while accuracy is good measure for performance evaluation, recall is an equally important measure because of the highly unbalanced nature of the data. In other words catching all the attacks is more important than correctly classifying the genuine requests.

4 Methodology

Figure 3 provides the overall architechture for the proposed model involving the use of SVM and k-NN. The entire process is divided into two steps.

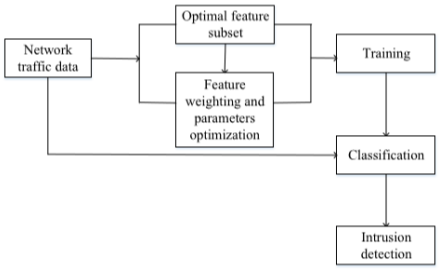

Fig. 3: Architecture of the proposed model

We use parameters w and b to write our classifier as

$$h_{w,b}(x) = g(w^x + b)$$

The functional margin of (w, b) with respect to the training example

$$\gamma^{(i)} = y^{(i)}(w^T x + b)$$

Now we define the geometric margin of (w, b) with respect to the training example as

$$\gamma^{(i)} = y^{(i)} \left(\left(\frac{w}{||w||} \right)^T x^{(i)} + \frac{b}{||w||} \right)$$

The larger the geometric margin, greater is the probability of our classification to be correct.

4.1 Step 1

In the first step we use Support Vector Machine to determine which class the input connection belongs to. In the given dataset there are five classes - Normal, DoS, Probe, R2L and U2R. These classes are mapped to 0, 1, 2, 3 and 4 respectively. After the classification via SVM of the given class, we find the certainty

Testing set	Method	Accuracy	Precision	DR	F1-score	FAR
$KDDTest^{+}$	C4.5	81.01%	96.67%	69.02%	80.54%	3.14%
	RF	77.06%	96.49%	61.96%	75.46%	2.98%
	k-NN	76.70%	97.21%	60.81%	74.82%	2.31%
	BPNN	75.34%	91.74%	62.29%	74.20%	7.41%
	NB	77.75%	93.36%	65.57%	77.04%	6.16%
	SVM + k-NN	**94.92%**	**98.72%**	**92.28%**	**95.39%**	**1.59%**
$KDDTest^{-21}$	C4.5	63.95%	95.08%	59.00%	72.82%	13.75%
	RF	55.34%	94.27%	48.37%	63.93%	13.24%
	k-NN	55.73%	95.54%	48.15%	64.03%	10.13%
	BPNN	57.46%	93.30%	51.73%	66.56%	16.73%
	NB	58.12%	90.20%	54.77%	68.16%	26.81%
	SVM + k-NN	**91.35%**	**98.76%**	**90.57%**	**94.49%**	**5.11%**

Table 3: A comparison of performance between the proposed method and other popular methods in recent works

of our classification as the geometric margin of our given example from the hyperplane. If this value exceeds a given threshold, we call this classification as certain and otherwise as uncertain. Now the connections for which classification came ou to be uncertain are forwarded to step 2 for further evaluation.

4.2 Step 2

In the second step we only receive those records as inputs which were classified as being uncertain along with their predicted class. We then apply the process of classification via k-Nearest Neighbor approach to classify the given record into the desired class. After the process is complete we finally output the desired class which the connection must belong to, i.e., normal or anomaly.

5 Experiment and Analysis

To analyze the performance of the proposed model, we conducted a series of experiments over the NSL-KDD dataset and compared the performance with some other recent works.

Figure 4 shows the comparison of accuracy for different values of k in k-NN classification. In both of the test sets, the KDDTest and the KDDTest21, we observe that the best value of accuracy is found from the values k = 10 to k = 16. Thus choosing a value of k in this range would yield best results.

Table 4 denotes the confusion matrix in case of classification on KDDTest dataset and shows a quantitative comparison between the approaches using only SVM, only k-NN and both.

While table 5 shows the similar comparison for the KDDTest21 test set.

The Table 6 Confusion matrix stating True and false detections for all the 5 given classes.

Finally the Table 3 presents a quantitative comparison between the performance of the proposed model with some general methods as well as some hybrid approaches used in some of the recent works.

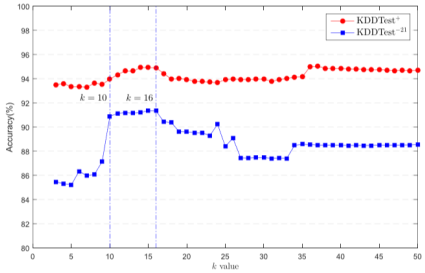

Fig. 4: Value of k vs accuracy for the test set

	Predicted	Step 1		Step 2		Step 1 & 2	
		Attack	Normal	Attack	Normal	Attack	Normal
Actual	Attack	8,634	201	3,208	790	11,842	991
	Normal	41	2,138	113	7,419	154	9,557

Table 4: Confusion matrix for classification on KDDTest

	Predicted	Step 1		Step 2		Step 1 & 2	
		Attack	Normal	Attack	Normal	Attack	Normal
Actual	Attack	5,499	201	3,284	714	8,783	915
	Normal	35	540	75	1,502	110	2,042

Table 5: Confusion matrix for classification on KDDTest21

Predicted class / actual class	Normal	Dos	Probe	R2L	U2R
Normal	9107	372	217	11	4
Dos	1051	6206	96	103	2
Probe	332	91	1982	14	2
R2L	2005	53	93	597	6
U2R	136	13	16	9	26

Table 6: Confusion matrix for classification of all the given classes using the proposed appraoch

6 Conclusion and Future Works

In this paper we presented an effective two step approach for implementing an Intrusion Detection System. In step 1 we used Support Vector Machines to try to classify a given connection as a normal connection or an attack. The incoming connections which could not be classified into a desired class with certainty were forwarded to the second step in order to further attempt to clsasify with the help of k-Nearest Neighbor algorithm. We then with several analysis and experimentation analyzed the performance gained by the proposed model against the regular approaches as well as some hybrid approaches in some recent works.
In future steps can be taken in order to improve the model further. Different hybrid models may be simulated based on different classification techniques to improve the results. Also one may involve deep learning approaches and attempt to improve the performance of the system with multiple layers of neurons combining them with other binary classifiers for optimized results.

References

1. K. Wu, Z. Chen and W. Li, "A Novel Intrusion Detection Model for a Massive Network Using Convolutional Neural Networks," in IEEE Access, vol. 6, pp. 50850-50859, 2018. doi: 10.1109/ACCESS.2018.2868993
2. S. Naseer et al., "Enhanced Network Anomaly Detection Based on Deep Neural Networks," in IEEE Access, vol. 6, pp. 48231-48246, 2018. doi: 10.1109/ACCESS.2018.2863036
3. L. Chen, Y. Gao, G. Chen and H. Zhang, "Metric All-k-Nearest-Neighbor Search," in IEEE Transactions on Knowledge and Data Engineering, vol. 28, no. 1, pp. 98-112, 1 Jan. 2016. doi: 10.1109/TKDE.2015.2453954
4. Jaewook Lee and Daewon Lee, "An improved cluster labeling method for support vector clustering," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 3, pp. 461-464, March 2005.
5. I. Ahmad, M. Basheri, M. J. Iqbal and A. Rahim, "Performance Comparison of Support Vector Machine, Random Forest, and Extreme Learning Machine for Intrusion Detection," in IEEE Access, vol. 6, pp. 33789-33795, 2018.
6. Xiujuan Wang, Chenxi Zhang and Kangfeng Zheng, "Intrusion detection algorithm based on density, cluster centers, and nearest neighbors," in China Communications, vol. 13, no. 7, pp. 24-31, July 2016.
7. A. Bryant and K. Cios, "RNN-DBSCAN: A Density-Based Clustering Algorithm Using Reverse Nearest Neighbor Density Estimates," in IEEE Transactions on Knowledge and Data Engineering, vol. 30, no. 6, pp. 1109-1121, 1 June 2018.
8. F. A. Khan, A. Gumaei, A. Derhab and A. Hussain, "A Novel Two-Stage Deep Learning Model for Efficient Network Intrusion Detection," in IEEE Access, vol. 7, pp. 30373-30385, 2019.
9. Z. Wang, "Deep Learning-Based Intrusion Detection With Adversaries," in IEEE Access, vol. 6, pp. 38367-38384, 2018.
10. F. Angiulli and A. Astorino, "Scaling Up Support Vector Machines Using Nearest Neighbor Condensation," in IEEE Transactions on Neural Networks, vol. 21, no. 2, pp. 351-357, Feb. 2010.

11. D. Yu, G. Liu, M. Guo, X. Liu and S. Yao, "Density Peaks Clustering Based on Weighted Local Density Sequence and Nearest Neighbor Assignment," in IEEE Access, vol. 7, pp. 34301-34317, 2019.
12. P. Tao, Z. Sun and Z. Sun, "An Improved Intrusion Detection Algorithm Based on GA and SVM," in IEEE Access, vol. 6, pp. 13624-13631, 2018.
13. S. Pan, T. Morris and U. Adhikari, "Developing a Hybrid Intrusion Detection System Using Data Mining for Power Systems," in IEEE Transactions on Smart Grid, vol. 6, no. 6, pp. 3104-3113, Nov. 2015.
14. J. R. Yost, "The March of IDES: Early History of Intrusion-Detection Expert Systems," in IEEE Annals of the History of Computing, vol. 38, no. 4, pp. 42-54, Oct.-Dec. 2016.