Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 04, No. 03, December 2018.

FEATURE PRUNING METHOD FOR HIDDEN MARKOV MODEL-BASED ANOMALY DETECTION: A COMPARISON OF PERFORMANCE

Sulaiman Alhaidari and Mohamed Zohdy

(Received: 10-Oct.-2018, Accepted: 10-Nov.-2018)

ABSTRACT

Selecting effective and significant features for Hidden Markov Model (HMM) is very important for detecting anomalies in databases. The goal of this research is to identify the most salient and important features in building HMM. In order to improve the performance of HMM, an approach of feature pruning is proposed. This approach is effective in detecting and classifying anomalies, very simple and easily implemented. Also, it is able to reduce computational complexity and time without compromising the model accuracy. In this work, the proposed approach is applied to NSL-KDD (the new version of KDD Cup 99), DDoS, IoTPOT and UNSW_NB15 data sets. Those data sets are used to perform a comparative study that involves full feature set and a subset of significant features. The experimental results show better performance in terms of efficiency and providing higher accuracy and lower false positive rate with reduced number of features, as well as eliminating irrelevant redundant or noisy features.

KEYWORDS

Anomaly detection; Feature pruning; Hidden Markov Model; NSL-KDD; DDoS; UNSW_NB15; IoTPOT.

1. INTRODUCTION

Increasing size of data and network traffic has made information security more complex and challenging than at any time in the past. Information security is intended to protect information (data) and information systems from any malicious activities and unauthorized access [1]-[9]. However, the increasing size of data maximizes number of computations and minimizes detection accuracy rates [9]. Therefore, in recent years, many researchers have worked on this problem and applied the feature pruning method on several data sets to improve the detection performance and obtain faster and more cost-effective results. Using a feature pruning method for machine learning is a way to solve this problem.

This research examines the full feature set on NSL-KDD, DDoS, IoTPOT and UNSW_NB15 to reduce the number of features and identify the most salient and significant features that give higher accuracy and lower false positive rate. Therefore, a subset of significant features in detecting anomalies can be obtained by using one of the most popular machine learning techniques, HMM. These significant features can then be used in building an HMM that can effectively achieve much better prediction and detection performance. The remainder of this paper is organized as follows: Following the introduction to our work in Section 1, Section 2 reviews HMM and its problems. Section 3 describes the approach, followed by Section 4 which illustrates the experiments of this study. At the end, conclusions are presented in Section 5.

2. HIDDEN MARKOV MODELS

HMMs have been extensively utilized in many applications, such as speech recognition, finance, computer vision and bioinformatics. Hidden Markov Models can be defined as a tool for representing different probability distributions over sequences of observed variables [1]. In HMM, the sequence of observations is $O = \{O_1, O_2, \dots, O_T\}$, where O_T is one of the observations symbols and T is the number of observations in the sequence. This sequence that goes through over time is observable and it's

S. Alhaidari* and M. Zohdy are with School of Engineering and Computer Science, Oakland University, Rochester, MI 48309, U.S.A. Email: *salhaidari@oakland.edu.

generated by a stochastic process. In contrast, the sequence of states X= {X1, X2…XN}, where N the number of states in the model, is not visible to the observer; therefore, it's not directly observed in this process. In a Markov chain, the probability of each state at time t is predicted based only on the previous state at time t-1 as shown in Equation (1). Figure 1 shows the first-order Hidden Markov model.

$$P = (X_{t+1}|X_1,..X_N) = P(X_{t+1}|X_t) \quad (1)$$

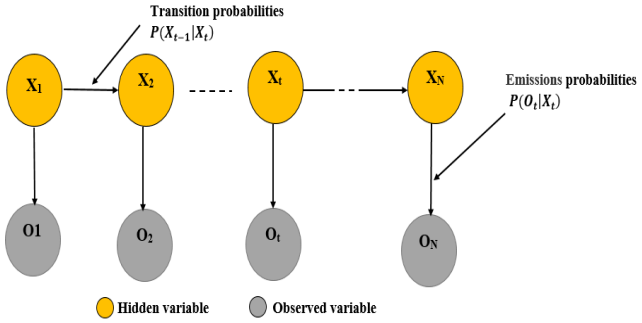

Figure 1. First-order hidden Markov model.

Mathematically, an HMM is defined as:

$$\lambda = (A, B, \pi) \quad (2)$$

where,
A is the state transition probability distribution. It's a single matrix N × N (N is the number of states), with each element a_{ij} representing the probability transitioning from state X_{t-1}, i to X_t,j as shown in Figure 2. It can be written as:

$$a_{ij} = P(X_{t-1}, j = 1|X_t, i = 1) \quad (3)$$

P is the emission probability. It is a single matrix N × M (N is the number of states and M is the number of emissions), where each element b_{kj} represents the probability of making observation O_t,k given state X_t,j as shown in Figure 2. It can be written as:

$$b_{Kj} = P(O_t = K|X_t = i) \quad (4)$$

$\boldsymbol{\pi}$ is the initial state distribution. I is an initial vector that contains the probabilities of starting in each of the states. It can be written as:

$$P(X_1|\pi) = \prod_{i=1}^{N} \pi_i \quad (5)$$

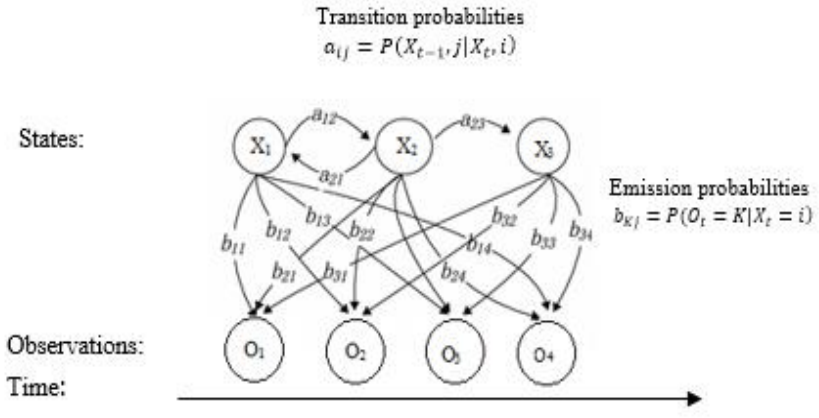

Figure 2. Structure of a hidden Markov model system, where the arrows between states X_1, X_2 and X_3 represent state transition probabilities and the arrows from states to emissions represent emission probabilities O_1,O_2,O_3,O_4.

Thus, an HMM can be characterized by a set of three parameters: the probability distribution of the initial states (π), the probability matrix of transition probabilities between states (A) and the probability matrix of emission probabilities for each state (B). HMM can be used in real-world applications by solving the following three fundamental problems:

- **The Evaluation Problem:** Given an HMM $\lambda = (A, B, \pi)$ and a sequence of observations $O = \{O_1, O_2, \dots, O_T\}$, compute the probability (likelihood)) of the observed sequence that was generated from the system given the model $P = (O|\lambda)$. This problem can be solved by two algorithms; namely, the forward algorithm and the backward algorithm.
- **The Decoding Problem:** Given an HMM $\lambda = (A, B, \pi)$ and a sequence of observations $O = \{O_1, O_2, \dots, O_T\}$, find the most likely hidden state sequence X_1, X_2,…X_N that produced the observed sequence. This problem is solved using Viterbi algorithm.
- **The Learning Problem:** Given an HMM **(**$\lambda = (A, B, \pi)$ and a sequence of observations $O = \{O_1, O_2, \dots, O_T\}$, adjust the model parameters (A, B, π) to maximize $P = (O|\lambda)$ to obtain the most descriptive model for the system. The Baum-Welch algorithm is used to solve this problem.

More expansion of these problems mentioned above and their solutions can be found in [2]

3. FEATURE PRUNING METHOD

Feature pruning method is a method of eliminating features from the original dataset to obtain a subset of features. Reducing features from the full data set will not only reduce the data size that is required to process and the computational complexity, but selecting a good feature set also helps to improve the efficiency and accuracy of the detection model approach. It plays a key role in building detection models. On the other hand, using all features without applying feature pruning method might increase the overhead of the model, which leads to increase the time to build the model [7]. In this study, we present a feature pruning method which was built to be used to choose certain features from a given feature set. Our aim here is to find the significant feature set that gives the highest accuracy. In order to perform feature-pruning method, we first need to standardize the data to a normal distribution and then combine the standardized data to one sequence of observation, which then could then be used afterward with Viterbi algorithm to compute the most likely state sequence and then be compared to the actual sequence of states to determine the accuracy of the feature. The feature pruning algorithm that we implemented automates this process, eliminates each feature from the full set of the features and then checks the accuracy of the subset of features. More features that are least significant are eliminated if the obtained accuracy is within a certain tolerance of the accuracy, equal or higher than the previous accuracy of every feature combined. This process continues until no improvement of the accuracy is observed on elimination of features.

4. EXPERIMENTS

4.1 Method of Performance Testing

To evaluate the performance of our proposed model and how accurate the model classifies and predicts the class label of attack and non-attack, we need to know the following four terms: True Positive (TP): the number of attack instances classified as attacks; True Negative (TN): the number of non-attack instances classified as non-attacks; False Negative (FN): the number of attack instances classified as non-attacks; False Positive (FP): the number of non-attack instances classified as attacks. For this study, we used the following performance measures to test the performance of the proposed model:

Accuracy: It is the ratio of the total number of correctly predicted instances to the total number of all instances. In our study, accuracy is measured by using the Viterbi algorithm to generate a likely state sequence and compare it to the known state sequence to get TP, FP, FN and TN. The accuracy can be calculated by using the following equation:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

Error rate: It is the ratio of the total number of misclassifications to the total number of all predictions.

Algorithm 1. Feature pruning algorithm for HMM.

1: Begin
2: feature_pruning (X, L, Already_Cheched)
3: **Input**: X-The number of desired features
L-List of N feature Vectors
Already_Cheched- A set containing already checked combinations of features
4: **Output**: R-The reduced feature set of length X
5: **if** Already_Cheched contains (L) **then**
6: Return

7: **end**

8: **if** N-X=0 **then** //L contains the desired of features so test accuracy

9: Return L, evaluate accuracy ()

10: **else**

11: A_i= {All features in L except feature i} for all i=1,…, N //Create subsets of L with one less feature
12: Return max (feature_pruning (X,A_1), feature_pruning (X,A_2), …, feature_pruning (X,A_N) //Return the subset with best accuracy

13: **end**

14: End

The error rate can be calculated by using the following equation:

$$Error\ rate = \frac{FP + FN}{TP + FP + TN + FN} \tag{7}$$

Fall-out: It is the ratio of the number of detected false positives to the total number of predictions. The fall-out can be calculated by using the following equation:

$$Fall\ out = \frac{FP}{FP + TN} \tag{8}$$

Sensitivity: It is the ratio of the total number of detected true positive instances that are correctly identified as attacks to the total number of positive instances. Sensitivity can be calculated by using the following equation:

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{9}$$

Specificity: It is the ratio of the total number of detected true negative instances that are correctly identified as non-attacks to all the negative instances. Specificity can be calculated by using the following equation:

$$Specificity = \frac{TN}{TN + FP} \tag{10}$$

Precision and recall: Precision and recall measures are widely used for performance evaluation of machine-learning classification methods. Precision is the ratio of the total number of positive instances that are correctly identified as attacks to the total number of attacks. Recall is the ratio of the total number of instances that are correctly identified as attacks to the total number of all the instances that are correctly identified as attacks and misidentified attacks (it is the same as the sensitivity). Precision and recall can be calculated by using the following equations:

$$Precision = \frac{TP}{TP + FP} \quad (11)$$

$$Recall = \frac{TP}{TP + FN} \quad (12)$$

F-measure: F-measure is a testing score that tests the accuracy of the model and considers both precision and recall. F-measure can be calculated by using the following equation:

$$F - meaure = 2 * \frac{Precision * recall}{Precision + reacall} \quad (13)$$

4.2 Datasets and Discussion

4.2.1 NSL-KDD Dataset

The NSL-KDD 2009 dataset is a revision of the KDD'99 dataset that is extracted from the KDD'99 dataset to solve some of the inherent problems [3]. The size of NSL-KDD dataset is smaller than that of the original KDD'99. In each record of the set, there are 41 different features of the flow and one more attribute for class assigned to each record as either an attack type or a normal type as shown in Table 1. The NSL-KDD dataset contains 23 types of attack in the training set and 17 additional attack types in the testing set (New attacks that are not included in the training set) that are classified into four major categories: DoS, Probe, R2L and U2R, as shown in Table 2. The features in bold in Table 1 are the significant features obtained by the feature pruning algorithm used in our experiment on NSL-KDD dataset. Note: some features such as IP addresses, protocol type and source/destination port numbers were ignored from the initial feature set to guarantee that the results of the detection model are not dependent on particular acquisition biases. Table 2 summarizes the results of the experiment and Figure 3 shows the results in a graphical way.

Table 1. List of features of NSL-KDD dataset.

No.	Feature Name	No.	Feature Name
1	Duration	22	is_guest_login
2	protocol type	23	**count**
3	service	24	srv_count
4	**flag**	25	**serror_rate**
5	src_bytes	26	**srv_serror_rate**
6	**dst_bytes**	27	**rerror_rate**
7	land	28	srv_rerror_rate
8	wrong_fragment	29	**same_srv_rate**
9	urgent	30	diff_srv_rate
10	hot	31	srv_diff_host_rate
11	num_failed_logins	32	dst_host_count
12	logged_in	33	**dst_host_srv_count**
13	num_compromised	34	dst_host_same_srv_rate
14	root_shell	35	dst_host_diff_srv_rate
15	su_attempted	36	dst_host_same_src_port_rate
16	num_root	37	dst_host_srv_diff_host_rate
17	num_file_creations	38	**dst_host_serror_rate**
18	num_shells	39	**dst_host_srv_serror_rate**
19	num_access_files	40	dst_host_rerror_rate
20	num_outbound_cmds	41	dst_host_srv_rerror_rate
21	is_host_login	42	

Table 2. Summary of the experiment results for NSL_KDD dataset.

Measure	Training/Testing (80/20 %)	
	All features	**Obtained features**
accuracy	0.8431	0.8815
error rate	0.1569	0.1185
fall-out	0.1374	0.0477
sensitivity/ recall	0.8207	0.8009
specificity	0.8626	0.9523
precision	0.8388	0.9365
F-measure	0.8296	0.8634

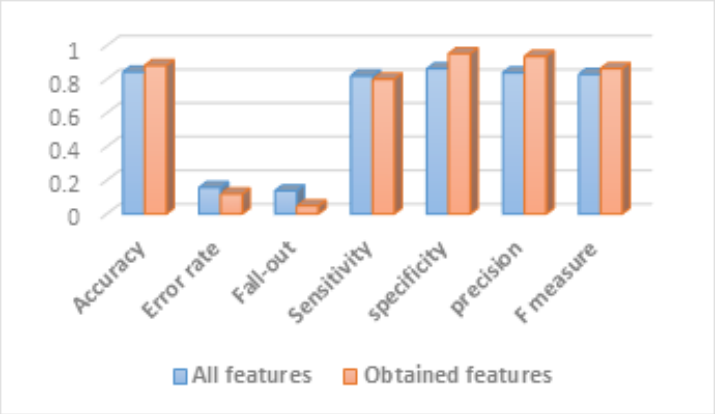

Figure 3. Comparison chart of the performance of all features and obtained features for NSL-KDD dataset.

4.2.2 DDoS Dataset

DDoS dataset is used for the evaluation [4]. It contains 27 features, which are labeled as either normal or an attack, as shown in Table 3. The DDoS dataset includes four types of the DDoS attack, which are: Smurf, UDP-Flood, HTTP-Flood and SIDDOS. From this dataset, a small portion of training and testing data is selected for experimentation. The features in bold in Table 3 are the significant features obtained by the feature pruning algorithm used in our experiment on DDoS dataset. Certain features were ignored from the initial feature set due to the same reason mentioned for the NSL-KDD dataset case. Table 4 summarizes the results of the experiment and Figure 4 shows the results in a graphical way.

Table 3. List of features of DDoS dataset.

No.	Feature Name	No.	Feature Name	No.	Feature Name	No.	Feature Name
1	src add	8	flags	15	pkt in	22	utilization
2	des add	9	fid	16	pktout	23	pkt delay
3	pkt id	10	**seq number**	17	pktr	24	pkt send time
4	from node	11	number of pkt	18	pkt delay node	25	pkt reseved time
5	to node	12	number of byte	19	**pktrate**	26	first pkt sent
6	pkt type	13	node name from	20	**byte rate**	27	**last pkt reseved**
7	**pkt size**	14	node name to	21	pkt avg size	28	

4.2.3 IoTPOT Dataset

IoTPOT dataset is Telnet IoT honeypot that analyzes malware attacks against various IoT devices, such as: IP Camera, DVR, Wireless Router, Customer Premises Equipment, Industrial Video Server, TV

Table 4. Summary of the experiment results for DDoS dataset.

Measure	Training/Testing (80/20 %)	
	All features	**Obtained features**
accuracy	0.8985	0.9741
error rate	0.1015	0.0259
fall-out	0.0098	0.0104
sensitivity/ recall	0.1071	0.8413
specificity	0.9902	0.9896
precision	0.5583	0.9038
F-measure	0.1797	0.8714

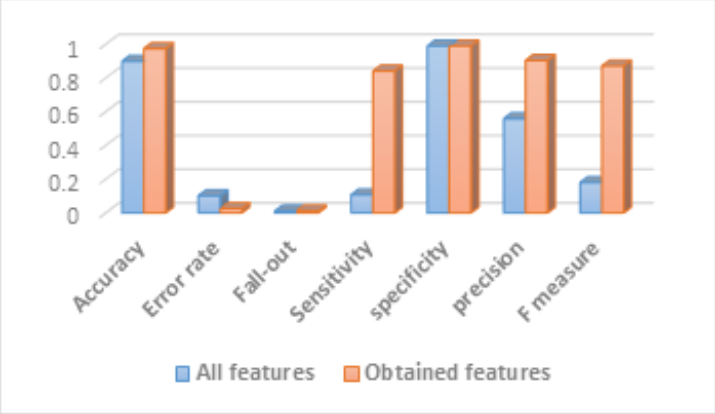

Figure 4. Comparison chart of the performance of all features and obtained features for DDoS dataset.

Receiver, Heat Pump, Environment Monitoring Unit (EMU system), Digital Video Scalar, home routers [4]. It analyzes the increase in Telnet-based attacks. This dataset contains 9 attack categories and 41 features, as shown in Table 5. The features are extracted by using NetMate tool set. Table 6 summarizes the results of the experiment and Figure 5 shows the results in a graphical way.

Table 5. List of features of IoTPOT dataset.

No.	Feature Name	No.	Feature Name	No.	Feature Name	No.	Feature Name
1	srcip	12	**sflow_fbytes**	23	std_biat	34	**min_biat**
2	srcport	13	**sflow_bpackets**	24	std_active	35	**mean_biat**
3	dstip	14	**sflow_bbytes**	25	**min_fpktl**	36	**max_biat**
4	dstport	15	fpsh_cnt	26	**mean_fpktl**	37	**duration**
5	**total_fpackets**	16	furg_cnt	27	**min_bpktl**	38	**min_active**
6	**total_fvolume**	17	bpsh_cnt	28	**mean_bpktl**	39	**mean_active**
7	**total_bpackets**	18	burg_cnt	29	**max_bpktl**	40	**max_active**
8	**total_bvolume**	19	**std_fpktl**	30	**max_fpktl**	41	**min_idle**
9	**total_fhlen**	20	**std_bpktl**	31	**min_fiat**	42	**mean_idle**
10	**total_bhlen**	21	std_fiat	32	**mean_fiat**	43	**max_idle**
11	**sflow_fpackets**	22	std_idle	33	**max_fiat**	44	Proto

4.2.4 UNSW_NB15 Dataset

The UNSW-NB 15 dataset was published in 2015 [5]. This dataset contains 9 attack categories and 48 features (shown in Table 7) which are categorized into 6 groups; namely: Flow Features, Basic Features, Content Features, Time Features, Additional Generated Features (General Purpose Features and Connection Features) and Labelled Features. Table 8 summarizes the results of the experiment and Figure 6 shows the results in a graphical way.

Figure 7 shows the ROC curves of the performance of HMM for the obtained features using the following datasets: NSL-KDD, DDoS, IoTPOT and UNSW_NB15. In terms of accuracy and the area

Table 6. Summary of the experiment results for IoTPOT dataset.

Measure	Training/Testing (80/20 %)	
	All features	**Obtained features**
accuracy	0.9222	0.9467
rrror rate	0.0778	0.0533
fall-out	0.0069	0.0188
sensitivity/ recall	0.1312	0.4786
specificity	0.9931	0.9812
precision	0.6316	0.6518
F-measure	0.2173	0.5519

Figure 5. Comparison chart of the performance of all features and obtained features for IoTPOT dataset.

under the ROC curve (AUC), the DDoS dataset achieves the best results. The AUC estimates when the feature pruning method is applied to the datasets are shown in Figure 8.

Table 7. Summary of the experiment results for UNSW_NB15 dataset.

No.	**Feature Name**	**No.**	**Feature Name**	**No.**	**Feature Name**	**No.**	**Feature Name**
1	srcip	13	**dloss**	25	**trans_depth**	37	**ct_state_ttl**
2	**sport**	14	service	26	res_bdy_len	38	**ct_flw_http_mthd**
3	dstip	15	sload	27	sjit	39	**is_ftp_login**
4	dsport	16	dload	28	djit	40	ct_ftp_cmd
5	proto	17	**spkts**	29	stime	41	ct_srv_src
6	state	18	dpkts	30	ltime	42	ct_srv_dst
7	dur	19	**swin**	31	sintpkt	43	ct_dst_ltm
8	**sbytes**	20	**dwin**	32	**dintpkt**	44	ct_src_ ltm
9	**dbytes**	21	**stcpb**	33	tcprtt	45	ct_src_dport_ltm
10	sttl	22	**dtcpb**	34	synack	46	ct_dst_sport_ltm
11	**dttl**	23	**smeans**	35	ackdat	47	ct_dst_src_ltm
12	sloss	24	dmeans	36	is_sm_ips_ports	48	attack_cat

Table 8. Summary of the experiment results for UNSW_NB15 dataset.

Measure	Training/Testing (80/20 %)	
	All features	**Obtained features**
accuracy	0.8303	0.9641
error rate	0.1697	0.0359
fall-out	0.1916	0.0038
sensitivity/ recall	0.8459	0.9414
specificity	0.8084	0.9962
precision	0.8616	0.9971
F-measure	0.8536	0.9685

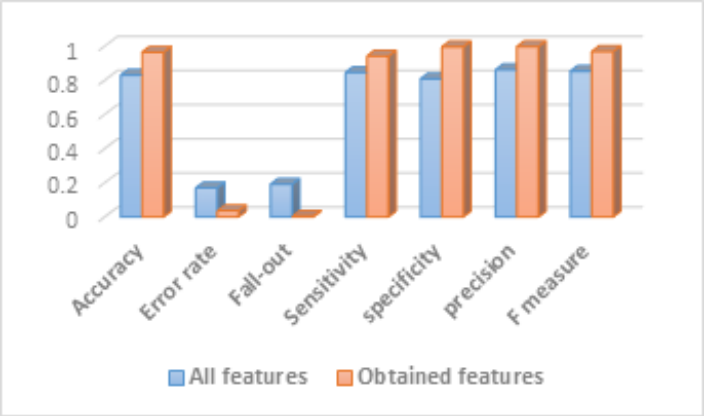

Figure 6. Comparison chart of the performance of all features and obtained features for UNSW_NB15 dataset.

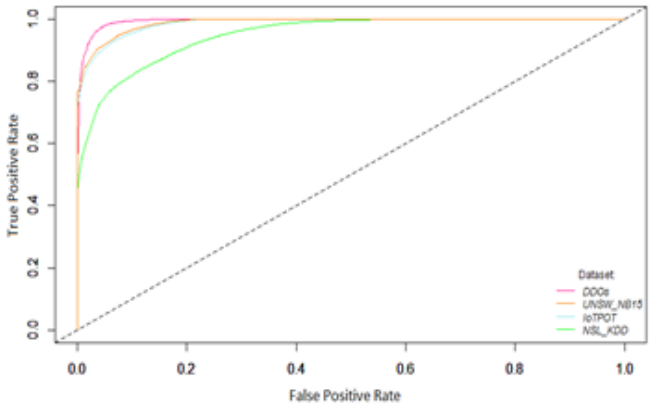

Figure 7. ROC curve of the HMM performance using the selected features on NSL-KDD, DDoS, IoTPOT and UNSW_NB15 datasets.

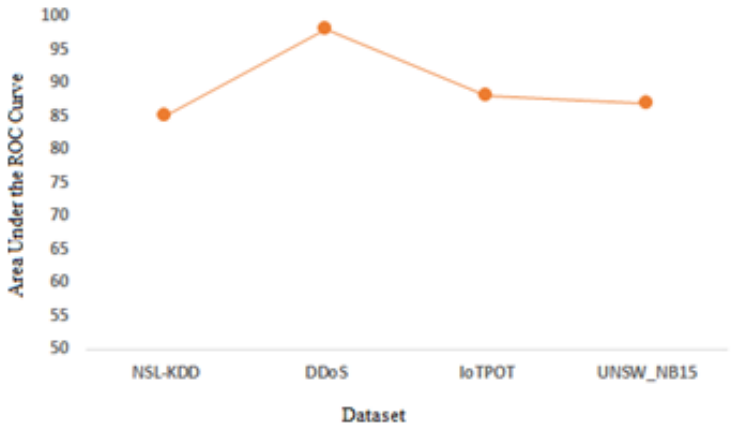

Figure 8. Performance (AUC) when feature-pruning method is applied on NSL-KDD, DDoS, IoTPOT, UNSW_NB15 datasets.

5. CONCLUSION

This paper proposed a feature pruning method for Hidden Markov Models to reduce the number of features and eliminate irrelevant, redundant or noisy features to overcome performance problems and improve the accuracy rate. In addition, this feature pruning method can effectively identify and determine the most significant feature set to be used for classification purposes. Experiment results were tested on four datasets: the NSL_KDD 2009, DDoS 2016, IoTPOT 2016 and UNSW_NB15 2015

datasets to show the superiority of our approach. The experiments demonstrated that the detection accuracy rate of using our approach is higher than the detection accuracy rate of full feature sets. In addition, false positive rate is lower than in full feature set. NSL_KDD 2009 produces 88.15% accuracy with 10 features. DDoS 2016 achieves 97.41% accuracy with 5 features. IoTPOT achieves 94.67% accuracy with 31 features. UNSW_NB15 achieves 96.41% accuracy with 16 features. As for the future work, we will focus on comparing the results derived from this study with other alternative machine learning methods. Meanwhile, we will continue to explore other datasets and flow-based features that could be used in order to improve the performance and achieve higher accuracy.

REFERENCES

[1] Z. Ghahramani, "An Introduction to Hidden Markov Models and Bayesian Networks," International Journal of Pattern Recognition and Artificial Intelligence, vol. 15, no. 1, pp. 9-42, 2001.

[2] Sulaiman Alhaidari, Ali Alharbi and Mohamed Zohdy, "Detecting Distributed Denial of Service Attacks Using Hidden Markov Models," International Journal of Computer Science Issues (IJCSI), vol. 15, no. 5, 2018.

[3] NSL-KDD Dataset, [online], Available: http://nsl.cs.unb.ca/nsl-kdd/.

[4] Pa, Yin Minn Pa et al., "Iotpot: A Novel Honeypot for Revealing Current IoT Threats," Journal of Information Processing, vol. 24, no. 3, pp. 522-533, 2016.

[5] N. Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Dataset for Network Intrusion Detection Systems," Proc. of the IEEE Military Communications and Information Systems Conference (MilCIS), Australia, 2015.

[6] A. Alharbi, S. Alhaidari and M. Zohdy, "Denial-of-Service, Probing, User to Root (U2R) and Remote to User (R2L) Attack Detection Using Hidden Markov Models," International Journal of Computer and Information Technology, 2018 (Submitted).

[7] X. Zeng, Y.-W. Chen, C. Tao and D. Alphen, "Feature Selection Using Recursive Feature Elimination for Handwritten Digit Recognition," Proc. of the 5th International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Kyoto, pp. 1205-1208, 2009.

[8] A. Alshammari et al., "Security Threats and Challenges in Cloud Computing," Proc. of the IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud), NY, USA, pp. 46-51, 2017.

[9] A. Alharbi et al., "Sybil Attacks and Defenses in Internet of Things and Mobile Social Networks," CyberHunt 2018: IEEE International Workshop on Big Data Analytics for Cyber Threat Hunting, Westin Seattle, WA, USA, 2018 (Submitted).

ملخص البحث:

إنّ انتقاء مزايا فعّالة وذات مغزى لنموذج ماركوف المخفيّ أمر ذو أهمية كبيرة لكشف الشذوذ في قواعد البيانات. يهدف هذا البحث إلى تحديد أبرز المزايا وأهمها في بناء نموذج ماركوف المخفيّ. ومن أجل تحسين أداء نموذج ماركوف المخفيّ، يقترح هذا البحث طريقة لتشذيب المزايا. والطريقة المقترحة فعالة في كشف أوجه الشذوذ وتصنيفها، وهي إلى جانب ذلك بسيطة جداً وسهلة التطبيق. ومن ناحية أخرى، فهي قادرة على خفض التعقيد الحسابي وتقليل الوقت دون الإضرار بدقة النموذج. في هذا العمل، يتم تطبيق الطريقة المقترحة على أربعٍ من مجموعات البيانات (UNSW_NB15; IoTPOT; DDoS; NSL-KDD) التي جرى استخدامها لإجراء دراسة مقارنة تتضمن مجموعة البيانات ذات المزايا الكاملة ومجموعة البيانات الفرعية التي تشتمل على المزايا المهمة. وقد أسفرت النتائج التجريبية عن أداءٍ أفضل من حيث الفعالية، ودقة أعلى، ومعدّل أقل للخطأ في الكشف عن المزايا، بالإضافة الى حذف المزايا المتكررة أو المشوشة.